Texture Classification for Rail Surface Condition Evaluation

Ke Ma[1], Tomás F. Yago Vicente[1], Dimitris Samaras[1], Michael Petrucci[2], and Daniel L. Magnus[2]

[1]Stony Brook University

{kemma, tyagovicente, samaras}@cs.stonybrook.edu

[2]KLD Labs, Inc.

{mpetrucci, dmagnus}@kldlabs.com

Abstract

Rail surface defects threaten train and passenger safety. Hence rail surfaces must be restored using different processes depending on measurement of the severity of the defects. In this paper, we propose a new method for automatic classification of rail surface defect severity from images collected by rail inspection vehicles. It contains 2 components: a rail surface segmentation module, which utilizes structured random forests to generate an edge map and a Generalized Hough Transform to locate the boundaries of the rail surface; and a defect severity classification module, which combines multiple classifiers through a stacked ensemble model. The first-level learners are trained using descriptors of the rail surface images extracted by texton forests and texton dictionaries, with χ^2-kernel SVM classifiers. The probability estimation output of the first-level learners is the input to a second level linear-kernel SVM. Our experiments on a dataset of 939 images categorized into 8 severity levels achieved 82% accuracy.

1. Introduction

According to the Federal Railroad Administration (FRA), the expected duration of rail service is influenced mainly by the following factors: chemical composition of the rail, track maintenance programs, speed and tonnage traveling through. These factors contribute to the development of rail defects such as cracking, pitting, spalling, plastic flow and rail deformation [10].

Defects in the rail surface can develop into subsurface rail cracking eventually causing rail failures. There were 2,033 accidents caused by track problems from 2012 to Aug 31st, 2015 based on the FRA safety database[1]. Among these 2,033 accidents, there were 163 accidents closely related to rail surface defects, making them the second major accident cause. Therefore, the rail surface defects threaten the safety of trains and passengers, and can have large economic costs.

[1]http://safetydata.fra.dot.gov

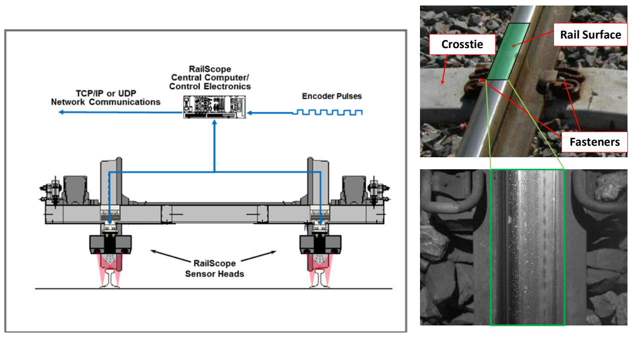

RailScope Central Computer/ Control Electronics
TCP/IP or UDP Network Communications
Encoder Pulses
RailScope Sensor Heads
Crosstie
Rail Surface
Fasteners

Figure 1: **Image Acquisition System**. The RailScope system consists of two rail surface image sensor heads, one for each rail, and a central computer/control electronics. Inside each sensor head are a laser light source that is used to illuminate the rail and a high resolution digital camera to capture the rail surface images. As the vehicle travels down the track high resolution images of the rail surface are acquired.

A key component in rail maintenance programs is periodic rail grinding. The level of grinding is adjusted to match the rail profile cross-section and the defect severity that the rail surface presents. Current practices involve assessment of the rail profile cross-section and rail surface. The rail profile cross-section is assessed typically by automated laser based machine vision systems. For the rail surface, this is typically performed by either physical inspection of the rail track or by reviewing rail surface image data. In both cases the process is subjective, prone to user biases, and in the case of track walking increased risk and costs.

In this paper, we propose a system for automatic rail surface defect severity classification. The system takes as

