sensors

Article

An Adaptive Track Segmentation Algorithm for a Railway Intrusion Detection System

Yang Wang [1,2], Liqiang Zhu [1,2,*], Zujun Yu [1,2] and Baoqing Guo [1,2]

[1] School of Mechanical, Electronic and Control Engineering, Beijing Jiaotong University, Beijing 100044, China; 12116331@bjtu.edu.cn (Y.W.); zjyu@bjtu.edu.cn (Z.Y.); bqguo@bjtu.edu.cn (B.G.)

[2] Key Laboratory of Vehicle Advanced Manufacturing, Measuring and Control Technology (Beijing Jiaotong University), Ministry of Education, Beijing 100044, China

* Correspondence: lqzhu@bjtu.edu.cn; Tel.: +86-10-51684151

Received: 6 May 2019; Accepted: 3 June 2019; Published: 6 June 2019

Abstract: Video surveillance-based intrusion detection has been widely used in modern railway systems. Objects inside the alarm region, or the track area, can be detected by image processing algorithms. With the increasing number of surveillance cameras, manual labeling of alarm regions for each camera has become time-consuming and is sometimes not feasible at all, especially for pan-tilt-zoom (PTZ) cameras which may change their monitoring area at any time. To automatically label the track area for all cameras, video surveillance system requires an accurate track segmentation algorithm with small memory footprint and short inference delay. In this paper, we propose an adaptive segmentation algorithm to delineate the boundary of the track area with very light computation burden. The proposed algorithm includes three steps. Firstly, the image is segmented into fragmented regions. To reduce the redundant calculation in the evaluation of the boundary weight for generating the fragmented regions, an optimal set of Gaussian kernels with adaptive directions for each specific scene is calculated using Hough transformation. Secondly, the fragmented regions are combined into local areas by using a new clustering rule, based on the region's boundary weight and size. Finally, a classification network is used to recognize the track area among all local areas. To achieve a fast and accurate classification, a simplified CNN network is designed by using pre-trained convolution kernels and a loss function that can enhance the diversity of the feature maps. Experimental results show that the proposed method finds an effective balance between the segmentation precision, calculation time, and hardware cost of the system.

Keywords: railway intrusion detection; scene segmentation; scene recognition; adaptive feature extractor; convolutional neural networks

1. Introduction

With a continuous increase in the public's expectation for railway safety, railway intrusion detection systems require more effective technology to detect objects intruding into the track area and to provide real-time alarm information for the command center [1]. Railway intrusion behavior is defined as an object intruding into the track area and endangering the safe operation of trains. Typical intruding objects include rocks falling from a hill beside railway line or a tunnel entrance, pedestrians, vehiclesand their cargo staying in the railroad crossing area or falling from the bridge over the railway.

Depending on the detecting principle, railway intrusion detection systems can be divided into two categories: the contact type and the non-contact type. A representative of the contact type is the protective metal net installed along the line to block an object from intruding into the clearance, and the system will send the alarm information when the physical deformation of the net is measured by a dual-power sensor [2] or fiber grating sensor [3,4]. The systems based on the non-contact measurement technology use infrared sensor [5] or laser scanner [6,7] to get the size and location of the intruding object [8]. Video surveillance is also widely used as another kind of non-contact intrusion detection systems because of the large monitoring area, convenient installation, maintenance, and good observable results [9]. As shown in Figure 1, we established an intrusion detection system for the Shanghai–Hangzhou high-speed railway in China. The system contains data process servers, communication networks, and 1550 cameras, including both of fixed and PTZ cameras.

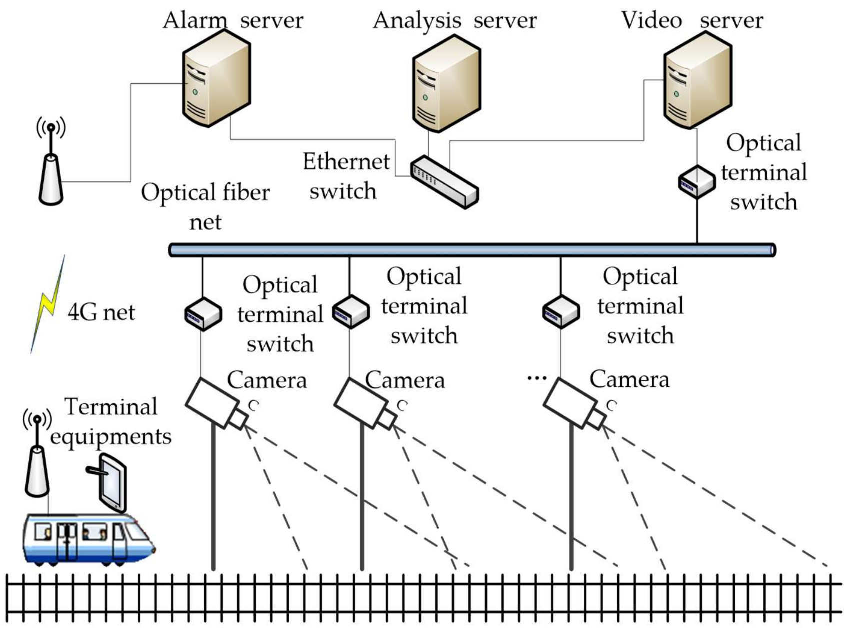


Figure 1. Structure of the railway intrusion detection system.

The threat level of an intrusion behavior will be evaluated by the category, location, and moving trajectory of the object with respect to the track area. The information of the intruding object can be extracted by image processing algorithms, e.g., density-based spatial clustering of applications with noise (DBSCAN) [10], fast background subtraction (FBS) [11], Kalman filtering [12], principal components analysis (PCA) [13]. DBSCAN uses extremum points of scan sequence as core objects of clustering, and the movement and distribution characters are used to judge whether the cluster is a train or other foreground object. FBS projects the scene image into one dimension (x or y dimension) to locate position of the foreground object by the change of the peak value. KF classifies the objects acquired via image background subtraction by support vector machine (SVM), and then using the Kalman-filter tracking algorithm to analyze the behavior and moving trend of the objects. PCA projects the statistic of the scene images and the successive images in a transformation space and calculates the Euclidean distance, which is greater than a threshold, is considered like belonging to motion objects. Most of the above-mentioned algorithms only focus on the foreground object, rather than the track area in the background. Therefore, the position and boundary of track area are still delineated manually in advance, as shown in Figure 2.

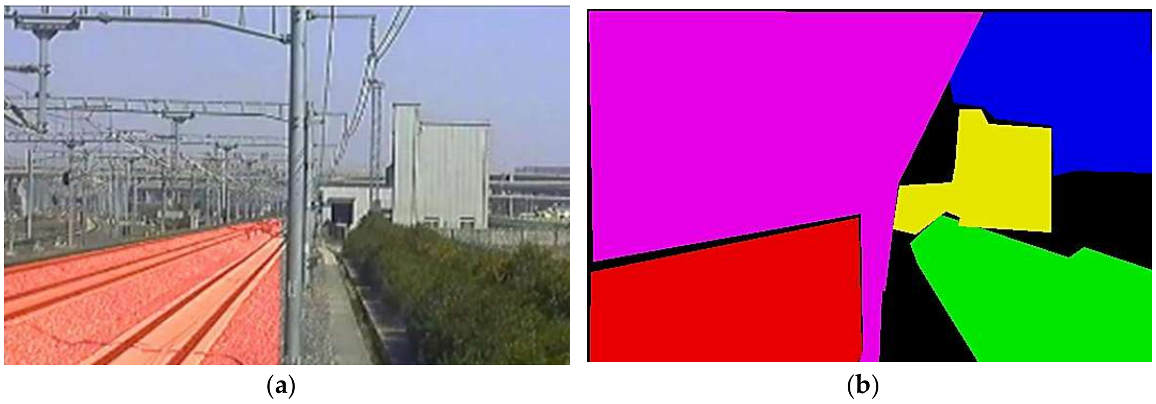


Figure 2. Railway scene and the local areas, labeled manually. The image quality is susceptible to external influences, such as the illumination, weather, and even the dust on the lens. (**a**) The red area is the track area to be surveilled. The track area includes the rails, sleepers, subgrades, or high-speed railway slabs. (**b**) Labeling the different area of the railway scene with different colors by manual, including track area (**red**), sky (**blue**), catenary system (**purple**), green belt (**green**), and ancillary buildings (**yellow**). The precision depends on the patience of the manual operator.

The precision of the track area boundary directly affects the reliability of intrusion detection. With an increasing number of surveillance cameras along the railway line, especially as some PTZ cameras will change their focal lengths and angles temporarily for different applications, manual labeling has become time-consuming and laborious. Thus, for the efficiency of the railway intrusion detection system, a scene segmentation algorithm is needed to recognize the track area and delineate the boundary automatically. The algorithm will be applied to initialize surveillance areas after the installation of all cameras, and to relearn them when the operator adjusts PTZ cameras. Meanwhile, the practical engineering application has many requirements: the relevant image parsing algorithm should not only have good segmentation precision and classification accuracy, but also be able to process temporarily changing scenes quickly. In addition, the algorithm should have small number of parameters and can be easily applied into the data processing servers with different hardware configurations and even into the embedded surveillance equipment in the field.

Currently, there are two ways to parse a scene. The traditional way will segment the scene image into superpixels, ultrametric contour maps (UCM), or other fragmented segment regions [14,15], and then combine them into candidates of objects or local areas based on Markov random fields (MRFs), conditional random fields (CRFs), multiscale combinatorial grouping (MCG), or other rules [16–18]. These traditional methods will generate fragmented regions with precise boundaries and require time-consuming iterative calculations to form a best candidate of an object or a local area. In addition, category information of objects cannot be produced. The second way relies on deep neural networks, e.g., fully convolutional networks (FCN) [19,20], to process the feature extraction, combination, segmentation, and recognition at the same time. A FCN can achieve the segmentation and recognition in a single process. One drawback of FCNs is that the boundary line generated is usually a smooth curve, which will miss the corner of the track area. In addition, FCN has big memory footprint and needs a GPU to accelerate its large amount of computation.

