A COMPREHENSIVE FRAMEWORK FOR EVALUATING VISION-BASED ON-BOARD RAIL TRACK DETECTION

Markus Ziegler Vishal Mhasawade Martin Köppel Philipp Neumaier Volker Eiselein
Digitale Schiene Deutschland
DB Netz AG
Berlin, Germany
markus-franz.ziegler@deutschebahn.com

***Abstract*—In this work a CNN-based rail track detection algorithm and two novel evaluation metrics are proposed. Rails define the region of interest for object detection and localization algorithms of rail-bound vehicles, like lane markings do for automotive driver assistance functions. Looking at the analogies in significance and appearance of both, it becomes apparent that rail and lane marking detection could be solved similarly. Hence, this paper firstly introduces rail detection using an adopted version of PINet, a regression net for lane marking detection. The network is completely re-trained using a novel loss function and our own railway dataset. Secondly, a post-processing approach for clustering the detected rails into tracks using geometric constraints is proposed. Finally, two track detection metrics are introduced: The rail position offset metric (RPOM) and the track centerline offset metric (TCOM), which allow precise assessment of rail and track centerline detection results and can be cornerstones to foster future developments in this area.**

***Index Terms*—railway, track detection, rail detection, regression network, detection metrics**

I. Introduction

Convolutional Neural Networks (CNN) have become the state-of-the-art in many computer vision tasks. Whether it is object detection, image segmentation, scene understanding or object tracking – the development of automotive advanced driver assistance systems (ADAS) and robotics has been and remains one of the main drivers for scientific advancements in these research areas. Other sectors are able to build on these developments while still having to cope with their individual requirements when transferring knowledge from automotive.

An important example to mention here is the rail sector where ADAS will presumably be one tool to reduce disruptions and to improve the system performance, especially the system capacity, on the way towards an increasingly automated future. With railway as a highly climate-friendly means of transportation, these developments are necessary in order to reduce the world's carbon footprint by shifting more goods and passengers to rail.

On-board ADAS systems for trains need to monitor the train environment in order to understand if an object is on the tracks or approaching/leaving them and derive a suitable reaction for the given situation. It is therefore indispensable to automatically determine the track course in a certain, velocity-dependent distance from the train, the so-called *ego-track*. While incidents on the ego-track might only be mitigated due to the long breaking distance of the train, monitoring the neighboring tracks can be used to warn other trains well ahead and thus to avoid accidents.

Fig. 1. Rail track detection result with the proposed framework. On the left, the rails, on the right the derived center line. Depicted in green are the outer rail edges of the ego-track, in blue those of the right neighboring rail track and in magenta the ground truth.

Calculating the track courses can partially be covered using a high precision map and localizing the vehicle on it by fusing satellite-, inertial- and odometry information with detected landmarks. However, when assessing the risk imposed by detected objects, deviations in lateral position and orientation can easily lead to unnecessary breaking maneuvers. A vision-based track detection can provide reliable information on the train's lateral position and orientation if the track is used as a landmark, and also allows a direct first risk assessment. In order to evaluate the absolute precision of the detector, an adequate metric is required that offers unambiguous indicators, like real-world deviations in centimeters.

Until today, only a few rail detection algorithms have been published. They mostly use segmentation networks or traditional feature extraction approaches and they are typically limited to ego-track detection. Although these papers usually use well established detection metrics, the measurements are hard to interpret as they are made in image-space only, e.g. [1].

In this work, we therefore propose a camera-based framework to detect the ego-track and its neighboring tracks from RGB camera images recorded at the train front (Figures 1 and 2). Based on a thorough state-of-the-art review (Chapter II), we adopt PINet [2], a CNN for lane-marking detection, for our task (Chapter III). The underlying assumption is that both, lane and track detection tasks share a similar structure because both rails and lane-markings usually come in pairs and visually stand out against their surroundings. In order to obtain meaningful tracks from individual rails, they

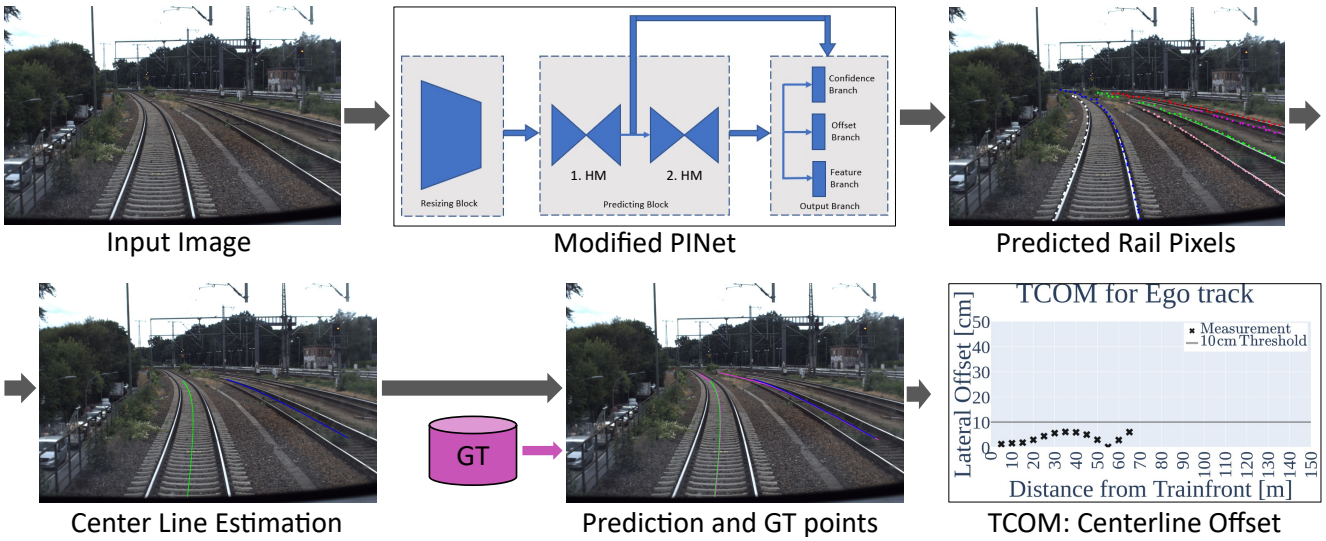

Fig. 2. Our proposed framework uses a modified version of PINet [2] to predict pixels along the rails. In an additional post-processing step our algorithm then estimates the centerline for the ego-track and its first neighbouring tracks. Our proposed novel TCOM metric determines the centerline deviation from ground truth and can be evaluated for different distances from the train front.