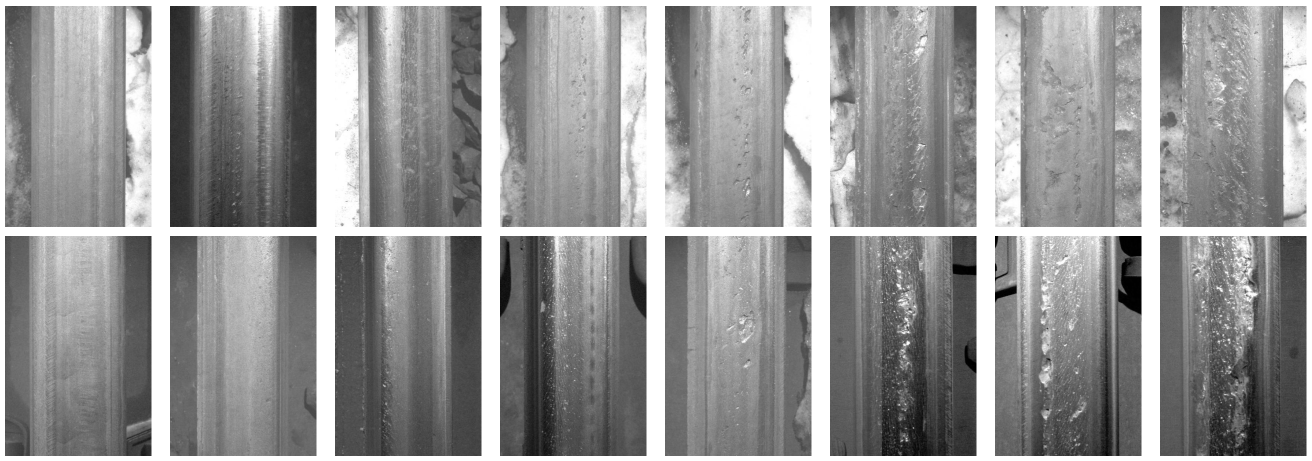


Figure 2: **Rail surface images.** Examples of the 8 different defect severity levels. 0 means no defects, 7 means the most serious defects.

input high resolution rail surface images acquired by the RailScope System [14] installed underneath a moving railway vehicle and positioned over the center of the rail as shown in Figure 1. The RailScope System is a state-of-the-art high resolution image acquisition system used to collect and display top of rail (rail head) images from a moving rail or road/rail vehicle. This system captures blur free images of the rail surface in real-time with sufficient resolution to detail pitting and surface cracking on the top of the rail surface. The system utilizes a laser light source unit to illuminate the rail head and a high resolution CCD digital camera to capture the rail head images. All the images collected contain the meta-information including railroad name, division name, track number, milepost number and timestamp. With this information, we can retrieve when and where the images were taken for rail maintenance.

The segmentation component of our system automatically extracts the rail surface region of the input image. We propose a segmentation method based on computing the Generalized Hough transform on the edge map of the image. To detect the relevant edges we train a structured output random forest on hand segmented images.

The defect severity assessment component of our system classifies rail surface images into 8 categories of defect severity (0-7), in a classification designed by experts in the field. Figure 2 shows examples of categories 0 (no damage), 4 (medium damage), and 7 (high damage). The classification of the defect severity of the rail surface images is based on the two most prevalent texture representations: We build multiple texton dictionaries and texton forests to learn different texture representations from the data. For each texture representation we train a χ^2-kernel Support Vector Machine (SVM) with probabilistic output [5]. We combine the outputs of the different classifiers with a second level SVM that predicts the final category level for the rail surface image.

We report experiments on a dataset of 939 rail surface images annotated by experts in the field. Our proposed rail surface segmentation method achieves a Jaccard index of 98.64% between the segmented rail surface and the ground truth annotation. For classification, our method correctly predicts the damage severity category for the rail surface with 82% accuracy. An additional 13.6% of rail surfaces are classified within an adjacent category (-1 or +1 from the ground truth annotation) which is acceptable, given annotation error and the inherent smoothing between sections, in the rail grinding process.

To the best of our knowledge this is the first fully automatic system that classifies the overall condition of rail surfaces, and not just individual effects, based on visual inspection, and produces information that can be directly acted up on, for rail maintenance. The benefits of such a system will be multiple: Such an automatic system would enable much more frequent inspection of the entirety of the installed rail track. Thus small defects could be addressed before, through additional wear and tear, they become larger and deeper and potentially threaten train and passenger safety. Furthermore deteriorated rail surfaces create ride quality issues, accelerated wear of components and cause additional friction which slows trains down and increases fuel consumption. Hence maintaining rail in good condition has significant economic and environmental benefits.

The rest of the paper is organized as follows: We describe related previous work in Section 2. In Section 3 we present our automatic rail surface segmentation method. In

Section 4, we define our rail surface classification method. In Section 5 we present our experimental results. Section 6 concludes the paper.

2. Previous Work

Computer vision techniques have been applied before in the railway inspection industry. Different methods and devices have been designed and proposed for different components of the rail [19, 20, 21], such as fasteners [7, 12] and joint bars [1]. In [17] and [27], methods to detect tie-related components including ties, tie plates and anchors, are proposed.

One of the earliest working applications of computer vision was by Magnus et al. for analyzing the rail specifically targeting the wear condition [19]. The system focused on collecting rail cross-sections using a combination of video cameras and light sources. The basis of the analysis was to acquire video images of the rail and then apply a threshold for edge detection to extract the rail profile. This was further improved in work by KLD presented at the Machine Vision in Wheel/Rail Maintenance Seminar in 1987 [21] where an edge detection algorithm helped to more accurately define the profile. Later in 1995, D. L. Magnus [20] showed how peak detection for edge detection applied to the Gaussian light distribution, further defined the rail surface condition.

