Combining Boundary and Region Information with Bolt Prior for Rail Surface Detection

Yaping HUANG†, Nonmember, Siwei LUO†, Member, and Shengchun WANG†, Nonmember

SUMMARY  Railway inspection is important in railway maintenance. There are several tasks in railway inspection, e.g., defect detection and bolt detection. For those inspection tasks, the detection of rail surface is a fundamental and key issue. In order to detect rail defects and missing bolts, one must know the exact location of the rail surface. To deal with this problem, we propose an efficient Rail Surface Detection (RSD) algorithm that combines boundary and region information in a uniform formulation. Moreover, we reevaluate the rail location by introducing the top-down information—bolt location prior. The experimental results show that the proposed algorithm can detect the rail surface efficiently.

key words: railway inspection, boundary information, region information, line segmentation detection, template matching

1. Introduction

Recently, automatic railway maintenance is attracting more and more interests. Many companies, such as Benntec Systemtechnik Gmbh, Vision Light Tech, Sperry Rail, and researchers [1]–[4] are interested in proposing a suitable vision-based automatic detection system to perform railway inspection. The rail surface detection is a fundamental problem in performing the tasks shown in Fig. 1. For the rail defect detection task, only when the rail surface is located correctly, it is possible to find the defects correctly. With regard to the bolt detection task, we can observe that the bolts are always close to the rail and the distance between rail and bolts is constant. Therefore, if we can predict the location of the rail, the search region that may contain the possible bolts is extremely reduced.

To the best of our knowledge, there are only few literatures discussed on this problem. In Italy VISyR system, a patented real time Visual Inspection System for Railway maintenance, they proposed an approach, named Rail Detection & Tracking Block (RD&TB), which uses Principal Component Analysis (PCA) to extract the feature for a candidate window and Multi-layer Perceptron Neural Network (MLPs) to classify whether it is a center of the rail surface [5]. However, there is a critical disadvantage—laborious training work. More seriously, such training work must be performed manually once again when applied to a new railway. Lin et al. [1] used project analysis method to locate the rail surface. Projection method is primarily useful for good quality railway images, where the rail region has different contrast with other regions. However, this method has limited success when the image has similar contrast.

In this paper, we develop an efficient Rail Surface Detection (RSD) algorithm to detect the rail surface. Particularly, we propose a new efficient cost function that takes a ratio form: the numerator reflects the boundary information, and the denominator reflects the region information of the candidate region. The use of the ratio form makes the results biased to detecting longer and more homogeneous boundaries. Therefore, it leads to the robustness to image noise and confusing texture. This cost function can be easily expanded to include other boundary information, such as boundary continuity (smoothness), and region information such as region intensity contrast. However, there are some railway images where the rail region is not so distinctive relative to the background. In order to overcome this difficulty, by observing the fact that the bolts are always close to the rail, we introduce the top-down information of bolt location prior to further reevaluate the rail region. The experimental results show that our proposed algorithm can achieve the accuracy rate of 99.5% in detecting rail surface.

2. Railway Image Acquisition

A railway video is acquired using DALSA Spyder2 line scan camera mounted under a diagnostic train. Each image in the video is 800 × 1230. Figure 2 shows an example image.

3. RSD (Rail Surface Detection) Algorithm

From Fig. 2 we can see that the boundary of the rail surface is vertical to x-axis. Thus, the goal of RSD is to find two parallel lines as the left and right side of the rail region.
3.1 Line Segments

First, a set of line segments must be constructed from the original image. Indeed, there are many successful line detection methods. Recently, von Gioi et al. [6] proposed a fast line segmentation detection (LSD) algorithm. LSD merges all edges in a small patch that share the same gradient angle up to a certain tolerance using region-growing method, then finds the line segment that best approximates each line support region. Moreover, LSD is a very fast line detection method with linear time. The average time consuming of LSD algorithm for an 800×1230 image is 1.1 s. Since the rail region always locates nearing the center of the image, we can deal with the middle region such that the average time consuming can be decreased to 0.6 s. Therefore, RSD uses LSD to detect the vertical line segments of a railway image. But it is still not fast enough so as to work in real time. Fortunately, the rail defect detection and bolt detection tasks in our application are performed at off-line working style. In other words, the detection tasks will be performed after the video sequence has been acquired. However, a fast algorithm is better. Some strategies can be used to reduce the time consuming. We are planning to introduce GPGPU (General-Purpose computation on Graphics Processing Units) technology to speed up the performance.

Figure 3 shows the vertical line segments using LSD indicated by red lines. In order to avoid the noise, we only preserve the line segments whose length is larger than 5 pixels.