are post-processed using a clustering approach in the bird's eye view (BEV) which enables us to easily exploit the geometrical constraints that rail tracks adhere to: Parallel rails with a fixed distance between them. Our data and evaluation procedure is shown in Section IV. To evaluate the performance in meaningful real-world measurement units we propose two novel metrics as another key contribution in this work: Rail Position Offset Metric (RPOM) and Track Centerline Offset Metric (TCOM). Using also the BEV, they allow calculating the absolute offset between a detected rail or track centerline and the ground truth and thus enable an objective and accurate evaluation (Chapter V). We conclude the paper with an outlook on open challenges and future work.

II. Problem Statement and Related Work

In this section, we provide a short review of state-of-the-art approaches for our application. For the remainder of this paper, we refer to the term *rail detection* for identifying a single rail in an image or data frame while *track detection* covers the task of finding a complete track, i.e. two parallel rails which can be used by one train. Given this definition, a valid output for track detection is either lines along the edges of the rails of a track or the *centerline* of a track defined as the virtual line in the middle of its two rails. We distinguish between the *ego-track* used by an automated train and *neighboring tracks* which could exist to both sides of it.

As rail track detection has not yet been extensively covered in the literature, we also present state-of-the-art lane detection approaches for automotive applications which are a technically similar field. In both areas, traditional computer vision (CV) methods have advantages from a safety perspective thanks to their explainability. On the other hand, deep learning-based approaches usually perform better but will introduce additional effort for homologation of safety-critical systems such as trains. We conclude the chapter with a discussion of related metrics and datasets.

In the literature, there are a few examples on camera-based rail track detection which could be used for driver assistance systems. A recent and comprehensive overview is given in [3] which also covers further aspects as e.g. object detection in rail contexts. In [4], track detection using dynamic programming is proposed. [5], [6] use a recursive approach and a monofocal camera to detect turnouts and to improve the localization of the train. Recently, a photogrammetric approach has been presented in [7] which fuses known map data with track estimates from the train. As a similar approach, an inverse projection mapping can be used for track detection from a train front camera performing the track detection in a bird's eye view and with geometric constraints [8], [9], respectively.

Apart from train-mounted cameras, satellite and airborne imagery have been used for maintenance [10] [11] and inspection of rail tracks [12]. Also, other modalities can be used, e.g. an adaptive edge detection by a genetic algorithm for infra-red images proposed in [13].

Next to classical CV methods, Deep Neural Network (DNN) approaches have been developed for rail detection, e.g. segmentation approaches. Wang *et al.* [14] use semantic segmentation and polygon fitting to detect especially the ego-track. They enhance the well-known SegNet [15] by adding dilated cascade connections and cascade downsampling. A similar approach to rail detection was taken by Wang *et al.* in RailNet [1] who enhanced the ResNet [16] feature extractor by including a pyramid aggregation scheme. In a recent work, Yang *et al.* [17] combine rail detection and train detection to propose an active safety system for railways. However, state-of-the-art rail detection methods still focus a lot on ego-track detection. To our knowledge, there are no approaches aiming specifically at the detection of neighboring tracks.

While rail track detection is a rather young field of research, lane detection approaches for automotive are technically similar and have attracted more interest in the past, i.e. [18] and [19]. Respective DNN methods can be clustered into two categories: Semantic segmentation networks and regression networks. Many researchers

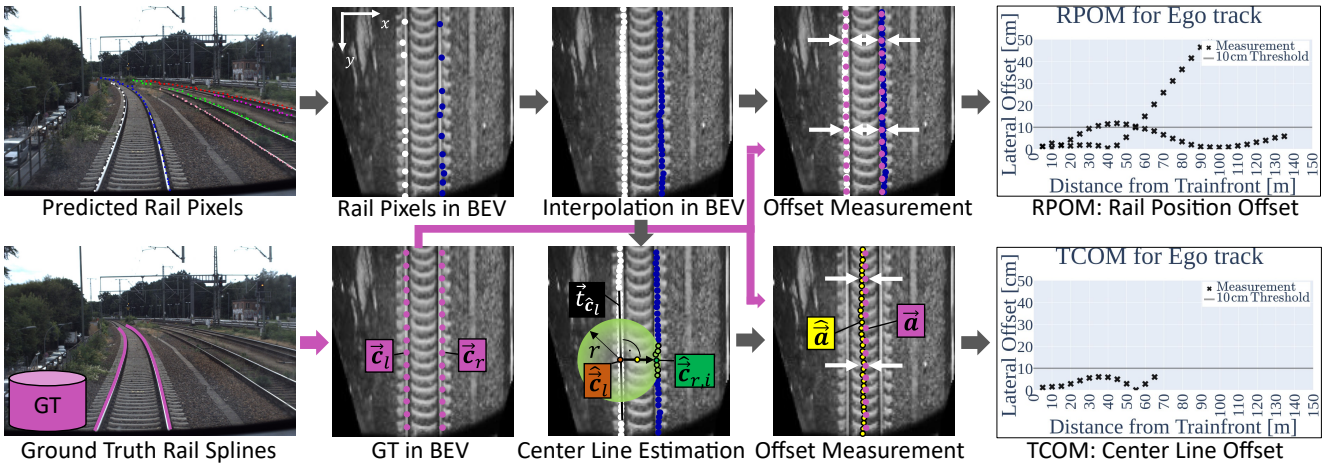

Fig. 3. Process of RPOM and TCOM calculation in BEV. The GT centerline is calculated in the same way as the centerline derived from the predictions. The RCOM graph shows that one of the rail detections of the ego track drifts off, while the other stays close to the GT. For the TCOM this means that the centerline calculation is stopped at 65 m, where the offset between the rails starts getting too big to be physically possible.

have extended segmentation techniques for lane detection, i.e. [2], [20]–[22], [22]–[27].

