applied sciences

Article

Track Line Recognition Based on Morphological Thinning Algorithm †

Weilong Niu [1], Zan Chen [1], Yihui Zhu [2], Xiaoguang Sun [3] and Xuan Li [4,*]

1 School of Rail Transportation, Soochow University, Suzhou 215137, China
2 School of Automation, Nanjing University of Information Science and Technology, Nanjing 210044, China
3 School of Mechanical and Electrical Engineering, Yangzhou University, Yangzhou 225127, China
4 School of Mechanical and Electrical Engineering, Soochow University, Suzhou 215137, China
* Correspondence: xuanli@suda.edu.cn

† This paper is an extended version of a paper published in the 2021 IEEE Conference on Cognitive and Computational Aspects of Situation Management, CogSIMA held in Tallinn, Estonia, 14–22 May 2021.

Featured Application: In this study, an improved ZS thinning algorithm combined with a de-noising algorithm is proposed for track line detection; this algorithm can realize the detection function under complex road conditions, different weather and other scene conditions, and has higher feasibility and effectiveness.

Abstract: In the field of intelligent driving of freight trains, determining the track line ahead of the train is an important function in the autopilot technology of such trains. Combining the characteristics of freight railway tracks, we conduct an in-depth analysis of the shortcomings of object detection technology in extracting track lines and propose an improved Zhang–Suen (ZS) thinning theory for a railway track line recognition algorithm. Through image preprocessing and single pixel thinning steps, a continuous track line is obtained and then processed by a denoising algorithm to obtain a complete track line. Experimental results show that the track extracted by our method has good continuity and less noise. It can simultaneously perform track detection on straight roads, curves and turnouts, and is suitable for changing weather conditions such as sunny daytime, mild rainy daytime, cloudy daytime, night with lamp lighting and night without lamp lighting conditions.

Keywords: track line recognition; object detection; improved ZS thinning; single pixel; continuity

Citation: Niu, W.; Chen, Z.; Zhu, Y.; Sun, X.; Li, X. Track Line Recognition Based on Morphological Thinning Algorithm. *Appl. Sci.* **2022**, *12*, 11320. https://doi.org/10.3390/app122211320

Academic Editor: Giancarlo Mauri

Received: 3 October 2022
Accepted: 6 November 2022
Published: 8 November 2022

1. Introduction

In the field of intelligent driving of freight trains, track line identification ahead of the train plays a crucial role and involves more functions; this creates higher requirements for its real-time performance and accuracy [1]. The track line in this paper are two rails on the railway, used as a reference line for the boundary of the railway; the accuracy of recognition directly affects the train's perception of the range of environment factors ahead of it [2]. The dynamic and complex background interference caused by fasteners, transponders, gravel pavements, sleepers as shown in Figure 1, and natural light and shadows within the limits of the rails is the main constraint on the robustness and accuracy of track line recognition.

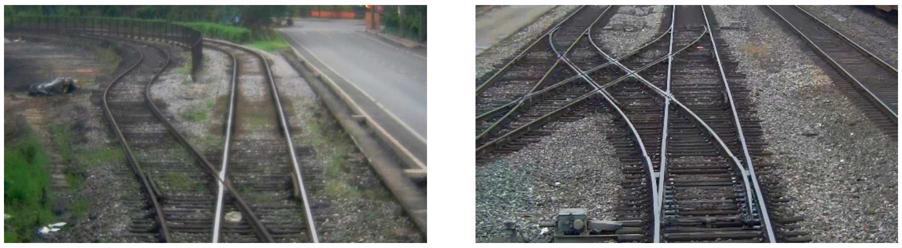


Figure 1. The complex environment of train tracks.

In the past decade, the development of deep learning and convolutional neural networks led to great progress in object detection technology. It includes a wide range of fields, for example, intelligent video surveillance, vehicle automatic driving, robot environment perception, and visual human–computer interaction. However, there are still many unresolved problems. Combining the current research by domestic and foreign scholars in the field of target detection and the scenarios applicable to this paper, the existing problems can be summarized as described below.

The narrow and long object problem: Current mainstream object detection algorithms include the one-step SSD (Single Shot MultiBox Detector) algorithm [3], the YOLO (You Only Look Once) algorithm [4], and the two-step Faster R-CNN (Region-CNN) algorithm [5]. Two types of object detection algorithms have limitations in terms of narrow and long object detection.

If the object is too narrow and long to become a small target, the ground truth box cannot find the corresponding default box to match it during training, so the detection effect is not reliable. The SSD network trains the anchor boxes, with each feature map randomly corresponding to several anchor boxes. If the corresponding anchor boxes are relatively small, it is difficult to obtain sufficient training using the pixels on the corresponding feature map. During testing, the predicted results may be inaccurate, which will greatly interfere with the normal results [6]. YOLO directly obtains prediction results through global features and depends entirely on data accumulation. It is not effective for detecting objects with weak characteristics. The two-step detection algorithm based on R-CNN does not scale the original image, resulting in a much slower detection speed than the one-step detection algorithm. Moreover, if there are many objects in the image and the aggregation is strong, the detection effect of the two-step detection algorithm is not ideal.

We define the proportion of effective information in the ground truth box as *PEI* in Equation (1): this can be used as a measure of the proportion of the object to be examined in the ground truth box, evaluating the feasibility of training and prediction results. In the object detection models based on deep learning, most deep learning models train images with a proportion of effective information higher than 50%, the accuracy of the model being even higher. In other words, the higher the proportion of effective information, the more accurate the model training.

$$PEI = \frac{con(k \in Effective|\bigcup k)}{con(\text{Ground truth box})} \tag{1}$$

where the ground truth box represents the true bounding box, *con* represents the amount of information contained, *k* represents the pixels, Effective represents the effective pixels (i.e., the pixels contained in the target to be checked), $\bigcup k$ represents the set of all pixels, *PEI* represents the proportion of the effective information in the true bounding box (i.e., the proportion of pixels contained in the target to be checked and all pixels in the bounding box).

As shown in Figure 2, the green rectangular box is the ground truth box. When the target to be detected is a person or signal lamp, $PEI > 0.5$ will be more ideal for training and prediction using the method of object detection. When the target to be detected is a narrow track line, $PEI < 0.5$ shows that the invalid information accounts for a large proportion in the ground truth box, which will cause great interference in later training and prediction, resulting in a poor final effect.

Relative entropy (*KL* divergence) is introduced to measure the difference between valid information and invalid information. As shown in Equation (2)

$$D_{KL}(P||Q) = \sum_{i=1}^{n} P(x_i)log_2(\frac{P(x_i)}{Q(x_i)}) \tag{2}$$

where D_{KL} represents the relative entropy and *KL* represents divergence which is used to measure the difference between valid information and invalid information. $P(x)$ represents the distribution of valid information and $Q(x)$ represents the distribution of invalid information; the difference between them can be measured by *P(x)/Q*(*x*), with *i* representing the random variable (1 . . . *n*). If the ratio is 1, it shows that they are very similar. The greater the interference of invalid information on effective information, the relatively poorer the object detection.

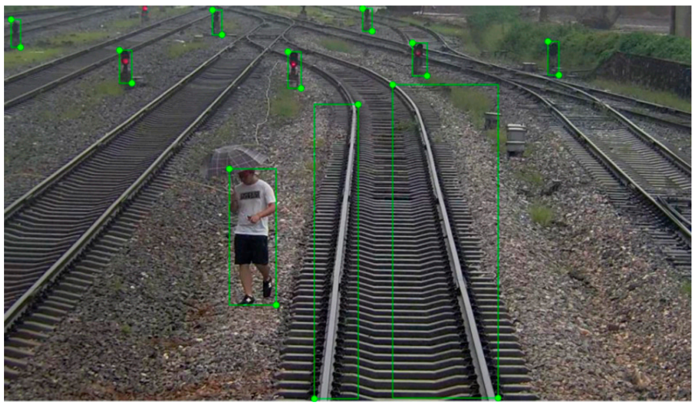


Figure 2. PEI contrast.

Rail line targets are relatively long and narrow. The straight parts are trapezoid in shape, with short upper and bottom edges and long lateral sides. The curved part presents various irregular shapes due to having different curvatures. The shape of the turnouts is more complicated and variable, and much redundant information exists. It is, therefore, difficult to use the corresponding labeling method of the object detection network for accurate data calibration; accordingly, the object detection method is less applicable to the track line detection problem.

Real-time detection: Since the object to be detected may be located anywhere in the image, and the size of the target is uncertain, it is usually necessary to construct a feature pyramid of the detection image [7]. A picture passes through the feature pyramid ($1\times, 2\times, 3\times$), three scales are enlarged and the calculation will be fourteen times larger. Sliding the window on multiple scales to search for the position of the object in an exhaustive manner results in a huge number of candidate regions. Many scholars realized effective improvements in this respect. Singh et al. [8] proposed an algorithm named SNIPER that performed efficient multi-scale training in instance-level visual recognition tasks. Their algorithm benefitted from batch normalization during training without the need to synchronize batch normalization statistics across GPUs and processed up to five images per second when reasoning with a single GPU. Shen et al. [9] proposed a multi-scale graph convolution network based on a spectral-graph wavelet framework to improve multi-scale representation learning. This network flexibly utilized continuously scaled multi-scale adjacent information to enhance the recognition ability of the learned multi-scale representation and provided stable feature extraction under the guarantee of the theoretical framework. Huang et al. [10] combined the Birch algorithm with an added multi-scale prediction of three-scale detection to improve real-time performance and accuracy. However, the detection speed was still slow and the prerequisite for the application of the object detection algorithm in the mobile terminal, autonomous driving, and embedded fields was to achieve lightweight processing. Deep neural networks are usually accompanied by problems such as large model size and high resource consumption. Implementing lightweight processing of object detection algorithm based on deep learning is still a key issue to be urgently solved.