3.2 Problem Formulation

Given a railway image \( w \times h \), where \( w \) and \( h \) are the width and the height of the image respectively, and \( w_r \) denotes the width of the rail, we obtain a set of vertical line segments whose length is larger than 5 pixel using LSD. Since the rail region is vertical to \( x \)-axis, we use an interval at \( x \)-axis \([l, r] \) to represent a candidate region whose left and right coordinates are \( l \) and \( r \) respectively. Let \( \{0, 1, 2, \ldots, w-1 \} \) denotes the \( x \)-coordinates of the image, our task is to find the optimal interval \([l, r], l \in (0, w-w_r-1), r \in (w_r-1, w-1) \) within the set of line segments that satisfies the following cost function:

\[
\arg \max_{l,r} f = \frac{B([l, r])}{R([l, r])}
\]  

(1)

with the constrain of \( (r - l) = w_r \). The numerator and the denominator terms measure the boundary information and region homogeneity of the region \([l, r] \), respectively. The cost function with ratio form makes the results biased to detecting longer and more homogeneous regions. Thus, RSD is more robust to image noise and confusing texture.

Note that the optimization is time-consuming if the number of the candidate region is very large. Fortunately, our application limits the number of line segments, only several vertical and long parallel lines with fixed distance are preserved. Thus, the optimization problem can be solved through exhaustive search quickly.

3.3 Boundary and Region Information

For a candidate rail region that is denoted by an interval \([l, r] \), it is clear that the total length is the first important boundary measurement to detect the rail. But if two regions have the same boundary length, the region with less line segments would be better. Therefore, we introduce the following length measurement:

\[
B([l, r]) = \frac{TL(l)}{LN(l)} + \frac{TL(r)}{LN(r)}
\]  

(2)

where \( TL(l) \) and \( TL(r) \) mean that the total length of the line segments whose \( x \)-coordinates equal to \( l \) and \( r \) respectively. \( LN(l) \) and \( LN(r) \) mean that the total number of the line segments whose \( x \)-coordinates equal to \( l \) and \( r \) respectively.

As we discussed above, it is not enough if only considered boundary information. For some railway images, e.g., Fig. 4 (a), we mistakenly detect the rail region (indicated by two red parallel lines in Fig. 4(b)) only using boundary information. In this situation, we can observe that the gray value statistics on the error-detecting region are much different to the ground truth (Fig. 4(c)). Thus, it is a better choice that combining both boundary and region information.

We use the standard deviation of the intensity values to measure the gray-scale statistics of a region, i.e., the denominator term in Eq. (1). Note that here we use normalized gray-scale value to calculate the standard deviation. In fact, a normalization over this term sets a preference to produce more homogeneous region, which improves the robustness of RSD by avoiding a bias toward a region with longer boundaries but heterogeneous texture. Thus, the cost function can be rewritten as


\[
\arg \max_{I, r} f = \frac{TL(l)/LN(l) + TL(r)/LN(r)}{\sigma^2([l, r])}
\]

where \(\sigma^2(\cdot)\) denotes the standard deviation of a region.

4. Rail Location Reevaluation Using a Bolt Prior

4.1 Motivation

By combining the boundary and region information, we can deal with most railway images. However, some railway images, e.g., Fig. 5 (a), are more challenging. The detection result using Eq. (3) is shown in Fig. 5 (b). The reason maybe that there is a bright line at the center of the rail region. We are entirely misled by the middle line and another line parallel to this middle line. Unfortunately, it happens that the distance statistics between the two parallel lines equals to \(w_r\). What is worse that the intensity statistics between the two parallel lines are not much different comparing with the background.

Even for people, it is a very difficult task to detect the rail region correctly if only rail surface information is considered. However, people can still recognize the rail surface correctly when depending on the context information—the bolt locations. Indeed, we can see that the bolts are always close to the rail surface. Thus, we can utilize the top-down information to detect the rail surface. Guided by top-down information, we can further verify the rail location results using a bolt prior.

4.2 Bolt Detection

In the literature [4], Marino et al. adopted 2-D Discrete Wavelet Transform (DWT) to describe the feature of a bolt and Multi-layer Perceptron Neural Networks (MLPs) to determine presence or absence of a bolt. However, given the video of a new railway, it needs to train MLPs classifier again to learn the weights and then the weights is downloaded into the FPGA for each new video. It is very inconvenient for users. They have to collect positive and negative samples from the railway video, and wait a long time for training.

To avoid laborious training work, we propose a template matching-based algorithm to detect bolts based on two observations. Firstly, due to the limit categories of bolts, we can use template matching method to detect the bolts. Moreover, there is an obvious fact that the number of the missing bolts is much less than the number of visible bolts (1:500 in our dataset) and it must have one visible bolt in an image at least. Thus, we can easily find the first bolt using a powerful feature descriptor and a proper distance measurement.

Before detecting the bolts for railway images, we first collect bolt images with typical shapes to create a basic template library (some examples can be seen in Fig. 6). Then, the templates can be dynamically added to the library when processed a new railway video. User can choose left and right bolt samples from the first frame as new templates.