From the mentioned segmentation-based publications only [20], [26], [27] are capable of detecting a variable number of lanes which is important in the rail context with potentially multiple parallel tracks. As [20] and [26] use binary classification, their output requires furthermore heavy post-processing before it is made parametric. Hence, also regression-based approaches have been investigated to provide a scalable approach. A prominent regression-based method are polynomial regression networks like [28], [29]. In [29] Tabelini *et al.* proposed LaneATT [30] which uses an anchor-based pooling mechanism for multi-scale feature extraction. In order to detect a variable number of rails a regression-based network was selected for this paper.

To our knowledge, the semantic segmentation offering dataset RailSem19 [31] is the only publicly available railway dataset. It offers many annotated object classes but does not focus specifically on rail tracks. This might be why no distinct metrics are available and why papers use indicators like Intersection over Union (IoU), precision and recall on rail-pixel basis [1]. However, two aspects make RailSem19 unsuitable for our use case: Firstly, the dataset consists of videos recorded out of drivers' cabins from all over the world and is therefore missing external and internal camera calibration information. Secondly, the rail labels are arbitrarily thickened splines along the rails, so the precise locations of the tracks' rail edges are unknown. Both facts hinder measuring precise absolute deviations.

For automotive lane detection, benchmarks like CVPR 2017 TuSimple [32] and CULane [22] have been introduced. Commonly, the evaluation follows the principles from [22] where the lane markings are evaluated calculating the IoU between arbitrarily thickened lines of constant width (e.g. 30 pixels). This might be a suitable metric for automotive applications but in not useful for a train case were high precision even at long ranges is necessary, e.g. to determine whether a pedestrian is too close to the tracks or not. Additionally, when using detected tracks as landmarks to accurately localize the ego train on a high-precision map, the 30-pixel width for ground truth used in the evaluation introduces undesired bias.

As rail tracks always have the same width in contrast to lanes on different roads, it appears also questionable to consider IoU errors for estimating one single rail in the metric for a track as a whole. In contrast to automotive applications, the individual rails can always be extracted using the fixed track width and an accurate centerline.

Other metrics appear even less suitable for the rail context: While a region of interest is obviously only a very rough form of ground truth for a long and potentially curved object such as a rail track, a segmentation mask also has flaws. It offers the benefit of an easy pixel-wise accuracy computation but the exact position of a rail track centerline does only slightly depend on its segmentation mask. One or more additional pixels at the border of the mask may influence its IoU score or accuracy value but usually have a very limited effect on the centerline which is the actual geometric result of interest. Additional difficulties are quantization issues and bias introduced especially in the far field where a single mask pixel would affect the IoU very little but attributes much more weight to the centerline estimate than in areas close to the train. This problem is similar for the CULane metric.

Overall, we conclude that a new evaluation scheme and new metrics are needed for rail track detection. Our proposal is outlined in Chapter IV.

III. Proposed Track Detection Framework

The proposed rail detection framework uses the Point Instance Network (PINet) [2] as its backbone. In its original form, PINet makes use of keypoint estimation and instance segmentation approaches to detect pixels along lane markings in camera images. For our framework we adopted and re-trained the network to detect points on rails. Indeed, the network detects points on the outer edge of a railhead to take advantage of the higher contrast as compared to the inner edge, but for simplicity, we will use the terms *rail* and *outer rail edge* interchangeably here.

The detected rail pixels are postprocessed to filter out false positives and clustered into meaningful tracks using the geometrical constraints of a rail network. The final output can be either points along the outer rail edges or points along the centerline, in each case

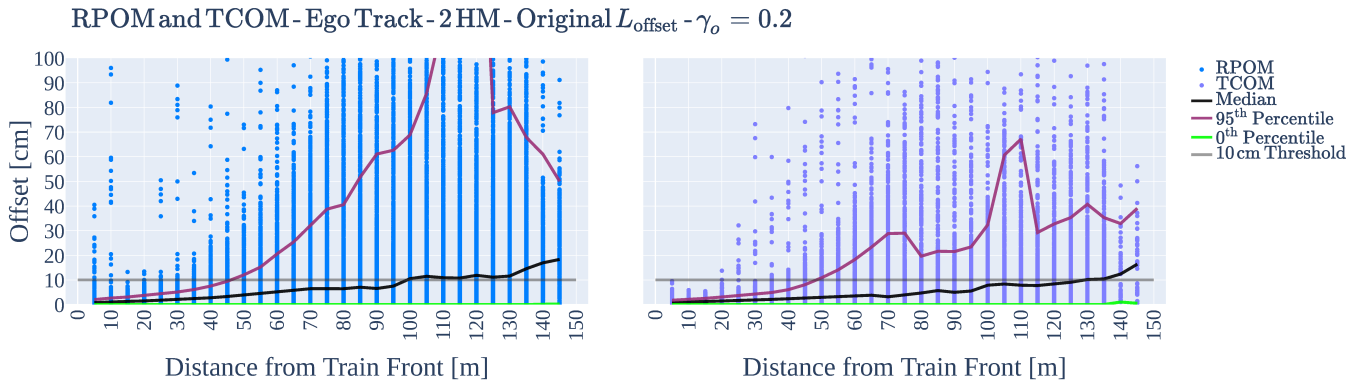

Fig. 4. RPOM (right) and TCOM (left) plots with median, 0$^{\text{th}}$ and 95$^{\text{th}}$ percentile at every 5 m from the train front. Within the first 50 m, 95% of the TCOM values are within 10 cm.

for 1-3 tracks: the ego-track and – if present – the first neighbouring track on each side.

The resizing block of PINet [2] takes an RGB image of resolution 512×256 px as input and compresses it to a feature map of size 64×32. The following prediction block consists of a number of encoder-decoder blocks (*hourglass modules*) made up of three types of bottleneck blocks, namely *Down*, *Same* and *Up* bottleneck used for downsampling, feature aggregation and decoding: We adopt these bottleneck blocks but add a dropout layer before the last convolution layer (size=1, padding=0, stride=1) to reduce overfitting issues and reduce the number of hourglass modules from 4 to 2 to account for the yet limited size of our dataset. We keep the original output branches for confidence, offset and feature prediction.