Sample data set: Against the background of deep learning, the object detection range is increasingly widely used because of the lack of various training sample data. At present, the mainstream method for obtaining sample data involves manual methods for large-scale corpus labeling. For example, the ImageNet database established by Li Feifei and others, and the general large-scale labeling datasets such as PASCAL VOC and MS COCO. However, there is no clear sample of a railway track dataset. Our application scenario is a modern freight railway of nearly 10 km in length. On the one hand, using supervised learning methods makes the cost of manual labeling too large, and it is impossible to label all scenarios. On the other hand, the complicated road conditions such as straight roads, curves, and turnouts, as well as material and color are different. The marked data cannot adaptively and accurately identify new categories of objects in different scenarios. Therefore, in recent years, methods such as transfer learning or reinforcement learning based on weak supervision [11] or unsupervised [12,13] were proposed to train object detection models; however, combining prior knowledge to achieve adaptive object recognition in different scenarios is still a problem.

To address these issues, in our scenarios, we consider the shortcomings of existing deep learning-based object detection technology. Our requirement is to show a clear track line, rather than directly giving the track area, which means that it cannot be segmented by a semantic method. We, therefore, propose an improved ZS thinning theory of the railway track line recognition algorithm. This method refines the track line into a single pixel, and there are no overlapping pixels, which makes the connection ability of the track line stronger. In addition, this method does not require sample data and only relies on the shallow features of the image to complete the track line extraction. It does not require much computing power and has good real-time performance. It can simultaneously perform track detection on straight roads, curves and turnouts, and is suitable for changing weather conditions.

The remainder of this paper is organized as follows. In Section 2, we review the related works on track line detection. In Section 3, we describe our method, introducing the concrete steps of track line identification in the improved ZS thinning algorithm, and expounding the contour-based denoising algorithm. After showing the experimental results in Section 4, we finally draw our conclusion in Section 5.

2. Related Work

In the field of intelligent driving of trains, determining the track line in front of the train is an important issue in autopilot technology. To detect track lines, it is convenient to divide the track area, reduce the obstacle detection range, and improve the calculation speed and accuracy of target recognition. Currently, there are many methods and systems to help drivers drive safely [14,15]. With the development of intelligent train research technology and computer vision, new technologies are increasingly applied to track line detection ahead of trains.

Researchers such as Kaleli and Akgul [16] proposed a vision-based track extraction algorithm using dynamic programming. They used dynamic programming to calculate the optimal path: this extracted rail space with minimal cost, and then used dynamic programming to extract left and right rails at the same time. Gschwandtner et al. [17] proposed a simple and effective geometric constraint, which was derived from the unique properties of the rail, in order to achieve fast and reliable track detection. Nassu, Ukai et al. [18] introduced a method to perform rail extraction by matching edge features with candidate rail patterns modeled as parabolic segment sequences. Qi et al. [19] and others proposed a track recognition algorithm based on HOG features. Firstly, the image was divided into a series of cells, then the cell located at the bottom of the image was taken as the seed point, and the seed region growth was carried out by comparing the similarity of HOG features of adjacent cells. Finally, the track was extracted from the seed region growth results. In [20], a cost-effective visual turnout detection method was proposed. Berg [21] and other researchers first established a curvature map and a direction map according to camera parameters, then calculated the most likely track curvature according to the edge information of the image, obtained the initial mask of the near track from the curvature, and then corrected the initial mask by the learning method to obtain the near track. Espino et al. [22] first set up a sliding window at the starting position of the track. The point with the largest gradient change in the window was regarded as the characteristic point of the track, and the window slid upward along the gradient direction to continue searching for the characteristic point of the track. In [23], the cumulative value of pixel gradient was first calculated in blocks and the bottom track feature points were determined by combining prior knowledge. A method based on angle alignment measurement was proposed to extract the track feature points upward. Work by [24] implemented the recursive estimation method to extract track lines. Bettemir [25] and other scholars used a heuristic algorithm to detect rails and sleepers with high accuracy; the algorithm then processed the image through Gaussian filtering and edge detection. However, the detection distance of this heuristic algorithm was extremely limited. In [26], an automatic rail detection method was proposed; it used Hu moment invariant features to realize track line searches and B-spline curves as fitting models. It was necessary to fit the near and far view regions, respectively. The various methods proposed in the above research achieved certain effects for specific scenes. In reference [27], by moving the three-dimensional laser scanning technology, a region growth-fitting algorithm was realized by using the tracking points with clear intensity filtering and the strategy of K-means clustering fusion, so as to extract the vector line of the railway track, and then further distinguish the situations of bends and turnouts.

Compared with high-speed railway tracks, tram tracks [28] and subway tracks, however, the appearance of gravel roads, irregular sleepers, road junctions and ravine fences make the surrounding environment more complex; furthermore, there are curves with larger curvatures and complex turnout conditions. The commonly used track recognition system based on an edge detection operator [29] cannot adapt to a complex road environment that includes different light scenes and multiple turnout track routes, so real-time performance cannot meet specific requirements. An algorithm based on deep learning also requires high computing power equipment, which is not available for freight trains, and is costly.

Beginning with the autopilot technology of freight trains, we study track line detection based on shallow features and propose a track line extraction algorithm based on morphological thinning theory. Our algorithm is suitable for complex road conditions and changeable weather scenes, and ensures the continuity of track lines.

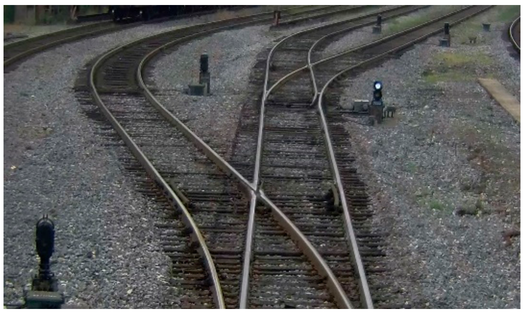
3. Methods

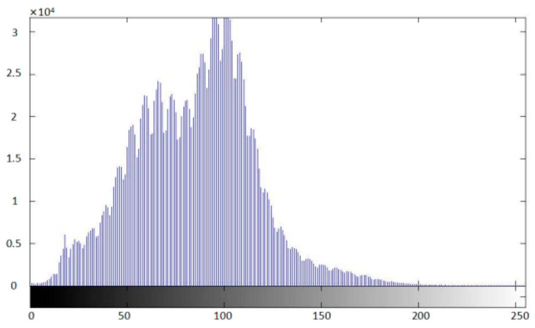
3.1. Image Preprocessing

The image obtained by the sensor contains much invalid information, so it is necessary to preprocess the entire image and reduce the calculations. The preprocessing in our method includes two sections: histogram equalization and image threshold segmentation.

When acquiring images by camera, factors such as uneven illumination and environmental noise affect the final image quality. The necessary image-enhancement process is conducive to improving the image effect. Since image enhancement refers to the degraded image features, it can process the edges, contours and contrast to improve the visual effect of the image, thereby improving the clarity of the image, highlighting the effective information in the image, compressing the invalid information and transforming the image into a form more suitable for human or computer analysis and processing. A basic algorithm in image-enhancement processing, histogram equalization is simple in principle, mature in technology, and can significantly improve image quality. The essence of histogram equalization is to perform non-linear stretching on an image with uneven pixel distribution, redistribute the pixel values of the image, and finally evenly distribute the pixel values in the entire grayscale range. In this paper, we use the method of global histogram equalization. The original image is shown in Figure 3 and the processing effect is shown in Figure 4.

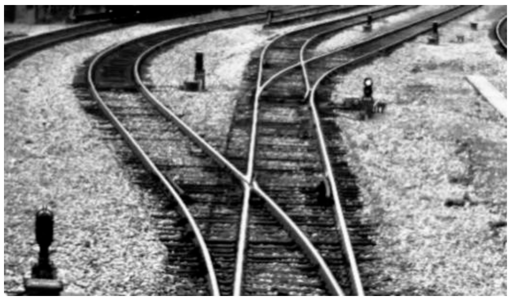
Image threshold segmentation is a widely used segmentation technique in image processing. It uses the difference in grayscale characteristics of the target area to be extracted from the image and its background to treat the image as a combination of two types of areas with different grayscale levels. It is reasonable to choose one threshold to determine whether each pixel in the image should belong to the target or the background, thus generating a corresponding binary image. Considering that the scene studied in this paper is complex and changeable, the target and background will change greatly; as a result, a single fixed threshold poses great limitations. In this paper, we use the adaptive threshold method, in which the optimal threshold should be adaptively generated according to each image frame. In this method, we iterate through different thresholds and calculate the intraclass variance between background and target for different thresholds. When the intraclass variance is maximized, the corresponding threshold will be the desired threshold [30]. The calculation process is as follows:

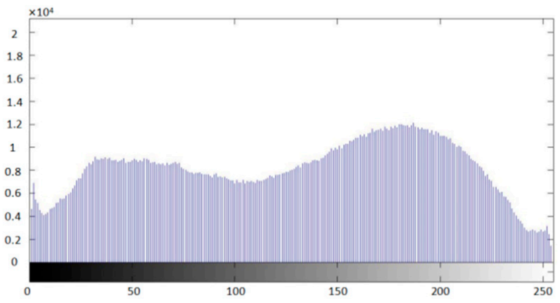




(**a**) (**b**)

Figure 3. (**a**) presents the original image of freight track and (**b**) shows the gray histogram image of freight track.





(**a**) (**b**)

Figure 4. (**a**) presents the image of histogram equalization effect and (**b**) shows gray histogram image of freight track.