There are previous studies on defects on the rail surface, that typically target a specific defect. In [22], Mandriota et al. propose a filter-based rail surface defect detection system, focused on corrugation. The texture is extracted by Gabor filters, wavelets and Gabor wavelet filters, statistical features are computed and detection results using KNN and SVM are compared. Deutschl et al. [8] design a novel lighting system to facilitate the detection of defects. They illuminate the rail surface with 2 light sources of different colors from different positions, hence defects are expected to be on the shaded part under this lighting system. The appropriate shading features are input to a decision tree. Li et al. [16] target 2 kinds of defects based on area size. They localize defects by aggregating intensities along the longitudinal and transversal axes and detecting the defect signatures. Liu et al. [18] describe a spalling defect detection system, in which candidate spalling defect regions are found through relative thresholding of the defect regions. In an example of surface classification, Huber-Mörk et al. [13] describe a system using a specialized illuminant configuration, which uses Gabor filter banks to extract 2D texture descriptions and categorize defects using a Gaussian mixture model. More comprehensive surveys can be found in [3, 24].

Compared to previous work which either addresses specific type of defects, or models the defects classes based on clustering, we cooperate closely with field experts to study the rail surface defects comprehensively and quantize the defect severity levels to 8 classes. Meanwhile, rather than rely on only one prediction model, our classification module combines several classifiers, forming an ensemble model, which boosts the result a step further. Moreover, our system is an end-to-end system which includes a segmentation module and a classification module. It can directly take a RailScope image as input and output the defect severity level.

3. Automatic Rail Surface Segmentation

We propose a method to automatically segment the rail surface from high resolution images captured with the RailScope imaging system. Besides the rail surface, the captured images also include some background environment elements like rocks, mud, and other rail components such as crossties or fasteners, see figure 3.a for an example image. We want to extract the rail surface from the raw image and exclude the background. Direct segmentation of railway components is non-trivial [12]. It is worth noting that all rail surfaces are bounded by 2 straight lines. These lines correspond to the boundaries of the surface and are almost vertical. We pose rail surface segmentation as the problem of finding this pair of parallel straight lines. To solve it, first we compute an edge map of the image using a structured output random forest trained on hand segmented images, see figure 3.b. Then, we apply a Generalized Hough transform to find the pair of parallel lines corresponding to the rail surface boundaries, see Figure 3.e. Similarly, Li et al. [17] use a Generalized Hough transform on a Sobel edge map to detect rail ties.

3.1. Generalized Hough Transform for Rail Surface Segmentation

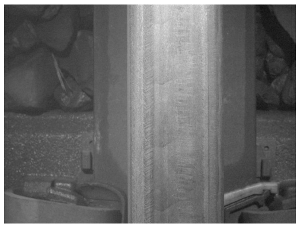
A pair of parallel straight lines, corresponding to the rail boundaries, defines the rail surface region in the image. We parameterize this region with 3 parameters, θ, ρ and $width$ as shown in Figure 4.a. The parameters are defined as:

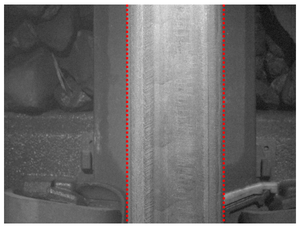
- θ indicates the parallel line direction since the rail surface boundaries are not perfectly vertical in the image.
- ρ represents the distance of the left boundary of the rail surface from the left top point of the image.
- $width$ is the minimal distance between 2 parallel straight lines.

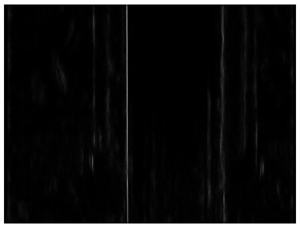
Parallel lines are defined in terms of θ, ρ and width as:

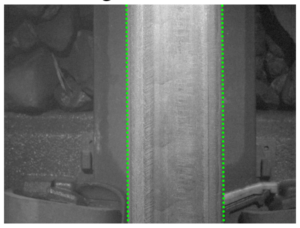
$$\begin{cases} x * \cos(\theta) + y * \sin(\theta) - \rho = 0 \\ x * \cos(\theta) + y * \sin(\theta) - \rho - width = 0 \end{cases} \quad (1)$$

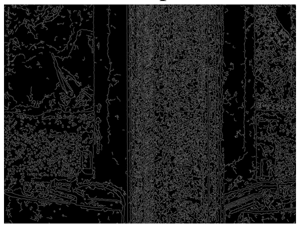
We apply a Generalized Hough transform on the edge map of the image using this parameterization. The coordinates of the most voted point in parameter space correspond


(a) Input image


(b) Hand annotated segmentation


(c) Random Forest edge map


(d) Segmentation from RF edge map


(e) Canny edge map

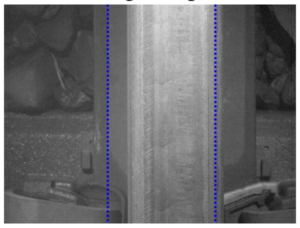
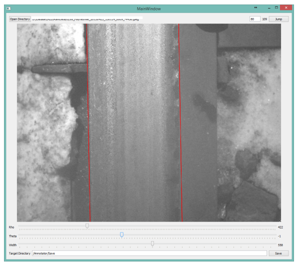