Each branch has its own loss function, however in contrast to the original PINet, the total loss L_{total} is the weighted sum of only three loss components:

$$\begin{aligned} L_{\text{total}} = & \underbrace{\gamma_{\text{e}}\, L_{\text{exist}} + \gamma_{\text{n}}\, L_{\text{non-exist}}}_{= L_{\text{confidence}}} + \gamma_{\text{o}}\, L_{\text{offset}} \\ & + \underbrace{\gamma_{\alpha}\, L_{\text{SISC}} + \gamma_{\beta}\, L_{\text{DISC}}}_{= L_{\text{feature}}} + \cancel{\gamma_{\text{d}}\, L_{\text{distillation}}}\,, \end{aligned} \tag{1}$$

with $\gamma_{\text{e}}, \gamma_{\text{n}}, \gamma_{\text{o}}, \gamma_{\text{f}}, \gamma_{\alpha}$, and γ_{β} being the respective weight factors. We remove the original $L_{\text{distillation}}$, because its application for *teacher-student* training is out of our scope. $L_{\text{confidence}}$ and L_{feature} are the original confidence and feature loss functions for which we refer to [2], while we propose to alter the offset loss L_{offset} for our application.

In order to account for the fact that in the camera perspective an offset between a predicted rail pixel $\hat{\vec{c}} = (\hat{c}_x, \hat{c}_y)$ and a GT rail pixel $\vec{c} = (c_x, c_y)$ generates a bigger error far from the train than the same pixel offset in close distance, we apply a weighting based on the rail pixel's distance to the train front (Equation 2). It is achieved by deploying a weight grid W of the same size as the ground truth grid. The loss is applied to cells G_e containing GT points and normalized using the number N_e of these cells:

$$\begin{aligned} L_{\text{offset}} = & \frac{1}{N_e} \left(\sum_{c_x \in G_e} \left((c_x - \hat{c}_x)\, w_{c_x} \right)^2 + \sum_{c_y \in G_e} (c_y - \hat{c}_y)^2 \right), \\ & w_{c_x} \in \text{W} = \underset{32 \times 64}{\begin{bmatrix} 2.53 & \dots & 2.53 \\ \vdots & & \vdots \\ 1.0 & \dots & 1.0 \end{bmatrix}}. \end{aligned} \tag{2}$$

Because the BEV transformation is linear, all columns of W are set to be equal and increase linearly. We arbitrarily choose them to increase from 1.0 to 2.53, resulting in a factor of 1.5 in 150 m longitudinal distance from the train front.

The output of PINet is a set of clustered points on the rails. Using a flat-world assumption and an extrinsic sensor calibration, we transform these points $\hat{\vec{c}}_i$ into points $\hat{\vec{\mathbf{c}}}_i$ in a bird's eye view (BEV) at railhead level. Rails are extracted in the BEV by fitting a cubic b-spline onto the predictions including outlier effect reduction using a smoothing factor [33] that increases with the distance from the train front.

For track extraction, the rails are sorted from left to right and we associate right to left rails in order to form pairs. The pairing-condition is that a potential right rail needs to have at least 30 points in a distance close to the track width to their left counterparts. We only allow for unambiguous assignments, otherwise the left rail is discarded completely, leading to single false positive rails being excluded from further processing.

Finally, the track center lines are estimated: For every predicted point $\hat{\vec{\mathbf{c}}}_l$ on the left rail, a radius search using the known geometric track constraints identifies all n predicted points on the right rail $\mathcal{R} = [\hat{\vec{\mathbf{c}}}_{r,1} \dots \hat{\vec{\mathbf{c}}}_{r,n}]$ and their connecting vectors $\overrightarrow{\hat{\vec{\mathbf{c}}}_l \hat{\vec{\mathbf{c}}}_{r,i}}$. The vector $0.5 \cdot \overrightarrow{\hat{\vec{\mathbf{c}}}_l \hat{\vec{\mathbf{c}}}_{r,i}}$ forming the closest $90^\circ \pm 2^\circ$ angle with the rail-tangent $\vec{t}_{\hat{\mathbf{c}}_l}$ in $\hat{\vec{\mathbf{c}}}_l$ then determines the respective centerline point estimate $\hat{\vec{\mathbf{a}}}$ (Figure 3). The extracted tracks are assigned the labels "ego" / "right" / "left" based on their lateral centerline position in BEV close to the train.

IV. Dataset and Evaluation Metrics

For evaluation of our method, we use data obtained from the aTL (advanced TrainLab) [34] which is a modified SIEMENS/Bombardier series BR605 (ICE-TE) test train serving as a platform for mounting synchronized sensors of four modalities

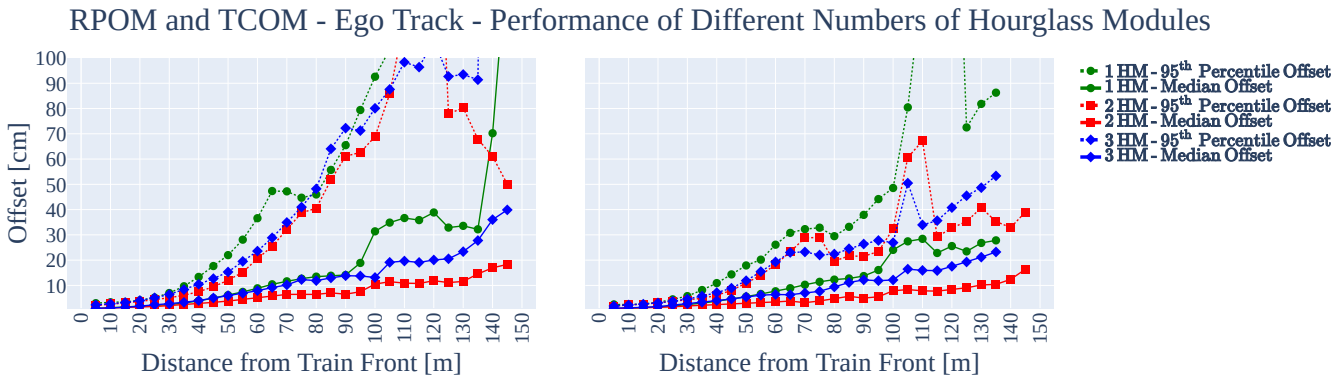

Fig. 5. RPOM (right) and TCOM (left) graphs with median and 95$^{\text{th}}$ percentile offset values for different numbers of hourglass modules. Using two hourglass modules shows the best performance.

