This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945633, IEEE Access

RailNet: a Segmentation Network for Railroad Detection

Yin Wang[1], Lide Wang[1], Yu Hen Hu[2], Ji Qiu[1]
[1]School of Electrical Engineering, Beijing Jiaotong University, Beijing, 100044, China.
[2]Dept. Electrical and Computer Engineering, University of Wisconsin-Madison, Madison, 53705, USA

Corresponding author: Lide Wang (ldwang@bjtu.edu.cn).

This work was supported by the Fundamental Research Funds for the Central Universities, China. [NO. E17JB00150]

ABSTRACT Future trains will use more computer vision aids to help achieve fully autonomous driving. One of the most important parts of the train's visual function is the detection of railroad obstacles. This makes it important to identify and segment the railroad region within each video frame as it allows the train to identify the driving area so that it can do effective obstacle detection. Traditional railroad detection methods rely on hand-crafted features or highly specialized equipment such as lidar, which typically require expensive equipment to be maintained and are less reliable in scene changes. RailNet is a deep learning segmentation algorithm for railroad detection for videos captured by the front-view on-board cameras. RailNet provides an end-to-end solution that combines feature extraction and segmentation. We have modified the backbone network to extract multi-convolution features and use a pyramid structure to make the features have a top-to-bottom propagation. Our model can detect the railroad without generating large numbers of regions, which greatly increases the detection speed. Tested on a railroad segmentation dataset (RSDS) which we have built, RailNet exhibits very good performance while achieving 20 frames per second processing speed.

INDEX TERMS Railroad Detection; Deep Learning; Segmentation;

I. INTRODUCTION

Low-speed fully autonomous trains will become a very important means of transportation in the future. Front-view cameras mounted on locomotives facilitate visual monitoring of the railroad to detect obstacles. At present, the detection of obstacles of railroad mainly relies on trackside equipment and manual inspection. With the development of computer vision, the camera-based obstacles detection system will replace the original expensive method. The goal of the detection is to fully perceive the environment in front of the train. A critical pre-processing step of environment perception is to identify and segment the railroad region within each video frame as it allows the train to identify the driving area so that it can do effective obstacle detection. As such, performing accurate camera-based railroad detection is a key enabler of fully autonomous trains.

Several existing works have been reported to detect and segment the railroad from a given image using the linear features of a railroad image [6][7][35]. The performance of these traditional methods, however, may suffer due to varying lighting conditions and complex backgrounds. Recently, Saux et al. [1] and Gilbert et al [2] proposed to use the convolutional neural networks (CNN) for railroad track or track elements detection. These demonstrate the effectiveness of the CNN structure in extracting railroad or basic track elements. In view of recent progress using deep neural networks for object detection and segmentation, the CNN approach has shown great promise.

In this paper, we propose RailNet, which is an end-to-end deep learning-based railroad track segmentation algorithm. This algorithm consists of a feature extraction network and a segmentation network. The feature extraction network uses a pyramid structure to propagate features from top to bottom to obtain a hybrid feature vector. The segmentation network is a convolutional network for generating the segmentation map of the railroad. We also developed a Railroad Segmentation Dataset (RSDS), which consists of 3000 images. These images are derived from the actual train operating environment.

Using the RSDS dataset, we compare the performance of RailNet against that of modified version FCN [17] and Mask-RCNN [3]. We observe that RailNet achieves the best performance in terms of both the accuracy metrics (pixel accuracy, mean intersection over union ratio, F-measure value) and efficiency (frame rate).

The rest of the paper is organized as follows: In section II, related works are reviewed. The main idea and details of the RailNet algorithm are developed in Section III. Experimental implementation and results are presented in Section IV. In Section V, conclusion and future works are discussed.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945633, IEEE Access

II. RELATED WORK

To our best knowledge, there have few methods that using learning algorithms to detect railroad and few related works can be directly compared to our work. In fact, RailNet is a specialized segmentation network designed for railroad detection, including railroad feature extraction and railroad segmentation. There are several aspects of previous works related to this paper, as discussed below.