(f) Segmentation from Canny edge map

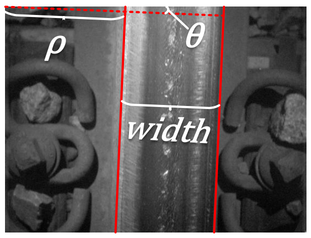
Figure 3: **Rail surface segmentation results.** Comparison of segmentation results using structured edge detection (Random Forests) and Canny edge detection on rail images. (a) Original input image. (b) Hand annotated rail boundaries in **red**. (c) Edge map obtained by structured edge detection with Random Forests. (d) Segmentation results in **green** of the Generalized Hough Transform on the Random Forest edge map of (c). (e) Canny edge map from an image smoothed by a guided filter. (f) Segmentation results in **blue** of the Generalized Hough Transform on the Canny edge map of (e). The segmentation in (d) has a Jaccard index of 0.9887 with the ground truth (b) while the segmentation in (f) has 0.7606.

to the optimal values for θ, ρ and $width$. To accelerate computation, we restrict the parameter range to correspond to the ranges found in human-segmented images.

3.2. Annotation Tool

We developed a graphical user interface (GUI) for manual annotation of the rail surface region in images. The main window of the annotator is shown in figure 4. Users can easily drag 3 slider bars, that correspond to the 3 parameters θ, ρ and $width$ used by the Generalized Hough transform, to properly align the 2 red straight lines to the rail surface boundaries in the image.


(a)



(b)

Figure 4: **Rail surface region annotation.** (a) Annotation tool GUI. (b) Rail surface region defined by 3 parameters θ, ρ and width. Note that the dashed line is perpendicular to the 2 solid lines.

3.3. Edge Map Generation

To segment the rail surface we apply the Generalized Hough transform on the edge map of the image. We first tried the Canny edge detector [4] to obtain the edge map. As we can see in Figure 3.c, Canny finds confounding edges in the background regions. These often lead to poor rail segmentation as shown in Figure 3.f. To generate better edge maps that contain only semantically relevant edges, we learn a rail edge detector from labeled data. We train a random forest with structured output[9]. We use the human segmented rail images as input ground-truth. We use rail image intensity and gradient features. In Figure 3.b we show the edge map obtained with the trained random forest. We can clearly see how most of the detected edges correspond to rail edges. Moreover, very few background edges are detected. In Figure 3.e we can see the rail surface segmentation obtained by applying the Generalized Hough transform on the random forest edge map.

4. Classification of Defect Severity Level

In this Section we describe our two level classification system using texton dictionaries, texton forests and an ensemble classifier. Classifying the severity of the segmented rail surface images (Sec. 3) is a texture classification task. Rail surface defects exhibit distinctive textural characteristics, depending on the severity level. A widely-used method for classifying images is the Bag of Words model (BoW) [11]. It has been successful on multiple image classification datasets in computer vision [31]. This model generates image descriptors of images based on the histogram of visual words. In our system, we apply texton dictionaries and texton forests to form the visual words.

4.1. Texton Dictionary

Texton is a term describing the response of a linear filter bank. A filter bank separates the input signals into multiple components. We use the Full Maximum Response filter bank (FMR8) [28, 29], which contains 38 low dimensional and rotationally invariant filters.

The detailed process is as follows: every pixel of the image is convolved by all 38 filters, which include 6 orientations at 3 scales for 2 oriented filters (edge filters and bar filters), plus 2 isotropic filters: a Gaussian filter and a LoG filter. This process is repeated for all images, and the responses for each pixel form a 38-dimensional feature vector. Then the representative features or visual words can be obtained by clustering all feature vectors into k centers. We use k-means to construct a dictionary of k words (each centroid is a visual word). Thus, for each image, the descriptor will be a histogram of k bins, normalized for image size. Each bin contains the number of pixels that are closest to the corresponding visual word. Using these image descriptors, we train a χ^2-kernel SVM on the training set and use it to predict the labels of the images on the test set.

In practice, using all pixels within an image and all images in a category requires too much memory. We randomly sample $p\%$ pixels per image and $m\%$ images per category.

4.2. Texton Forest

Texton forests were proposed by Shotton et al. [26] as a variant of random forests. The novelty is that instead of training a random forest to predict the label of the test data, texton forest trains a random forest model to extract an image descriptor, by making use of both the leaf nodes and internal nodes of each decision tree.

Texton forests are trained like conventional random forests. Each decision tree in the forest is trained with a subset of the training set and at every split of the tree, only a small number of the features are taken into consideration. We assume $h(x, \theta_j)$ is the split function for data x at node j and θ is the parameter set. For example, using $\theta = \{m, n, \tau\}$, and the difference between 2 feature values in x, $d(x, m, n) = x_m - x_n$, we can define a split function as:

$$h(x, \theta) = \begin{cases} 0 & \text{if } d(x, m, n) < \tau \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

Other split functions used in texton forest include the raw feature value $d(x, m) = x_m$ and the absolute difference between 2 features $d(x, m, n) = |x_m - x_n|$. The feature vector x is an image patch and m, n are the coordinates of the pixels.

Using this trained forest model, each image patch generates a traversal of the forest. Thus we compute an image descriptor as a histogram which counts how many times each (non-root) node has been traversed. As with Texton Dictionaries, we train a χ^2-kernel SVM on the texton forest descriptors.