Assume that the gray range of threshold t is 0, 1, 2..., $L-1$, and let the probability S_i of pixels with gray level i be:

$$S_i = \frac{n_i}{N} \tag{3}$$

$$\sum_{i=1}^{L-1} S_i = 1 \tag{4}$$

The probability value of the target is:

$$\omega_0(t) = \sum_{i=0}^{t} S_i \tag{5}$$

The mean value of target is:

$$\mu_0(t) = \sum_{i=0}^{t} \frac{iS_i}{\omega_0} \tag{6}$$

The probability value of the background is:

$$\omega_1(t) = \sum_{i=t+1}^{L-1} S_i \tag{7}$$

The mean value of background is:

$$\mu_1(t) = \sum_{i=t+1}^{L-1} \frac{iS_i}{\omega_1} \tag{8}$$

The intraclass variance value is:

$$f(t) = \omega_0(t)\omega_1(t)(\mu_0(t) - \mu_1(t))^2 \tag{9}$$

Traverse all the thresholds and find the corresponding t value that maximizes the intraclass $f(t)$, which is the threshold required. The effect of the adaptive threshold method after the histogram equalization in Figure 4 is shown in Figure 5.

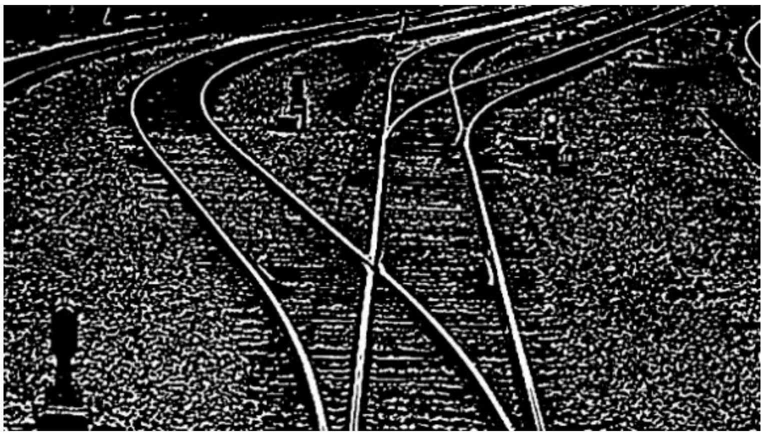


Figure 5. The result after adaptive threshold segmentation.

3.2. Improved ZS Thinning Algorithm

The ZS thinning algorithm proposed by Zhang and Suen [31] in 1984 is a very popular and fast parallel thinning algorithm [32–35]; it has the advantages of fast speed, good connectivity, simple principles, etc., [36]. It involves the following basic definitions: a pixel at location (x, y) has two horizontal and two vertical neighbors. The set of four pixels, noted $N_4(p)$, is called the 4-neighborhood of the pixel, as shown in Figure 6c. As is shown in Figure 6d, the four diagonal pixels, denoted $N_D(p)$, represent the diagonal neighborhood of the pixel. The points of $N_4(p)$ and $N_D(p)$ together are called the 8-neighborhood of the pixel, as shown in Figure 6b.

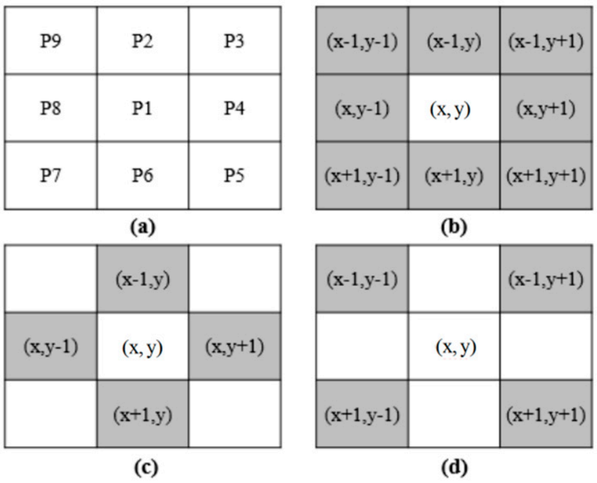


Figure 6. (**a**) 3 × 3 neighborhood; (**b**) 8-neighborhood; (**c**) 4-neighborhood; (**d**) D-neighborhood.

Its function is to perform two iterations on the entire binary map in the 3 × 3 neighborhood boundary points that must be deleted by operation search. However, the thinning result cannot be guaranteed to be a single pixel; moreover, different degrees of noise exist, which affects recognition of the track line.

Making full use of the advantages of the ZS thinning algorithm, we start by thinning into a single pixel and adapting to various complex scenes. An improved ZS thinning algorithm is proposed for track line detection. The specific process is as follows:

Set the target point to 1 and the background point to 0. The algorithm performs the following operations on the boundary points:

(1) In the binary image obtained after the preprocessing described in Section 3.1, the algorithm looks for 8-neighborhoods centered on the boundary point, denotes the center point as P1, and the adjacent points in the clockwise direction as P2, P3,..., P9, where P2 is directly above P1, as shown in Figure 6a.

Num(P1) represents the number of non-zero neighbors in P1's 8-neighborhood.

$$\mathrm{Num(P1)} = \sum_{i=2}^{9} P_i \tag{10}$$

S(P1) is the number of changes from 0 to 1 in the sequence P2, P3,..., P9, and the range is 0–4, which can be expressed in the following form.

$$\mathrm{S(P1)} = \sum_{i=1}^{4} A_i \tag{11}$$

$$A_i = \begin{cases} 1, & P_{2i-1} = 0 \; and \; (P_{2i} = 1 \; or \; P_{2i+1} = 1) \\ & 0, else \end{cases} \tag{12}$$

The reason for the processing result of the ZS parallel thinning algorithm being a non-single pixel width is that there are pixels with an angle of 90°, that is, existing points that are not deleted because they do not meet the conditions for deletion. Here, these points are defined as redundant points. The redundant points have four main directions, shown by the red dotted frames in Figure 7.

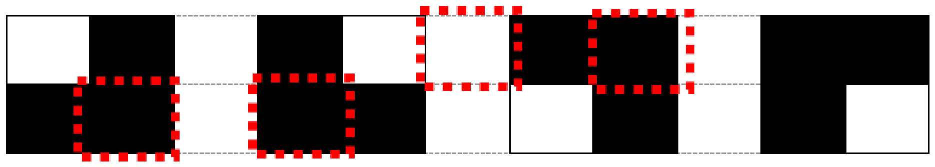


Figure 7. Redundant point direction.

Combining the four directions in which redundant points are generated, the specific reasons for their occurrence are analyzed in the 4-neighborhood. The number of non-zero neighbors in the 4-neighborhood of the central point P1.

$$N_{(4)} = \sum_{i}^{9} P_i \qquad (i = 2, 4, 6, 8) \tag{13}$$

Redundant points appear, $N_{(4)} = 2$. Express this in the form of a neighborhood template as, '*' represents 0 or 1, as follows:

0	1	*
1	1	0
*	0	0

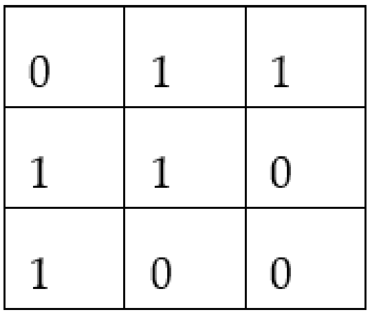
*	1	0
0	1	1
0	0	*

*	0	0
1	1	0
0	1	*

0	0	*
0	1	1
*	1	0

(2) Extraction of redundant points. In order to facilitate the extraction of redundant points, the 8-neighborhood points of P1 are binary coded in clockwise order in Step (1). P2, P3,..., P9 correspond to one binary bit. If the neighborhood point value is 1, the corresponding binary bit is 1. If the neighborhood point value is 0, the corre-

sponding binary bit is 0. As shown in Figure 8, the binary code corresponding to the eight neighborhoods of P1 is shown in Table 1.



0	1	1
1	1	0
1	0	0

Figure 8. 8-neighborhood of P1.

Table 1. Binary codes corresponding to the eight neighbors of P1.

P2	P3	P4	P5	P6	P7	P8	P9
1	1	0	0	0	1	1	0

The mathematical expression for calculating the 8-neighborhood binary coding of P1 is as follows:

$$B(P1) = \sum_{n=2}^{9} P_n * 2^{n-2} \tag{14}$$

The set of all values of $B(P1)$ is Σ.

(3) Mark the points that meet all the following conditions:

$$\begin{cases} a.\ 2 \le \mathrm{Num}(P1) \le 6 \\ b.S(P1) = 1 \ || \ B(P1) \in \Sigma \\ c.\ P2 * P4 * P6 = 0 \\ d.\ P4 * P6 * P8 = 0 \end{cases} \tag{15}$$

(4) Follow Step (3) and mark the points that meet all the following conditions:

$$\begin{cases} a.\ 2 \le \mathrm{Num}(P1) \le 6 \\ b.\ S(P1) = 1 \ || \ B(P1) \in \Sigma \\ c.\ P2 * P4 * P8 = 0 \\ d.P2 * P6 * P8 = 0 \end{cases} \tag{16}$$

After checking all boundary points, remove all marked points.

(5) Steps (3) and (4) constitute an iteration until no boundary points satisfy the marking conditions, and the area composed of the remaining points is the final result.

In order to show the experimental results more intuitively, we selected the same small segment of track to conduct ZS refinement and improve ZS refinement, respectively. The experimental results are shown in Figure 9. Figure 9a,c shows the original refinement results, while Figure 9b,d shows the corresponding refinement results for a single pixel. According to the experimental results, the track line is refined into a single pixel, and there are no overlapping pixels, which makes the connection ability of the track line stronger, demonstrating that our method can extract more continuous track lines.