A. Railroad Detection

There are many methods of railroad or rail extraction in recent years. In [5], an algorithm that uses dynamic programming in railroad environments to extract the training course and railroad track in front of the train. They made this method in three steps: First, the method uses the Sobel operator to compute the gradient of the input image. Next, a Hough transform process is applied to the binary image for the detection of rail line. Finally, it use Dynamic Programming to extract the railroad. Qi et al. [6] proposed to use the HOG features and establish integral images, and then extracted railway tracks by region-growing algorithm. Nassu et al. [7] introduced an approach that performs rail extraction by matching edge features to candidate rail patterns modeled as sequences of parabola segments. Purica et al. [8] proposed a railroad detection algorithm use Hough Transform to detect lines and perform a line clustering in the Rho and Theta space. Teng et al. [9] proposed a visual railway detection method based on super-pixels rather than pixels. This method uses the support vector machine (SVM) to classify the features transformed by TF-IDF(term frequency-inverse document frequency) and uses intracellular decision scheme to make decisions on a super-pixel by using predictions of features within the super-pixel. Other proprietary algorithms [10], [11], [12] have been developed to incorporate other sensing modalities such as Laser, Lidar or remote sensing data to detect the railroad tracks.

B. Methods of Using Multi-features

In a deep neural network, features are extracted at every layer except the final fully connected classification layers. For object detection and image segmentation, features from multiple network layers are leveraged to yield better results. For example, in FCN [17], sums of partial scores of each category on multiple scales are incorporated for the purpose of segmentation. In[22], a feature pyramid network is used to improve the feature extraction performance in several applications. Other methods such as Laplacian pyramid [23], U-Net [24], etc. also draw on this idea to use multi-scale features. Mask RCNN [3] uses the pyramid structure features to help achieve the state of the art in instance segmentation and human pose estimation on the COCO dataset. Many papers have shown that using multi-features can significantly improve the performance of the model.

C. Segmentation

Similar to the lane detection [25] problem, railroad detection requires the segmentation of the region that contains the railroad from the rest of the video frame. Shelhamer et al. [17] proposed a fully convolutional network (FCN) to perform image segmentation. SegNet [26] uses Encoder-Decoder architecture to improve the resolution of segmentation. Yu et al. [27] uses dilated convolutions to systematically aggregate multi-scale contextual information without losing resolution. Chen et al. [28] proposed an atrous spatial pyramid pooling (ASPP) method based on dilated convolution to robustly segment objects at multiple scales. Zhao et al. [30] proposed a pyramid scene parsing network (PSPNet) that uses global context information. Xu et al. [29] presented a dynamic video segmentation network (DVSNet) for fast and efficient semantic video segmentation. Another series of solutions [31][32] for semantic segmentation improved the performance of the model.

III. RAILNET

Traditional methods that use the line or edge features to detect railroad maybe have a good performance in one fixed scene but performance decrease fast after scene change. Especially in the course of the train driving, the background is constantly changing, and the hand-crafted features cannot meet the requirements. We convert the railroad detection task from railroad line detection or edge detection problems to segmentation problems. Our railroad detection Network, called RailNet, is an end-to-end training network. RailNet combines the ResNet50 backbone network with a fully convolutional network to compute segmentation. We built an independent multi-level feature extraction network outside the backbone network to make full use of the features in the network without changing the network mainframe structure. Figure.1 shows the structure of the RailNet. The further explanation is as follows.

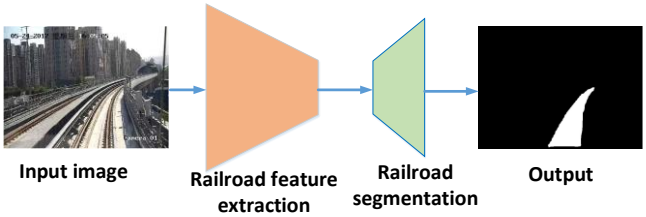

FIGURE 1. **RailNet structure**

A. RAILROAD FEATURE EXTRACTION

Like image classification and recognition tasks et al, railroad detection also needs to extract the high and low-level features of the image. The purpose of a feature extraction network is to leverage the powerful feature extraction capability of a convolution neural network trained on the dataset. VGG16 [20] and ResNet50 [21] have been quite popular choices of backbone feature extraction networks for various computer vision tasks. In this work, we choose ResNet50 because it has demonstrated stronger

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945633, IEEE Access

feature extraction capabilities. It also has fewer parameters and hence is computationally efficient.

Our model takes an arbitrary size image as input and outputs feature maps through the backbone network. These feature maps come from different network layers and have different sizes and dimensions. The multi-level features bring the benefits of local and global perception due to multiple receptive filed sizes. The ResNet50 layers are grouped into 5 stages, namely, stage1, stage2, stage3, stage4, and stage5. Each stage contains several convolutional layers and batch normalization layers. These stages bring richer features for further processing. It is reported [17] that features from lower layers (closer to the input) are related to low-level visual features; while from higher layers, features extracted are often linked to global, abstract concepts. In order to make the best use of these richer features, we design an independent structure outside the backbone network to process the fusion of multiple features. Figure.2 shows this structure on topology. And we borrowed the ideas from the feature pyramid network [22] to use the feature propagation from top to bottom. For the feature extraction network, our modifications on the ResNet50 can be described as following:

