RC25435 (WAT1312-044) December 19, 2013
Computer Science

IBM Research Report

Rail Component Detection, Optimization and Assessment for Automatic Rail Track Inspection

Ying Li, Hoang Trinh, Norman Haas, Charles Otto, Sharath Pankanti
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598
USA

Research Division
Almaden – Austin – Beijing – Cambridge – Dublin - Haifa – India – Melbourne - T.J. Watson – Tokyo - Zurich

Rail Component Detection, Optimization and Assessment for Automatic Rail Track Inspection

Ying Li, *Senior Member, IEEE,* Hoang Trinh, *Member, IEEE,* Norman Haas, Charles Otto, *Student Member, IEEE,* and Sharath Pankanti, *Fellow, IEEE*

***Abstract*—In this paper, we present a real-time automatic vision-based rail inspection system, which performs inspections at 16 km/h with a frame rate of 20 fps. The system robustly detects important rail components such as ties, tie plates and anchors with high accuracy and efficiency. To achieve this goal, we first develop a set of image and video analytics, then propose a novel global optimization framework to combine evidence from multiple cameras, GPS (Global Positioning System) and DMI (Distance Measurement Instrument) to further improve the detection performance. Moreover, as the anchor is an important type of rail fastener, we have thus advanced the effort to detect anchor exceptions, which includes assessing the anchor conditions at tie level and identifying anchor pattern exceptions at compliance level.**

Quantitative analysis performed on a large video data set captured with different track and lighting conditions as well as on a real-time field test, have demonstrated very encouraging performance on both rail component detection and anchor exception detection. Specifically, an average of 94.67% precision and 93% recall rate has been achieved on detecting all three rail components, and a 100% detection rate is achieved for compliance-level anchor exception with 3 false positives per hour. To our best knowledge, our system is the first to address and solve both component and exception detection problems in this rail inspection area.

***Index Terms*—Railroad track inspection, rail component detection, anchor exception detection, multi-sensor evidence integration, machine-vision technology**

I. Introduction

To maintain safe and efficient operations, railroads must inspect their tracks for physical defects on a regular basis. It is not only required by Federal Railroad Administration (FRA is a USA agency) regulations, but also enforced by individual railroad companies, usually with more stringent requirements, to maintain track health to a higher standard. Such track inspection normally covers a wide spectrum, ranging from detecting surface cracks in the rail, measuring rail profile and gauge (or rail spacing), to monitoring the conditions of joint bars, spikes and anchors. Figure 1 shows an illustration of a rail fastening system containing components such as tie (aka sleeper), tie plate, spike, joint bar and anchor. Specifically, *Ties* are used as a base to support and fix railroad tracks, as well as to transfer the load from rails to the underlying ballasts and sub-grades. *Tie plates* are then placed on ties (one for each rail), and held in place by steel spikes to prevent the rail from latitudinal movements. Since each tie plate on a tie fastens one rail, it forms two sides, namely, gauge side and field side. To prevent the rail from longitudinal movements, *anchors* are placed around each of the two rails, which wrap around the bottom and sides of the rail base. Consequently, there are at most 4 anchors for each tie, with 2 anchors on each side of the tie (see Fig. 1 (a)).

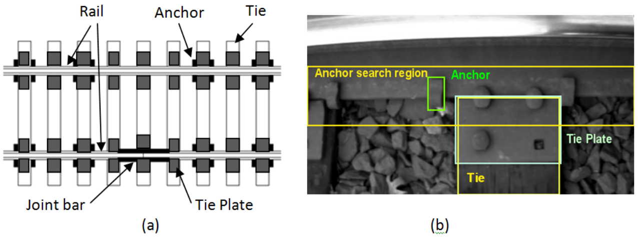


Fig. 1. An illustration of (a) rail fastening system; and (b) targeted rail components: Tie plate (in cyan), tie (in yellow) and anchor (in green). The round objects on the tie plate are spikes that fasten it to the ground.