In this paper, we propose an adaptive segmentation algorithm that can take advantage of both methods while avoiding their shortcomings. Like the existing traditional methods, we extract the texture distribution of the image to generate the boundary point with different weight for segmenting the image into small fragmented regions, and then the regions are combined into local areas with precise boundaries; finally, we apply a specially designed convolutional neural network (CNN) for the area's classification without the need of GPU. Our main contributions include:

- To accelerate the generation of small fragmented regions, we propose a method to find the optimal set of Gaussian kernels with adaptive directions for each specific scene. By making full use of the straight-line characters of the railway scene, a smaller number of adaptive directions are calculated according to the maximum points in Hough transformation rather than being chosen from a set of fixed angles in the traditional way. As a result, the calculation time for the boundary extraction and fragmented region generation is cut in half;
- A new clustering rule based on the boundary weight and the size of the region is set up to accelerate the combination of the regions into local areas. The number of regions is reduced in the process of weak boundary point removal by filtration, and the smallest remaining region is combined with its neighbor region, which shares the weakest boundary;
- We propose a specially designed CNN model to achieve the fast classification of local areas without the need of GPU. The local areas are divided into two categories: the track area which is used to judge the intrusion behavior and the rest area which is unrelated to the intrusion. The convolution kernels are pre-trained, and a sparsity penalty term is added into the loss function to enhance the diversity of the convolutional feature maps.

The rest of this paper is organized as follows. In Section 2, we review the related works on image parsing algorithms. Section 3 explains the proposed fast image segmentation process. Section 4 explains the proposed simplified CNN network structure and the optimization process. Section 5 presents the experimental results and discusses them. The last section summarizes our conclusions.

2. Related Work

2.1. Image Parsing by Traditional Methods

To segment an image using the traditional methods, the first step is to calculate the correlation between the adjacent pixels in the scene image, and then segment the image into fragmented regions by a certain convergence criterion [21,22]. The superpixel algorithm, for example, converts the image from the RGB color space to CIE-Lab color space to form a five-dimensional vector (brightness, color A, color B, and position x, position y), and the vector distance between two pixels representing their similarity, is used to generate the small segments patches [14,23]. A spatial pyramid descriptor fuses the gray, colored and edge gradient into one feature vector for the SVM classifier to recognize a traffic sign [24]. The image can also be converted into the YCbCr color space, and the local texture features in different channels are matched with the artificially designed template to locate the position of the traffic sign [25]. Therefore, converting the image from the RGB color space into another feature space can obtain more dimensional information channels: brightness, texture, and other feature maps besides RGB color.

To achieve the final segmentation, the fragmented regions need to be combined. The internal correlations among the adjacent regions are calculated according to different rules, and the regions are combined into local areas according to their correlation values. For example, the K-means clustering rules are used in different practical engineering applications, such as object detection for the synthetic aperture radar (SAR) image and the sea scene [26–28]. The MCG algorithm is another grouping strategy using random forests to combine the multiscale regions into highly accurate object candidates.

MCG can process one image (pixel size 90×150) in 7 s, and the mean Intersection over Union (IU) is about 80% [18,29,30]. The clustering rules influence the combination precision, which is also directly proportional to the calculation time; as a result, MCG is suitable for the initial or post processing of a fixed scene, not for real-time processing of temporarily changing scenes. Therefore, to accelerate the whole scene segmentation process, we choose to improve the traditional methods in both generation and combination of the fragmented regions while maintaining the segmentation precision.

2.2. Image Parsing by Deep Learning Methods

Deep learning methods have also been widely used in image parsing recently, e.g., various convolutional networks, which have better robustness to image translation, rotation, scaling, and distortion. Deep learning methods can be divided into three types: image classification [19], object detection [31], and pixelwise prediction [20]; and the complexity of their network structures increases from image-wise to pixel-wise.

For the pixelwise segmentation of a scene, the convolutional networks can be combined with the superpixels, the random effect model and the texture segmentation to generate the pixelwise labels [32], and also can be used as a classifier to classify the feature maps containing RGB and depth information [33–35]. FCN can even process feature extraction, combination, segmentation, and recognition at the same time, also achieving a pixelwise prediction [20].

Depending on the details of different FCN structures, the mean IU of FCN is about 80%, the accuracy is about 90%, and the quantity of the parameter is about 57 M to 134 M. The massive number of parameters and computation need a GPU with big memory to handle the operation, leading to a high cost for practical applications. Therefore, we choose to use the traditional methods to get the precise boundary of the local area first, and then use a simplified CNN only to classify the local areas without the need of GPU. However, the reduction of the network size causes low accuracy in the classification, so extra care has to be taken in optimizing the network structure and the training process.

3. Railway Scene Segmentation

As shown in Figure 2b, typical railway scene consists of different areas, including track area, sky, catenary system, green belt, and ancillary buildings. The precision of the track area boundary directly affects the reliability of the judgement about whether the intrusion occurs or not. The track area is defined as the clearance area including rails, sleepers, subgrades or high-speed railway slabs, as shown in Figure 2a. To avoid manual labeling, a fast and precise railway scene segmentation algorithm is proposed.

Figure 3 illustrates the outline of the proposed algorithm. We first calculate the feature distribution in a small image patch (pixel size 15×15) representing the central pixel of the patch, then evaluate the central pixel's probability of being a boundary point, and finally use the boundary weights to segment the image according to a fast combination rule. Unlike the traditional method, we use a smaller set of adaptive Gaussian kernels to extract the pixel color (PC) distribution and pixel similarity (PS) distribution of the image in different channels C and by different scales S. The Gaussian kernels are rotated by a set of adaptive θs, calculated from Hough transformation. The detailed procedure of boundary weight generation is described in the remainder of this section.

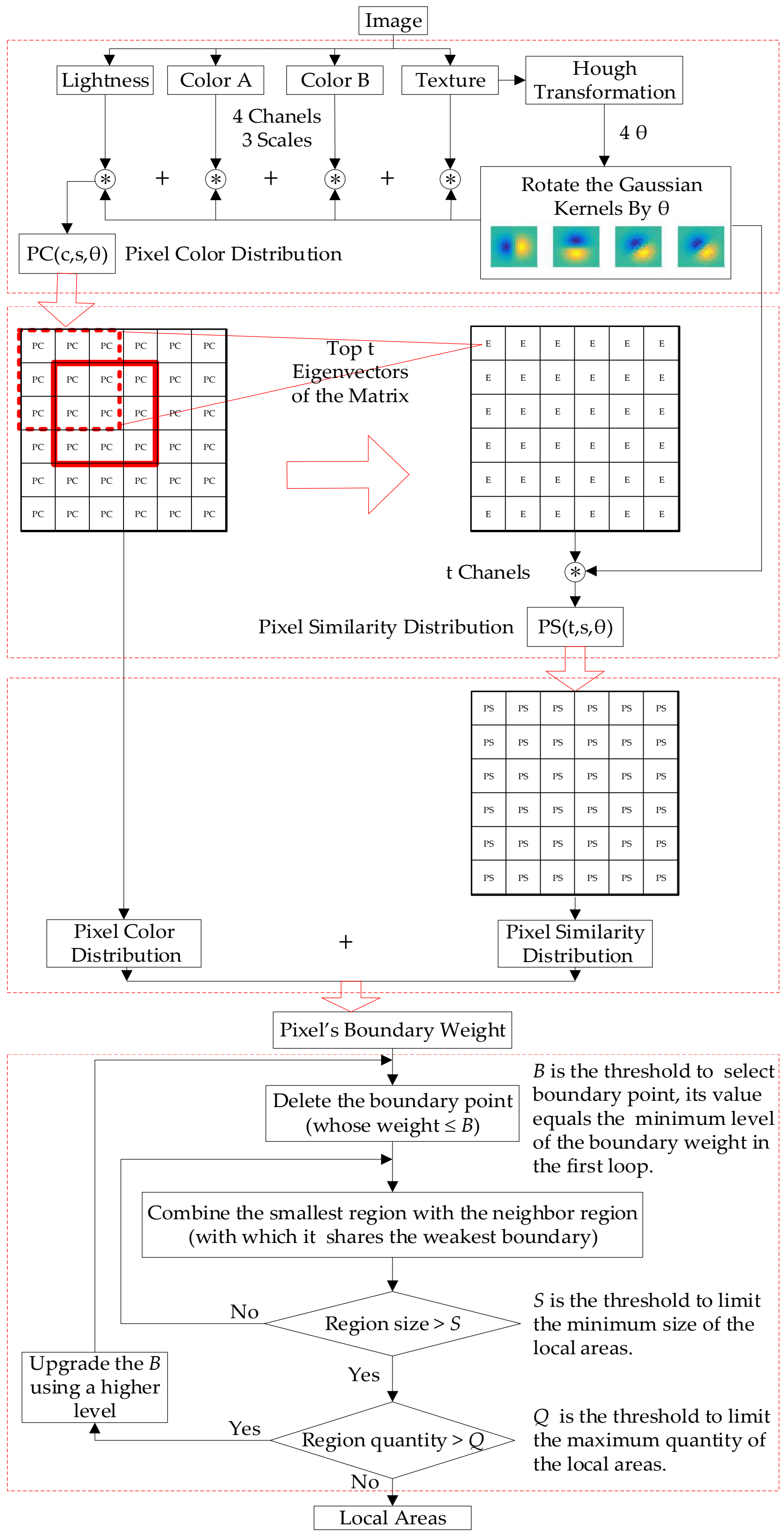


Image
Lightness
Color A
Color B
Texture
Hough Transformation
4 Chanels 3 Scales
4 θ
Rotate the Gaussian Kernels By θ
PC(c,s,θ)
Pixel Color Distribution
Top t Eigenvectors of the Matrix
t Chanels
Pixel Similarity Distribution
PS(t,s,θ)
Pixel Color Distribution
Pixel Similarity Distribution
Pixel's Boundary Weight
Delete the boundary point (whose weight ≤ B)
B is the threshold to select boundary point, its value equals the minimum level of the boundary weight in the first loop.
Combine the smallest region with the neighbor region (with which it shares the weakest boundary)
Region size > S
No
Yes
S is the threshold to limit the minimum size of the local areas.
Upgrade the B using a higher level
Region quantity > Q
Yes
No
Q is the threshold to limit the maximum quantity of the local areas.
Local Areas

Figure 3. The procedure to segment an image into fragmented regions and combine them into local areas.

