Application of Rail Segmentation in the Monitoring of Autonomous Train's Frontal Environment

Mohamed Amine Hadded, Ankur Mahtani, Sébastien Ambellouis, Jacques Boonaert, Hazem Wannous

Application of Rail Segmentation in The Monitoring of Autonomous Train's Frontal Environment*

Mohamed Amine Hadded[1][0000−0002−0375−1809], Ankur Mahtani[1][0000−0003−2738−4705], Sébastien Ambellouis[2][0000−0002−3719−1934], Jacques Boonaert[3][0000−0002−8594−6959], and Hazem Wannous[4][0000−0001−8475−4309]

[1] FCS Railenium, F-59300 Famars, France
mohamed-amine.hadded,ankur.mahtani@railenium.eu
[2] Gustave Eiffel University, Villeneuve d'Ascq, Lille, France
sebastien.ambellouis@univ-eiffel.fr
[3] IMT Lille-Douai, Douai, France
jacques.boonaert@imt-lille-douai.fr
[4] IMT Lille Douai, Villeneuve-d'Ascq, France
hazem.wannous@imt-lille-douai.fr

Abstract. One of the key factors in achieving an autonomous vehicle is understanding and modeling the driving environment. This step requires a considerable amount of data acquired from a wide range of sensors. To bridge the gap between the Roadway and Railway fields in terms of datasets and experimentation, we provide a new dataset called RailSet as the second large dataset after Railsem19, specialized in Rail segmentation. In this paper we present a multiple semantic segmentation using two deep networks UNET and FRNN trained on different data configuration involving RailSet and Railsem19 datasets. We show comparable results and promising performance to be applicable in monitoring autonomous train's ego perspective view.

Keywords: Semantic segmentation · Rail segmentation · Frontal train monitoring · Railway.

1 Introduction

For decades, the industry has sought to replace or augment drivers in various modes of transportation with autonomous, programmed computers that would be more efficient, cost-effective, and safer than human operators.

The concept of developing self-driving trains is not new; indeed, the "Victoria" metro line (inaugurated in 1968 in London-UK) is considered to be the first large-scale automatic rail system [1] [4]. This train is currently classified as GoA2 [3]: the driver is still present in the cab but only deals with closing doors and starting orders, and the train moves automatically to the next stop. As of July 2016, 789 km of automated metro in operation consisting of 53 lines and 822 stations in 36 cities around the world according to the International Union of Public Transport (UITP) [4]. Half of the automated metro

* IRT Railenium, SNCF, Alstom Crespin, Thales, Bosch, and SpirOps.
For now only 10% of our dataset RailSet are provided in this link, will publish the rest shortly.

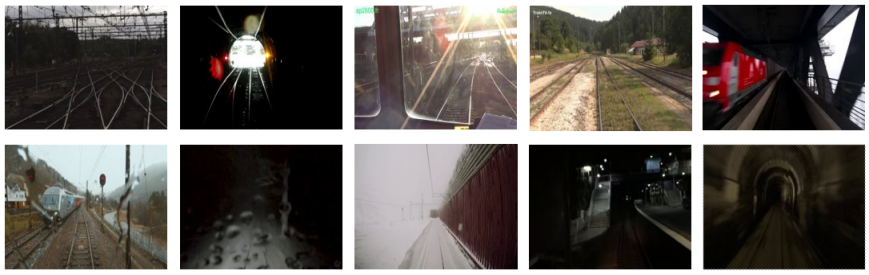


Fig. 1: Examples from RailSet: front views of train taken from different setups, climatic conditions and different acquisition periods.

infrastructure is concentrated in four countries: France, South Korea, Singapore and the United Arab Emirates [UITP, 2016] [4], but these sophisticated systems are limited and located in urban areas relatively isolated from certain potential hazards. The most sophisticated driverless trains, particularly in urban guided transport, are automatic but not autonomous.

The absence of a driver in the cabin is therefore not enough to classify a train as autonomous. It is its on-board decision-making capacity that makes a train autonomous. Moreover, autonomous trains are different from automatic trains regarding the fact that autonomous trains evolve in an open and dynamic environment. Some elements of the environment have an unpredictable behavior that can lead to unexpected and potentially hazardous situations. These autonomous rolling stocks will encounter a very heterogeneous environment and will operate on tracks equipped with various signaling systems. Therefore, one of the most important aspects for ensuring a vehicle's autonomy is to remain self-aware of its surroundings, first by perceiving (identifying and classifying) all external risks, possible anomalies or hazards; then by acting on the information through the control system to position themselves correctly in their environment in order to better adapt to external conditions [5], this implies the use of more advanced applications and technologies.