Some of the track inspections such as measuring the track's curvature and alignment, as well as the crosslevel of the two rails, have already been automated using a track geometry car, yet other inspections such as monitoring the spiking and anchor patterns and detecting raised or missing spikes and anchors, are still manually and visually conducted by railroad track inspectors. It is thus of great interest to railroad companies to enhance the current manual inspection process using machine vision technology for more efficient, effective and objective inspections. It also helps them lower maintenance costs and increase track capacity.

This paper describes our recent engagement with a railroad company to develop a real-time automatic vision-based rail inspection system. In particular, by taking into account the lack of available technology and severity of defects, we jointly identified the following two key tasks where our technologies can provide the most value.

1) *Tie-level exception detection*, which refers to the detection of any shifted or spread anchors, raised spikes or deadhead spikes.
2) *Compliance-level exception detection*, which refers to the scenario where there are more than 15% of ties having abnormal anchor patterns, or there are more than 25% ties having abnormal spiking patterns, within a 100-foot track segment. Operating trains over track having non-compliant spiking or anchor patterns could potentially lead to derailment.

Manuscript received on xxx. All authors were with IBM T.J. Watson Research Center, NY USA, when this work was conducted. E-mail: {yingli,sharat}@us.ibm.com.

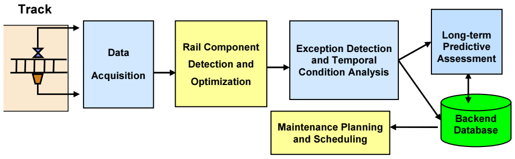

Track
Data Acquisition
Rail Component Detection and Optimization
Exception Detection and Temporal Condition Analysis
Long-term Predictive Assessment
Backend Database
Maintenance Planning and Scheduling

Fig. 2. Overall architecture of the proposed rail inspection system.

A. *Proposed Solution*

Figure 2 shows the overall architecture of our solution. Specifically, the *data acquisition module* is responsible for capturing videos using multiple cameras that are mounted on a moving track inspection vehicle. It also records positioning and speed information and synchronizes them with the video. Various image and video analytics are then launched by the *rail component detection and optimization module* to detect important rail components such as tie plates, ties, spikes and anchors. Next, exceptions at both tie-level and compliance-level which warrant immediate reporting, are identified in the *exception detection module*. Such exception information is then stored and used to guide any *maintenance planning and scheduling*. Finally, a comparative and trend analysis of track component condition is performed in the *long-term predictive assessment module*. This helps lead to more informed preventative maintenance strategies and a greater understanding of track structure degradation and failure modes.

Due to limited space, this paper will mainly focus on the anchor-related component detection and exception recognition, as anchor is a very important rail component. Fig. 3 shows the block diagram of the overall data process. Specifically, given four video streams captured by cameras focusing on four different views of the rails, namely, *left field view*, *left gauge view*, *right gauge view* and *right field view* (please refer to Fig. 4 for the view illustration and Fig. 14 for images of different views), we first detect all necessary components from each of them by applying various image and video analytics. Note that as the heads of anchors only appear in gauge views, anchors are only detected from the gauge view streams; yet both tie and tie plate will be detected from all four video streams. Next, we integrate the evidence from multiple information sources including cameras, GPS and DMI, and apply a global optimization approach to further improve the component detection accuracy. Both cross-object spatial constraint as enforced by the sequential structure of rail tracks, and cross-frame and cross-view constraints in camera streams are applied during this optimization process. Finally, anchor conditions are assessed and anchor pattern exceptions are detected, with the assistance of geo-reference data.

B. *Related Work*

Applying machine vision technology to assist rail track inspection has attracted much interest from the industry. So far, various systems have been proposed, prototyped, and even applied for various specific tasks. Examples include the $VisiRail^{TM}$ Joint Bar Inspection System, developed by ENSCO with high-resolution scan line cameras and laser sensors [1]; the AURORA system, developed by Georgetown Rail, for inspecting wood ties, rail seat abrasion, tie plates, anchors and spikes [2] (yet no technical details or performance report are available about this system); the system developed by MERMEC Group, for detecting track surface defects with high-speed line-scan cameras [3], and the TrackVue system, developed by RailVision, for measuring rail wear, track gauge, curvature, rail cant and vegetation cover using an array of cameras and laser equipment [4].