3.1. Generation of Fragmented Regions

Firstly, we convert the image into the CIE lab color space, getting 3 channels: brightness, color A, and color B. Images in different channels will be scaled by $s = (0.5, 1, 2)$. In each channel, the image is convoluted with Gaussian kernels to get the color value distribution; each kernel has a special orientation angle θ. Define $G(x, y, \theta, c, s)$ as the convolution result at pixel $P(x, y)$, with angle θ, in channel c, by scale s. Then PC, the pixel's color distribution, can be obtained by

$$PC(x, y, \theta) = \sum_{s} \sum_{c} \alpha_{c,s} G(x, y, \theta, \mathrm{c}, s) \tag{1}$$

where $\alpha_{c,s}$ is a weighting coefficient.

Secondly, define $S_{imilarity}(i, j)$ as the maximum PC value of all pixels on the line $l_{i,j}$ connecting two pixels i and j in an small image patch by Equation (2), representing the similarity between pixel i and j.

$$S_{imilarity}(i, j) = \exp(-Max\left\{PC(x, y)\middle|(x, y) \in l_{i,j}\right\}) \tag{2}$$

Calculate the similarity of each pixel $i_{x,y}$ in the patch and the central pixel j_{center}, assign $S_{imilarity}(i_{x,y}, j_{center})$ to each element $MS(x, y)$ of the Matrix of Similarity MS, and assemble MS representing the similarity matrix between each pixel in the image patch and the central pixel.

Calculate the top t eigenvalues and eigenvectors of MS. Assign the eigenvector to the central pixel $P(x, y)$ marked as $e(x, y, t)$, forming a feature map E of the image, representing the similarity of the adjacent points. Again, in each dimension of the feature map, convolute $E(t)$ with Gaussian kernel of orientation θ to get the similarity distribution. Define $g(x, y, \theta, t, S)$ as the convolutional result at location $E(x, y)$, with angle θ, in dimension t, by scale s. Then, the pixel's similarity value distribution can be obtained as

$$PS(x, y, \theta) = \sum_{s} \sum_{t} \beta_{c,s} g(x, y, \theta, t, s) \tag{3}$$

where $\beta_{c,s}$ is a weighting coefficient.

Finally, $B(x, y)$, the possibility of the pixel $P(x, y)$ being a boundary point, can be estimated by

$$B(x, y) = \sum_{\theta} PC(x, y, \theta) + \sum_{\theta} PS(x, y, \theta) \tag{4}$$

3.2. Finding the Optimal Set of Gaussian Kernels

It can be found that, in the process of estimating $B(x, y)$, convolution operations (Equations (1) and (3)) using Gaussian kernels with different orientation angles θ cost most of the computation, which can be reduced if a smaller set of Gaussian kernels are used. The traditional UCM algorithms choose fixed size of $\theta = (\theta_1, \theta_2, \theta_3 ...)$ with 8 or 16 values uniformly distributed from 0 to π. Here we propose to utilize the characteristics of the railway scene to find a much smaller set of useful orientation angles and thus a smaller set of Gaussian kernels. Usually, in railway scene, there is a clear vanishing point (VP), and the boundaries of many local areas are lines passing through the VP. Therefore, if we can automatically adjust the candidate θ for each specific scene to enhance the weights of the line boundary points of the relevant areas, then we will be able to use a smaller set of θ to accelerate the process.

We propose to find the candidate θ by filtering the original image with a Canny kernel [36], and then convert the obtained texture feature into the Hough coordinate system using

$$\rho = x \cos \theta' + y \sin \theta', -\frac{\pi}{2} < \theta' < \frac{\pi}{2} \tag{5}$$

As shown in Figure 4a, each curve in the Hough coordinate system stands for one point in the Cartesian coordinate system. If the curves (colorful curve lines in Figure 4a) have one intersection point in the Hough coordinate system, then the corresponding points (blue point in Figure 4a) in the Cartesian coordinate system are collinear.

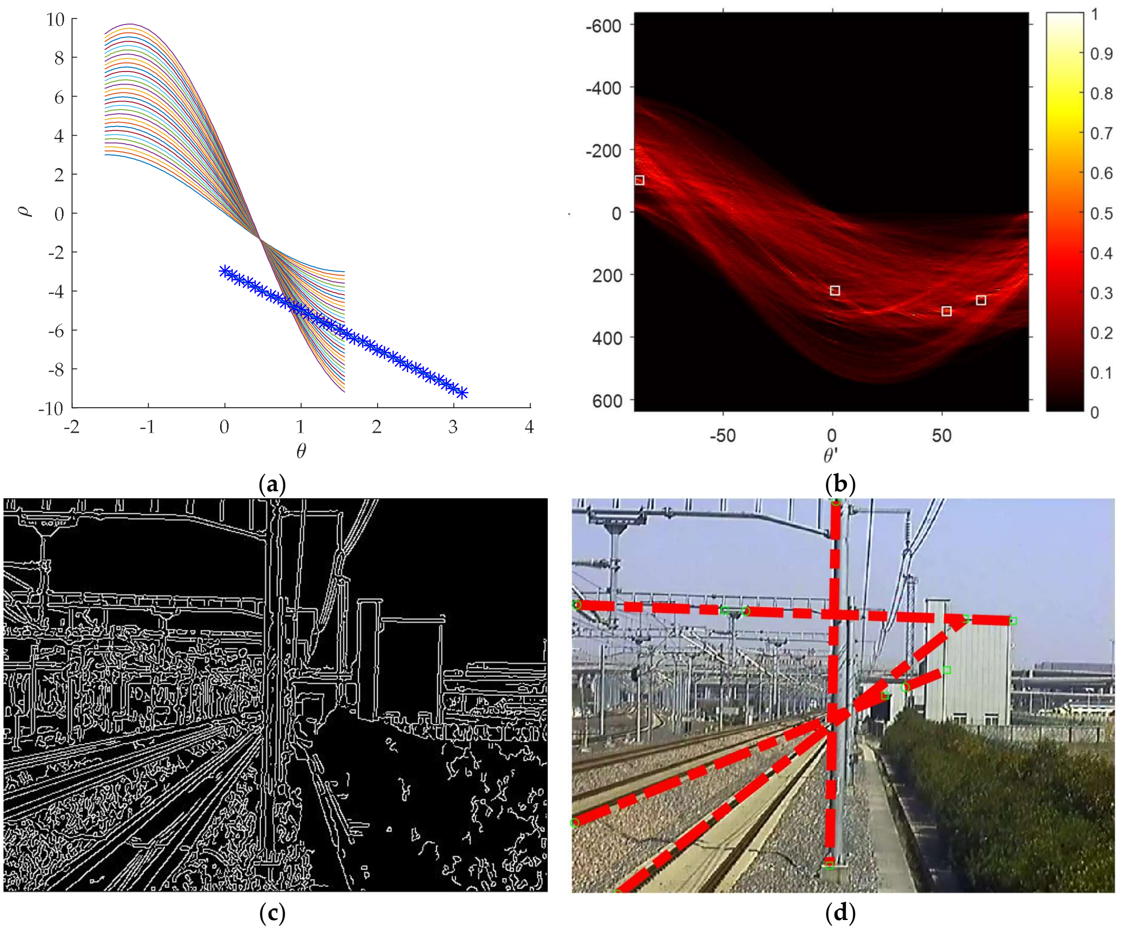


Figure 4. Using Hough transformation to detect the most significant lines in the Hough coordinate system. (**a**) The intersection point of a group of curves in the Hough coordinate system means there are a group of collinear points in the Cartesian coordinate system. (**b**) The more the curves intersect in the Hough coordinate system, the lighter the intersection point is, meaning that there are more collinear points along this line in the Cartesian coordinate system. (**c**) The texture feature maps filtered by canny filter. (**d**) The top four significant lines.