Inspection and detection of the structural condition of the railway infrastructure [6] [7] [8] [9] is crucial to ensure safety and to enable more advanced applications such as driver automation. At the very heart of this challenge comes the rail and the rail track, which constitute one of the most important components of the railway environment, as they are in direct contact with the train's wheels determining its trajectory. Any alteration in the structure of these components [9] through wheels interaction [8], imperfections in either of them due to severe loading conditions heavier axles and vehicles, due to traffic growth, gives rise to severe dynamic effects such as vibrations and track deformations that can lead to catastrophic vehicle derailment, the consequences of which can result in fatalities, injuries and economic losses. Alternatively, external objects such as obstructions and passengers at the vicinity of the rails have the same bad impact on the normal flow of the train. This makes detecting Rails a central key in monitoring frontal environment of the train and in decision making in autonomous trains, similar to detecting road lanes for autonomous vehicles in order to localize its flow in reference to other users.

In this paper, we discuss various semantic segmentation experiments based on UNET [10] and FRNN [11] methods to assess their performance on rail segmentation from various train frontal scenes taken from two railway datasets including the largest datasets, Railsem19 scenes and our RailSet dataset with 8500 and 6600 scenes respectively, see Fig.1. When creating our RailSet dataset, we strived to bring more complex scenes in terms of rail visibility, weather conditions, and long distance rail detection, not just noisy images due to vibration and flow movement. This brings more scenes to Railsem19 [12], more semantic use cases, and helps enrich public rail datasets and test scenarios like the semantic experiences detailed in this paper, which summarize our contribution in this work.

2 Related works

2.1 Dataset

Railsem19 [12] is the first public contribution to the segmentation task of multiple objects in the railway environment. It contains 8500 annotated short sequences, including various scenes taken from the perspective of the train and streetcar driver. Yin et al. [13] proposed a railroad segmentation dataset called RSDS, which consists of 3000 images taken in the environment of low-speed trains in China. Yin et al. [13] employed manual annotation with the VIA tool [14] to label only the active railroad track on which the train is moving. The data is not made available to the public.

Then comes our RailSet dataset as a second public dataset consisting of 6600 manual annotations using the open source software CVAT [15]. We process all visible rails in the scenes, where only rail and track classes are labeled.

However, with road-oriented datasets such as Cityscapes [16], Mapillary vistas [17], COCO-stuff [18], and KITTI [19], some railway elements can be found in the images. The rail scenes in these datasets represent a combination of interior views, road views, and pedestrian views, but perspective scenes of the driver are almost non-existent.

2.2 Segmentation

Our proposed dataset, RailSet, is intended to solve the task of segmenting the rails from the train's ego view. This task is important for monitoring the railway environment from an onboard video surveillance system at the front of the train. The rail segmentation task has been the subject of some existing research. Among these works, Kaleli and Akgul [20] propose a driver assistance solution, in which they use dynamic programming to extract the active track ahead of the train. Ukai and Tomoyuki [21] also focus on front track extraction, an important step to detect fixed or moving obstacles in front of the train. To extract the railroad tracks, they divide the scene into a near and far area and apply a set of image processing techniques and an iterative algorithm for each to extract the rail parts.

With the emergence of deep learning and convolution neural network reasoning, a number of works have begun to exploit the semantically segmented neural network for rail detection in frontal train scenes. Zendel et al. [12] propose a full-resolution residual network (FRNN) [11] as a benchmark method for fully segmenting Railway scenes

from an ego-perspective view. This model is first trained on the Cityscape dataset and then refined on the Railsem19 dataset. This approach achieves a good level of segmentation on the different classes in the rail and streetcar configurations, and in particular on the rail-specific labels, with 71.5% mIoU on elevated railroads, 45% mIoU on embedded railroads, 81.9% mIoU on railroads, and 40.1% mIoU on streetcar tracks. Yin et al. proposed RailNet, a custom convolutional model designed to segment the active track region of low-speed trains. Yin et al. [13] proposed RailNet, a custom end-to-end model that combines feature extraction and segmentation, where they integrate multi-convolution feature extraction based on pyramid structure to make the features have a top-bottom propagation. It achieves better performance that fully convolutional FCN and detection based methods Mask RCNN. Gibert and Patel [22] demonstrate a custom full convolution network to segment railroad track scenes for visual inspection of rail components. This approach is tested on the Northeast Corridor dataset by ENSCO Rail's Comprehensive Track Inspection Vehicle (CTIV) and shows good accuracy and low false positive segmentation rates.