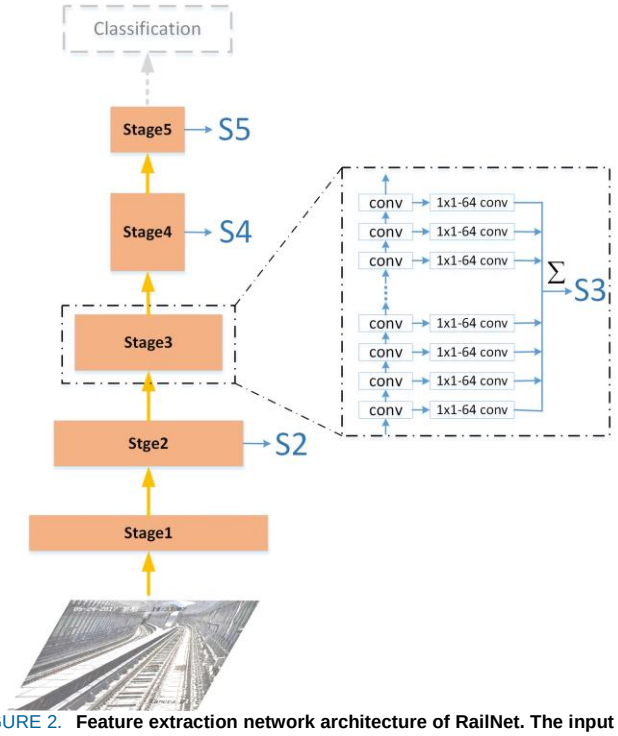

FIGURE 2. **Feature extraction network architecture of RailNet. The input of the network is one image or a batch of images with arbitrary size, and the output is the hybrid features {S2, S3, S4, S5}. In stage2, stage4 and stage5 have a similar structure to stage3.**

- We cut off all layers beyond stage5, including the fully connected layer, average pooling layer, classification layer, which are useless for our task.
- Inside the stage2 to stage5, we connect the output of each convolution layer using a convolution layer with a kernel size of 1x1 and a depth of 64. Since the size of the features does not change inside each stage. We can concatenate these several outputs as the output features of each stage and obtain the feature sets {S2, S3, S4, S5}

In this way, we make full use of the features from most convolution layers. It covers from high-level abstract features to low-level specific features. But even so, we still isolate the features of different level, {S2, S3, S4, S5} represent the different dimensions features obtained in the process of forward propagation. We can't merge them into a joint feature directly. Then we add a Top-down pathway to make the feature have a backward propagation. As shown in Figure. 3. This pyramid structure feature[22] has been proved to improve the effect of feature extraction significantly. Details are as following:

- Four 1x1-256 convolution layers, denoted by { C2, C3, C4, C5}, are attached to {S2, S3, S4, S5} respectively.
- From top to bottom, the features {C2, C3, C4, C5} are connected to an up-sampling layer and merged with the features of the previous layer respectively.

Each stage of the network has a scaling step of 2, so we up-sampling the feature maps by a factor of 2. After adding these two aspects of the independent structure to the backbone network, we obtained the final fused feature maps {P2, P3, P4, P5}. Since C5 is at the top layer, it cannot be backpropagated and merged with the features of the upper layer, so P5 is C5.

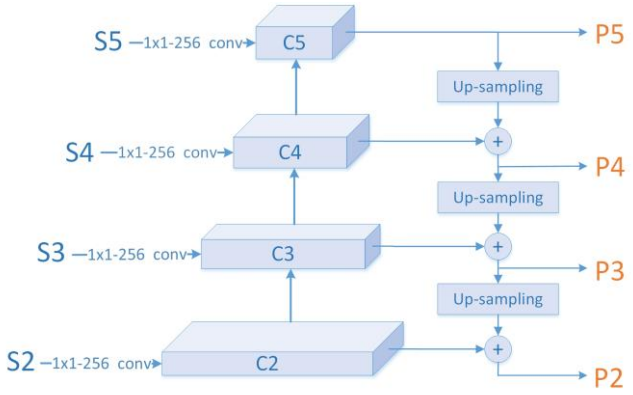

FIGURE 3. **The top-down pyramid structure of feature processing.**

Our goal is to use this hybrid features for railroad segmentation. We still need a fully convolutional network to make pixel-level predictions for feature maps. In the next section, we will introduce a segmentation network that connects the feature extraction networks.