Let $H(\theta', \rho)$ be the number of curves intersecting at point (θ', ρ) and find the point with maximum $H(\theta', \rho)$, where there are the largest number of points which are collinear on the corresponding line in the Cartesian coordinate system. The line can be expressed as

$$y = -\frac{1}{\tan\theta'}x + \frac{\rho}{\sin\theta'} = kx + b \tag{6}$$

To find a small set of four orientation angles, one can take the top four maximum θ in $H(\theta', \rho)$, e.g., the points with highest 'lightness' in Figure 4b: $\theta' = 68°, 52°, 0°$ and $-88°$. Here we change the $\theta = 90° - \theta' = 22°, 38°, 90°$, and $178°$ in order to obtain a range of values from $0°$ to $180°$ (0–π). Based on the selected set of orientation angles, the Gaussian kernels can be constructed correspondingly by rotating the Gaussian. As shown in Figure 5, in the Cartesian coordinate system X-O-Y, point $P(x,y)$ rotates around the point $o(\frac{W}{2},\frac{W}{2})$ an angle θ to $P'(x',y')$, which can be formulated as

$$\begin{aligned}\begin{bmatrix} x & y & 1 \end{bmatrix} &= \begin{bmatrix} x' & y' & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ -0.5W & 0.5W & 1 \end{bmatrix}\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0.5W & 0.5W & 1 \end{bmatrix} \\ &= \begin{bmatrix} x' & y' & 1 \end{bmatrix}\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ -0.5W(\cos\theta-\sin\theta-1) & -0.5W(\sin\theta+\cos\theta-1) & 1 \end{bmatrix}\end{aligned} \tag{7}$$

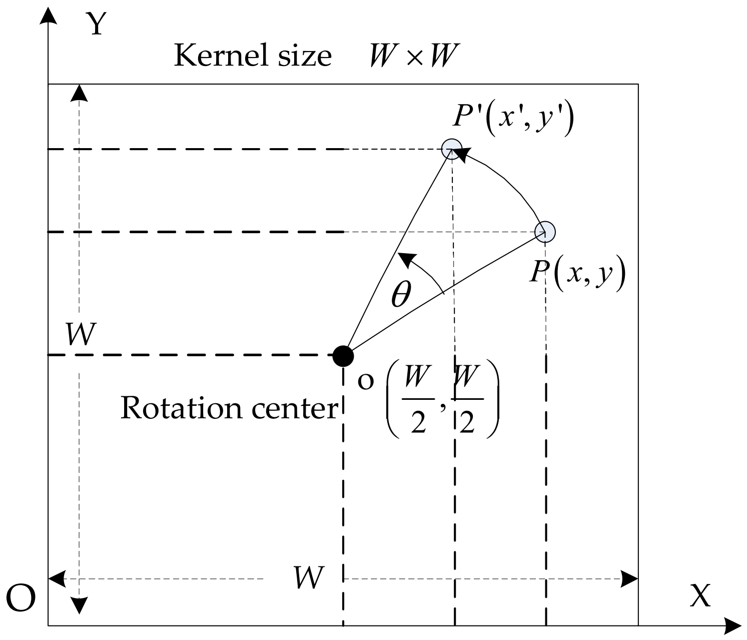


Figure 5. Calculating the rotation matrix of the Gaussian kernel. The rotation center is on the kernel center.

Figure 6 shows several Gaussian kernels rotated by the optimal set of θ =22°, 38°, 90°, and 178° obtained above and $\theta = 112.5°$, one of the eight uniformly-distributed values commonly used in traditional UCM algorithms, respectively. The results show that the features of the horizontal catenary bracket, the vertical catenary column and the declining track are strengthened obviously in the first four filters, contrasting with the feature extraction equality in the fifth filter. The universality of using 8 or 16 uniform values in different angle θ causes a redundant calculation when applied to the railway scene. Therefore, adjusting a smaller number of θ adaptively to filter the feature map can accelerate the boundary weighting to generate the fragmented regions.

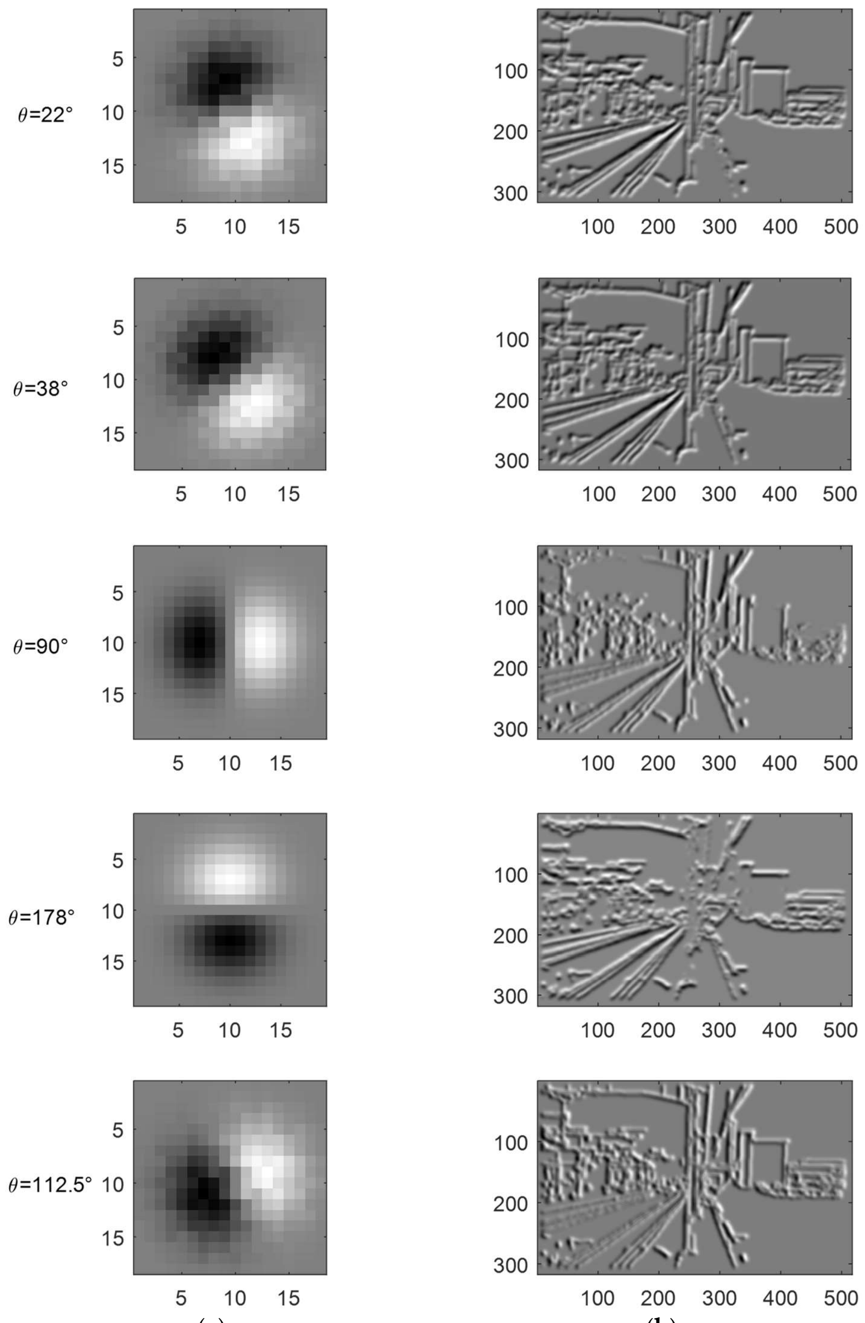


Figure 6. Different kernels and the convolution results of CIE-lab color L channel. (**a**) First order derivative Gaussian kernels rotated by five angles. (**b**) Results of the Gaussian convolution.

3.3. Combination Rule

The fragmented regions generated by the adaptive boundary detection are shown in Figure 7a. The higher the boundary weight is, the brighter the point is shown in the gray feature map, indicating that the point is more likely to become a boundary point.

A clustering rule based on both of the boundary weight and the region size is proposed to combine the fragmented regions into local areas. The number of the regions will be reduced in the process of weak boundary point removal by filtration. The smallest remaining region will be combined with its neighbor region, with which it shares the weakest boundary. Repeat this iteration until the statistical parameters meet the requirements. The process is as follows:

1. Let $B(m)$ be the normalized value of the boundary point's weight $B(x_m, y_m)$, where $m = 1, 2, 3 \ldots M$, and $M$is the total number of boundary points:

$$B(m) = sigmoid(B(x_m, y_m)) = \frac{1}{1 + e^{-B(x_m, y_m)}} \tag{8}$$

2. The statistical distribution of the boundary point weight $B(i)$ is shown in Figure 7b. There are many levels of boundary point weights. Choose the minimum level B as the threshold to delete the weak boundary points $B(m) \leq B$;
3. The fragmented regions will be reduced by reconnecting the breakpoints of the boundary line using expansion and corrosion operations, as shown in Figure 7c. The new regions are shown in Figure 7d;
4. The statistical distribution of region size $f(n)$ is shown in Figure 7e, where $n = 1, 2, 3 \ldots N$, and N is the serial number of the regions. Choose the smallest region along its boundary line and find the neighbor region which shares the weakest boundary with it. Then combine them into a new region. As shown in Figure 7d, regions in number 1, 2, 3, and 4 are combined as one new region in Figure 7f;
5. Repeat Step 4 to reduce N until the area of the smallest region is larger than a threshold S, which is used to limit the minimum area of the remained regions;
6. Compare the final N with another threshold Q to limit the minimum quantity of the remained regions. If $N > Q$, select the second minimum level weight B and go back to Step 2;

Figure 7g is the original railway scene image, and the Figure 7h is the result of our segmentation algorithm. The railway scene only contains five categories of areas, and the shape of the area is usually in a large and radial pattern. Therefore, we set the minimum area threshold S to 10% of the whole image and the maximum quantity threshold Q to 10, which will prevent the remained regions from being too fragmented. The remained regions will be adjusted into a standard size of 64×64 and RGB 3 channels, after being classified by the CNN in Section 4, the remaining regions with the same labels will be combined as one local area.