3 Rail segmentation

3.1 RailSet

We selected 23 videos from the train driver's perspective to create the RailSet dataset. Of these videos, 22 are accessible via the YouTube platform[5], one of which is owned by the SNCF[6]. These videos cover more than 34 hours of train traffic from different countries and in various conditions: weather conditions, camera models, editing positions and lighting conditions. To avoid redundancy and overlapping scenes, we choose an average of 280 frames per video, with each frame relatively distinct from the previous one. We also adjust an SSIM metric [23] to measure the difference in structure between the selected frames and filter out repetitive scenes. We include empty scenes where the rails are invisible, such as in tunnels with no lighting, and snowy scenes where the rails are fully or partially covered, in order to evaluate the consistency of existing models over more variable situations (see Table 1).

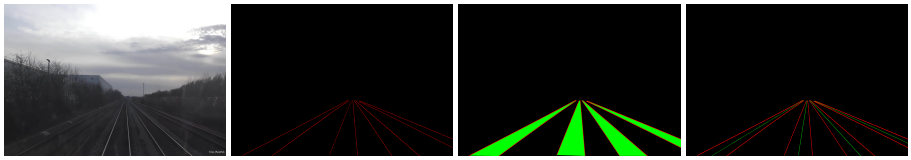
Unlike Railsem19, we provide links to the collected videos and the index of each processed scene. This allows us to provide more similar scenes for testing purposes, and also serves as pseudo-labels where we can take a sequence of unlabeled images around the labeled scenes for semi-supervised learning and for temporal processing like tracking and matching.

We annotate the data based on the basic concept of rail structure, where the rails are the components in direct contact with the train wheels, while the areas between each pair of rails define the rail tracks. The rail class is annotated using the CVAT computer vision annotation tool [15]. Each rail is annotated with a polyline extended to the widest range of visibility of the rail, then smoothed with spline interpolation to remove wobbles and sharp edges. The rail track class is automatically inferred from the

[5] These videos are licensed permissive or free to use except for commercial purposes or with explicit consent to allow experimentation and dissemination

[6] Société Nationale des Chemins de fer Français

rail labels by applying rail pairing. Rail pairing consists of grouping two rails belonging to the same track which we performed manually using the CVAT tool [15]. We represent the rail track class in two forms, either by a midline or by an enclosing polygon. To form the polygonal representation, we simply connect each pair of lines and then fill in the surrounding space. For the midline shape, we fit a line between each pair of rails by taking the midpoint coordinates of the points that lie on the same horizontal level (Fig.2). The purpose of using different shapes of the rails class is to detect which of them will allow the models to achieve better performance in detecting rails and neglecting other objects semantically similar to rails.



(a) (b) (c) (d)

Fig. 2: RailSet different annotations masks. **(a)** raw-image, **(b)** rail class, **(c)** rail and rail-track (enclosing polygon representation) classes and **(d)** rail and rail-track (mid-line representation) classes.

Railsem19 [12] labeling policy contains more rail-specific labels than in our work. For the SBDS data, only the active rail track is annotated, which is different from our dataset and Railsem19, and since it is private data, we cannot consider it in our comparison. To ensure comparability between the RailSet and Railsem19 datasets in our experiment we take four of these labeled classes and we transform them following the policy described previously taking the rail-raised (Fig.6 (a)) and the rail-embedded (Fig.6 (b)) labels as rail class , while the rail-track (Fig.6 (a)) and the tram-track (Fig.6 (b)) labels as rail-track class.

Table 1: RailSet dataset properties

Number of Data	Number_of_videos	Acquisition time	Number of rails	less visibility
6600	23	Day light=4376, Sunny=56, Rain=395, Snow=1033, Dusk time=299, Night=441	Pairs=15223, Single-tracks=139, Total=30585, >4 rails per frame=25%, <4 rails per frame=75%	Before bridge or tunnel=393 Inside tunnels=604 Snow=517 Night=320

Annotation shapes	Presence in frames	Empty_images	Percentage_covered_areas	Mean_frames_per_video
rail=polyline rail-track=polygon rail-track=poly-line(mid-line)	Rails=98.74% Rail-track(polygon, mid-line)=98.28%	83	Rails=0.89% Rail-tracks=8.26%	280 $\simeq$ 0.41%

3.2 Experiments

We have two semantic segmentation networks deployed to solve the rail track segmentation task, UNET [10] and FRNN [11] which represent the baselines for our experiments.

The training, validation, and testing distributions are based on different combinations of data, including the RailSet and Railsem19 [12] datasets. Both architectures, UNET [10] and FRNN [11] are configured for an input shape of 512x512x3 and a batch size of 20 images, with the Adam optimizer (learning rate=0.01, decay rate=0.9) [24]. As a model evaluation metric, we use the Jaccard index, called the IoU metric. This metric provides the amount of overlap between the predicted labels and the ground truth labels. The models are implemented in the NVIDIA V100 GPU.