4.3. Ensemble Model

Accuracy can be further improved by fusing the decisions from several models through an ensemble model. There are many ways to train an ensemble model or combine the results of several models together such as boosting, bagging and mixture of experts. Zhou [32] provides a comprehensive survey of ensemble methods.

Here, we apply stacking [30][2], which is a general "combining by learning" framework to learn another model based on the outputs of the existing models. We call the existing models the *first-level learners* and the model that combines them the *second-level learner* or *meta-learner*[32]. The meta-learner concatenates the outputs of the first-level learners into an input feature vector with the same ground-truth label for supervised learning.

It is meaningless to create an ensemble of identical learners. So the first-level learners are decorrelated by using classifiers trained with different descriptors using different subsets of the data and different parameters. In our system, we train 5 texton forest SVM classifiers and 4 texton dictionary SVM classifiers. Each classifier estimates the probability p_i the input belongs to class i. Hence for our 8 classes, the classifier outputs $P = [p_1, p_2, ..., p_8]$ for every input x. The concatenation of the 9 probability estimate vectors generated by the first-level classifiers for x forms a new feature vector of length 72 which is the input of the meta-learner. Finally, we train a linear-kernel SVM as the meta-learner to further improve classification results.

5. Experimental Results

For our experiments we use a dataset of 939 rail images. The images are categorized by field experts into 8 defect severity levels: from category 0 for no defects, up to category 7 for serious defects. The distribution of images per defect category is shown in Table 1. The dataset also contains ground-truth rail surface masks resulting from human segmentation of the rail images.

Category	0	1	2	3	4	5	6	7
Number	110	111	101	148	214	92	105	58

Table 1: **Category distribution in the dataset.** Number of images in each defect severity level.

We evaluate our approach using 4-fold cross validation on the dataset. We divide the dataset into 4 disjoint splits of equal size. The images are randomly drawn from each category uniformly so as to keep each split with a similar label distribution.

Method	mean JI	std JI
SE + Hough	**0.986**	**0.031**
Canny + Hough	0.858	0.138

Table 2: **Rail surface segmentation results.** Segmentation results measured by the Jaccard Index(JI) between the segmented rail region and the ground-truth region mask.

5.1. Automatic Segmentation Results

We evaluate the rail surface segmentation using the Jaccard index, computed as follows:

$$J(R_s, R_g) = \frac{|R_s \cap R_g|}{|R_s \cup R_g|} \tag{3}$$

where R_g is the ground truth rail surface region and R_s is the segmented region. The Jaccard index penalizes both background pixels wrongly segmented as rail, and rail pixels wrongly segmented as background.

We evaluate two versions of our segmentation method based on the Generalized Hough transform, using the Canny edge map (denoted as Canny+Hough), and using the edge map obtained by the trained random forest with structure output (SE+Hough). Table 2 shows the segmentation results measured by the mean Jaccard Index (mean JI) for all images. Our segmentation method using Canny edge maps yields a mean JI of 0.858. Our segmentation method using edge maps from the structured output random forest achieves a mean JI of 0.986.

In terms of computational time, our segmentation method with the random forest edge map implemented as a mex function in Matlab takes 4.8s to segment a 1600×1200 image. The segmentation method using the Canny edge map (computed with the Matlab built-in function `edge`) takes 4.2s per image. Experiments were performed on a desktop computer with I3 CPU at 3.6GHz and 8GB RAM.

5.2. First Level Classification Results

We train a texton forest with 12 trees of maximal depth of 10. The forest uses patches of size 32 $\times$ 32 pixels with a stride of 16. We perform data augmentation by randomly rotating, scaling, and adding Gaussian noise to the original patches. We use the trained forest to generate a texture descriptor for the data. Using this texture descriptor we train a χ^2-kernel SVM classifier for rail images. We denote this method as TF.

To create a texton dictionary, we first apply the full MR8 filter bank[28] on the segmented rail surface images. Then, we subsample the filter responses by randomly selecting $m\%$ images per category and then randomly choosing $p\%$ pixels of the selected images. Finally, we cluster the subsampled filter responses using k-means to generate a dictionary of k visual words. For our experiments we set m to 25 images, p to 20% of the pixels, and k to 512 words. Using this texton dictionary we compute a histogram of textons to describe the texture of the rail surface images. Using this descriptor we train a χ^2-kernel SVM classifier for rail images. We denote this method as TD.

We evaluate our classification result by the main diagonal accuracy, which is the sum of the diagonal elements of the confusion matrix divided by the sum of the elements of the whole matrix. Additionally, we report the tridiagonal accuracy, which is the sum of the elements of the tridiagonal band of the confusion matrix over the sum of elements of the whole matrix. This looser definition of accuracy considers correct predictions that are one level up or down from the ground truth. This is often a reasonable relaxation, given annotation error and the inherent smoothing between rail sections, in the rail grinding process.