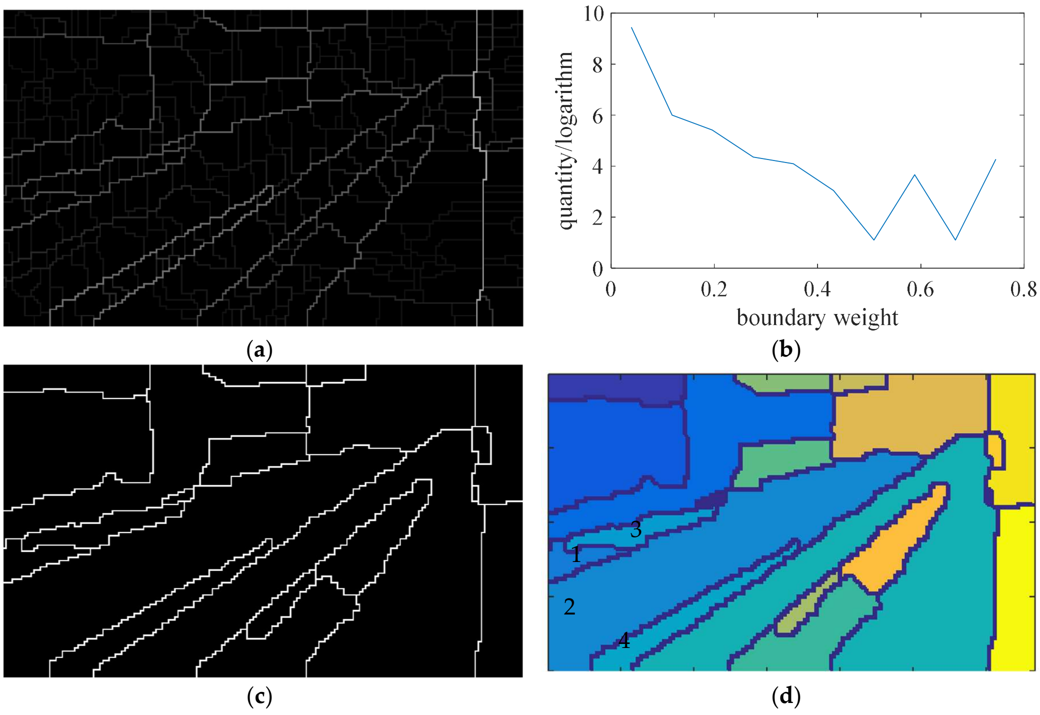
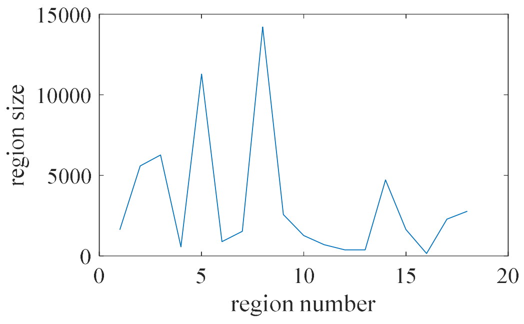
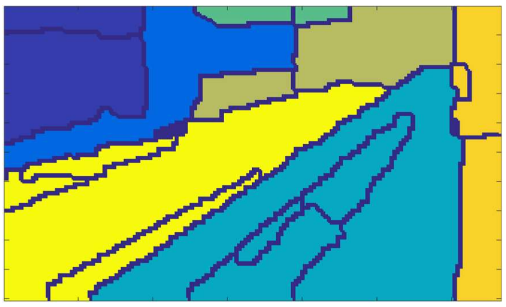


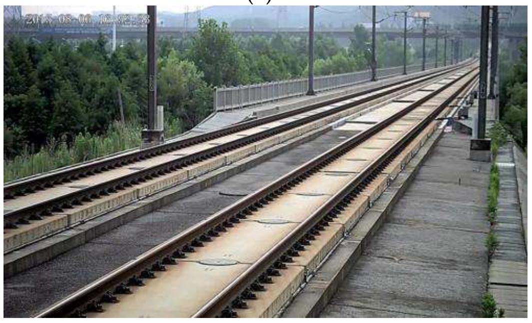
Figure 7. *Cont.*



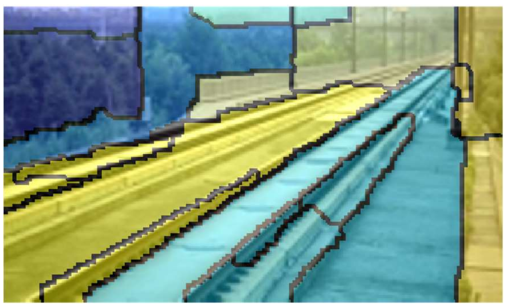
(**e**)



(**f**)



(**g**)



(**h**)

Figure 7. The procedures of combining the fragmented regions into local areas. According to the adjustment and experiments, for the railway scene, the scene image is set to a pixel size of 90 × 150, the number of adaptive θ is reduced to 4, the number of reserved areas Q is set to 10, and the smallest fragmented area S is set to 10% of the total size of the image. (**a**) Boundary with weight. (**b**) Distribution of boundary weight and quantity. (**c**) Delete the weak boundary. (**d**) Fragmented regions. (**e**) Distribution of the region size and serial number. (**f**) Local areas after the fragmented regions are combined. (**g**) The original railway scene image, (**h**) is the result.

4. Local Area Recognition in Railway Scene

To automatically label the local areas in real time without the help a GPU, we design a simplified CNN with less layers and kernels. To compensate the reduced accuracy, the convolution kernels are pre-trained, and a sparsity penalty term is added into the loss function to enhance the diversity of the feature maps.

4.1. Structure of Simplified CNN

Before designing and applying a simplified CNN, we first construct a dataset of local area images for training it. As shown in Figure 8, there are mainly five basic categories of elements in a typical railway scene, including track area, sky, catenary system, green belt, and ancillary buildings. To sample the dataset, five solid line rectangles are manually defined to cover the five different areas. We program a simple extraction code to take the image patches using the dotted-line box as samples with the same category of the outer rectangle. We set up a group of constraint parameters to control the dotted box to extract the patches at a random position, by a random scale, maintaining inside of each rectangle. The image patches are adjusted into a pixel size of 64 × 64 and RGB 3 channels to assemble our five-category datasets of railway local area. However, for the specific application of this paper, our target is focused on the track area for judging intrusion behavior, so besides the 'track' label, we merge the other four elements into one category labeled as 'others'. There are 9000 image patches in total, in which 5000 images are used for training our net, 2000 images are used for cross-validation, and 2000 images are used for testing.

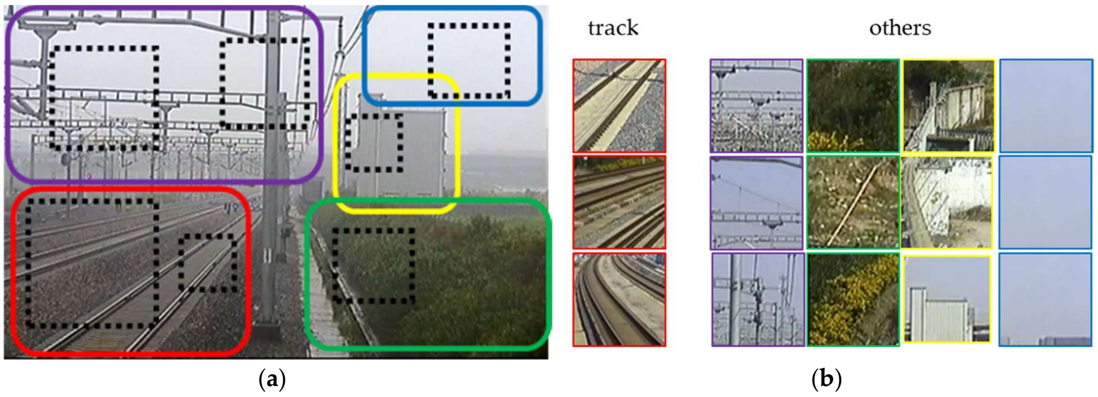


Figure 8. Collecting samples of local areas for CNN training. (**a**) Solid-line rectangles are delineated by manual with labels, including the track area (red), sky (blue), catenary system (purple), green belt (green), ancillary buildings (yellow). The dotted-line boxes are extractor windows. (**b**) The dataset containing two categories for training the CNN.

A simplified CNN structure is designed for fast recognition, which consists of an input layer, two convolution layers *C1* and *C2*, two mean pooling layers *S1* and *S2*, and a logistic classification layer, as shown in Figure 9.

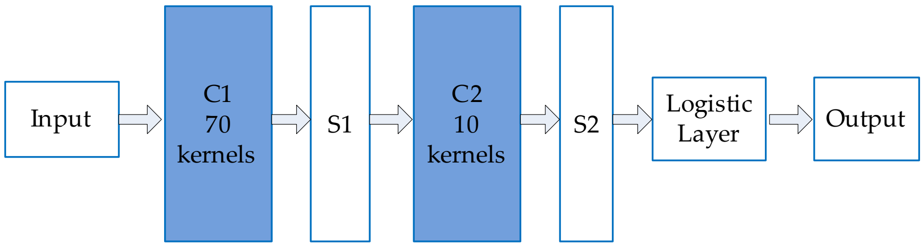


Figure 9. Structure of the simplified CNN. The size of the input image is a pixel size of 64 × 64 with RGB 3 channels. The output is one of the two category labels.

As shown in Table 1, we conducted five experiments with different kernel quantities and sizes. It can be seen that increasing the kernel size and quantity may increase the accuracy, but the accuracy is still less than 80%. Although the railway scene is very simple, only containing several typical area categories, the shapes, color, and texture features of the area belonging to the same category are still very complex and different. Therefore, the training process must be optimized to increase the accuracy.

Table 1. Experimental results of different CNN network structures.

Kernel Size	**Kernel Quantity**		**Calculation Time (s)**	**Accuracy**
	C1	**C2**		
3 × 3	50	10	0.00372	72.25%
	70	10	0.00495	73%
	100	10	0.00689	75%
5 × 5	100	10	0.0125	76%
7 × 7	100	10	0.0217	76.5%

4.2. *Optimization of the Simplified CNN*

To increase the accuracy, kernels are pre-trained to extract better low-level features. The pre-training strategy is based on autoencoder-decoder network; and the $W^1_{i,3\times3\times3}$ after training in first layer is applied as the convolution kernel in the first convolution layer *C1*, as shown in Figure 10 for the case with kernel size of 3×3 and in RGB 3 channels. During the training, 3×3patches in RGB 3 channels are randomly selected from random railway scene images, as shown in Figure 11a. The result of the pre-trained kernels is shown in Figure 11b, where the patches and the kernels are all in RGB 3 channels.