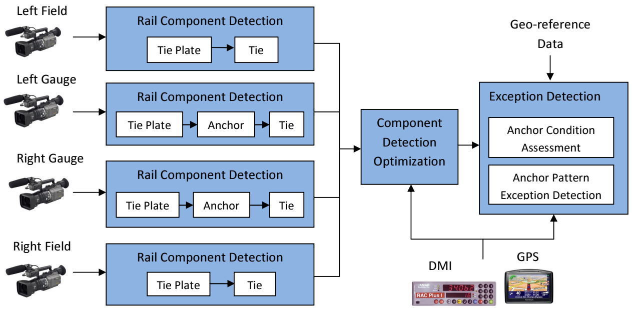


Fig. 3. Block diagram of the overall data process.

There are however, not many reported efforts on detecting rail fastener components including anchors. In [5], the authors applied some image processing approach to inspect elastic rail clips. A recognition rate of 77% was reported for broken clips on concrete track. Similar efforts were also reported in [6] for finding broken and new clips using edge and color information. The two most related work are perhaps from [7] and [8]. Specifically, Babenko proposed to detect steel fasteners, e-clips and fast clips, which are common on concrete ties, using SIFT features and a correlation-based matching approach [7]. Experiments conducted on 2436 fasteners have achieved around 90% detection rate. However, that work did not address the detection of other types of rail fasteners. Another important work is from UIUC, which developed a track inspection system to detect tie plates, spikes and specific types of anchors in both lateral and over-the-rail views of the track [8], [9]. Color, edge and Gabor features were applied along with some heuristics to accomplish the task. Nevertheless, the cart on which the cameras were mounted was hand-pushed, thus traveling at a very low speed. Consequently, such prototype system is not practical for a real deployment. Moreover, the proposed algorithms have not been validated by any quantitative experiments. In our previous work [10], detections on various rail components including tie plate, spike, spike hole and anchors, were described. Nevertheless, no global optimization was performed to improve detection accuracy by integrating multiple information sources. It also did not discuss anchor exception detection.

Anchor is a very vital type of rail fastener component, consequently, assessing its condition and detecting any exceptions caused by non-functioning anchors (such as shifted, spread or even missing) are very critical to maintain railroad safety. To our best knowledge, the system proposed in this work is the first one to address these important issues as well as to offer practical solutions.

For the rest of paper, we present the camera setup in Section

II, and elaborate on the detection and optimization of various rail components in Section III. In Section IV, we discuss the anchor condition assessment as well as the compliance-level anchor pattern exception detection. Performance of the proposed system achieved at a real-time field test is reported in Section V. We conclude the paper in Section VI with a layout of future work.

II. Camera Setup

The cameras chosen for this project are Point Grey Dragonfly2, which provide many flexibilities in terms of image resolution, frame rate, color mode and the option of using region-of-interest. Four cameras are used in total, imaging lateral views of the gauge and field sides of both rails. We have chosen the field of view to be 24 inches to obtain 50% overlap of images when traveling at 10 miles per hour, thus ensuring total coverage of the track elements we are inspecting.

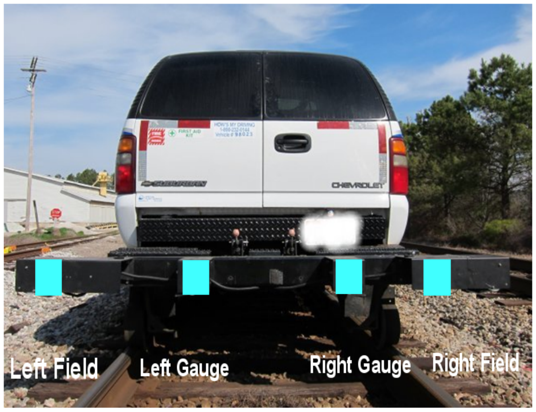


Fig. 4. The four cameras (indicated as rectangles in blue) mounted on a hi-rail inspection vehicle, along with their field of view.