In table 3, when the classification module is trained and tested using ground truth segmented rail surface images, namely, hand segmented images, texton forests (TF-GTseg) can achieve 79.5% main diagonal accuracy and 94.8% tridiagonal accuracy. Texton dictionaries (TD-GTseg) perform slightly worse, with 78.2% main diagonal accuracy and 94.2% tridiagonal accuracy.

Furthermore, as a baseline, we include the results of Histogram of oriented Gradients (HOG) [6] with SVM classifier and Local Binary Patterns (LBP) [23] with k-nearest neighbors classifier (KNN). We also implemented a Convolutional Neural Network (CNN) classification method based on the AlexNet deep learning architecture [15]. In particular, the CNN is trained on 400 by 400 regions randomly extracted from the rail surface images in each iteration. At test time, we apply the CNN on five regions (top left, top right, bottom left, bottom right, center) and average the predicted probabilities for each class.

On our dataset, HOG+SVM achieves a main diagonal accuracy of 73.6% and tridiagonal accuracy of 89.8%, which is lower than both texton forest and texton dictionaries. LBP+KNN performance is close to texton dictionaries. The CNN model (AlexNet-GTseg) does not perform as well as texton forests or texton dictionaries with a main diagonal accuracy of 75.2% and tridiagonal accuracy of 92.2%. This may be because our dataset is too small to train a good CNN model.

The results of both texton forests and texton dictionaries drop when they are trained and tested using automatically segmented rail surface images. The main diagonal accuracy of texton forests (TF-AutoSeg) drops 1.8%, to 77.7%, and the tridiagonal accuracy drops 1.3% to 93.5%. Texton dictionary (TD-AutoSeg) accuracy drops 1% on the main diagonal to 77.2%, and drops 0.8% to 93.4% on tridiagonal accuracy. This demonstrates that imperfect segmentation still influences adversely the classification module, both because included background pixels may pollute the

Method	Main Diag AC	Tridiag AC
TF-GTseg	**79.5**	**94.8**
TD-GTseg	78.2	94.2
TF-AutoSeg	77.7	93.5
TD-AutoSeg	77.2	93.4
AlexNet-GTseg	75.2	92.2
HOG+SVM	73.6	89.8
LBP+KNN	78.2	94.2

Table 3: **Defect severity level classification results (single classifier).** TD: Texton Dictionary. TF: Texton Forest. Methods marked "-AutoSeg" are trained and tested with automatically segmented rail surface images. Methods trained and tested using manually segmented rail surface images are marked "-GTseg". Main Diag AC: Main Diagonal Accuracy. Tridiag AC: Tridiagonal Accuracy. HOG+SVM and LBP+KNN are tested with manually segmented rail surface images.

Method	Main Diag AC	Tridiag AC
4TD-GTseg	78.0	94.2
5TF-GTseg	79.6	94.4
4TD5TF-GTseg	80.4	94.8
ES-GTseg	**82.0**	**95.6**
ES-AutoSeg	79.9	94.1

Table 4: **Defect severity level classification results (ensemble model).** ES: EnSemble model. 4TD: first-level learners are 4 SVMs using Texton Dictionary descriptors. 5TF: first-level learners are 5 SVMs using Texton Forest descriptors. Methods marked "-AutoSeg" are trained and tested with automatically segmented rail surface images. Methods trained and tested using manually segmented rail surface images are marked "-GTseg". Main Diag AC: Main Diagonal Accuracy. Tridiag AC: Tridiagonal Accuracy.

		Predicted severity level							
		0	1	2	3	4	5	6	7
Actual severity level	0	94	8	1	1	3	1	0	0
	1	12	88	6	1	1	0	0	0
	2	6	7	85	2	0	0	0	0
	3	0	2	0	117	29	0	0	0
	4	1	1	2	12	190	5	0	1
	5	0	0	0	1	4	60	22	5
	6	1	0	0	0	3	7	86	7
	7	0	0	1	0	2	7	5	41

(a) Confusion matrix on hand segmented images

		Predicted severity level							
		0	1	2	3	4	5	6	7
Actual severity level	0	96	7	1	0	3	1	0	0
	1	17	83	5	1	1	0	0	1
	2	6	9	83	2	0	0	0	0
	3	0	1	0	113	33	1	0	0
	4	2	1	1	8	195	4	0	1
	5	1	0	0	0	6	57	18	10
	6	0	0	0	0	3	11	83	7
	7	0	2	0	0	2	16	5	31

(b) Confusion matrix on automatically segmented images

Figure 5: **Confusion matrices of the ensemble model results.**

Category	Recall	Precision	F-measure
0	87.0	82.4	84.7
1	81.5	83.0	82.2
2	85.0	89.5	87.2
3	79.0	87.3	83.0
4	89.6	81.9	85.6
5	65.2	75.0	69.8
6	82.7	76.1	79.3
7	73.2	75.9	74.5
mean	80.4	81.4	80.8
std	7.9	5.4	6.0