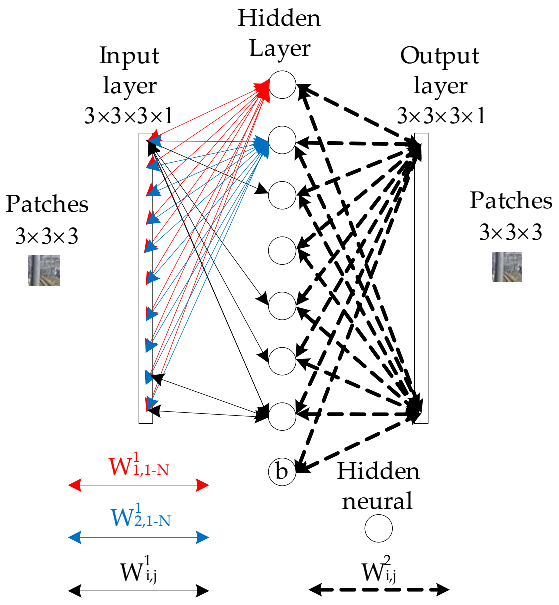


Figure 10. Structure of the autoencoder-decoder network. The hidden layer contains 70 hidden neurons; *W* denotes the weight associated with the connection between neurons; and the network is trained to produce output the same as its input.

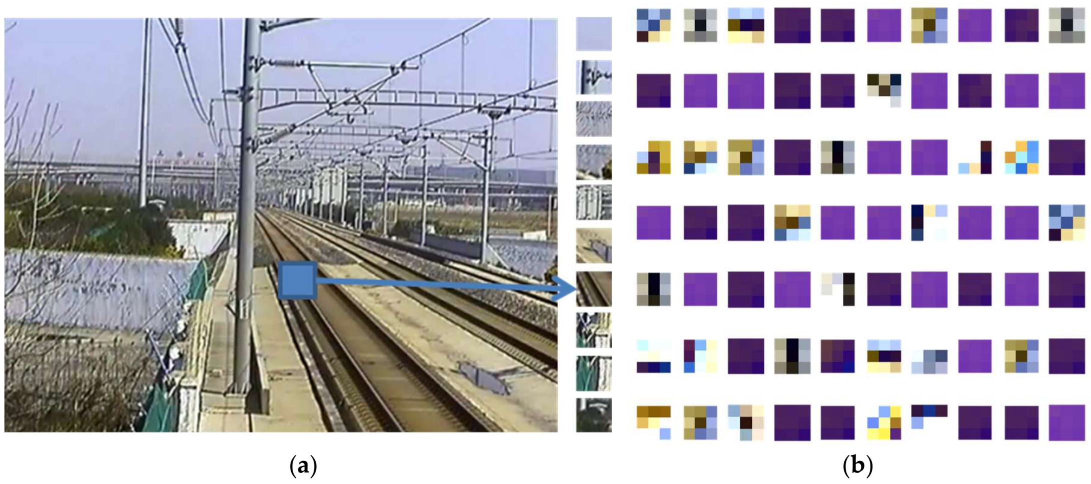


Figure 11. Pre-trained convolution kernels using the autoencoder-decoder algorithm. (**a**) The image patches are extracted from the left railway scene image for the kernel training. (**b**) The pre-trained kernels used in convolution layer *C1*.

After pre-training, the input weights of each neuron in the hidden layer are used as the initial weights of kernels in the first convolution layer *C1* in Figure 9. The rest of CNN in Figure 9 are randomly initialized and then trained by using a backpropagation algorithm (stochastic gradient descent, SGD). To enhance the diversity of the feature maps, a sparsity penalty term is added into the loss function *J* as

$$J = \left\{ \frac{1}{P} \sum_{p=1}^{P} \frac{1}{2} \left[h\left(e_p\right) - l_p \right]^2 \right\} + \tau \sum_{f=1}^{10} \left[\chi \lg \frac{\chi}{\eta_f} + (1-\chi) lg \frac{1-\chi}{1-\eta_f} \right] \tag{9}$$

where

$$\eta_f = \frac{1}{P}\sum_{p=1}^{P}\sum_{u=1}^{29}\sum_{v=1}^{29} O^{(2)}_{f,e_p}(u,v) \tag{10}$$

e_p is the p-th input image, l_p is the ground truth label, there are totally P images in the dataset, $h(\mathrm{e}_p)$ is the output label, τ is the weight of the sparsity penalty term, χ is the sparsity parameter (a smaller value close to 0, e.g., 0.05), η_f is the average output of the f-th feature map in convolution layer *C2* (averaged over the training dataset), and $O^{(2)}_{f,e_p}(u,v)$ is the value at position (u,v) in the f-th feature map of the input e_p in the second convolutional layer *C2*, the size of the feature map is 29 × 29 pixels.

In the process of backpropagation, the sparsity penalty item will suppress the average output of all feature maps in the second convolutional layer *C2*, but enforce the output of one feature map at the same time, so as to enhance the diversity of the feature maps and improve the accuracy. The learning rate is set to 0.1, and the decay of the learning rate is 0.001 after each iteration, the final value of J should be less than 0.05.

4.3. Performance of the Simplified CNN

As shown in Table 2, the accuracies of the simplified CNNs with different structures are all increased by using the proposed optimization method, compared with the results of traditional training method shown in Table 1, e.g., the simplified CNN with 70 kernels (3 × 3, 3 channels) in *C1* and 10 kernels (3 × 3, 70 channels) in *C2* is used for the proposed segmentation algorithm. The quantity of the network parameters is only 0.02912M. After the railway scene is segmented and classified, the regions with track labels can be combined together as the final track areas.

Table 2. Experiment results of different CNN network structures after the optimization.

Kernel Size	Kernel Quantity		Accuracy
	C1	*C2*	
3 × 3	50	10	98%
	70	10	98.5
	100	10	98.5%
5 × 5	100	10	98.75%
7 × 7	100	10	99.25%

5. Experiments and Results

5.1. Railway Scene Dataset

We collect images from 16 PTZ cameras at straight lines, curves and bridges in the high-speed railway from Shanghai to Hangzhou, China. For each camera, images are collected from 10 different shooting angles, lenses, and under different illumination conditions from 8:00 a.m. to 5:00 p.m. Examples are shown in Figure 12a. There are totally 1760 scene images in the dataset, in which 1000 images are used in the training dataset, 400 images are used in the cross-validation dataset and 360 images are used in the test dataset. These datasets are used to generate the datasets for our simplified CNN (Section 4.1) and the dataset for training the FCN for the comparison experiments.

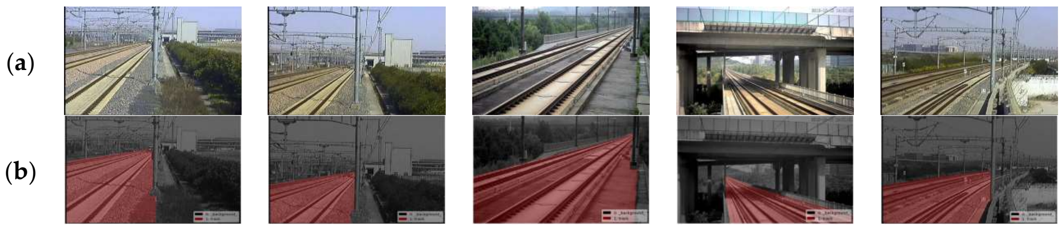
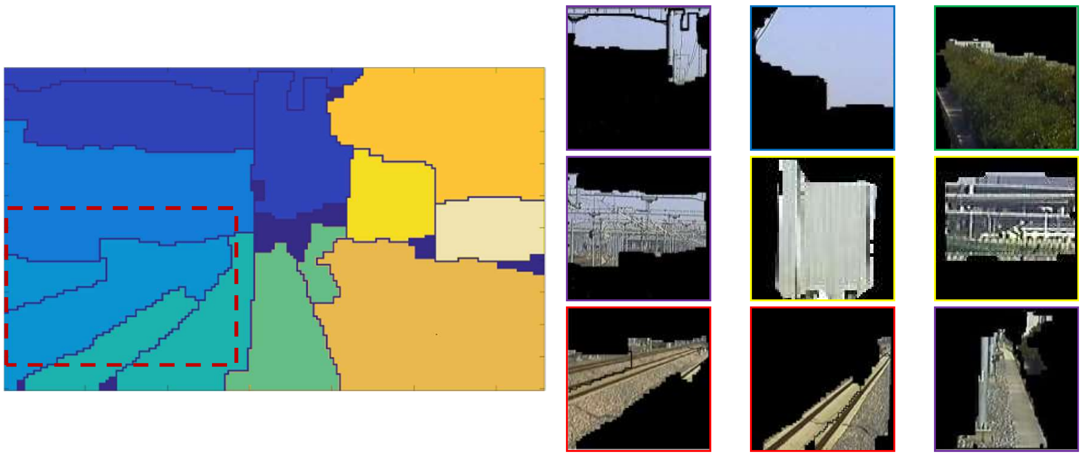


Figure 12. Samples in railway dataset. (**a**) Images from PTZ cameras under different conditions. (**b**) Ground truth of the track area.