B. RAILROAD SEGMENTATION

In order to separate the railroad pixels from the other parts, we built an underlying network of the RailNet for railroad segmentation. This network is trained to output a binary segmentation map, that indicate which pixels belong to the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945633, IEEE Access

railroad area and which do not. Most popular detect-and-segment methods (e.g. Mask R-CNN, SegNet) are not suitable for railroad segmentation. Since bounding box detection is suitable for compact objects, the railroad does not. And these detect-and-segment algorithms perform global image detection, generating thousands of proposals for classification and segmentation. This makes a lot of computing time spent on the unused proposals. For RailNet, we use a fully convolutional network to compute segmentation that is not for the regions but for the entire image. This can significantly reduce the computing time spent on regions.

We need to convert the features to a fixed size before sending the features to the segmentation network. Unlike detect-and-segment networks, our model does not generate regions. We can't use RoIPool [19] or RoIAlign [3] to complete this step directly. Based on the characteristics of the output features, we have proposed a method that takes into account two situations. Define the output feature size as $n \times n - d$, and the converted size is $m \times m - d$. If $n/m \geq 2$ then consider the feature map as one region and use the method similar to RoIPool[19] to reduce feature size. If $n/m < 2$, use bilinear interpolation [33] to calculate new feature values. See Figure. 4 for details.

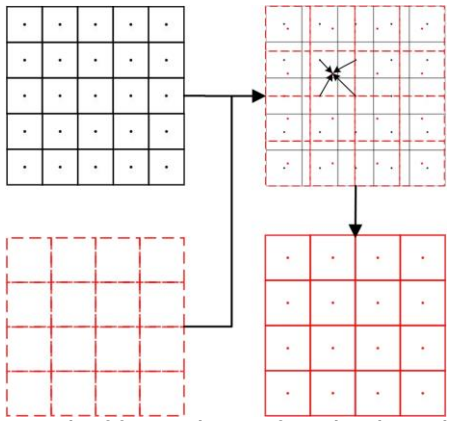

FIGURE 4. **Example of feature size transformation. (Here, the feature size of 5x5 is converted to 4x4) Each small square represents a pixel, and the center point of the square represents the value of the pixel. We can calculate a new value from four adjacent pixels use bilinear interpolation for the new feature maps.**

After several layers of *Conv* and *Deconv*, the features will be calculated as a one-dimensional matrix, called Seg-map, each value represent the probability of belong to the railroad area. We can get the final railroad segmentation map by adjusting the threshold.

C. LOSS FUNCTION

For RailNet, we compute two loss values for the entire network, which are the loss of the entire image L_{image} and the railroad Loss L_{Railroad} . As shown in Figure.5. The coordinates of the railroad region box can be determined by using the annotations of the ground truth. The railroad region box determined by ground truth can derive the actual position of the railroad of the RailNet output Seg-map. L_{Railroad} is the loss between ground truth railroad region and the corresponding area of the Seg-map. If the coordinates of the railroad region box in the ground truth are (a,b,l,h) , and the scaling between the ground truth and the Seg-map is k. The quantification coordinates of railroad box in the Seg-map are $([a/k],[b/k],[l/k],[h/k])$. The value represents the coordinates of the upper left corner and the length and width of the box. For each loss, training samples can be expressed as: $\{(X_n, Y_n), n=1,2,...,m\}$, where X_n denotes the original feature tensor of the input image, $Y_n = \{y_j^{(n)}, j = 1, 2, ..., |X_n|\}, y_j^{(n)} \in \{0,1\}$ denotes the label of the input image. $\overline{Y_n} = \{\overline{y_j}^{(n)}, j = 1, 2, ..., |X_n|\}, \overline{y_j}^{(n)} \in [0,1]$ denotes the output of the network. We compare the sample label Y_n with the actual output $\overline{Y_n}$ of the network to calculate the loss. We define the weight of the network as W, and calculate the loss for each pixel:

$$l(X_j, W) = \begin{cases} \beta \log P(X_j, W), & \text{if } y_j = 1 \\ (1-\beta) \log(1 - P(X_j, W)), & \text{if } y_j = 0 \end{cases} \quad (1)$$

In which $\beta = |Y_-| / |Y_+ + Y_-|$, $1 - \beta = |Y_+| / |Y_+ + Y_-|$, Y_+ , Y_- denote railroad pixels and background pixels sets, respectively. $P(X_j, W)$ is computed using the sigmoid function on the activation value at a pixel j . The total loss of each sample can be denoted as:

$$L_{image} = \sum_{j=1}^{N} l(X_j, W) \quad (2)$$

Use the same method to get the railroad loss L_{Railroad} . We define the total loss function as $L = \alpha L_{image} + \beta L_{\text{Railroad}}$ α and β denote the weights of the two losses, respectively.