Table 5: **Performance of ensemble model in each category**

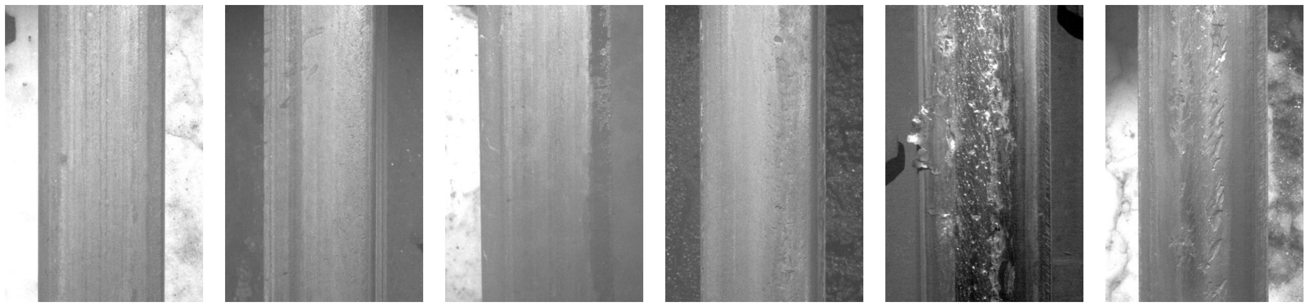
visual words and because missing rail surface pixels may include discriminant texture patterns.

5.3. Ensemble Model Classification Results

The first level learners in our proposed ensemble model are 5 texton forest SVM classifiers and 4 texton dictionary SVM classifiers. The texton forest models are trained on different subsets of the data (bagging), so that the classifiers are decorrelated. We create 3 texton dictionaries with 256, 512 and 1024 words, respectively. We also create a fourth dictionary, by clustering the filter responses of each category separately into 64 clusters. Then, we aggregate the 64 words from each category to create a 512-word dictionary.

The proposed meta-learner is a linear-kernel SVM trained with the probability estimation output of all 9 first-level learners. In Table 4, we show classification results for the meta-learner. The meta-learner trained with ground-truth segmentation (ES-GTseg) accurately predicts 82% of the rail surface images. That is a 3% increase with respect to a single Texton Forest classifier (TF in Table 3. Furthermore, the tridiagonal accuracy of the meta-learner is 95.8%, (94.9% for the single Texton Forest classifier). In figure 5.b we show the confusion matrix results of the meta-learner.

We compare the performance of the SVM meta-learner with the results of averaging the probability output of the first level learners in table 4. The SVM meta-learner clearly outperforms averaging texton dictionary classifiers (4TD-GTseg), texton forest classifiers(5TF-GTseg), and also averaging all first level classifiers(4TD5TF-GTseg).



(a) actual level: 0 predicted level: 3 (b) actual level: 1 predicted level: 4 (c) actual level: 2 predicted level: 0 (d) actual level: 3 predicted level: 1 (e) actual level: 5 predicted level: 7 (f) actual level: 7 predicted level: 2

Figure 6: **Failure cases.**

We also measured the performance of the proposed SVM meta-learner when using first level learners trained on automatically segmented images (ES-AutoSeg). As we can see in table 4, there is a small performance drop when compared to using ground truth segmentation. Main diagonal accuracy drops by 2% while tridiagonal accuracy drops by 1.5%. In Figure 5 we compare the confusion matrices for the SVM meta-learner using ground-truth segmentation (Figure 5.a) and automatic segmentation segmentation (Figure 5.b).

Moreover, we evaluate the performance of the meta-learner in each category in table 5. The definition of recall, precision and F-measure follows that in [25]. The performance of the meta-learner in each category varies within a relatively small range. Recall in category 0 is highest, which is 87.0%, while category 5 has the lowest recall of 65.2%. Category 2 has the highest precision of 89.5%, and precision in category 5 is lowest, that is 75.0%. The meta-learner achieves the highest F-measure of 87.2% in category 2 and the lowest of 69.8% in category 5.

5.4. Classification Failures

Some failure cases are shown in figure 6. The defect severity level is over-estimated in figure 6.a and figure 6.b possibly because dirt on the surface is mistakenly classified as a defect. In figure 6.c, the defect severity level is under-estimated because the lower-right part of the rail surface is under-lit and the texture is suppressed. The severity level in figure 6.d is underestimated because while there are numerous small-scale defects scattered on the surface, individual textons do not contain enough defects.

Figure 6.e and figure 6.f are challenging cases. Though it appears that the severity level of figure 6.e is higher than that of figure 6.f, the depth of the defect in figure 6.f might be deeper, which might require more grinding to restore.

6. Conclusions and Future Work

In this paper, we presented an automatic rail surface defect severity level classification system. This system contains 2 modules: a rail surface segmentation module and a defect severity classification module. To reduce the influence of the background pixels, the automatic segmentation module locates the boundaries of the rail surface by using Generalized Hough transform together with the edge map obtained by a trained structured random forest. The segmented rail surface images are classified using a stacked ensemble model. The first-level learners of this model are χ^2-kernel SVM classifiers which are trained on the descriptors extracted by 5 texton forest models and 4 texton dictionaries and the meta-learner is a linear kernel SVM built on the probability output of the first-level learners. Thus, we achieved a main diagonal accuracy of 82% and tridiagonal accuracy of 95.58% on our dataset.