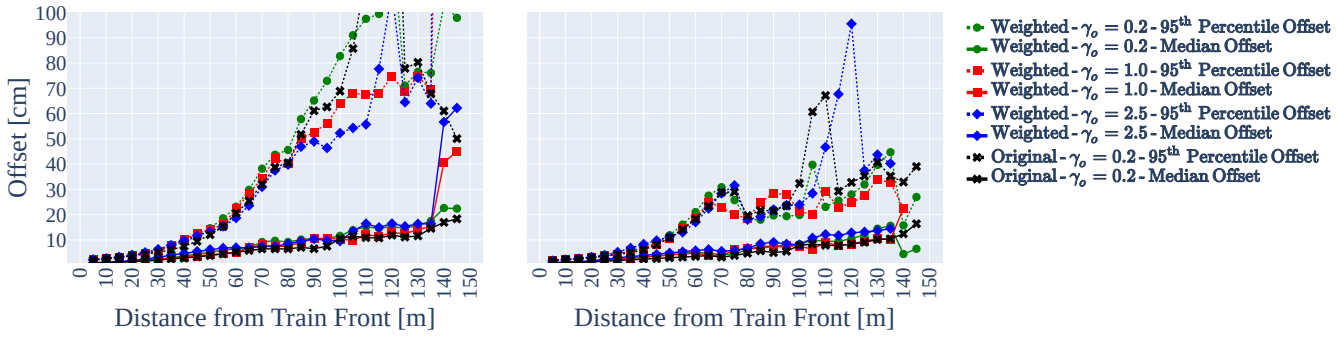

Fig. 6. RPOM (left) and TCOM values (right) for different offset loss configurations: The proposed weighted offset loss function with $\gamma_0 \geq 1.0$ tends to be more precise and shows less variation. The black curves depict the baseline performance (two hourglass modules and original offset loss function) that was determined in the first experiment (see Figure 5).

(RGB camera, IR camera, lidar and radar) which were intrinsically and extrinsically calibrated.

The dataset contains ten different scenes from rail tracks recorded in the area of Berlin, Germany, and shows suburban and city railway environment. In total, it consists of 18,043 video frames (≈ 30 min), of which 80 % were used as training data and 20 % served as test and validation data.

In addition to railway-related objects like tracks, switches, transitions, passengers, poles, trains etc. a multitude of non-railway-related objects in surrounded areas were recorded and annotated, resulting in a realistic challenge for computer vision tasks in a rail environment. The rail edges were annotated with centripetal Catmull-Rom-Splines [35] and the track centerlines were derived from those edges with the same method used in our track detection postprocessor.

As stated in Section II, we found the existing metrics to be insufficient for evaluating rail track detection using rail edges or track center lines. Therefore, we come up with a new evaluation framework including two novel metrics: The Rail Position Offset Metric (RPOM), which measures the lateral offset of the detected outer rail edges, and the Track Center line Offset Metric (TCOM) that measures the lateral offset of the track center line. An important advantage of those metrics is their output of a real world-coordinate accuracy (e.g. in cm) at predefined longitudinal distances from the train front which enables easy and meaningful statistical analyses and interpretation. In other words, the metrics are suited to evaluate rail track detection precision at all distances equally: Close to and far from the train front. The absolute value also helps to assess the system performance for high-precision landmark-based localization.

The metrics are computed in the calibrated BEV (see Section III) which constitutes an important cornerstone for the evaluation framework. By mapping both, center lines / rails and ground truth (GT) into the BEV, the deviations are simple to compute and straightforward.

For a track jointly present in GT and predictions, we define RPOM for the predicted rail point $\hat{\vec{\mathbf{c}}} = (\hat{\mathbf{c}}_x, \hat{\mathbf{c}}_y)$ as the lateral offset Δx to the ground truth point $\vec{\mathbf{c}} = (\mathbf{c}_x, \mathbf{c}_y)$ (see Figure 3):

$$\text{RPOM}(\mathbf{c}_y) = |\mathbf{c}_x - \hat{\mathbf{c}}_x|_{\text{in BEV}} \,. \tag{3}$$

With the BEV-inherent world coordinate metrics, the RPOM at any given point directly translates into centimeters. Therefore it is easy to compute statistics, e.g. variances.

TCOM is similar to RPOM but uses the offset between a ground truth centerline point $\vec{\mathbf{a}} = (\mathbf{a}_x, \mathbf{a}_y)$ and an estimated centerline point $\hat{\vec{\mathbf{a}}} = (\hat{\mathbf{a}}_x, \hat{\mathbf{a}}_y)$ of matching tracks:

$$\text{TCOM}(\mathbf{a}_y) = |\mathbf{a}_x - \hat{\mathbf{a}}_x|_{\text{in BEV}} \,. \tag{4}$$

For analysis, RPOM and TCOM values are plotted over distance from the train front (single-frame results in Figures 2 and 3). All