For each segmentation model, we validate its performance and test it on both RailSet and Railsem19. The test set is composed of 1419 images from both datasets in a proportion of 10% each. For the training set, depending on the experience, it varies at proportion of 80% of the involved data. All these sets are selected equitably by taking into account various level of complexity presented on the number of rails, climatic conditions and visibility.

We apply to the RailSet and Railsem19 datasets an augmentation using two transformations which are mirroring with horizontal shift and zooming by warping the perspective of the region that contains all the rails to the full size image, see fig.3. We also add 1290 crops of objects semantically similar to the rails, such as poles and wires, to the empty images where the rails are not visible due to lighting, such as in tunnels. The goal is to test whether this improves the discrimination ability of the models, thereby reducing false positives.

Our data present a strong intrinsic imbalance, and to avoid having biased segmentation models, we generate for each batch of images and masks the corresponding weights to penalize misclassifications.

(a) (b)

Fig. 3: Data augmentation. **(a)** Mirror via horizontal-shift augmentation, **(b)** Zooming to rails via image stretching transformation.

We compare eight different experimental scenarios in which we refer to the results of Zendel et al. [12] on the Railsem19 dataset. Since the relevant datasets are based on two classes, while in Railsem19 they have dealt with several rail-specific classes, we derive the average results of Zendel et al. [12] to obtain an overall segmentation performance on the rail and track classes. Thus, we introduce the following equation: 1, in which we use the presence rate in the images "in-frame" and the individual "mIoU" performance provided in [12].

$$mIoU(Rails) = \frac{In\text{-}Frames_{(\text{Rail-raised})} \times mIoU_{(\text{Rail-raised})} + In\text{-}Frames_{(\text{Rail-embedded})} \times mIoU_{(Rail\text{-}embedded)}}{(In\text{-}frames_{(Rail\text{-}raised)} + In\text{-}Frames_{(Rail\text{-}embedded)})} \quad (1)$$

In all different scenarios, we opted for the dice-coefficient loss function that allows us to obtain the best performance. The process of how we select the suitable loss function is based on an experiment with UNET [10] and FRNN [11] on the RailSet dataset, we compare different loss functions among many such as AsymLoss, Tversky Loss, Sensitivity-Specifity Loss, SoftDice loss and TopKLoss, see Table 2. Their implementations are described in [25].

Scenarios on Rail class In the scenario (1) and (2) we used RailSet and Railsem19 respectively with only the rail class, to study the impact of data augmentation on UNET and FRNN segmentation performance. Therefore, we tested 3 data configurations; without data augmentation, with data augmentation using only zooming and mirroring, then we add the empty crops and images. We see an overall performance improvement on both datasets, see Fig.4. Since we proved the importance of data augmentation applied on RailSet and Railsem19, that allows the models to learn more features and different situations, we opt for this technique in the remaining scenarios.

Under full data augmentation, UNET and FRNN models perform better overall more complex scenes in the test set when trained on RailSet than on Railsem19 (see Table 3), given that only 40% of the test set comes from our RailSet dataset. In the scenario (1), we also tested the models on only tramway scenes, that yielded to comparable results to Zendel et al. [12]. That shows the consistency of our dataset in more specific environment setups. Otherwise, in the scenario (2), we observe a drop in performance when testing the models on RailSet which is a dataset specific to only railways (see Fig.5). This drop of performance can be explained by the fact that training models on a more general dataset that contains mix of railways and tramways data would produce a domain shift[7]. We therefore set an assumption that neglecting these tramway scenes and training only with an in-domain data would improve performance. This assumption is treated in the next scenario (3).

In scenario (3), we addressed the previously mentioned assumption. We trained UNET and FRNN models on the Railsem19 dataset with only railway scenes, and tested it against the RailSet dataset. We note that there is no significant improvement in performance (see Fig.5). This is can be due to the high variability of RailSet scenes, which potentially contain a more complex environment configuration. This explains the importance of introducing this new dataset to handle more specific environment complexity in the railway domain.

Scenarios on Rail and Rail track classes Previously, only the rail class is treated. In the scenario (4) and (5), we include the rail-track class when training the models on RailSet and Railsem19 respectively. This additional class lead to a better segmentation performance and provides a visual representation of the rail track region. In the scenario (4), we observe an improvement in rail segmentation, especially the results on Railsem19 with almost 1,5% gain compared to scenario (1) where we trained with only

[7] A change in the data distribution between an algorithm's training dataset, and a dataset it encounters when deployed.