We choose Histogram of Oriented Gradient (HoG) [7] descriptors to extract the feature for a candidate window. The patch size of each bolt in an image is \(90 \times 100\). Then, the optimal parameters are found to be \(3 \times 3\) cell blocks of \(6 \times 6\) pixel cells with 4 normalization and 9 histogram channels. So a detection patch can be represented by \(4 \times 9 \times 4 \times 5 = 720\) bins. Moreover, we choose \(\chi^2\) distance to calculate the similarity between a candidate window and a template. Thus, the most similar candidate window is obtained using a nearest neighbor classifier and it is just the first bolt we want.

Let us return to the Fig. 5. When we get the bolt position indicated by red rectangle in Fig. 5 (c), we can reevaluate the rail detection result and get the exact rail position indicated by yellow parallel lines.

The Algorithm 1 illustrates the detailed information.

5. Experiments and Analysis

In this section, we demonstrate the efficiency of RSD al-
Our program is implemented using Visual Studio 2008 and carried out on Windows 7.0 platform with an Intel Core 2 (2.26 GHz CPU). We extract 13,190 images from 4 different railway videos. Figure 7 shows example images.

We compare RSD algorithm with other techniques, i.e., projection analysis method [1] and RD&TB [5], to demonstrate the efficiency of RSD. Table 1 shows the accuracy comparison on different railways.

On one hand, we can see that RSD achieves the accuracy rate of 98.2% when only used boundary information which is comparable to that of projection method. When combining the boundary and region feature, the accuracy is 98.9%. When using a bolt prior, the accuracy rate is improved to 99.5%. Furthermore, the Yi-wan line is the most difficult task among four railways because the contrast of the rail region in some images is not much distinctive in contrast to other railways, then the accuracy rate of projection method in Yi-wan line is much lower than that of other railways. Obviously, it is not enough that only boundary information is used. We must consider the region information or other context information, e.g., bolt location prior. Therefore, RSD introduces a bolt prior to reevaluate the rail surface results. In contrast to the bolt detection method proposed by [4] which has to collect training samples manually and train the classifier once again for a new railway, our approach is an unsupervised method and achieves 99.39% accuracy rate for detecting bolt.

On the other hand, we can see that supervised algorithms, i.e., RD&TB [5] and RSD III (it introduces the top-down information and can be somewhat regarded as a supervised algorithm), have higher accuracy than unsupervised algorithms, i.e., PA, RSD I and RSD II. Although the accuracy of RSD with a bolt prior and RD&TB are similar (RSD is just little higher), RD&TB has two disadvantages, i.e., manual collection of training data set and time-consuming training work. What worse is that the collection of training data set and the training process must be performed manually once again when applied to a new railway. Such training work is infeasible in our application. Thus, our performance is remarkable considering the fact that RSD does not need a training process and is more convenient for users.

### 6. Conclusion

In this article, we propose an efficient rail surface detection (RSD) algorithm. RSD combines the boundary and region information to detect the rail surface. Moreover, we introduce a bolt prior to reevaluate the result. The experimental results on several railway videos demonstrate the efficiency of RSD algorithm. However, there are still some problems that need to be solved. We will focus on improving the performance. The ongoing works will be reported in the future.

### Acknowledgements

This work is supported by National Nature Science Foundation of China (60975078), Fundamental Research Funds for the Central Universities (2011JBZ005) and R&D foundation of Shenzhen (JC201005280432A). We would like to thank the anonymous reviewer for helpful suggestions.

### References

[1] J. Lin, S.W. Luo, Q.Y. Li, H.Q. Zhang, and S.W. Ren, “Real-time rail head surface defect detection: a geometrical approach,” Proc. IEEE International Symposium on Industrial Electronics, pp.769–774, 2009.

[2] E. Stella, P.L. Mazzeo, M. Nitti, G. Cicirelli, A. Distante, and T. D’Orazio, “Visual recognition of missing fastening elements for railroad maintenance,” Proc. IEEE-ITSIC International Conference on Intelligent Transportation System, pp.94-99, 2002.

[3] P.L. Mazzeo, M. Nitti, E. Stella, and A. Distante, “Visual recognition of fastening bolts for railroad maintenance,” Pattern Recognit. Lett., vol.25, no.6, pp.669–677, 2004.

[4] F. Marino, A. Distante, P.L. Mazzeo, and E. Stella, “A real time visual inspection system for railway maintenance: automatic hexagonal headed bolts detection,” IEEE Trans. Syst. Man Cybern. C, Appl Rev., vol.37, no.3, pp.418–428, 2007.

[5] P. De Ruvo, G. De Ruvo, A. Distante, M. Nitti, E. Stella, and F. Marino, “A visual inspection system for rail detection & tracking in real time railway maintenance,” The Open Cybernetics and Systemics Journal, vol.2, pp.57-67, 2008.

[6] R.G. von Gioi, J. Jakubowicz, J.M. Morel, and G. Randall, “LSD: a fast line segment detector with a false detection control,” IEEE Trans. Pattern Anal. Mach. Intell., vol.32, no.4, pp.722–732, 2010.

[7] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp.886–893, 2005.