In Fig. 4, the four cameras, which are enclosed within steel channels for rigidity and weatherproofness, are shown as blue rectangles. All cameras are mounted on the rear bumper of a hi-rail truck and connected to the same FireWire bus, which controls the time synchronization between them with high accuracy. When the truck travels on the rail, the four captured video streams are first compressed, then written to the computer, along with GPS and DMI information. The computer we used is a x3650 Mark III with 12 Xeon processors (of 3.0 GHz), with each processor having 2 or 4GB of RAM. The computer is placed along with a UPS and an inverter right inside the truck. We have chosen to use an image resolution of 640 × 400 with 12 bits of monochrome intensity per pixel, and a frame rate of 20 FPS. With such camera setup, at each time instant, each side of each tie plate will be seen by only one camera, and the anchors will be seen only by two gauge view cameras.

III. Rail Component Detection

Below we describe the detailed approaches on detecting rail components including tie plate, tie and anchor.

A. Tie Plate Detection

Tie plate detection is the first step in our detection pipeline, since it provides information to define the regions of interest (ROI), in which other components can be located.

In our current imaging setup, the rail always occupies the upper portion of the image, and presents a very distinct horizontal dividing line from the rest (as shown in Fig. 1 (b)). On the other hand, when a tie plate is present, its bottom edge would present another approximately horizontal line. This observations naturally inspire us to use a Hough transform to detect these two lines. One such detection example is shown in Fig. 5 (a).

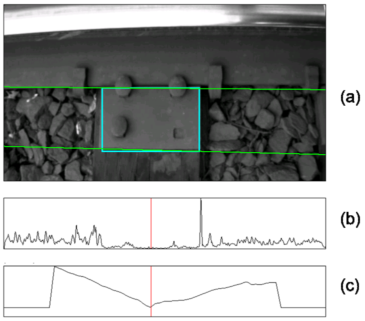


Fig. 5. Illustration of the tie plate detection process.

We then find the tie plate's vertical edges by distinguishing its region from the ballast area, as follows.

1) For the image region between the two horizontal lines, we first compute its edge map using the Sobel operator, then sum up the edge magnitude for each column. One such magnitude map is plotted in Fig. 5 (b).
2) For each column, sum up all magnitudes within a window that is centered on it. The window size approximately equals the width of a tie plate. Fig. 5 (c) shows the summation result, from which we see that there is a distinct local minimum, which exactly corresponds to the midpoint of the tie plate (indicated by the red line).
3) Find the minimum in the above plot, and derive the tie plate's left and right edges based on the window size. The final localization result is shown in Fig. 5 (a), indicated by the rectangle (in cyan).

B. Tie Detection

The key to detecting a tie is to localize its vertical edges, so our first step is to perform a Hough transform to detect near-vertical lines. To speed up the detection, we only apply the Hough transform in close proximity to the vertical edges of the detected tie plates in the frame. But if no tie plate was detected in the frame, we search the entire frame.

Next, considering the fact that for a real tie edge, its immediate left and right regions would present very different textures, as one contains wood with relatively smooth surface (vertically) and the other contains ballast of irregular surface, we measure the likelihood of *being-a-tie-edge* for each vertical line (found by the Hough transform) in terms of pixel intensity variation. Specifically, the intensities of pixels which lie on the vertical line going through the ballast tend to vary much due to its changing brightness. In contrast, they vary very little for the line going through the tie due to its smoothness. To measure such likelihood for a line L_0 (denote it as el_0), we first define two vertical lines L_1 and L_2 that are 5 pixels to its left and right, respectively. Then for line L_i $(i = 1, 2)$, we measure the intensity differences (absolute value) of pairs of pixels which are 20 pixels apart vertically, and sum them over the whole length of the line. We denote such measurement as id_i. Note that the parameter values used here (*i.e.* 5 and 20) are empirically determined. Now, we calculate el_0 as $id_1 - id_2$ where, if it is positive, we consider L_0 as a candidate for the left tie edge, otherwise, a candidate for the right edge. Moreover, the absolute value of el_0 indicates its edge strength or the likelihood of its being-a-tie-edge. Fig. 6 shows an example of such edge candidates.