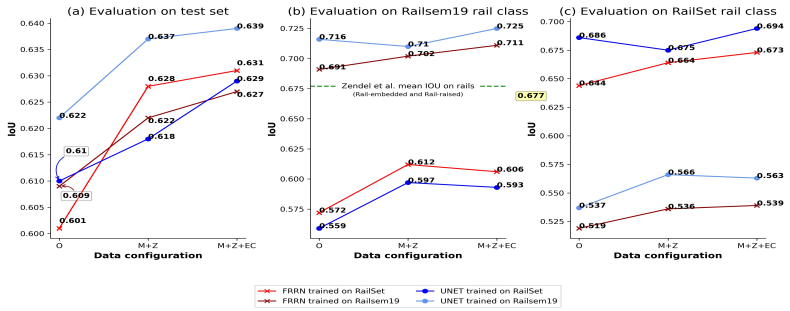


Fig. 4: Evaluation of FRNN and UNET on test set (a), rail class of Railsem19 dataset (b) and rail class of RailSet dataset (c) taken from the scenarios (1) and (2). The effect of the different data-augmentations O: without augmentation, M+Z: Mirror+Zoom, M+Z+EC:Mirror+Zoom+Empty-Crops(1290 scenes without rails+83 empty images).

the rail class, see Fig.5. In the scenario (5), we observe an enhancement when testing on RailSet with almost 1%. Moreover, rail-track class yields more direct local features on the rails and leads the models to focus on the low region of the images, which reduces the false positive rate, i.e., the detection of wires as rails. This improves rail detection in reduced visibility conditions in tunnels, at night and also in stormy conditions, as shown in the Table 3 of the test set.

In the scenario (6), we study the impact of changing the rail-track representation in the RailSet dataset. The rail-tracks in their standard representation as enclosing-polygons produce better performance results than the mid-line form, see Fig.5. This may be due to the area similarity covered by the mid-lines with the surrounding areas, resulting in more confusion and false positive segmentation. Such discontinuities or shifts in the mid-lines position can cause the same on the rails segments.

Scenarios on mix dataset RailSet and Railsem19 In the scenarios (7) and (8) we highlight the importance of mixing both datasets RailSet and Railsem19 when training models, in which we obtained the best results on the test set as shown in Fig.5. We also notice that training with the two classes rail and rail-track in scenario (8) achieves rails segmentation precision of 67% mIoU on test set, 70% mIoU on Railsem19 and 69% mIoU on RailSet almost similar performance results compared to training with only the rail class in scenario (7). In the Table 3, we can see in bold the best performances on various scene categories of the test set are obtained by training on both datasets, as well as adding the rail track class which improves the segmentation accuracy on less visible rails. This experiment confirms the need of obtaining more training data to get more scene situations and features. See Fig.7 for more visual results.

In perspective, along with the semantic segmentation, these data will serve for anomaly detection by taking the location of the rails and processing it by auto-encoders. Alternatively as in [26], we can take this data and explore the semantic labels, whether they are ground truth labels or pseudo-labels composed of a sequence of ground truth

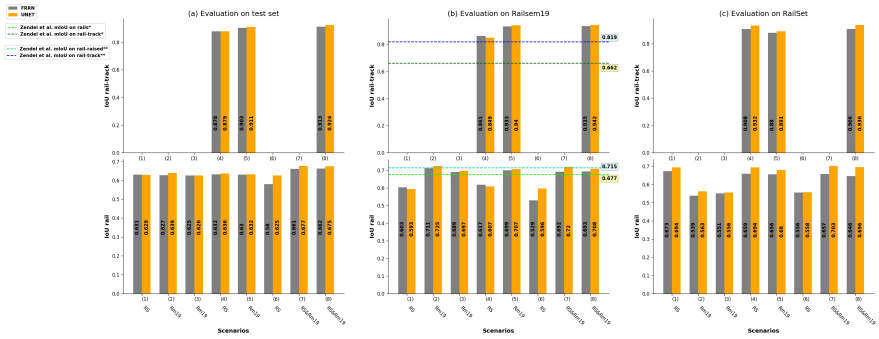


Fig. 5: Evaluation results of UNET and FRNN on test set (a), Railsem19 (b) and RailSet (c) across all the eight cases where all the data-augmentations were applied. The three lower plots deal with performance on rail class and the three upper plots with performance on rail-track class. **RS**=RailSet dataset, **Rm19**=Railsem19 dataset, **RS&Rm19**=RailSet dataset and Railsem19 dataset. In legends, * refers to results deduced from the equation (1), while ** refers to individual results taken from zendal et al. [12] paper.

labels and the underlying unlabeled scenes, as secondary guiding features for the 3D reconstruction of frontal train scenes to detect geometric deformation of the rails and holes in the rails and tracks.