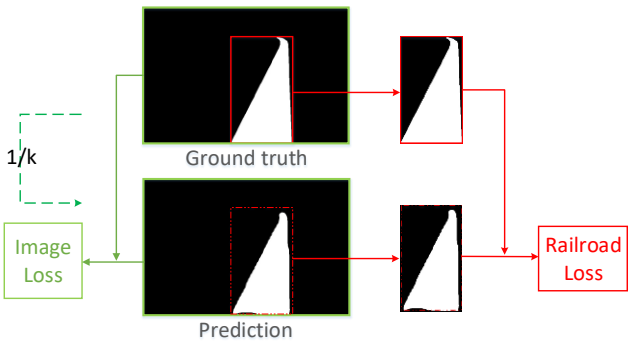

FIGURE 5. **Calculation area of the two losses. The red color calculates the loss of the entire image, and the blue color calculates the loss of the railroad region. The coordinate ratio between the ground truth and prediction is k:1.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945633, IEEE Access

IV. EXPERIMENTS

In this section, we will evaluate the performance of the RailNet for detecting railroad. In section A, we described the railroad dataset and the annotation of this dataset. The implementation details for training this network are described in section B. In section C, we discussed the evaluation method and the comparison of our model with other models in our dataset.

A. DATASET

As far as we know, there have no open-source datasets for railroad detection. In order to train and test our network. We built a railroad datasets for segmentation that include 3000 images, 2500 images for training, 200 images for validation and 300 images for test. We call it Railroad Segmentation Dataset (RSDS). All images come from a real train's driving environment. When labeling the ground truth of the datasets, the railroad is defined as all pixels between two rail lines. Due to the influence of different environments, we did not label the railway sleepers outside the two lines as the railroad. We use VIA [34] tool to manually draw a closed polygon region shape to define the ground truth. An example is shown in Figure. 6. All image size is 1920x1080, and we made the annotations as a JSON file. During training, we also applied the data augmentation algorithm to enhance training samples.

FIGURE 6. **An example of ground truth**

B. IMPLEMENTATION DETAILS

The network is implemented using the Keras Library with the Tensorflow as the backend. An NVIDIA GTX1080 GPU is used in a desktop Linux PC to perform all training. We initialize the weights of the entire network by two steps: the first step is to initialize the weights of RailNet backbone network (which is ResNet50) using the weights trained by the ImageNet Dataset. The weights can be download on Github. The second step is to randomly initialize other parameters. We use the RSDS training set to train the entire network and fine-tune the parameters in ResNet50. During the training phase, we use the ADAM stochastic gradient descent learning algorithm to training the network. Hyper-parameters during training are set to: mini-batch size = 8, learning rate = 5e-3, scheduled decay = 0.004, $\beta 1$ = 0.9, $\beta 2$ = 0.999, and ε = 1e-08. After 200 epochs training we adjusted the learning rate to 1e-3. The entire training was terminated after 300 epochs (nearly 150k iterations).

C. RESULTS

To evaluate the performance of the RailNet, we refer to the metrics from common semantic segmentation evaluation such as pixel accuracy and mean Intersection over Union (MIoU). Since the railroad detection is based on pixels, it is more like a classification task, so we can use the evaluation metrics in image classification: precision, recall, and F-Measure. We define three values: p_c, p_m, p_i , which denotes the numbers of correct identified railroad pixels, missing identified pixels and incorrectly identified pixels, respectively. This way we can build the following evaluation formulas. k is the number of test images.

- Pixel accuracy: $PA = \sum_k p_c^{(j)} / \sum_k (p_c^{(j)} + p_m^{(j)})$
- Mean Intersection over Union : $MIoU = (1/k)\sum_k p_c^{(j)} / \sum_k (p_c^{(j)} + p_m^{(j)} + p_i^{(j)})$
- Precision: $P_r = \sum_k p_c^{(j)} / \sum_k (p_c^{(j)} + p_i^{(j)})$
- Recall: $R_e = \sum_k p_c^{(j)} / \sum_k (p_c^{(j)} + p_m^{(j)}) = PA$
- F-measure: $F_m = 2P_r \cdot R_e / (P_r + R_e)$

Currently, there have no open-source data for performance comparison. And most of the railway detection work is to detect and fit the two railroad line, which is quite different from the work of this paper.

In order to compare the performance of our model, we modified several current significant scene segmentation algorithms to fit our detection task and tested them on the dataset proposed in this paper. By changing the threshold from 0 to 1, we can get different network outputs to calculate precision and recall. Results are shown in Figure. 7 and Table 1.