Once we obtain a set of candidates for both left and right tie edges within a frame, we need to pair them to define the tie. To accomplish this, each pair is scored based on both its edges' strengths and the spacing between them (since ties generally have a fixed width). Non-maximal suppression is then applied to pick the pair with the highest score among overlapping left-right candidate pairs.

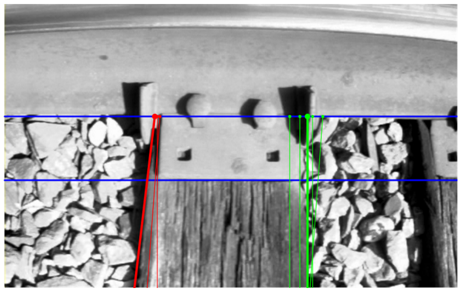


Fig. 6. Candidates of left vertical edges (in red) and right vertical edges (in green) of a tie. The thicker lines indicate the finally identified edges.

C. Anchor Detection

While an anchor can be seen from both gauge and field sides, the head of anchor resides in the gauge side, which will be our detection focus due to its larger size. Also, since anchors are supposed to be close to tie plates, searching anchors within the entire image region would be very inefficient. It would also increase false alarms as well. Consequently, we have defined an image ROI as its search region, particularly, the horizontal image stripe covering the base of the rail where anchors should be installed (see Fig. 1 (b)).

In this work, we developed a learning-based anchor detector based on Adaboost discriminative classifier. Because anchor appearance has a high variability due to the diversities in anchor type, size, shape, camera view, occlusion and lighting condition (see Fig. 7), a single cascade classifier, as introduced in [11], will not be able to capture all in-class variability. Consequently, we have chosen to apply multiple cascade classifiers [12]. Specifically, as depicted in Fig. 8, we first train multiple binary classifiers, each corresponding to a subclass of anchors. Then, during the detection stage, we apply a model-switching mechanism to find the optimal output for each sliding window within the search ROI. Particularly, while we keep all classifiers running simultaneously at all times, at any time instant we only return the detection results from one of them, the one that has the highest number of detections in the last 50 frames.

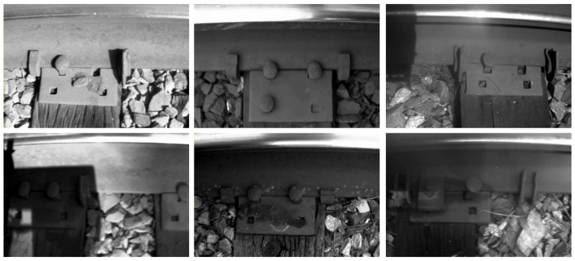


Fig. 7. Anchor appearance has a high variability due to the diversities in anchor type, size, shape, camera view, occlusion and lighting condition.

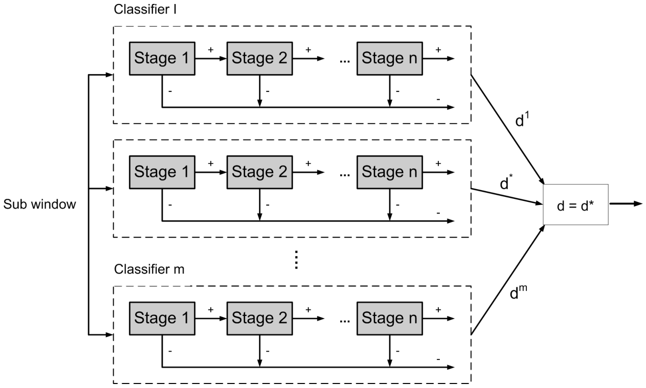


Fig. 8. Illustration of our model-switching mechanism for combining multiple cascade classifiers.

D. Component Detection Optimization

So far, we have been describing the rail component detection from a single camera view. Nevertheless, since we have video streams from four different camera views at any time, and since that each rail component can be seen from one or more views, we could potentially improve the detection performance by leveraging contextual information with cross-frame, cross-view and cross-object constraints. They are discussed below.