4 Conclusion

We present a new dataset for Rail and Rail-track segmentation in railway scenes: RailSet. Manual annotations are made to generate the Rail class masks, and from these initial annotations, the Rail-track labeling is generated automatically. We conducted multiple experiments using the UNET and FRNN architectures on different data configurations. These experiments yielded results comparable to those recorded on the Railsem19 dataset. That in turn proved the valuability of the RailSet dataset, along a number of pre-treatments such as the data augmentation, the inclusion of the Rail-track class in training and merging the two datasets. Adding temporal indexes of RailSet scenes would yield more data for testing, and would serve as input for other tasks such as semantic pattern matching and tracking. For now, only a part of the RailSet dataset is provided for public and will publish the remaining sooner.

Acknowledgements This research work contributes to the french collaborative project TASV (autonomous passengers service train), with SNCF, Alstom Crespin, Thales, Bosch, and SpirOps. It was carried out in the framework of IRT Railenium, Valenciennes, France, and therefore was granted public funds within the scope of the French Program “Investissements d’Avenir”.

References

1. J. Powell, A. Fraszczyk, C. Cheong, and H. Yeung. Potential benefits and obstacles of implementing driverless train operation on the tyne and wear metro: A simulation exercise. Urban

Rail Transit, pages 1–12, 12 (2016).
2. Y. Wang, M. Zhang, J. Ma, and X. S. Zhou. Survey on driverless train operation for urban rail transit systems. Urban Rail Transit, pages 1–8, 2016.
3. T. Stene. Automation of the rail-removing the human factor?, pages 1947–1955. 2018.
4. Y. Wang, M. Zhang, J. Ma, and X. S. Zhou. Survey on driverless train operation for urban rail transit systems. Urban Rail Transit, pages 1–8, 2016.
5. M. Carre. Autonomic framework for safety management in the au- tonomous vehicle. pages 18,19,39,40,67,69, 2019.
6. J. Sadeghi and H. Askarinejad. Influences of track structure, geometry and traffic parameters on railway deterioration. Transactions B: Appli- cations, pages 292–300, 2007.
7. G. Kumaran, D. Menon, and K. Nair. Dynamic studies of railtrack sleepers in a track structure system. Journal of Sound and Vibration, 268:488–490, 11 2003.
8. J. Bian, Y. Gu, and M. Murray. A dynamic wheel–rail impact analysis of railway track under wheel flat by finite element analysis. Vehicle System Dynamics, pages 1–16, 06 2013.
9. D. CANNON, K.-O. Edel, S. Grassie, and K. SAWLEY. Rail defects: An overview. Fatigue Fracture of Engineering Materials Structures, pages 865 – 886, 2003.
10. T. B. Olaf Ronneberger, Philipp Fischer. U-net: Convolutional networks for biomedical image segmentation. MICCAI, arxiv:1505.04597, pages 2,3,4, 2015.
11. T. Pohlen, A. Hermans, M. Mathias, and B. Leibe. Full-resolution residual networks for semantic segmentation in street scenes. CVPR, pages 1,4,5, 2016.
12. O.Zendel, M.Murschitz, M.Zeilinger, D.Steininger, S.Abbasi, and C.Beleznai. Railsem19: A dataset for semantic rail scene understanding. CVPRW, pages 1,5,6, 2019.
13. Y. Wang, L. Wang, Y. H. Hu, and J. Qiu. Railnet: A segmentation network for railroad detection. IEEE Access, 2019.
14. A. Dutta and A. Zisserman. The VGG image annotator (VIA). CoRR arXiv:1904.10699, 2019.
15. Sekachev and Boris. Computer vision annotation tool: A universal approach to data annotation, 2019.
16. M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benen- son, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. CoRR, pages 1,2,3, 2016.
17. G. Neuhold, T. Ollmann, S. Rota Bulo, and P. Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In ICCV, pages 1,2,4, 2017.
18. T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick. Microsoft COCO: common objects in context. CoRR, pages 5,6,7, 2014.
19. K. I. of Technology and T. T. I. at Chicago. Kitti.
20. F.Kaleli and Y.Akgul. Vision-based railroad track extraction using dynamic programming. ITSC, pages 4,5,6, 2009.
21. Ukai and Tomoyuki. Obstacle detection on railway track by fusing radar and image sensor. WCRR, page 12, 2011.
22. Gibert, V. M.Patel, and R. Chellappa. Semantic segmentation of railway track images with deep convolutional neural networks. (ICIP), pages 1,5, 2015.
23. D. Brunet, E. R. Vrscay, and Z. Wang. On the mathematical properties of the structural similarity index. IEEE Transactions on Image Processing, pages 1488–1499, 2012.
24. F. Chollet et al. Keras. `https://github.com/tensorflow/tensorflow/blob/master/tensorflow/python/training/adam.py`, 2015.
25. JunMa11. Segloss. `https://github.com/JunMa11/SegLoss/`, 2020.
26. V. Guizilini, R. Hou, J. Li, R. Ambrus, and A. Gaidon. Semantically- guided representation learning for self-supervised monocular depth. In International Conference on Learning Representations, 2020.