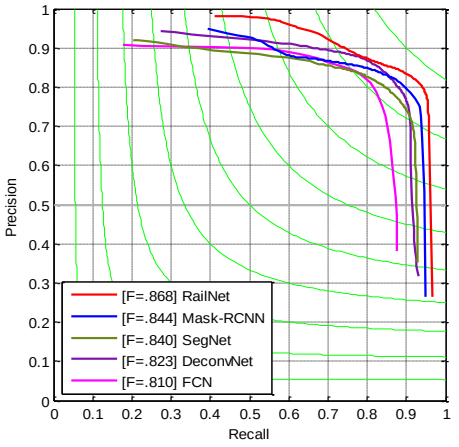

FIGURE 7. **The figure shows the precision-recall curves for different railroad detection methods. The X-axis is Recall, the Y-axis is Precision, and the green curve is F-measure** ($2 \cdot Precision \cdot Recall / Precision + Recall$). **From bottom to top, the F-measure values range from 0.1 to 0.9 respectively. The F-measure can reflect the accuracy of the model's detection. Our method has achieved the highest F-measure.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945633, IEEE Access

TABLE I
Detection results on RSDS. All calculations are implemented on the GPU.

	Pixel acc.	*Mean IoU*	*F-measure*	*Rate(fps)*
FCN[17]	0.807	0.773	0.810	8
SegNet[26]	0.866	0.843	0.840	6
DeconvNet[36]	0.851	0.827	0.823	6
Mask RCNN[3]	0.897	0.870	0.844	4
RailNet	**0.916**	**0.898**	**0.868**	**20**

Figure.8 shows some railroad detection results. Figure. 9 shows some of the comparisons results visually. All test images come from the test dataset of RSDS.

D. RESULT ANALYSIS

This paper mainly considers the railroad detection task of low-speed autonomous trains. According to the design specifications of low-speed trains in China. The maximum design speed is 80km/h, and the operating running speed is less than 60km/h. Emergency brake deceleration is not less than $1.2\,m/s^2$. Therefore, the emergency braking distance is approximately 118 meters. Our algorithm got a processing speed of 20 frames per second. That means the train runs a distance of 0.83 meters after processing one frame. The emergency braking distance is much larger than the train's run distance of processing for one frame. This means that whether the train can stop in time is mainly determined by the braking distance of the train and the effective detection distance. The algorithm proposed in this paper can meet the application requirements of railroad extraction accuracy. The speed of 20 frames/second can meet the basic detection requirements, but it does not meet the requirements of real-time detection of HD cameras. So future work will focus on improving detection speed and obstacle detection. The work in this paper is the basis of future research.

V. CONCLUSION

There are few methods for detecting railroad using learning algorithms. Because there is no relevant dataset and it is difficult to describe the railroad using a bounding box. In this paper, we have built a dataset (RSDS) for railroad segmentation. We propose a railroad detection network architecture that fully utilizes the features of the backbone network. We have modified other segmentation methods and compared them to our methods based on RSDS dataset. The experimental results show that the proposed method obtains the highest detection performance (Pixel accuracy, Mean IoU, F-measure) compared with other methods and has the fastest detection speed. Our method has stronger scene adaptability than traditional methods. However, this method is currently computationally intensive and requires a GPU to run. In the future, we hope to improve the algorithm to run the model on the onboard computer.

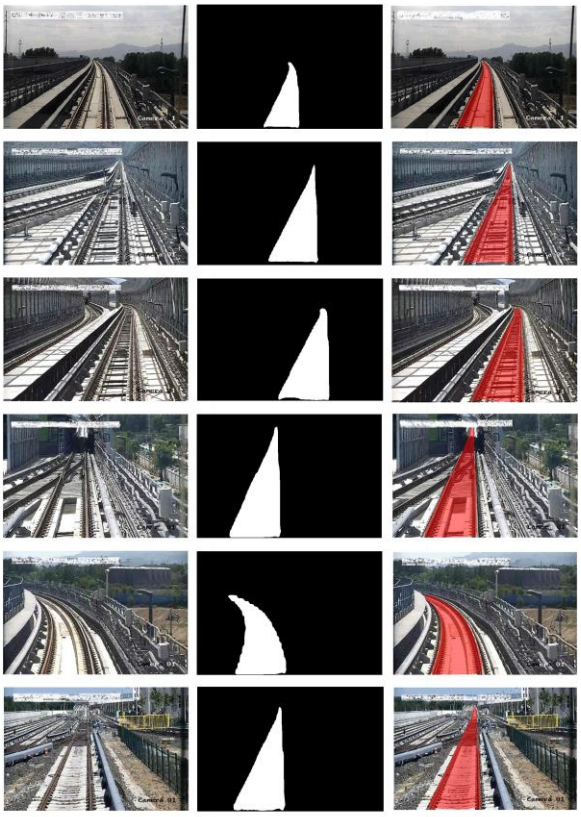

FIGURE 8. **RailNet results on the RSDS test set. From left to right: original images, output segmentation map, railroad mask.**