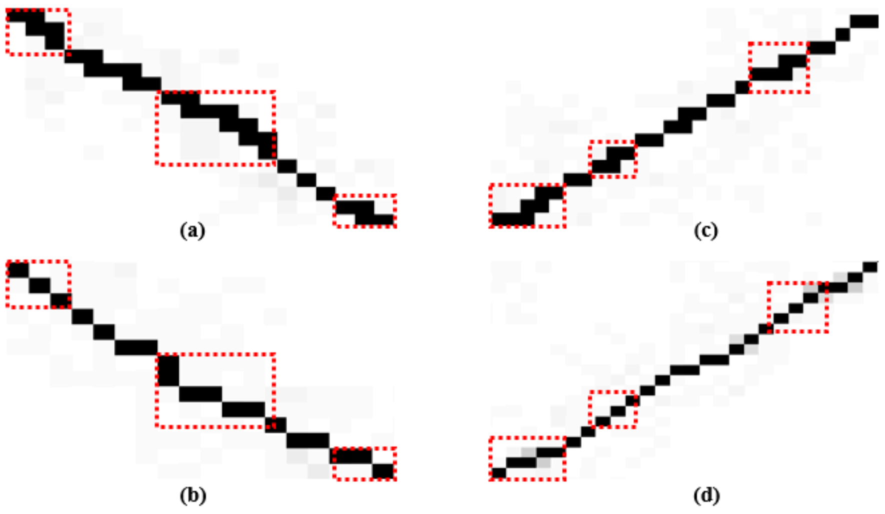


Figure 9. Single pixel rendering after macro-magnification. (**a**,**c**) is the original refinement results with the overlapping pixels, (**b**,**d**) is corresponding the refinement results for a single pixel.

In order to present a more significant visualization effect, we converted the white pixels to yellow without changing any line width. The effect of the improved ZS thinning algorithm is shown in Figure 10.

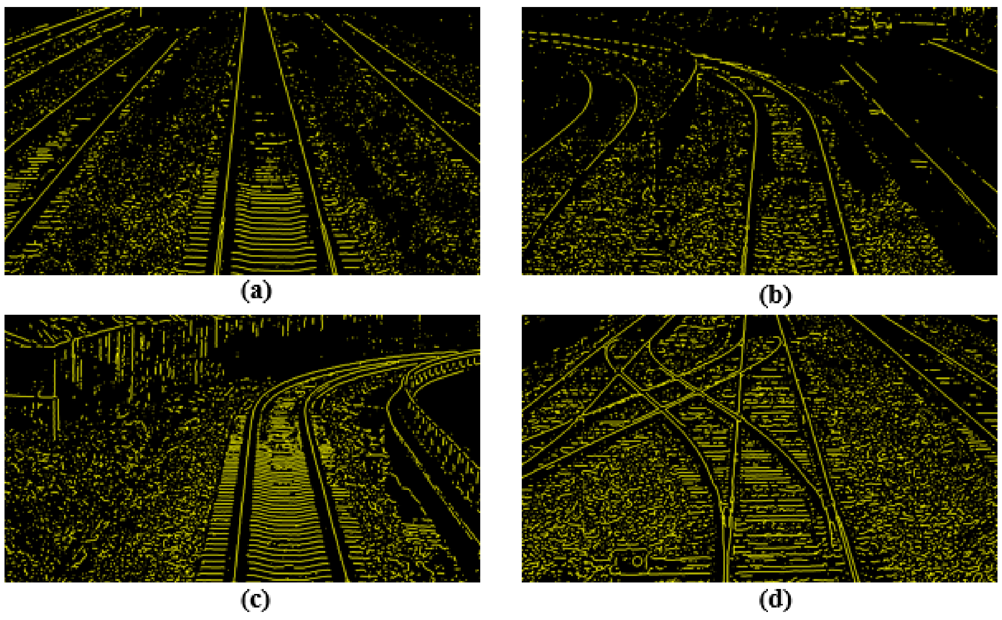


Figure 10. The result of freight railway after improved ZS algorithm processing. (**a**) is a multiple sets of straight track lines, (**b**) is curves with a small curvature track lines, (**c**) is the curved with a large curvature track line, (**d**) is the multiple turnout track lines.

3.3. Denoising Algorithm

The image obtained by the improved ZS thinning algorithm does not only include the track line, but also the interference from the factors beside the track, such as gravel, sleepers, gully fences, etc. Denoising must, therefore, be conducted to obtain the best track line. In this paper, a contour-based denoising method is proposed. According to the topology and distribution of the contour, the width, height and area of the contour are calculated using the minimum circumscribed rectangle. We then judge whether the contour is the boundary of the track profile we are looking for by the height, width and area of the minimum bounding rectangle of the contour, as shown in Figure 11.

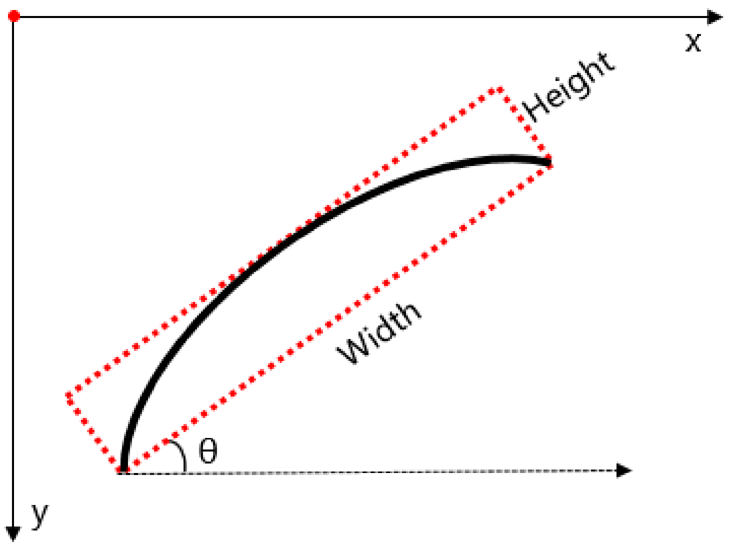


Figure 11. Schematic diagram of minimum circumscribed rectangle.

where θ is the included angle between the counterclockwise rotation of the horizontal axis and the first side of the encountered rectangle. The side length is width, and the top and bottom sides are height. The thinned image is I_1, imgW and imgH are the width and height of the original image, W and H are the width and height of the minimum circumscribed rectangle, T represents the width–height ratio of the minimum circumscribed rectangle, the length is L, and S represents the area.

$$\mathrm{T} = \mathrm{W}/\mathrm{H} \ \ W \in (0, imgW), H \in (0, imgH) \tag{17}$$

$$\mathrm{L} = \mathrm{Max}\{W, H\} \ W \in (0, imgW), H \in (0, imgH) \tag{18}$$

$$\mathrm{S} = \mathrm{W} * \mathrm{H} \ \ W \in (0, imgW), H \in (0, imgH) \tag{19}$$

The specific process is as follows:

(1) According to the length and area differences in track profile and side interference, find the minimum circumscribed rectangle of all profiles in I_1, as shown in Figure 12a. The blue parts in the figure represent the generated rectangular boxes. It can be seen from the figure that there are various large and small rectangular boxes. By setting the threshold, we screen out the rectangular boxes that meet the conditions, and all the pixel values in the rectangular boxes in this part will be retained; the remaining pixels are then set to zero. This method can eliminate most noise points. The specific steps are as follows: The length of each minimum circumscribed rectangle in I_1 is L_i. If L_i is less than the threshold parameter λ1, all pixels in its domain are set to 0. If it is larger than the parameter, the first λ2 bits are reserved in order from large to small. Many experiments demonstrated that most noise points can be eliminated when λ1 ∈ (40, 50) and λ2 ∈ (12, 15), and the processed image is recorded as I_2.

(2) Morphological dilation is performed on I_2 using 3 × 3 structural elements to eliminate small patches, holes, small discontinuities, etc. The processed result is recorded as I_3.

(3) Remember that λ is the length-screening coefficient of the minimum circumscribed rectangle, and δ is the area-screening coefficient of the minimum circumscribed rectangle. According to the geometric properties of image I_3, a filtering rule is defined:

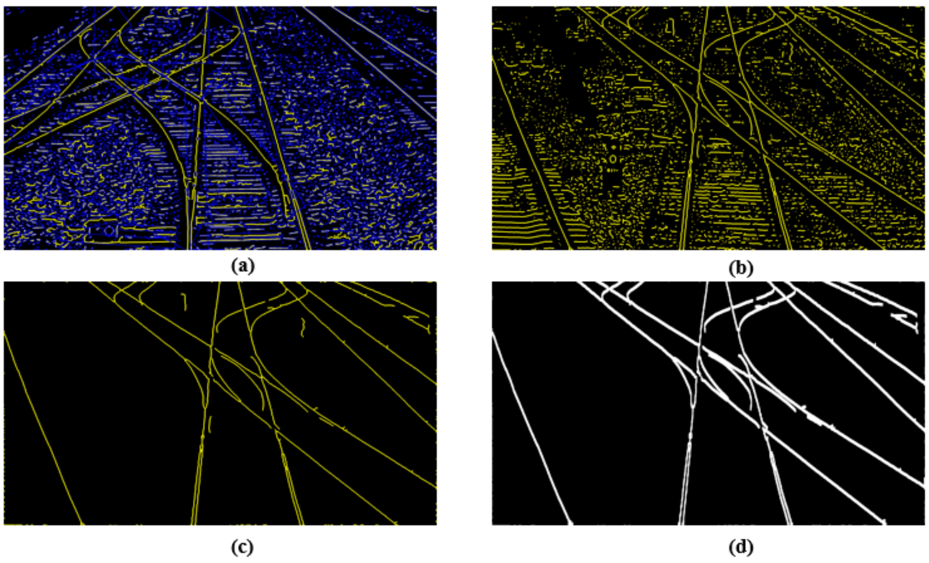


Figure 12. Effect of Denoising Algorithm. (**a**) is the minimum circumscribed rectangle of all profiles. (**b**) is the results of the improved ZS algorithm processing. (**c**) is the results of the denoising algorithm processing. (**d**) is the results of track line extraction.