- **Cross-frame constraints:** As the same object such as a tie plate, might be observed in more than one consecutive video frame, we apply temporal constraints on detections between neighboring frames for each camera view.
- **Cross-view constraints:** As some rail object such as a tie can be observed by multiple cameras simultaneously, we exploit spatial constraints between camera views to improve detection confidence.
- **Cross-object constraints:** Railroad tracks are sequential structures formed by a sequence of objects, whose installation conforms to specific designs. Therefore, there are certain spatial constraints across rail objects. For instance, the spacing between consecutive ties in a railroad track is close to a constant. Such a constraint makes the detection

of rail components different from that of objects in a general context [13], [14].

Following these ideas, we have developed a global optimization approach to improve object detections for each frame in the context of other detections in neighboring frames and other camera views. Specifically, the inputs to the approach include: 1) detections of a specific type of rail object (*e.g.* tie plate) from each frame of each camera view, along with their confidence scores; 2) DMI data; and 3) GPS data. We detail this approach below.

1) Optimization Formulation: Assume that we have 4 streams of object states ($\{S_1, .., S_4\}$), where each stream contains the detections of a specific type of rail object from one camera view for a duration of T. For illustration purpose, let us use S_k to indicate a sequence of object states $\left\{s_k^1, .., s_k^T\right\}$ over duration T, where k is the index of camera view. Here, an object state indicates the bounding box of a rail object detected in Section III A-C, consisting of location and size. Without loss of generality, we assume that there is only at most 1 object state per frame, although our approach can be directly applied to the multi-object states case. Fig. 9 illustrates our optimization problem in form of a graph. Here, each column indicates a video frame, each row indicates a camera view, and each round node indicates an object state. Note that the detector may find multiple detections per frame, which results in having multiple states per frame.

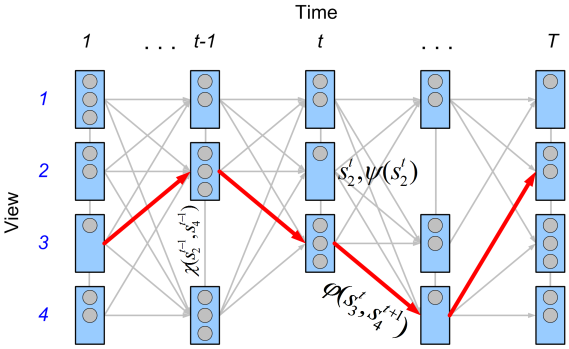


Fig. 9. Graphical representation of our detection optimization problem.

The goal here is to find a path from time 1 to time T, which consists of a set of object states $S^* = \left\{s_*^1, .., s_*^T\right\}$ and optimizes the following energy function:

$$S^* = \arg\max_S E = \prod_t \psi(s_k^t)\phi\left(s_k^t, s_l^{t+1}\right), \qquad (1)$$

where $\psi(s_k^t)$ is the potential function of an object state s_k^t and $\phi\left(s_k^t, s_l^{t+1}\right)$ indicates the cross-frame spatial constraints. s_k^t is the object state at node (k, t), which is initially the input object detection.

A. Applying the cross-view constraint. Let us first model the spatial constraint of different object states across camera views. Two assumptions are applied here: 1) all cameras' calibration parameters are fixed, which is indeed true in our case; and 2) given an object state s_k^t at view k, the state s_l^t at view l follows a Gaussian or Normal distribution. Subsequently, we represent this cross-view constraint as follows:

$$\chi(s_k^t, s_l^t) = \max\left(\begin{array}{ll} \mathcal{N}\left(\left|s_k^t - s_l^t\right|; \theta_{kl}\right) & , \\ \mathcal{N}\left(\left|s_k^t - s_l^t + \epsilon\right|; \theta_{kl}\right) & \end{array}\right) \qquad (2)$$

where $\theta_{kl} = [\mu_{cv}(k,l), \Sigma_{cv}(k,l)]$. Here, μ_{cv} is a 4x4 matrix of mean values and Σ_{cv} is a 4x4 covariance matrix. ϵ indicates the tie spacing constant, which is needed in case that the two object states s_k^t and s_l^t refer to adjacent objects in the sequence. Note that θ and ϵ can all be learnt from labeled training data.