ACKNOWLEDGMENT

This research was supported by the Fundamental Research Funds for the Central Universities, China.[NO. E17JB00150]

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945633, IEEE Access

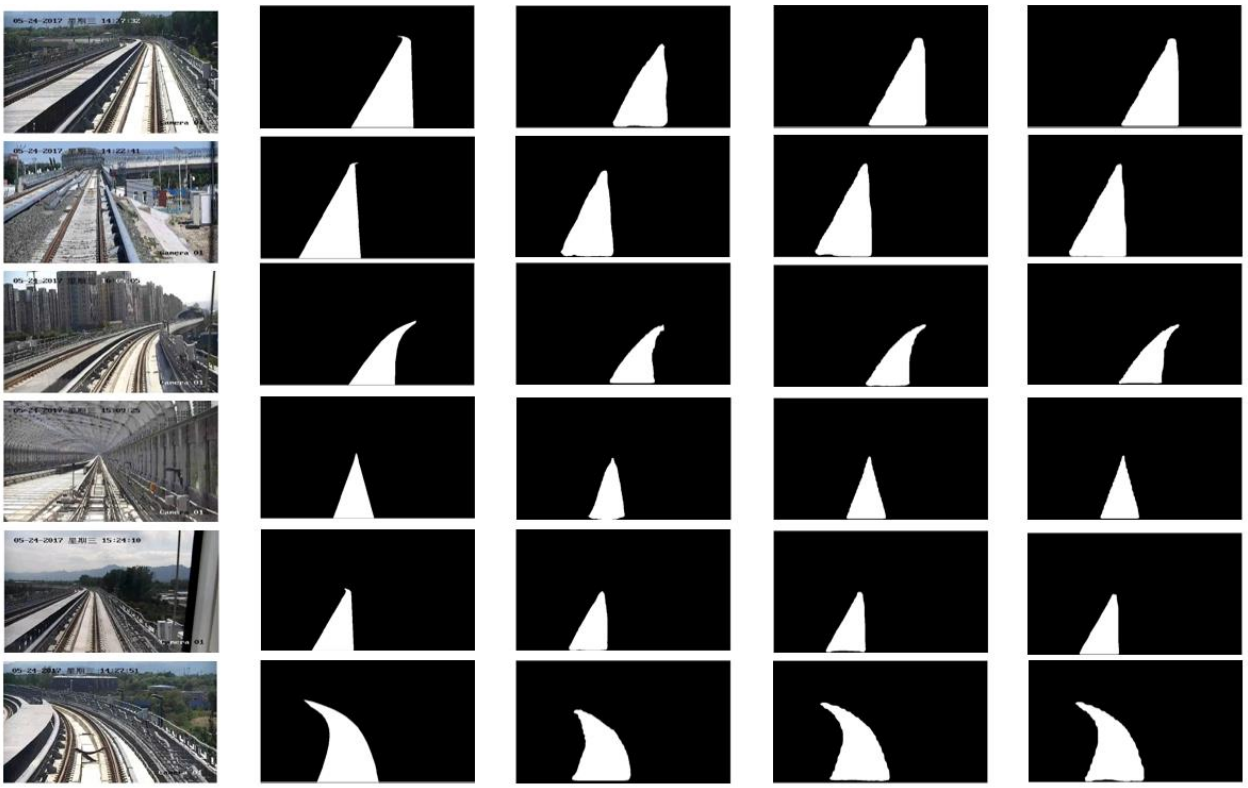

FIGURE 9. **Some comparison results on the RSDS test set. From left to right: original input images, ground truth, results of FCN, results of Mask RCNN and results of RailNet.**

REFERENCES

[1]. B. Le Saux, A. Beaupère, A. Boulch, J. Brossard, A. Manier and G. Villemin, "Railway Detection: From Filtering to Segmentation Networks," IEEE International Geoscience and Remote Sensing Symposium, Valencia, 2018, pp. 4819-4822.

[2]. X. Giben, V. M. Patel and R. Chellappa, "Material classification and semantic segmentation of railway track images with deep convolutional neural networks," IEEE International Conference on Image Processing (ICIP), 2015, pp. 621-625.

[3]. K. He, G. Gkioxari, P. Dollár and R. Girshick, "Mask R-CNN," 2017 IEEE International Conference on Computer Vision (ICCV),2017, pp. 2980-2988.