When the height, width and area of the minimum bounding rectangle of a contour do not meet the formula, the contour is excluded (all pixels in the contour are set to 0). After screening the length and area coefficients, the complete track area is finally obtained. The experimental results show that when $\lambda \in (400, 500)$ and $\delta \in (1400, 1600)$, the denoising effect is significant.

$$\begin{cases} \lambda < W < \text{imgW} \\ \lambda < H < \text{imgH} \end{cases} \text{and } \delta < \text{S} < \text{imgW} * \text{imgH} \tag{20}$$

As shown in Figure 12, Figure 12a is the minimum circumscribed rectangle of all profiles in the image. Figure 12b shows the results of the improved ZS algorithm processing. Figure 12c shows the results of the denoising algorithm processing. Figure 12d shows the results of track line extraction. As can be seen from Figure 12d, after the improved refinement algorithm and denoising processing, the obtained track line has good continuity and less noise, which is very consistent with the actual track line.

3.4. Track Region Extraction Algorithm

The algorithm for track region extraction based on track line matching is realized on the basis of track line recognition. The algorithm for track area extraction must first determine the starting point of the track, which may have multiple starting points. Secondly, each starting point is used as a reference point to search for new pixels, which connect to the starting point along its connected domain. The matching criterion is to find the matching points that meet the gauge characteristics in the transverse direction by using the position of the points found and the idea of connectivity, so as to determine the corresponding relationship between the track lines in the image—whether they belong to the same line. The interference track line is then eliminated. Finally, according to the complete matching relationship, the track area to be checked is obtained. The specific process is as follows, and the algorithm diagram is shown in Figure 13.

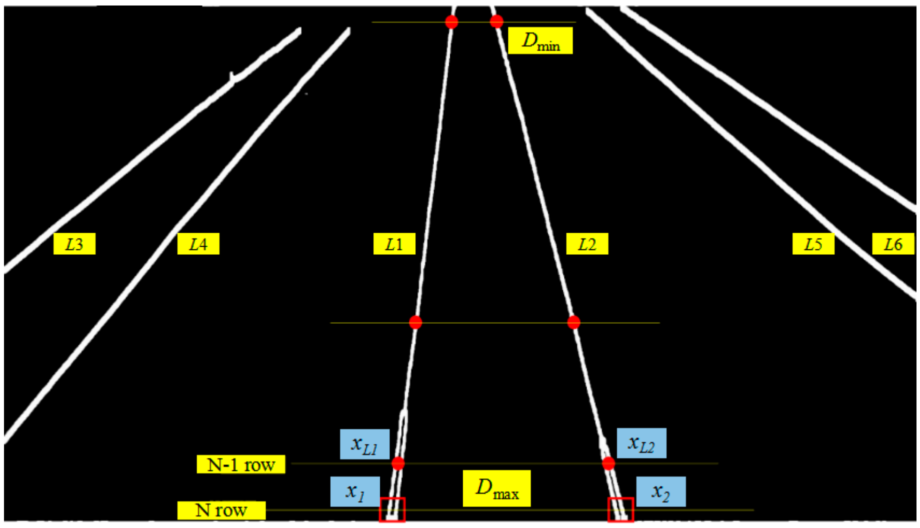


Figure 13. Schematic diagram of track region extraction algorithm based on track line matching.

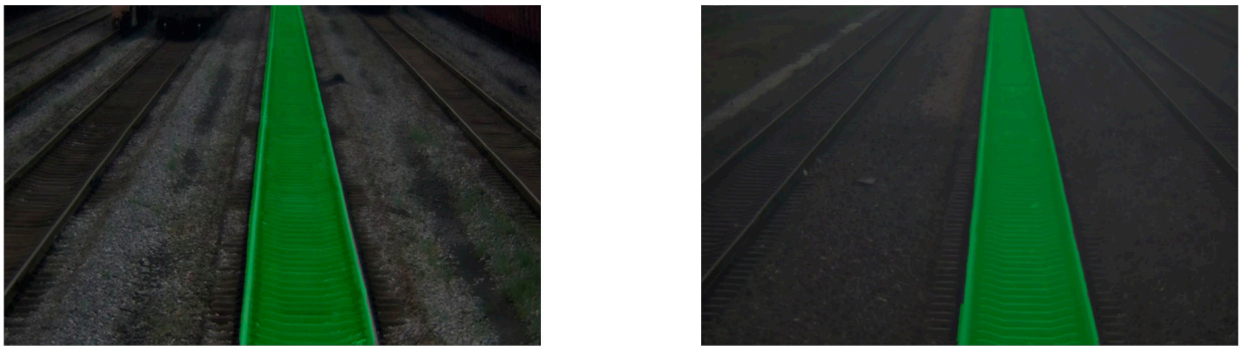
Determine the starting point. As the train is moving along a line, the image usually extends from the bottom up (excluding the lack of image visual field angle in the process of turning—blind area). Next, search for the starting point on the track line binary map from the *N* row (*N* starts at the bottom of the image). All points are considered as possible starting points, and the search area is the lateral width of the whole image. If a point x_i (i = 1, 2, 3 . . .) is found with a pixel value of 255 (white dot) and the pixel next to the left has a value of 0 (black dot), it is the starting point. If no possible starting point is found in the *N* row, then continue to n-1 row. By analogy, if the starting point cannot be found in the $N_{\max}$ row (indicating the maximum fracture line acceptable for the starting position), the search for the starting point in this connected domain will be abandoned and this step will be repeated in the next connected domain.

(1) Search for new pixels on the track line and match left and right in the transverse direction. After the initial point is determined in the first step, the new pixels x_{L_i} satisfying the conditions are searched upward in the connected area of each track line L_i (i = 1, 2, 3 . . .) in the f(x,y) image, and all the points (match points) on the same transverse surface with the point x_{L_i} are found on the left and right sides. At the same time, according to the trapezoidal characteristic of "near wide and far narrow" of the distance between the track lines, the maximum $D_{\max}$ and minimum $D_{\min}$ distance between the track lines are set, and the distance between the matching points is constantly updated in the process of matching from bottom to top, so as to reduce the mismatching rate. Considering the matching situation of track lines in complex side roads, this paper sets the limit that when the number of matching points on the same lateral surface exceeds the limit *T* number, the leftmost point is directly matched with the rightmost point to delimit the track area. By analogy, the track line is traversed from bottom to top, searching for all matching track lines.

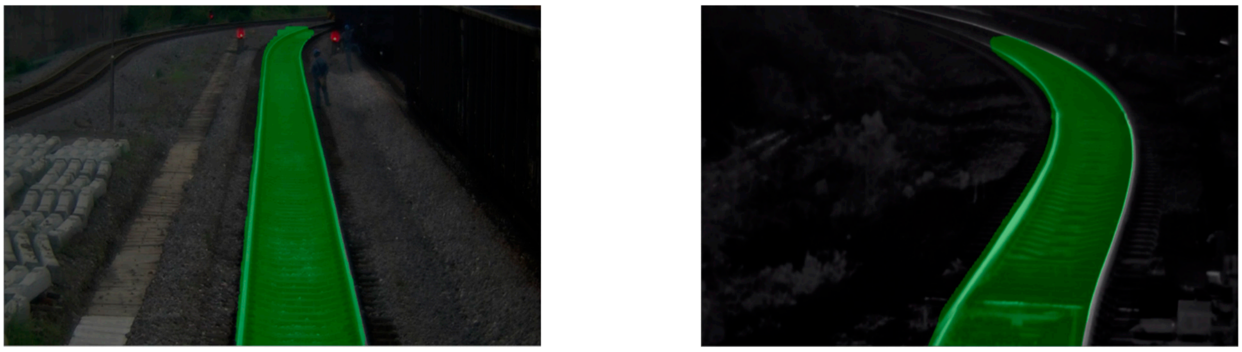
(2) Elimination of interference track lines. Not all the tracks in the image can match each other, and there will be tracks of different lengths. When searching for matching points on the track line, based on a large number of experiments, the track lines that can be fully matched are two or four matching points with little difference in the number of matching points (in the case of turnoff). Therefore, the most complete track lines are selected according to the number of matching points for the track area extraction.

(3) The matching points on the track line are obtained, which is the complete track running area of the train.

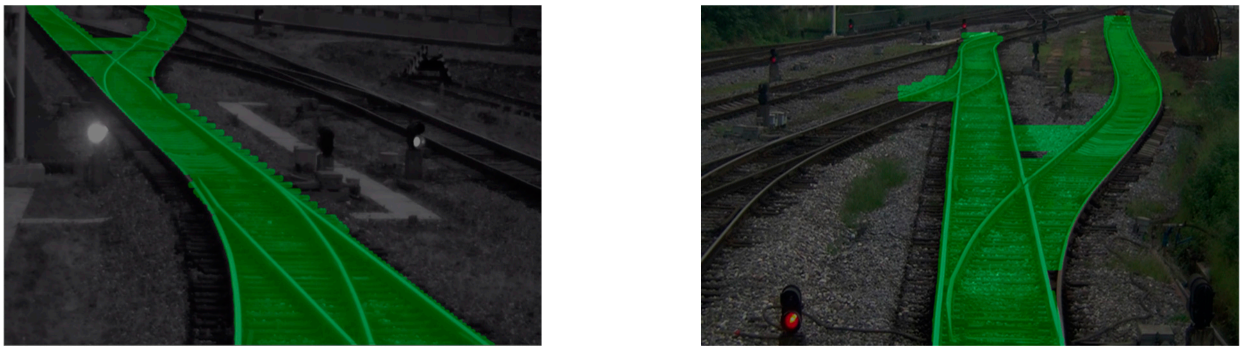
We proposed a method based on a track line of the track-line matching region extraction algorithm. This algorithm can accurately detect the track region, and is suitable for straight track lines, curved track lines and multiple turnout track lines; the testing effect is shown in Figure 14. Moreover, the algorithm is not affected by the influence of a downhill situation. Detection speed, minimal calculations and having the advantage of strong anti-interference mean that the algorithm satisfies real-time requirements.