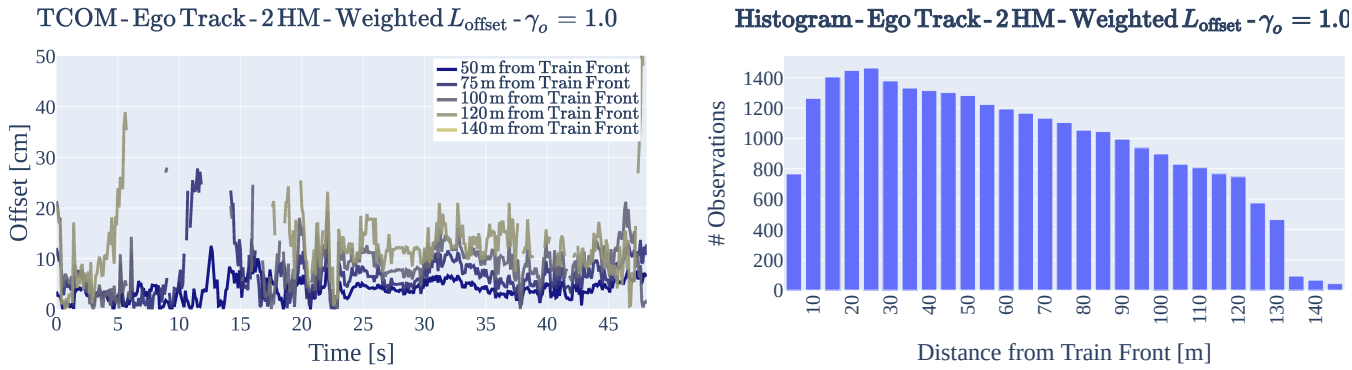

Fig. 7. Left: Plotting TCOM values for the ego-track in all frames of a video clip at distinct distances over time shows foresight and offset at a given time. Thus it allows to identify challenging situations, e.g. between 5 and 15 s. Right: Analysis of the maximum ego-track detection distance in all frames of the dataset by constructing a histogram of the farthest center line points that were calculated with TCOM.

results in Chapter V accumulate TCOM and RPOM values over the full dataset and thus allow for a detailed statistical analysis (see Figure 4). Analyzing a video clip for challenging situations can be done easily by plotting framewise RPOM- or TCOM offsets at given distances from the train front over time (see Figure 7). The diagram reveals video-sections with high offsets and also sections where only the first meters of a track where detected (then the brighter lines indicating farther distances are missing).

From our experience, TCOM values are smoother and less affected by outliers. This is based on the implicit averaging between left and right rail which evens out small errors done in rail detection. Both metrics together allow for a detailed analysis of a track detection system as the statistics for individual rails and final tracks can be assessed independently.

V. Experimental Results

Due to low contrast, every image in our dataset is enhanced using a CLAHE operation [36] to equalize the image histogram before randomly applying the following augmentation methods during training: Translation, rotation, flipping, change of intensity, shadowing and adding of Gaussian noise. Subsequently, all images are resized to the network input size,

Using experimentally determined hyperparameters, we trained for 850 epochs with a batch size of 6 images per GPU. An Adam optimizer, a cyclic learning rate scheduler with a learning rate $\in \left[1e^{-9}, 1e^{-5}\right]$, a weight decay of $4e^{-3}$ and a dropout rate of 0.5 for all dropout layers are used in the experiments. The weight factors of the different parts of the total loss function L_{total} (Equation (1)) are set to $\gamma_{\text{e}} = 2.5, \gamma_{\text{n}} = 1.0, \gamma_{\text{o}} = 0.2, \gamma_{\alpha} = 0.9$ and $\gamma_{\beta} = 0.5$. We choose the confidence threshold for predicted rail pixels as 0.5 and keep the distance threshold for distinguishing the rail instances at 0.08 as in original PINet.

In two experiments we investigate the effects of the proposed alterations: The reduced number of hourglass modules, and the configuration of the improved offset loss function. The influence of different numbers of houglass modules can be seen in Figure 5 showing the accumulated RPOM and TCOM values for the ego track at every 5 m from the train front. Over practically all distances, the two-hourglass module configuration delivers the most precise rail pixel positions. This is opposed to the results from the original PINet paper, where datasets with similar sizes as compared to ours performed best with three or four modules. We assume that the limited variance of our dataset quickly leads to overfitting. Figure 5 also supports our hypothesis that estimating a center line from a pair of rails increases the precision as single-rail errors cancel each other out.

The new offset loss function introduced in Equation (2) penalizes lateral prediction imprecision harder with increasing distance from the train front. Figure 6 shows the RPOM and TCOM results for different offset loss functions and values of γ_{o}. Using the two hourglass network configuration from the previous experiment as the new baseline, the figure shows that the proposed weighted offset loss function works as intended, improving precision at longer ranges, as long as $\gamma_{\text{o}} \geq 1.0$. For RPOM, $\gamma_{\text{o}} = 2.5$ is best at long ranges. For TCOM however, $\gamma_{\text{o}} = 1.0$ shows the smallest long-range offsets. We assume that this is caused by predictions with $\gamma_{\text{o}} = 2.5$ tending to have an offset to the same side which is not evened out.

Comparing RCOM and TCOM with standard metrics like IoU (see Table I) we observe a lack of variation in the ego-track IoU values. This clearly indicates that the one-dimensional metric is not an adequate performance measure to judge the course and precision of the detected track centerline.

Offset Loss Function (used γ_{o})	Average IoU		
	ego track	left neighbour	right neighbour
original (0.2)	0.806	0.776	0.540
weighted (0.2)	0.812	0.715	0.490
weighted (1.0)	0.829	0.764	0.481
weighted (2.5)	0.818	0.711	0.458

TABLE I
IoU for the area between the rails, calculated in BEV.

Figure 7 shows further meaningful, yet easily generated statistics from RPOM and TCOM. In the left graph, we analyze all frames of a video clip in succession and plot the TCOM offsets (y-axis) at certain distances from the train front over time (x-axis). We can see center line detections up to 120 m with $\leq$ 20 cm offset over almost the complete course of time. This foresight only drops to 50 m in the time window of 5 s to 20 s, where the ego track takes a sharp bend. We see that entering the curve at 5 s leads to increased offsets compared to the straight track in the remainder of the scene.

In the right graph of Figure 7 we see a histogram of the maximum detection distance in the frames of the complete dataset.

The ego track detection distance is usually bigger than 5 m and only rarely greater than 120 m. On the one hand this is due to PINet's difficulties to assign rail pixels at great distances to the correct rails, but on the other hand our post-processing discards all rail pixels outside the BEV image (i.e. farther than 150 m) which shortens the rail spline.

Figure 8 shows some more exemplary rail track detection results.

VI. Conclusion

We presented a framework for camera-based detection of multiple rail tracks and a consistent approach for its evaluation. We showed that the well-known PINet lane detector can be adapted to this task and trained it on our rail dataset. The improved offset loss function enhances the precision of rail detections in farther distances from the train front. Our simple post processing works, but needs improvement at spline interpolation and edge cases like rail track switches. Because existing detection metrics like IoU using segmentation masks fail to allow an unambiguous evaluation of the the track center line, we proposed the new comprehensive RPOM and TCOM metrics which evaluate track detection results in easily understandable real-world coordinates and deviations. Those metrics should enable researchers not only to benchmark rail track detection performance with clearly interpretable metrics, but also to develop precise rail track detectors and localization algorithms using the tracks as landmarks in high precision maps. Future research should enhance our metrics to also work in non-flat environments and investigate RPOM-like metrics in a loss function. We will build on this work by publishing a standard rail dataset in the future that will also include other sensor modalities.