5.2. Modification of the Workflow for the Case of Small Track Portion

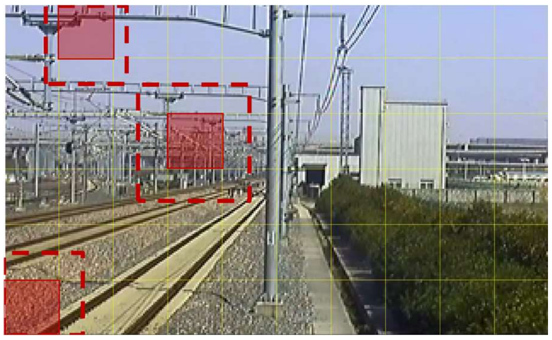
For cameras on line sections, track area only takes up a small portion of the scene image, while for the ones at tunnel entrances and bridges over railway line, track area usually takes up most of the scene. As shown in Figure 12b, the red track area takes about 25–70% of the whole scene image for different cameras. That means, the complete-processing workflow (Sections 3 and 4) would waste a lot of time calculating the boundaries between the 'others' areas (Figure 13b) rather than focusing on the potential track area as the red dotted line rectangle shown in Figure 13a. In order to find the potential track area and reduce the segmentation calculation furthermore, we design a partial-scanning workflow to locate the potential position of the track area before the segmentation and classification by scanning over the railway scene roughly using the proposed CNN. As shown in Figure 13c, we firstly divide the railway scene image into 6×10 cells (yellow cell); each cell and its peripheral zone (red dotted line rectangle) are resized to 64×64 pixels, define their classified labels as the representation of its central cell (red area in Figure 13c); the proposed CNN is used to classify these cells and the output labels are used to identify the potential track area roughly as the red area shown in Figure 13d; A minimum enclosing dotted line rectangle is used to adjust the potential track area into a regular shape as shown in Figure 13d.

The strategy of the partial-scanning workflow reduces the segmentation area, but spends extra scanning time. Thus, the overall processing time depends on the proportion of the track area to the railway scene, as shown in Table 3, the numbers on the left are the scene images in Figure 12, from the left to the right. If the track area takes over more than 88.1% of the railway scene, the performance of the partial-scanning workflow would be worse than the complete-processing workflow. These two workflows can be chosen for different cameras: for those with short focus lens and focus on the near scene full of track area, the complete-processing workflow should be used; for the ones with long focus lens and track area only take a small part of the scene, the partial-scanning workflow should be used.



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

Figure 13. *Cont.*


(**c**)

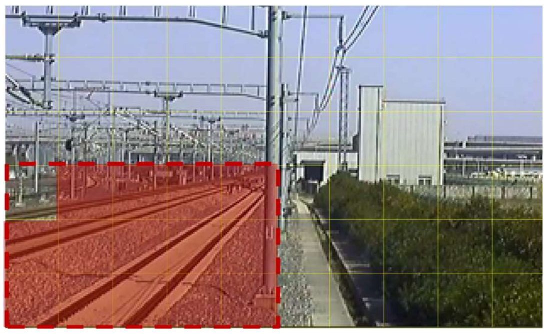

(**d**)

Figure 13. Rough scanning over the scene to find the potential track area. (**a**) Segmentation result of the whole railway scene images. (**b**) Different local areas. (**c**) Scanning the railway scene image roughly using proposed CNN. (**d**) Area in red dotted rectangle is the potential track area, which will reduce segmentation calculation by three-quarters.

Table 3. Calculation time of the comparison experiments with different workflow to segment the railway scene.

	Partial-Scanning Workflow				**Complete-Processing Workflow**
	Scan Time (s)	**Proportion of Track Area**	**Segmentation and Classification Time (s)**	**Total (s)**	**Time (s)**
1	0.297	41.7%	1.042	1.339	2.5
2	0.297	25%	0.625	0.922	2.5
3	0.297	75%	1.875	2.172	2.5
4	0.297	40%	1	1.927	2.5
5	0.297	30%	0.75	1.047	2.5

5.3. Metrics

To evaluate the segmentation performance, three criteria are used. The first one is the intersection over union (IU) generally defined as Equation (11), where L represents for the ground truth, R represents the segmentation result; the second one is the pixel accuracy (PA) defined in Equation (12) to evaluate the portion of the area which need to be surveilled are segmented; and the extra pixel (EP) as Equation (13) is used to evaluate the portion of segmented areas which do not need to be surveilled. PA would influence the missing part of the track area which would cause a missing alarm, and the EP would influence the extra part of the track area which would cause a false alarm.

$$IU = \frac{L \cap R}{L \cup R} \tag{11}$$

$$PA = \frac{L \cap R}{L} \tag{12}$$

$$EP = \frac{R - L \cap R}{L} \tag{13}$$

5.4. Performance of the Proposed Segmentation Algorithm

The proposed algorithm is compared with MCG and FCN using images from railway dataset and some examples are shown in Figure 14. In the experiment, the computation platform is equipped with an Intel i5-6500 CPU, 8 GB DDR3 memory, without GPU and MATLAB 2012, and images in the dataset are resized to 90×150. The MCG method is the pre-trained demo from [17]. The FCN network uses a standard VGG16 structure trained by VOC2012 dataset for the feature extracting, and upsampled the outputs of the third, fourth, and seventh convolution layers.

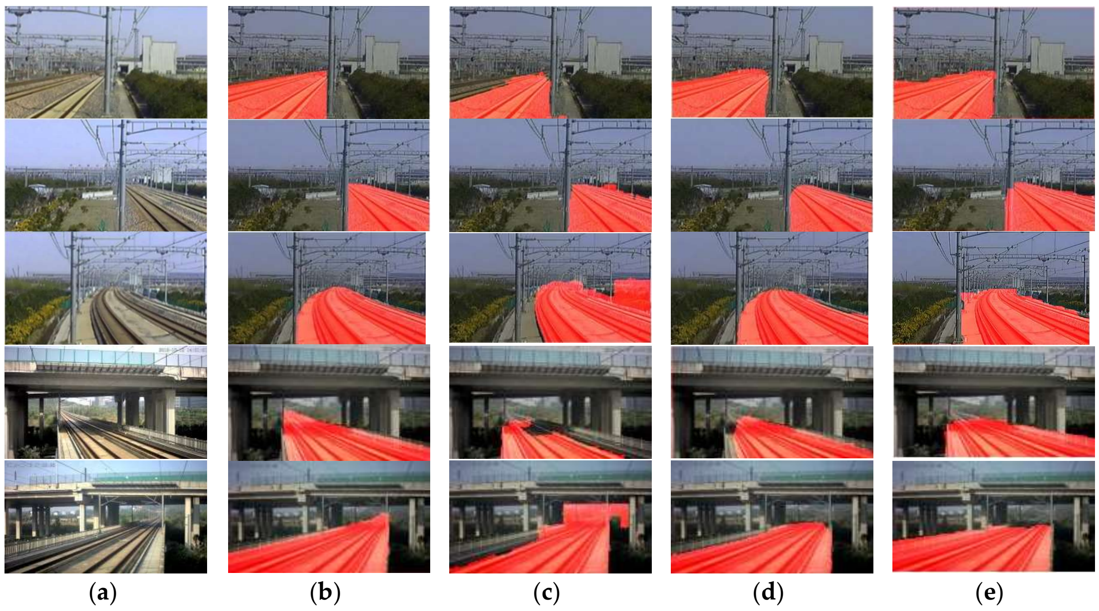


Figure 14. Using different algorithm to detect the track area. (**a**) The original railway scenes. (**b**) Ground truth of track areas. (**c**) Results of the MCG algorithm. (**d**) Results of the FCN algorithm. (**e**) Results of our algorithm.

The missing part and the extra part of the segmented track area are shown in Figure 15. For the MCG algorithm, it used the CRFs to combine the fragmented regions into one unified area based on the texture which caused the missing part (as shown in Figure 15e) because of the difference texture between the nearby track and the distant track. The performances of the FCN algorithms were improved slightly from their original results in [19] because of the monotonous railway scene and the small amount of categories; but not too significantly because the shape and color textures of the scene images sampled with different illuminations, weather, and in different seasons were still complex. As shown in Figure 15f,i, the smooth boundary line of the FCN algorithm was not suitable for our railway scene parsing because of the concave and convex shapes at the straight and sharp edge of the region, especially near the area with an acute angle and straight line. Concave and convex shapes caused both a missing part and an extra part of the track area when compared with the ground truth, which would release both the missing alarms and false alarms. For the engineering application, our system would rather release a false alarm than miss a true alarm.

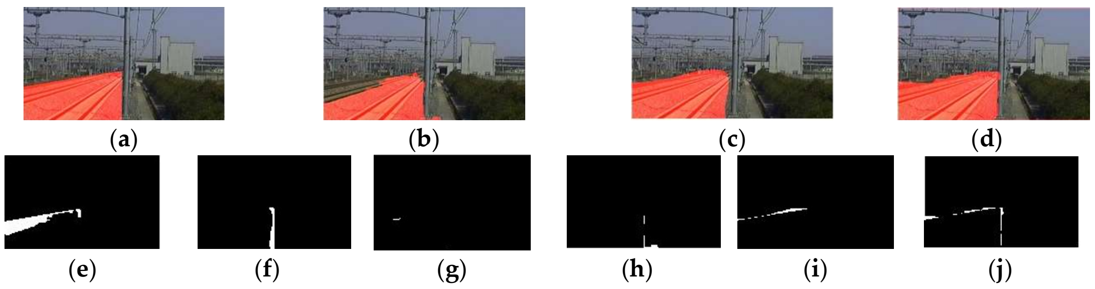


Figure 15. Missing and extra areas of different methods comparing with the ground truth. (**a**) Manual label of track areas. (**b**) Results of the MCG. (**c**) Results of the FCN. (**d**) Results of our method. (**e**) Missing part of MCG. (**f**) Missing part of FCN. (**g**) Missing part of our method. (**h**) Extra part of MCG. (**i**) Extra part of FCN. (**j**) Extra part of our method.