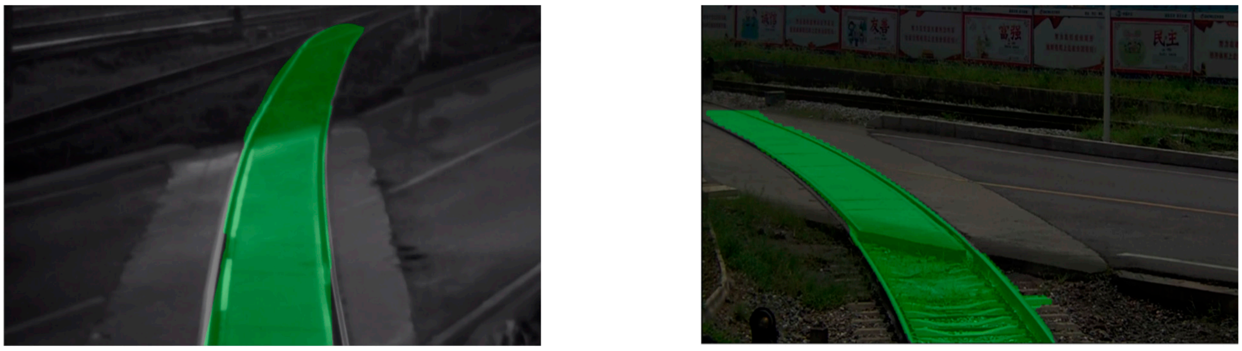
(**a**) straight track region extraction



(**b**) curved track region extraction



(**c**) Multiple turnout track region extraction



(**d**) Crossing track region extraction

Figure 14. Track region extraction based on track line matching under different lines and light conditions (the track region is colored in green).

4. Experiments and Results

After preprocessing with the improved ZS algorithm, the preliminary track line was obtained, and the complete track line was then also obtained using the denoising algorithm. Finally, the track region was extracted using the track line matching algorithm. In order to verify the effectiveness of the proposed algorithm under different routes and different lighting conditions, various environmental conditions were tested. The tracks of different routes are usually classified into a single set of straight track lines, multiple sets of straight track lines (only focusing on the current traffic routes), curved with a large curvature, curves with a small curvature, two turnout roads and multiple turnout track lines. After the above processing, we obtained the image of a single-pixel-wide track line and the track region, colored in green. Our experimental results for different scenes are shown in Figures 15 and 16.

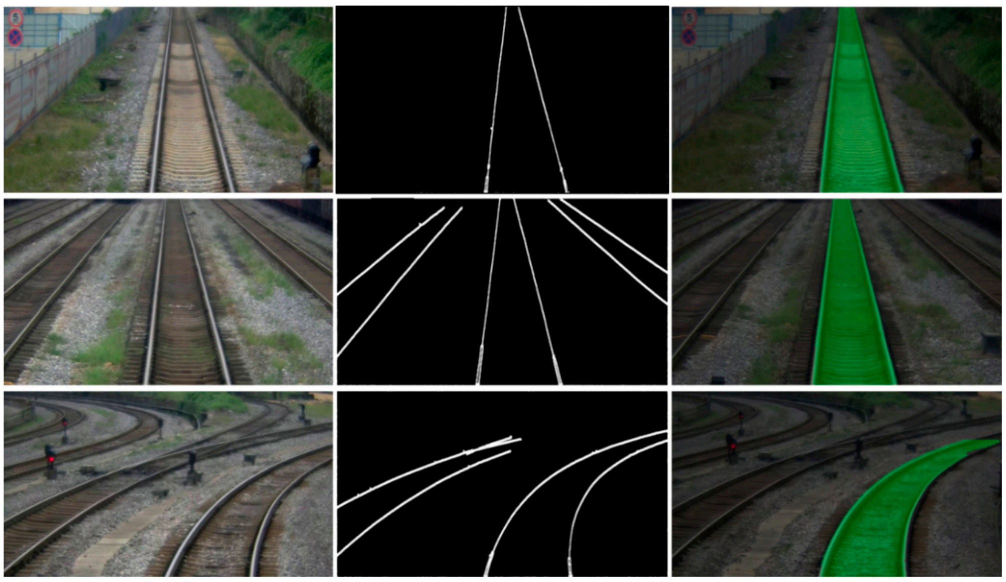


Figure 15. Single set of straight track lines and track region extraction.

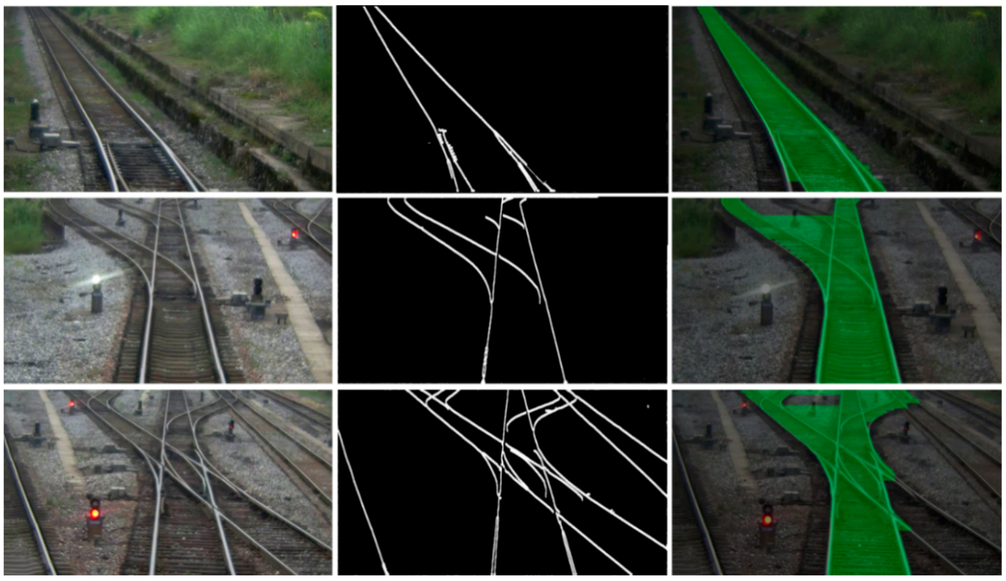


Figure 16. Multiple turnout track lines and track region extraction.

Different lighting scenes are usually divided into rainy weather straight roads, rainy weather curved roads, evening, evening without lamp lighting, evening with lamp lighting and crossing scenes. The effect is shown in Figures 17 and 18.

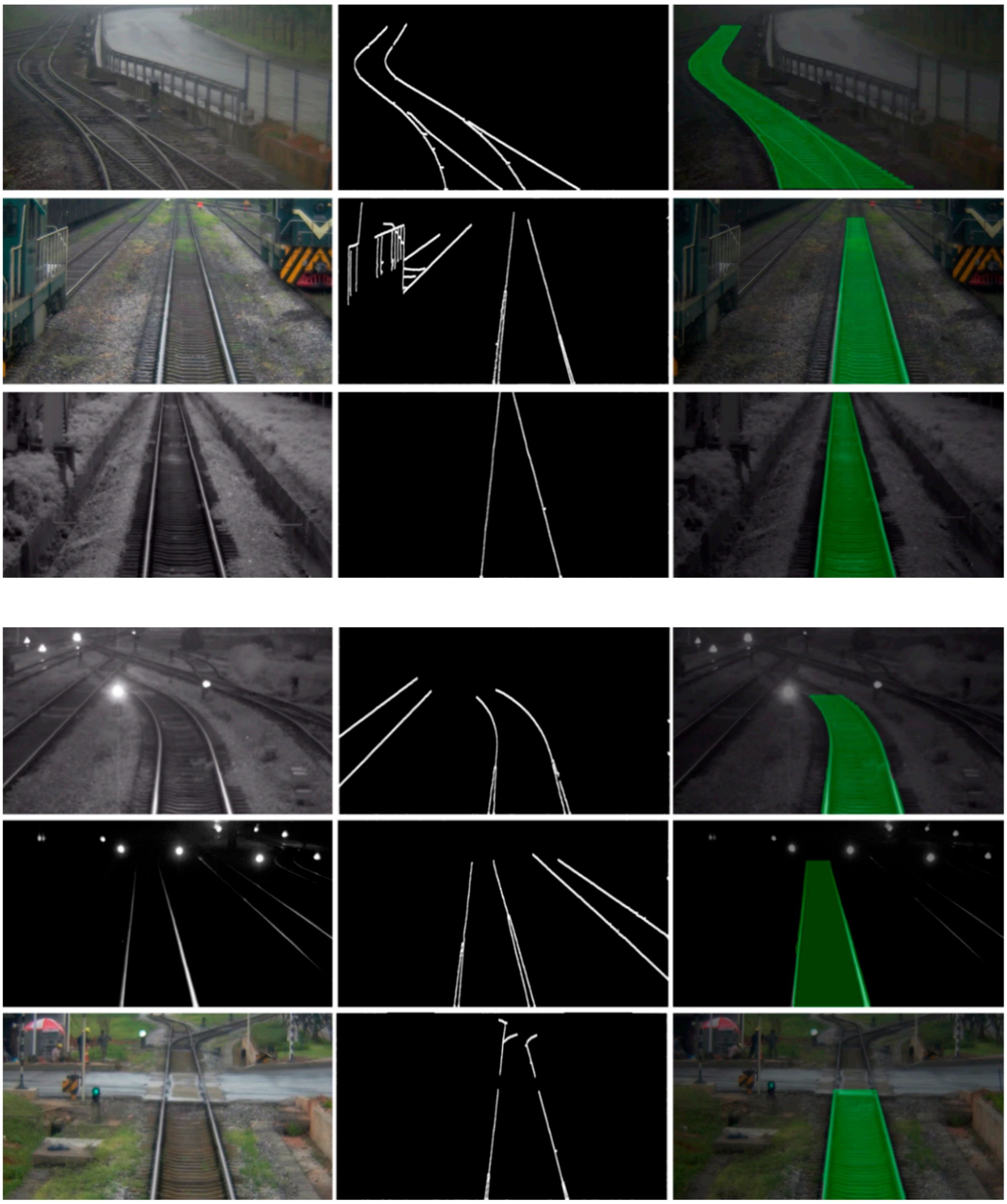


Figure 17. Different lighting scenes in the evening (including rainy weather, with lamp light and without lamp light).