We have demonstrated how the use of state-of-the-art techniques for texture analysis can produce information that can be useful to directly improve rail condition, with significant safety, financial and environmental benefits.

Currently, we are working to augment the size of the dataset, which could lead to training higher accuracy CNN models. Moreover, we will subdivide the rail surface into several zones, and try to investigate if more serious defects are correlated with certain parts of the rail surface.

Acknowledgments. We thank Ali Tajaddini from the Federal Railway Administration, Robert C. Harris from LORAM and Eric Magel from the National Research Council of Canada for their support of this project. This work was funded by the Federal Railroad Administration grant FRA DTFR5315C00011 and also by NSF IIS-1161876, IIS-1111047, the Subsample project from DIGITEO Institute, France, and a gift from Adobe Corp.

References

[1] A. Berry, B. Nejikovsky, X. Gilbert, and A. Tajaddini. High speed video inspection of joint bars using advanced image collection and processing techniques. In *World Congress on Railway Research*, 2008.

[2] L. Breiman. Stacked regressions. *Machine learning*, 1996.

[3] L. F. M. Camargo, J. R. Edwards, and C. P. Barkan. Emerg-

ing condition monitoring technologies for railway track components and special trackwork. In *Joint Rail Conference*. American Society of Mechanical Engineers, 2011.

[4] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986.

[5] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2011.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*. IEEE, 2005.

[7] P. De Ruvo, A. Distante, E. Stella, and F. Marino. A gpu-based vision system for real time detection of fastening elements in railway inspection. In *ICIP*. IEEE, 2009.

[8] E. Deutschl, C. Gasser, A. Niel, and J. Werschonig. Defect detection on rail surfaces by a vision based system. In *Intelligent Vehicles Symposium*, pages 507–511. IEEE, 2004.

[9] P. Dollár and C. L. Zitnick. Structured forests for fast edge detection. In *ICCV*. IEEE, 2013.

[10] Federal Railroad Administration. *Track Inspector Rail Defect Reference Manual*, July 2015.

[11] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *CVPR*. IEEE, 2005.

[12] X. Gibert, V. M. Patel, and R. Chellappa. Robust fastener detection for autonomous visual railway track inspection. In *WACV*. IEEE, 2015.

[13] R. Huber-Mörk, M. Nölle, A. Oberhauser, and E. Fischmeister. Statistical rail surface classification based on 2d and 21/2d image analysis. In *Advanced Concepts for Intelligent Vision Systems*. Springer, 2010.

[14] KLD Labs, Inc. Rail surface evaluation, 2012. `http://www.kldlabs.com/?page_id=63`.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 2012.

[16] Q. Li and S. Ren. A real-time visual inspection system for discrete surface defects of rail heads. *IEEE Transactions on Instrumentation and Measurement*, 2012.

[17] Y. Li, H. Trinh, N. Haas, C. Otto, and S. Pankanti. Rail component detection, optimization, and assessment for automatic rail track inspection. *IEEE Transactions on Intelligent Transportation Systems*, 2014.

[18] Z. Liu, W. Wang, X. Zhang, and W. Jia. Inspection of rail surface defects based on image processing. In *International Asia Conference on Informatics in Control, Automation and Robotics*. IEEE, 2010.

[19] D. Magnus and G. Westhoff. Development and testing of an automatic rail wear measuring system. Technical Report Report # DOT-TSC-1280, US Department of Transportation, 1978.

[20] D. L. Magnus. Noncontact technology for track speed rail measurements: Orian. *Proc. SPIE*, 1995.

[21] S. Magnus. Current technologies for rail profile measurement orian. Machine Vision in Wheel/Rail Maintenance Seminar, Ontario Ministry of Transportation and Communications, 1987.

[22] C. Mandriota, M. Nitti, N. Ancona, E. Stella, and A. Distante. Filter-based feature selection for rail defect detection. *Machine Vision and Applications*, 2004.

[23] T. Ojala, M. Pietikäinen, and D. Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 1996.

[24] M. P. Papaelias, C. Roberts, and C. Davis. A review on non-destructive evaluation of rails: state-of-the-art and future development. *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and rapid transit*, 2008.

[25] D. M. Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. 2011.

[26] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *CVPR*. IEEE, 2008.

[27] H. Trinh, N. Haas, Y. Li, C. Otto, and S. Pankanti. Enhanced rail component detection and consolidation for rail track inspection. In *WACV*. IEEE, 2012.

[28] M. Varma and A. Zisserman. Classifying images of materials: Achieving viewpoint and illumination independence. In *ECCV*. Springer, 2002.

[29] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *International Journal of Computer Vision*, 2005.

[30] D. H. Wolpert. Stacked generalization. *Neural networks*, 1992.

[31] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *International journal of computer vision*, 2007.

[32] Z.-H. Zhou. *Ensemble methods: foundations and algorithms*. CRC Press, 2012.