[4]. J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 6517-6525.

[5]. F. Kaleli and Y. S. Akgul, "Vision-based railroad track extraction using dynamic programming," 2009 12th International IEEE Conference on Intelligent Transportation Systems, St. Louis, MO, 2009, pp. 1-6.

[6]. Qi Z, Tian Y, Shi Y. “Efficient railway tracks detection and turnouts recognition method using HOG features”. Neural Computing and Applications, 2013, 23(1):245-254.

[7]. B. T. Nassu and M. Ukai, "Rail extraction for driver support in railways," 2011 IEEE Intelligent Vehicles Symposium (IV), Baden-Baden, 2011, pp. 83-88.

[8]. A. I. Purica, B. Pesquet-Popescu and F. Dufaux, "A railroad detection algorithm for infrastructure surveillance using enduring airborne systems," 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017, pp. 2187-2191.

[9]. Teng Z, Liu F, Zhang B. “Visual railway detection by superpixel based intracellular decisions”. Multimedia Tools and Applications, 2016, 75(5):2473-2486.

[10]. Arastounia M. “Automated Recognition of Railroad Infrastructure in Rural Areas from LIDAR Data”. Remote Sensing, 2015, 7(11):14916-14938.

[11]. B. Le Saux, A. Beaupère, A. Boulch, J. Brossard, A. Manier and G. Villemin, "Railway Detection: From Filtering to Segmentation Networks," IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium, Valencia, 2018, pp. 4819-4822.

[12]. B. Yang and L. Fang, "Automated Extraction of 3-D Railway Tracks from Mobile Laser Scanning Point Clouds," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2014, pp. 4750-4761.

[13]. J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. “Selective search for object recognition”. IJCV, 2013

[14]. I. Endres and D. Hoiem, "Category-Independent Object Proposals with Diverse Ranking," in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2014, pp. 222-234, Feb. 2014.

[15]. J. Pont-Tuset, P. Arbeláez, J. T. Barron, F. Marques and J. Malik, "Multiscale Combinatorial Grouping for Image Segmentation and Object Proposal Generation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, pp. 128-140.

[16]. Zitnick C.L., Dollár P. Edge Boxes: Locating Object Proposals from Edges[C]. ECCV 2014.

[17]. Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. “Visualizing higher-layer features of a deep network”. Technical Report 1341, University of Montreal, June 2009.

[18]. E. Shelhamer, J. Long and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2014, pp. 640-651.

[19]. R. Girshick, "Fast R-CNN," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 1440-1448.

[20]. K. Simonyan and A. Zisserman. “Very deep convolutional networks for large-scale image recognition”. ICLR, 2015

[21]. K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778.

[22]. T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature Pyramid Networks for Object Detection," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 936-944.
[23]. G. Ghiasi and C. C. Fowlkes. “Laplacian pyramid reconstruction and refinement for semantic segmentation”. In ECCV, 2016.
[24]. O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation[C]. In MICCAI, 2015
[25]. 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), 2018, pp. 286-291.
[26]. V. Badrinarayanan, A. Kendall and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, pp. 2481-2495.
[27]. Yu F, Koltun V. “Multi-Scale Context Aggregation by Dilated Convolutions”. ICLR. 2016.
[28]. Chen L C, Zhu Y, Papandreou G, et al. “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation”. ECCV, 2018.
[29]. Y. Xu, T. Fu, H. Yang and C. Lee, "Dynamic Video Segmentation Network," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 6556-6565.
[30]. H. Zhao, J. Shi, X. Qi, X. Wang and J. Jia, "Pyramid Scene Parsing Network," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6230-6239.
[31]. H. Zhang et al., "Context Encoding for Semantic Segmentation," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 7151-7160.
[32]. P. Lyu, C. Yao, W. Wu, S. Yan and X. Bai, "Multi-oriented Scene Text Detection via Corner Localization and Region Segmentation," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 7553-7563.
[33]. M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. “Spatial transformer networks”. In NIPS, 2015
[34]. Dutta, Abhishek and Zisserman, Andrew. The VIA Annotation Software for Images, Audio and Video. arXiv:1904.10699, 2019
[35]. Rodriguez, LA Fonseca, Jonny Alexander Uribe, and JF Vargas Bonilla. "Obstacle detection over rails using hough transform." In 2012 XVII Symposium of Image, Signal Processing, and Artificial Vision (STSIVA), pp. 317-322. IEEE, 2012.
[36]. Noh H , Hong S , Han B . "Learning Deconvolution Network for Semantic Segmentation". ICCV, 2015.