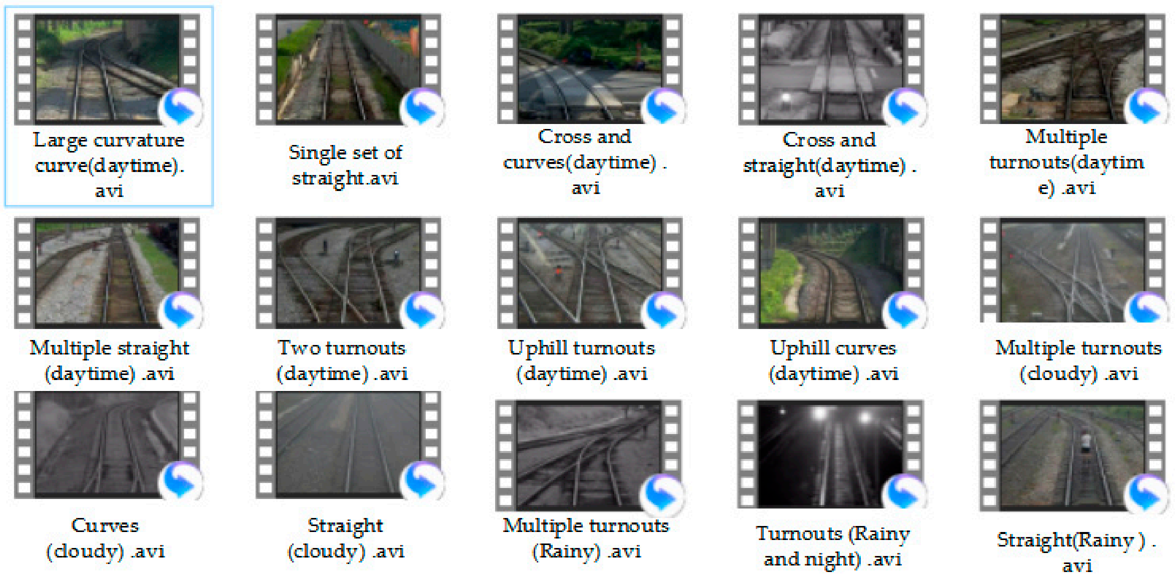

Large curvature curve(daytime). avi
Single set of straight.avi
Cross and curves(daytime) . avi
Cross and straight(daytime) . avi
Multiple turnouts(daytim e) .avi
Multiple straight (daytime) .avi
Two turnouts (daytime) .avi
Uphill turnouts (daytime) .avi
Uphill curves (daytime) .avi
Multiple turnouts (cloudy) .avi
Curves (cloudy) .avi
Straight (cloudy) .avi
Multiple turnouts (Rainy) .avi
Turnouts (Rainy and night) .avi
Straight(Rainy) . avi

Figure 18. Driving routes under different lighting conditions and directions.

In order to verify the effectiveness of our method more comprehensively, a test dataset was established. Our test data were collected in Shaoguan City, Guangdong Province, China. We built a railway track video library comprising 380 items, including videos of different lighting conditions and directions of different driving routes. The resolution of each frame was 1920×1080, and the frame rate was 10 fps. Several typical video frames in the video library are shown in Figure 18. Trains were traveling at a speed of 30 km/h, and the range of images captured by the camera was 10–150 m, which is sufficient to meet driver reaction needs. The algorithm code was implemented in the Visual Studio 2017 platform and runs in a 2 GB memory Intel Core I5 CPU. The speed at which the algorithm processes a 1920×1080 single-frame image is 150 ms.

To demonstrate the advantages of our method, we compared it with other methods; we tested the methods used in [30,32] and the related theoretical methods used in [36]. We randomly selected 10 videos from the dataset and processed them with these methods. Table 1 shows the results of the comparative experiment. In Table 2, TNF means total number of frames and GNF means total number of good frames (we removed the frames with discontinuities and noisy track lines). The method in [30] was based on Sobel. Research in [32] was based on ZS. Skeleton is a continuous thinning of the skeleton through morphological erosion and opening operations mentioned in the literature [36]. As can be seen from the table, under the same conditions, more frames with better extraction track lines are obtained through our method, and our processing accuracy is also higher. On sunny days, when the light was better, our accuracy is as high as 94.74%; however, in the evenings or on rainy days, our accuracy is a little lower; nevertheless, accuracy is still around 90%.

Table 2 shows the results of the comparison between the running time of the track line extraction algorithm and the track region extraction algorithm under different lighting and different route scenes. It can be seen from Table 3, with our algorithm and according to the different conditions, each frame-image-processing time is slightly different. When dealing with complex track lines and rainy weather conditions, the processing time is relatively long (five frames per second); however, when dealing with good light and a simple track line, the processing time is slightly shorter (seven frames per second). Nevertheless, there is not much difference between them in terms of processing speed. The algorithm also meets the real-time requirements of freight railway. (More than five frames per second is sufficient for freight railroads, which do not have fast trains.)

Table 2. The results of the comparative experiment under freight railway scenario.

	Scenes	TNF	GNF							
			Sobel		Skeleton		ZS		Ours	
			Good Frames	Accuracy Rate	Good Frames	Accuracy Rate	Good Frames	Accuracy Rate	Good Frames	Accuracy Rate
Video 1	rain, straights	445	331	74.38%	399	89.66%	387	86.96%	410	90.11%
Video 2	evening, turnouts, curves	331	156	47.12%	273	82.47%	281	83.63%	286	86.40%
Video 3	sunny, turnouts, curves	476	367	77.10%	432	90.75%	434	91.17%	451	94.74%
Video 4	evening, turnouts, straights	273	244	89.37%	236	86.44%	243	89.01%	245	89.74%
Video 5	sunny, straights turnouts, curves	281	215	76.51%	251	89.32%	244	86.83%	262	93.23%
Video 6	cloudy, turnouts, curves	479	435	90.81%	424	88.51%	438	91.44%	441	92.06%
Video 7	rain, curves	396	297	75.42%	340	86.03%	356	90.16%	364	92.02%
Video 8	evening, turnouts,	366	289	79.01%	322	88.65%	318	87.24%	329	90.23%
Video 9	sunny, straights turnouts,	421	374	89.24%	357	85.02%	378	90.16%	391	93.24%
Video 10	evening turnouts, curves	356	306	86.21%	316	89.19%	320	90.16%	327	92.65%

Table 3. Comparison of the extraction time for track areas in different scenes under the condition of finite computational force.

Scenes	Track Line Extraction Time (ms)	Track Region Extraction Time (ms)	Total Time (ms)
Simple track line (daytime)	40.3	110.5	150.8
Complex track line (daytime)	44.6	120.3	164.9
Simple track line (cloudy and rainy weather)	41.2	108.4	149.6
Complex track line (cloudy and rainy weather)	46.0	120.0	166.0
Multiple turnout track line (night)	39.7	107.2	146.9
Multiple turnout track line (headlights at night)	40.1	112.5	152.6

From the above, it can be concluded that our algorithm is better applicable to freight railways; however, in order to verify that this method is also applicable to other scenarios, we collected subway scene data which included videos of different directions of different driving routes. The resolution of each frame was also 1920 $\times$ 1080, and the frame rate was 25 fps; we extracted the track region using our algorithm, as shown in Figure 19.

We also chose video data from four different subway scenes and performed statistical analyses; the results are shown in Table 3. As can be seen in this table, we used the same conditions, retaining the images with a better processing effect and deleting the frames with discontinuities and noisy track lines, as shown by GNF. Table 4 shows that straight track line extraction accuracy achieves above 93%, curved track lines and multiple turnout line extraction also achieve more than 90% accuracy. Compared with freight railways in complex environments, subway lines are simple (no appearance of gravel roads, irregular sleepers, road junctions or ravine fences); we can, therefore, conclude that the method is also applicable to subway railways.

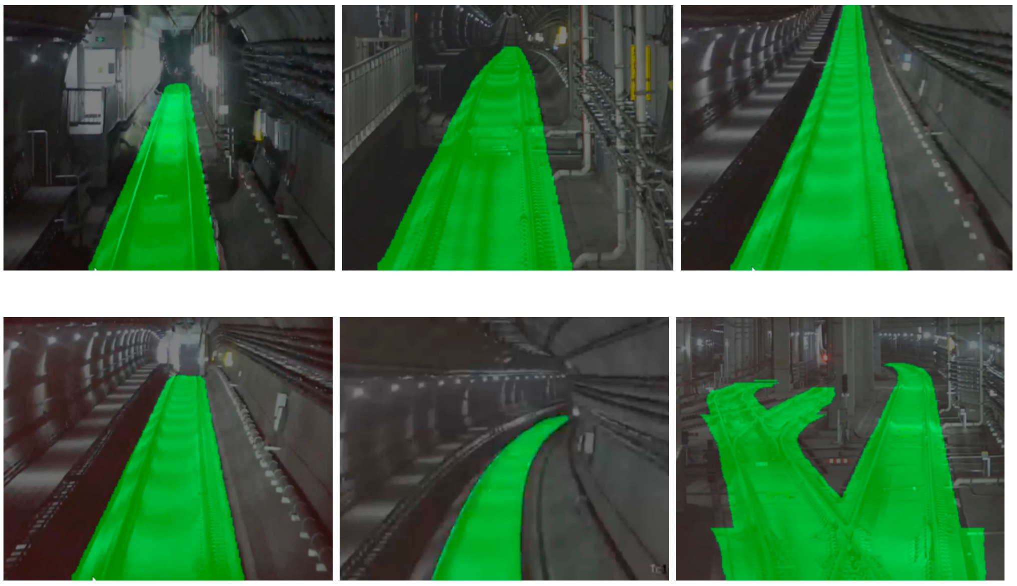


Figure 19. Track region extraction in different subway tunnel conditions (including uphill and downhill straight track line, curved track line, multiple turnouts).