References

[1] Y. Wang, L. Wang, Y. Hu, and J. Qiu, "Railnet: A segmentation network for railroad detection," *IEEE Access*, vol. 7, pp. 143 772–143 779, 2019.

[2] Y. Ko, Y. Lee, S. Azam, F. Munir, M. Jeon, and W. Pedrycz, "Key points estimation and point instance segmentation approach for lane detection," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10, 2021.

[3] D. Ristić-Durrant, M. Franke, and K. Michels, "A review of vision-based on-board obstacle detection and distance estimation in railways," *Sensors*, vol. 21, no. 10, 2021.

[4] F. Kaleli and Y. S. Akgul, "Vision-based railroad track extraction using dynamic programming," in *2009 12th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 2009.

[5] R. Ross, "Vision-based track estimation and turnout detection using recursive estimation," in *13th International IEEE Conference on Intelligent Transportation Systems*, 2010, pp. 1330–1335.

[6] ——, "Track and turnout detection in video-signals using probabilistic spline curves," in *2012 15th International IEEE Conference on Intelligent Transportation Systems*, 2012, pp. 294–299.

[7] P. Glira, K. Ölsböck, T. Kadiofsky, M. Schörghuber, J. Weichselbaum, C. Zinner, and L. Fel, "Photogrammetric 3d mobile mapping of rail tracks," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 183, pp. 352–362, 2022.

[8] Z. Wang, X. Wu, Y. Yan, C. Jia, B. Cai, Z. Huang, G. Wang, and T. Zhang, "An inverse projective mapping-based approach for robust rail track extraction," in *2015 8th International Congress on Image and Signal Processing (CISP)*. IEEE, 2015.

[9] Z. Wang, B. Cai, J. Chunxiao, C. Tao, Z. Zhang, Y. Wang, S. Li, F. Huang, S. Fu, and F. Zhang, "Geometry constraints-based visual rail track extraction," in *2016 12th World Congress on Intelligent Control and Automation (WCICA)*. IEEE, 2016.

[10] A. Javed, K. A. Qazi, M. Maqsood, and K. A. Shah, "Efficient algorithm for railway tracks detection using satellite imagery," *International Journal of Image, Graphics and Signal Processing*, vol. 4, no. 11, pp. 34–40, 2012.

[11] M. Neubert, R. Hecht, C. Gedrange, M. Trommler, H. Herold, T. Krüger, and F. Brimmer, "Extraction of railroad objects from very high resolution helicopter-borne lidar and ortho-image data," *GEOBIA*, pp. 6–7, 2008.

[12] M. Singh, S. Singh, J. Jaiswal, and J. Hempshall, "Autonomous rail track inspection using vision based system," in *2006 IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety*. IEEE, 2006.

[13] M. Pavlović, V. Nikolic, M. Simonovic, V. Mitrović, and I. Ciric, "Edge detection parameter optimization based on the genetic algorithm for rail track detection," in *Facta Universitatis - Series: Mechanical Engineering*, 2019.

[14] Z. Wang, X. Wu, G. Yu, and M. Li, "Efficient rail area detection using convolutional neural network," *IEEE Access*, vol. 6, pp. 77 656–77 664, 2018.

[15] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 2481–2495, 2015.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[17] Z. Yang, V. Cheung, C. Gao, and Q. Zhang, *Train Intelligent Detection System Based on Convolutional Neural Network*. Springer International Publishing, 2020, pp. 157–165.

[18] D. Liang, Y. Guo, S. Zhang, T.-J. Mu, and X. Huang, "Lane detection: A survey with new results," *Journal of Computer Science and Technology*, vol. 35, pp. 493–505, 2020.

[19] J. Tang, S. Li, and P. Liu, "A review of lane detection methods based on deep learning," *Pattern Recognition*, vol. 111, p. 107623, 2021.

[20] Q. Zou, H. Jiang, Q. Dai, Y. Yue, L. Chen, and Q. Wang, "Robust lane detection from continuous driving scenes using deep neural networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, p. 41–54, 2020.

[21] T. Zheng, H. Fang, Y. Zhang, W. Tang, Z. Yang, H. Liu, and D. Cai, "Resa: Recurrent feature-shift aggregator for lane detection," in *AAAI Conference on Artificial Intelligence*, 2020.

[22] X. Pan, J. Shi, P. Luo, X. Wang, and X. Tang, "Spatial as deep: Spatial cnn for traffic scene understanding," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[23] Y. Hou, Z. Ma, C. Liu, and C. Loy, "Learning lightweight lane detection cnns by self attention distillation," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE Computer Society, nov 2019, pp. 1013–1021.

[24] M. Ghafoorian, C. Nugteren, N. Baka, O. Booij, and M. Hofmann, "El-gan: Embedding loss driven generative adversarial networks for lane detection," in *Computer Vision – ECCV 2018 Workshops*, L. Leal-Taixé and S. Roth, Eds. Springer International Publishing, 2019, pp. 256–272.

[25] T. Michalke, D. Feng, C. Gläser, and F. Timm, "Where can i drive? a system approach: Deep ego corridor estimation for robust automated driving," 2021.

[26] S. Lee, J. Kim, J. S. Yoon, S. Shin, O. Bailo, N. Kim, T.-H. Lee, H. S. Hong, S.-H. Han, and I. S. Kweon, "Vpgnet: Vanishing point guided network for lane and road marking detection and recognition," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 1965–1973.

[27] D. Neven, B. D. Brabandere, S. Georgoulis, M. Proesmans, and L. V. Gool, "Towards end-to-end lane detection: an instance segmentation approach," *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 286–291, 2018.

[28] B. Wang, Z. Wang, and Y. Zhang, "Polynomial regression network for variable-number lane detection," in *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Springer International Publishing, 2020, pp. 719–734.