Table 2: Evaluation of UNET and FRNN models trained on RailSet under various state-of-the-art segmentation losses on the overall datasets. **EC**: Empty Crops with objects semantically similar to rails, **M+Z**: Mirror+Zooming. The evaluation metric is the mIoU. The shortcut SS-loss refers to Sensitivity-Specifity loss.

Model	Training-parameters					Test-data	Valid-data	Railsem19			RailSet
	Data	Loss	Class	Augmentation				Train-Scenes	Tram-Scenes	Shared Scene	
				M+Z	EC						
UNET	RailSet	Dice-coefficient	Rails	✓	✓	**0.642**	**0.637**	**0.627**	0.382	**0.621**	0.716
		Assymloss		✓	✓	0.585	0.574	0.587	0.335	0.587	0.597
		Tversky		✓	✓	0.639	0.636	0.607	0.382	0.596	**0.721**
		SS-loss		✓	✓	0.638	0.635	0.621	0.398	0.616	0.708
		DC_and_topk		✓	✓	0.629	0.625	0.615	0.373	0.61	0.695
FRNN	RailSet	Dice-coefficient	Rails	✓	✓	0.63	0.631	0.625	**0.418**	0.62	0.673
		Assymloss		✓	✓	0.603	0.605	0.564	0.399	0.558	0.692
		Tversky		✓	✓	0.548	0.555	0.512	0.35	0.505	0.631
		SS-loss		✓	✓	0.609	0.612	0.581	0.386	0.572	0.682
		DC_and_topk		✓	✓	0.622	0.623	0.587	0.395	0.583	0.699

Table 3: Detailed results of UNET and FRNN performance on various scenes in the test set, taking into account weather conditions and the number of rails per scene that alter rail visibility and make segmentation more complex. Training is performed on different data configurations and classes:"Rail" and "Rail Track" (polygon, centerline). <4 rails refers to scenes with less than four rails and >4 rails for scenes with more than four rails that represents 20% of test set. Values in bold indicate the best results on rail and track segmentation. For the rail midline configuration, we only shows the results on the rail class.

Model	Data	Classes	Test-data					
			Night	Snow	Tunnels	Rain	<4 rails	>4 rails
UNET	RailSet	Rail	0.58	0.647	0.544	0.66	0.627	0.633
		Rail | Rail track (midline)	0.546	0.6	0.514	0.621	0.593	0.594
		Rail | Rail track (polygon)	0.61 | 0.81	0.65 | 0.866	0.556 | 0.802	0.677 | 0.909	0.631 | 0.871	0.633 | 0.844
	Railsem19	Rail	0.562	0.559	0.525	0.616	0.617	0.657
		Rail | Rail track (polygon)	0.554 | 0.831	0.553 | 0.853	0.524 | 0.83	0.613 | 0.913	0.615 | 0.905	0.651 | 0.894
	RailSet + Railsem19	Rail	0.615	0.659	0.566	0.681	**0.677**	**0.675**
		Rail | Rail track (polygon)	**0.635** | **0.844**	**0.666** | **0.892**	**0.592** | **0.86**	**0.701** | **0.939**	0.675 | **0.923**	**0.675** | **0.903**
FRNN	RailSet	Rail	0.561	0.598	0.521	0.681	0.628	0.64
		Rail | Rail track (midline)	0.497	0.573	0.49	0.614	0.58	0.57
		Rail | Rail track (polygon)	0.552 | 0.763	0.592 | 0.823	0.535 | 0.801	0.682 | 0.897	0.634 | 0.88	0.639 | 0.869
	Railsem19	Rail	0.457	0.5	0.456	0.625	0.597	0.655
		Rail | Rail track (polygon)	0.505 | 0.763	0.522 | 0.814	0.494 | 0.787	0.615 | 0.909	0.613 | 0.895	0.645 | 0.894
	RailSet + Railsem19	Rail	0.561	0.619	0.541	0.662	0.655	0.674
		Rail | Rail track (polygon)	0.564 | 0.77	0.61 | 0.839	0.542 | 0.819	0.667 | 0.924	0.659 | 0.91	0.671 | 0.90