The performances of the three algorithms are shown in Table 4. It can be found that the proposed algorithm with four optimal Gaussian kernels achieves the highest score in PA, which means that the greatest portion of the surveillance area is found out and thus is preferred for applications.

Table 4. Experimental results of different algorithms.

Algorithm		Mean IU	Mean PA	Mean EP	Time (s)
MCG		72.05%	79.94%	10.63%	7
FCN		89.83%	91.26%	16.20%	41
Our Algorithm	Four optimal Gaussian kernels	81.94%	95.90%	18.17%	0.9–2.8
	Eight regular Gaussian kernels	85.23%	93.85%	17.56%	1.1–4.4

6. Conclusions

The proposed algorithm uses an adaptive feature distribution extractor for railway track segmentation by making full use of the strong linear characteristics of railway scenes and the typical categories of the local areas. A good balance between segmentation precision, recognition accuracy, calculation time, and complexity of manual operation can be achieved. By using the proposed algorithm, the railway intrusion detection system can automatically and accurately delimit the boundaries of a surveillance scene in real time and greatly improve the efficiency of the system operation. Considering the fact that, in China, there are over 29,000 km of high-speed railways and the average density of cameras on high-speed railway lines is about 2.92 cameras/km, the proposed algorithm is of great significance to improve the efficiency.

The proposed algorithm can be applied into the surveillance system of public places such as airport aprons, highway pavement, and squares. These places share some common characteristics: simple structure full of straight lines—such as airplane runways and different functional areas, vehicles and different lanes, pedestrians and sidewalk lines. Before applying this method, however, the training dataset of the simplified CNN has to include new categories in such scenes, then the proposed algorithm can segment the scene and label each local area.

Author Contributions: Conceptualization, Y.W., L.Z., and Z.Y.; Investigation, Y.W. and B.G.; Methodology, Y.W. and L.Z.; Project administration, Z.Y.; Software, Y.W.; Validation, Y.W.; Writing—original draft, Y.W.; Writing—review & editing, Y.W. and L.Z.

Funding: This research was funded by National Key Research and Development Program of China (2016YFB1200401).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, Y.; Yu, Z.; Zhu, L.; Guo, B. Fast feature extraction algorithm for high-speed railway clearance intruding objects based on CNN. *J. Sci. Instrum.* **2017**, *38*, 1267–1275.
2. Hou, G.; Yu, Z. Research on flexible protection technology of high speed railways. *J. Railw. Stand. Des.* **2006**, *11*, 16–18.
3. Cao, D.; Fang, H.; Wang, F.; Zhu, H.; Sun, M. A fiber bragg-grating-based miniature sensor for the fast detection of soil moisture profiles in highway slopes and subgrades. *Sensors* **2018**, *18*, 4431. [CrossRef] [PubMed]
4. Zhang, Y. Application study of fiber bragg-grating technology in disaster prevention of high-speed railway. *J. Railw. Signal. Commun.* **2009**, *45*, 48–50.
5. Oh, S.; Kim, G.; Lee, H. A monitoring system with ubiquitous sensors for passenger safety in railway platform. In Proceedings of the 7th International Conference on Power Electronics, Daegu, Korea, 22–26 October 2007; pp. 289–294.

6. Wang, Y.; Shi, H.; Zhu, L.; Guo, B. Research of surveillance system for intruding the existing railway lines clearance during Beijing-Shanghai high speed railway construction. In Proceedings of the 3rd International Symposium on Test Automation and Instrumentation, Xiamen, China, 22–25 May 2010; pp. 218–223.
7. Luy, M.; Cam, E.; Ulamis, F.; Uzun, I.; Akin, S.I. Initial results of testing a multilayer laser scanner in a collision avoidance system for light rail vehicles. *Appl. Sci.* **2018**, *8*, 475. [CrossRef]
8. Guo, B.; Yu, Z.; Zhang, N.; Zhu, L.; Gao, C. 3D point cloud segmentation, classification and recognition algorithm of railway scene. *Chin. J. Sci. Instrum.* **2017**, *38*, 2103–2111.
9. Zhan, D.; Jing, D.; Wu, M.; Zhang, D.; Yu, L.; Chen, T. An accurate and efficient vision measurement approach for railway catenary geometry parameters. *IEEE Trans. Instrum. Meas.* **2018**, *67*, 2841–2853. [CrossRef]
10. Guo, B.; Zhu, L.; Shi, H. Intrusion detection algorithm for railway clearance with rapid DBSCAN clustering. *J. Sci. Instrum.* **2012**, *33*, 241–247.
11. Guo, B.; Yang, L.; Shi, H.; Wang, Y.; Xu, X. High-speed railway clearance intrusion detection algorithm with fast background subtraction. *J. Sci. Instrum.* **2016**, *37*, 1371–1378.
12. Shi, H.; Chai, H.; Wang, Y. Study on railway embedded detection algorithm for railway intrusion based on object recognition and tracking. *J. China Railw. Soc.* **2015**, *37*, 58–65.
13. Vazquez, J.; Mazo, M.; Lazaro, J.L.; Luna, C.A.; Urena, J.; Garcia, J.J.; Hierrezuelo, L. Detection of moving objects in railway using vision. In Proceedings of the IEEE Intelligent Vehicles Symposium, Parma, Italy, 14–17 June 2004; pp. 872–875.
14. Achanta, R.; Shaji, A.; Smith, K.; Lucchi, A.; Fua, P.; Susstrunck, S. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 2274–2282. [CrossRef] [PubMed]
15. Arbeláez, P. Boundary extraction in natural images using ultrametric contour maps. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop, New York, NY, USA, 17–22 June 2006; p. 182.
16. Verbeek, J.; Triggs, B. Region classification with Markov field aspect models. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, USA, 17–22 June 2007; pp. 1–8.
17. Ladický, L.; Russell, C.; Kohli, P.; Torr, P. Associative hierarchical CRFs for object class image segmentation. In Proceedings of the 12th International Conference on Computer Vision, Kyoto, Japan, 29 September–2 October 2009; pp. 739–746.
18. Arbeláez, P.; Pont-Tuset, J.; Barron, J.; Marques, F.; Malik, J. Multiscale combinatorial grouping. In Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 328–335.
19. LeCun, Y.; Boser, B.; Denker, J.; Henderson, D.; Howard, R.E.; Hubbard, W.; Jackel, L. Handwritten digit recognition with a back-propagation network. In *Neural Information Processing Systems*; Morgan-Kaufmann: San Francisco, CA, USA, 1990; pp. 396–404.
20. Shelhamer, E.; Long, J.; Darrell, T. Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *39*, 640–651. [CrossRef] [PubMed]
21. Ren, X.; Malik, J. Learning a classification model for segmentation. In Proceedings of the 9th International Conference on Computer Vision, Nice, France, 13–16 October 2003; Volume 1, pp. 10–17.
22. Shi, J.; Malik, J. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 888–905.
23. Zhu, Y.; Luo, K.; Ma, C.; Liu, Q.; Jin, B. Superpixel segmentation based synthetic classifications with clear boundary information for a legged robot. *Sensors* **2018**, *18*, 2808. [CrossRef] [PubMed]
24. Liu, Y.; Chen, Y.; Zhang, S. Traffic sign recognition based on pyramid histogram fusion descriptor and HIK-SVM. *J. Transp. Syst. Eng. Inf. Technol.* **2017**, *17*, 220–226.
25. Fang, Z.; Duan, J.; Zheng, B. Traffic signs recognition and tracking based on feature color and SNCC algorithm. *J. Transp. Syst. Eng. Inf. Technol.* **2014**, *14*, 47–52.
26. Liu, K.; Ying, Z.; Cui, Y. SAR image target recognition based on unsupervised K-means feature and data augmentation. *J. Signal Process.* **2017**, *33*, 452–458.
27. Zhang, X.; Fan, J.; Xu, J.; Shi, X. Image super-resolution algorithm via K-means clustering and support vector data description. *J. Image Graph.* **2016**, *21*, 135–144.
28. Ma, G.; Tian, Y.; Li, X. Application of K-means clustering algorithm in color image segmentation of grouper in seawater background. *J. Comput. Appl. Softw.* **2016**, *33*, 192–195.

29. Arbeláez, P.; Maire, M.; Fowlkes, C.; Malik, J. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *33*, 898–916. [CrossRef] [PubMed]
30. Pont-Tuset, J.; Arbeláez, P.; Barron, J.; Marques, F.; Malik, J. Multiscale combinatorial grouping for image segmentation and object proposal generation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *39*, 128–140. [CrossRef] [PubMed]
31. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.; Berg, A. SSD: Single shot multibox detector. In Proceedings of the 14th European Conference on Computer Vision (ECCV 2016), Amsterdam, The Netherlands, 8–16 October 2016; Volume 9905, pp. 21–37.
32. Farabet, C.; Couprie, C.; Najman, L.; LeCun, Y. Learning hierarchical features for scene labeling. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1915–1929. [CrossRef] [PubMed]
33. Couprie, C.; Farabet, C.; Najman, L.; LeCun, Y. Indoor semantic segmentation using depth information. *arXiv*, 2013; arXiv:1301.3572.
34. Gupta, S.; Girshick, R.; Arbeláez, P.; Malik, J. Learning rich features from RGB-D Images for object detection and segmentation. In Proceedings of the 13th European Conference on Computer Vision (ECCV 2014), Zurich, Switzerland, 6–12 September 2014; Volume 8695, pp. 345–360.
35. Petrelli, A.; Pau, D.; Di Stefano, L. Analysis of compact features for RGB-D visual research. In Proceedings of the 18th International Conference on Image Analysis & Processing, Genoa, Italy, 7–11 September 2015; Volume 9280, pp. 14–24.
36. Canny, J.F. A computation approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1986**, *8*, 769–798.