[29] L. Tabelini, R. Berriel, T. M. Paixão, C. S. Badue, A. F. de Souza, and T. Oliveira-Santos, "Polylanenet: Lane estimation via deep polynomial regression," *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 6150–6156, 2020.

[30] L. Tabelini, R. Berriel, T. M. Paixao, C. Badue, A. F. D. Souza, and T. Oliveira-Santos, "Keep your eyes on the lane: Real-time attention-guided lane detection," in *2021 IEEE/CVF Conference on Computer*

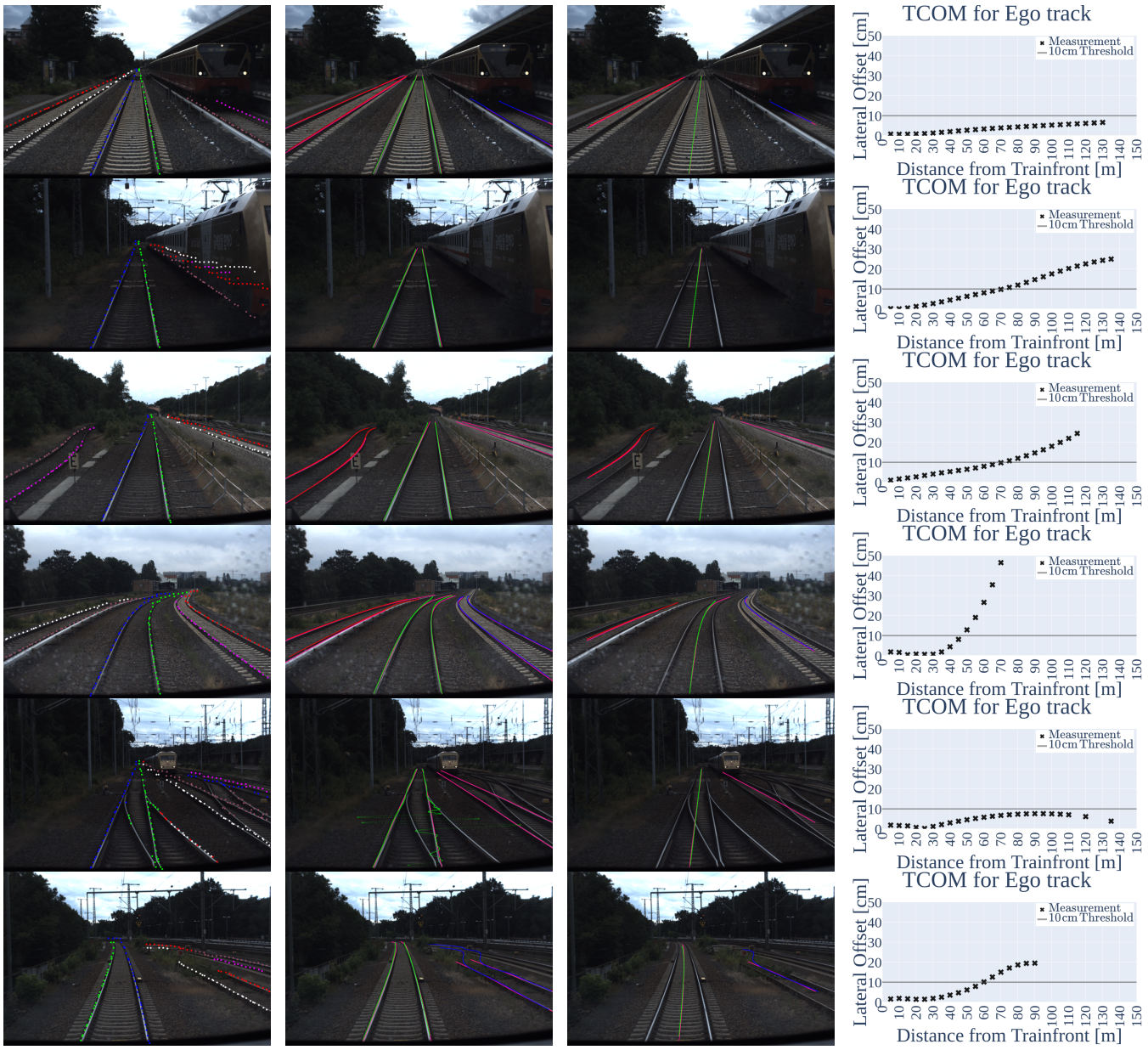

Fig. 8. Exemplary detections (left to right): Detected rail pixels, detected tracks, centerlines deduced and associated TCOM graph for that frame (GT in magenta). In row 1 all tracks are detected and the right neighbor track ends just right beneath the S-Bahn train. In row 2 PINet detects lots of rails on the side of a train, but our post processing filters them out due to their distances to each other. In row 3 we see good performance on the left double-S-shaped neighbor track but poor rail pixel clustering at the right neighbor track. Row 4 shows good detections at the front of the bend but the foresight drops due to faulty clustering of the rail pixels in the distance. Edge cases like switches (row 5) and buffer stops (row 6) need additional handling in the post-processing.

Vision and Pattern Recognition (CVPR). IEEE Computer Society, 2021, pp. 294–302.

[31] O. Zendel, M. Murschitz, M. Zeilinger, D. Steininger, S. Abbasi, and C. Beleznai, "Railsem19: A dataset for semantic rail scene understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[32] tusimple. (2017) TuSimple: Lane detection benchmark. [Online]. Available: https://github.com/TuSimple/tusimple-benchmark/tree/master/doc/lane_detection

[33] P. Dierckx, "Curve and surface fitting with splines," in *Monographs on numerical analysis*, 1994.

[34] Deutsche Bahn AG. (2021) advanced TrainLab: The fastest lab on rails. [Online]. Available: https://www.deutschebahn.com/en/Digitalization/The-fastest-lab-on-rails-6935126

[35] P. J. Barry and R. N. Goldman, "A recursive evaluation algorithm for a class of catmull-rom splines," in *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '88. Association for Computing Machinery, 1988, p. 199–204.

[36] K. Zuiderveld, "Contrast limited adaptive histogram equalization," in *Graphics Gems IV*. Academic Press Professional, Inc., 1994, p. 474–485.