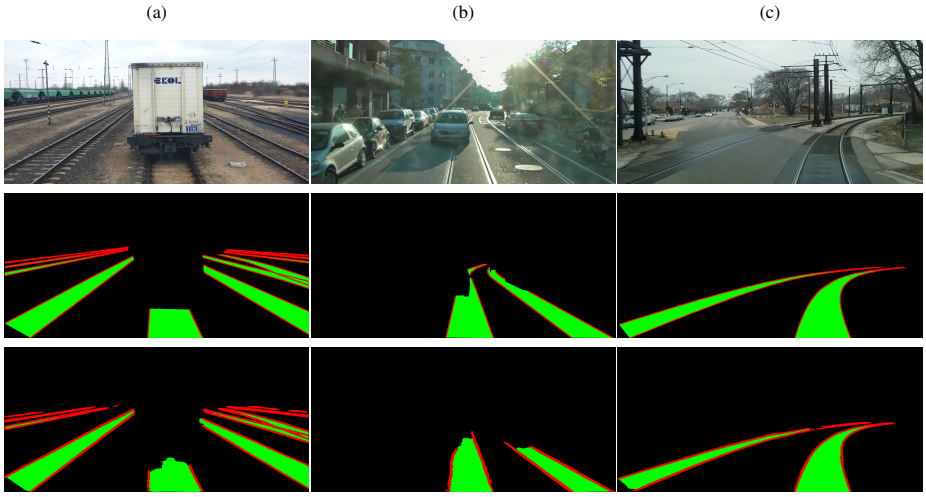


Fig. 6: FRNN test example on Railsem19's scenes. The first row displays the input images: (a) train scene, (b) tramway scene and (c) shared scene, the second row displays the ground truth of the input images and the last row displays the FRNN predictions.

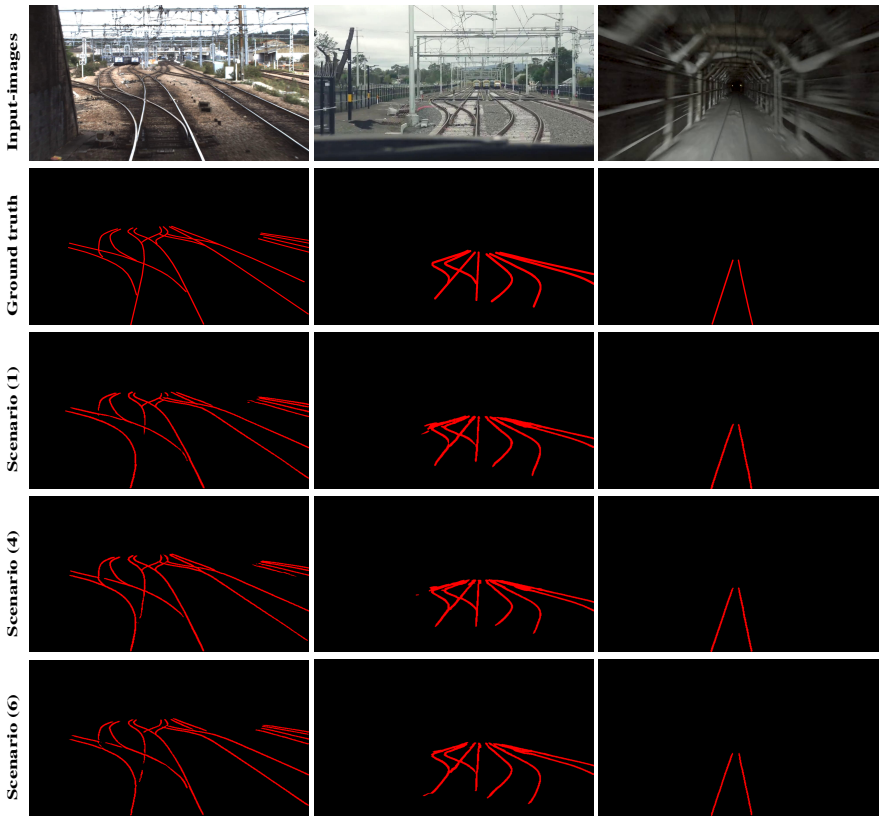


Fig. 7: Further examples of the results of UNET trained on different configurations of RailSet detailed in *scenario (1)*, *scenario (4)* and *scenario (6)*. Only the rail segmentation masks are displayed.