Table 4. The results of the comparative experiment under subway tunnel conditions scenario.

	Video 1		**Video 2**		**Video 3**		**Video 4**	
Scenes	straight (Uphill, Downhill)		straights, Curves		straight (Different Light)		straight, turnouts	
TNF	498		486		524		516	
GNF	good frames	accuracy rate	good frames	accuracy rate	good frames	accuracy rate	good frames	accuracy rate
Result	464	93.17%	443	91.15%	492	93.89%	465	90.11%

5. Discussion

Beginning with the extraction of medium- and low-speed freight railway track lines, this paper proposes an improved ZS thinning algorithm combined with a denoising algorithm for track line detection. Our algorithm can realize the detection function under complex road conditions, different weather and other scene conditions, and has a higher feasibility and better effectiveness than several others. However, due to uneven illumination, shooting angle, heavy snow and other severe weather conditions, the shallow layer feature information of the track in the image is insufficient, which undoubtedly has adverse effects on the effect of the algorithm. In addition, the time taken to process a single frame of image exceeds 150 ms, which does not meet the video-processing requirement of 10 frames per second on a fast railroad. The next work direction of the authors is to achieve wider adaptability of the optimization algorithm and reduce its processing time.

Author Contributions: Conceptualization, W.N. and Y.Z.; methodology, W.N. and Z.C.; software, W.N. and Z.C.; validation, W.N., X.S. and Y.Z.; formal analysis, W.N.; investigation, X.L.; resources, W.N.; data curation, W.N.; writing—original draft preparation, W.N.; writing—review and editing, Y.Z.; visualization, X.S.; supervision, X.S.; project administration, W.N.; funding acquisition, W.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Natural Science Foundation of Jiangsu Province (Grant number BK20210710).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest. Weilong Niu and Yihui Zhu contributed equally to this work and should be considered co-first authors.

References

1. Punekar, N.S.; Raut, A.A. Improving railway safety with obstacle detection and tracking system using GPS-GSM model. *Int. J. Sci. Eng. Res.* **2013**, *4*, 282–288.
2. Kazanskiy, N.L.; Popov, S.B. Integrated design technology for computer vision systems in railway transportation. *Pattern Recognit. Image Anal.* **2015**, *25*, 215–219. [CrossRef]
3. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016.
4. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016.
5. Uijlings, J.R.R.; Van De Sande, K.E.A.; Gevers, T.; Smeulders, A.W.M. Selective search for object recognition. *Int. J. Comput. Vis.* **2013**, *104*, 154–171. [CrossRef]
6. He, Y.; Zhang, X.; Sun, J. Channel pruning for accelerating very deep neural networks. In Proceedings of the IEEE international Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
7. Hu, P.; Ramanan, D. Finding tiny faces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
8. Singh, B.; Najibi, M.; Davis, L.S. Sniper: Efficient multi-scale training. In Proceedings of the Advances in Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018; Volume 31.
9. Shen, Y.; Dai, W.; Li, C.; Zou, J.; Xiong, H. Multi-Scale Graph Convolutional Network With Spectral Graph Wavelet Frame. *IEEE Trans. Signal Inf. Process. Over Netw.* **2021**, *7*, 595–610. [CrossRef]
10. Huang, B.; Lin, H.; Hu, Z.; Xiang, X.; Yao, J. An improved YOLOv3-tiny algorithm for vehicle detection in natural scenes. *IET Cyber-Syst. Robot.* **2021**, *3*, 256–264. [CrossRef]
11. Ren, M.; Triantafillou, E.; Ravi, S.; Snell, J.; Swersky, K.; Tenenbaum, J.B.; Larochelle, H.; Zemel, R.S. Meta-learning for semi-supervised few-shot classification. In Proceedings of the 6th International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
12. Rahman, S.; Khan, S.; Porikli, F. Zero-shot object detection: Learning to simultaneously recognize and localize novel concepts. In Proceedings of the Asian Conference on Computer Vision, Perth, Australia, 2–6 December 2018.
13. Demirel, B.; Cinbis, R.G.; Ikizler-Cinbis, N. Zero-shot object detection by hybrid region embedding. In Proceedings of the British Machine Vision Conference 2018, Newcastle, UK, 3–6 September 2018.
14. Ye, T.; Wang, B.; Song, P.; Li, J. Automatic railway traffic object detection system using feature fusion refine neural network under shunting mode. *Sensors* **2018**, *18*, 1916. [CrossRef] [PubMed]
15. Sinha, D.; Feroz, F. Obstacle detection on railway tracks using vibration sensors and signal filtering using Bayesian analysis. *IEEE Sens. J.* **2015**, *16*, 642–649. [CrossRef]
16. Kaleli, F.; Akgul, Y.S. Vision-based railroad track extraction using dynamic programming. In Proceedings of the 12th International IEEE Conference on Intelligent Transportation Systems, St. Louis, MO, USA, 4–7 October 2009.
17. Gschwandtner, M.; Pree, W.; Uhl, A. Track detection for autonomous trains. In *International Symposium on Visual Computing*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 19–28.
18. Nassu, B.T.; Ukai, M. Rail extraction for driver support in railways. In Proceedings of the 2011 IEEE Intelligent Vehicles Symposium (IV), Baden-Baden, Germany, 5–9 June 2011.
19. Qi, Z.; Tian, Y.; Shi, Y. Efficient railway tracks detection and turnouts recognition method using HOG features. *Neural Comput. Appl.* **2013**, *23*, 245–254. [CrossRef]
20. Wohlfeil, J. Vision based rail track and switch recognition for self-localization of trains in a rail network. In Proceedings of the 2011 IEEE Intelligent Vehicles Symposium (IV), Baden-Baden, Germany, 5–9 June 2011.
21. Berg, A.; Öfjäll, K.; Ahlberg, J.; Felsberg, M. Detecting rails and obstacles using a train-mounted thermal camera. In Proceedings of the Scandinavian Conference on Image Analysis, Copenhagen, Denmark, 15–17 June 2015.
22. Espino, J.C.; Stanciulescu, B. Rail extraction technique using gradient information and a priori shape model. In Proceedings of the 15th International IEEE Conference on Intelligent Transportation Systems, Anchorage, AK, USA, 16–19 September 2012.
23. Wu, H.; Siu, W.C. Real time railway extraction by angle alignment measure. In Proceedings of the 2015 IEEE International Conference on Image Processing (ICIP), Quebec City, QC, Canada, 27–30 September 2015.
24. Nassu, B.T.; Ukai, M. A vision-based approach for rail extraction and its application in a camera pan–tilt control system. *IEEE Trans. Intell. Transp. Syst.* **2012**, *13*, 1763–1771. [CrossRef]

25. Bettemir, Ö.H. Detection of railway track from image by heuristic method. In Proceedings of the 2015 23nd Signal Processing and Communications Applications Conference, Malatya, Turkey, 16–19 May 2015.
26. Dong, Y.; Guo, B. Railway track detection algorithm based on Hu invariant moment feature. *J. China Railw. Soc.* **2018**, *40*, 64–70.
27. Zou, R.; Fan, X.; Qian, C.; Ye, W.; Zhao, P.; Tang, J.; Liu, H. An efficient and accurate method for different configurations railway extraction based on mobile laser scanning. *Remote Sens.* **2019**, *11*, 2929. [CrossRef]
28. Qiang, X.; Zhang, Z.; Chen, Q.; Wu, C.; Wang, Y. Video-based adaptive railway recognition in complex scene. In Proceedings of the 2016 International Conference on Audio, Language and Image Processing (ICALIP), Shanghai, China, 11–12 July 2016.
29. Cai, S. Railway recognition based on edge detection. *Railw. Comput. Appl.* **2009**, *25*, 20.
30. Otsu, N. A threshold selection method from gray-level histograms. *IEEE Trans. Syst. Man Cybern.* **1979**, *9*, 62–66. [CrossRef]
31. Zhang, T.Y.; Suen, C.Y. A fast parallel algorithm for thinning digital patterns. *Commun. ACM* **1984**, *27*, 236–239. [CrossRef]
32. Chen, W.; Sui, L.; Xu, Z.; Lang, Y. Improved Zhang-Suen thinning algorithm in binary line drawing applications. In Proceedings of the 2012 International Conference on Systems and Informatics, Yantai, China, 19–20 May 2012.
33. Abu-Ain, T.; Abdullah, S.N.H.S.; Bataineh, B.; Omar, K. A Fast and Efficient Thinning Algorithm for Binary Images. *J. ICT Res. Appl.* **2013**, *7*, 205–216. [CrossRef]
34. Boudaoud, L.B.; Sider, A.; Tari, A. A new thinning algorithm for binary images. In Proceedings of the International Conference on Control, Engineering & Information Technology (CEIT), Tlemcen, Algeria, 25–27 May 2015.
35. Ben Boudaoud, L.; Solaiman, B.; Tari, A. A modified ZS thinning algorithm by a hybrid approach. *Vis. Comput.* **2018**, *34*, 689–706. [CrossRef]
36. Prakash, R.P.; Prakash, K.S.; Binu, V.P. Thinning algorithm using hypergraph based morphological operators. In Proceedings of the IEEE International Advance Computing Conference, Banglore, India, 12–13 June 2015.