Now, function $\psi(s_k^t)$ in Equation 1 can be defined as:

$$\psi(s_k^t) = f(s_k^t)\prod_{l \neq k} \chi(s_k^t, s_l^t), \qquad (3)$$

where $f(s_k^t)$ is the confidence score of object state s_k^t, which is assigned to each object by respective detection module to indicate the confidence level of detection.

B. Applying the Cross-frame Constraint. Now, let us model the temporal constraints of object states across consecutive frames. For both tie and tie plate detections, we assume that the spacing between consecutive ties in the rail track is relatively a constant. Now, for state s_k^t at frame t and s_l^{t+1} at frame $t+1$, where k and l may indicate different views, they could either refer to the same physical object, or two different yet adjacent physical objects. We represent the cross-frame constraint for either case as follows.

$$\phi\left(s_k^t, s_l^{t+1}\right) = \max\left(\begin{array}{ll} F\left(\left|s_k^t, s_l^{t+1}\right|; \lambda\right) & , \\ F\left(\left|s_k^t, s_l^{t+1} + \epsilon\right|; \lambda\right) & \end{array}\right) \qquad (4)$$

where $\lambda = [\mu_{cf}, \sigma_{cf}, \mu_{cv}, \Sigma_{cv}, \tau_{dmi}]$, and cf and cv indicate cross-frame and cross-view, respectively. $\langle\mu_{cf}, \sigma_{cf}\rangle$ models the Gaussian distribution of the object state at the next frame given its state at previous frame, and τ_{dmi} represents the DMI data. Both μ_{cf} and σ_{cf} can be learnt from labeled training data. F is a function of the distance between two object states s_k^t and s_l^{t+1}, and is defined as follows:

$$F(s_k^t, s_l^{t+1}) = \begin{array}{l} \mathcal{N}\left(s_k^t - s_l^t; \mu_{cv}, \Sigma_{cv}\right) \\ \times\mathcal{N}\left(s_l^t - s_l^{t+1} + \tau_{dmi}; \mu_{cf}, \sigma_{cf}\right) \end{array} \qquad (5)$$

As we can see, F becomes larger when the physical distance between the 2 bounding boxes of s_k^t and s_l^t gets smaller. τ_{dmi} indicates how much an object should have moved from time t to time $t+1$. In a summary, given s_k^t, the observation of an object at time t, and the parameter λ, F measures the confidence score of observing the same object again at time $t+1$ at the location of s_l^{t+1}.

Note that Equation 4 also handles cross-object constraints by using the constant ϵ, in case that the two object states s_k^t and s_l^{t+1} do not indicate the same physical object.

2) Real-time Algorithm: We have developed a real-time algorithm to implement the aforementioned global optimization approach, which can perform at the inspection speed of 16 km/h, with frame rate of 20 fps. At each time instant t, the real-time algorithm will compute the optimal path from time 1 up to the current time point t, given all object states from the beginning up to t. The algorithm is described below.

1) Compute a score for every node in the graph using dynamic programming:

$$x_k^1 = \psi(s_k^1). \tag{6}$$

$$x_k^t = \psi(s_k^t) \max_j \left(x_j^{t-1} \phi \left(s_k^t, s_j^{t-1} \right) \right). \tag{7}$$

2) At each time point t, select the optimal object state s_v^t, where

$$v = \arg\max_k \left(x_k^t \right).$$

3) Use the selected object states to infer/update suboptimal object states in other camera views at time t.
4) If no object detection is found at time t, restart the algorithm at time $t+1$.

3) Batch Algorithm: We have also developed a batch algorithm to compute the optimal path from the beginning (*i.e.* time 1) all the way up to the end (*i.e.* time T) in one shot. Specifically, given a set of object states from time 1 to time T, the batch algorithm works as follows.

1) Compute a score for each node in the graph using dynamic programming (same as the real-time algorithm).
2) For each node, store the predecessor with which it obtains the optimal score.
3) At time T, select the optimal object state.
4) Use the selected object state to infer/update detections in other camera views at time T.
5) Backtrack the stored predecessors at each time point to obtain the full path.

As we see, compared to the real-time algorithm, the batch algorithm takes into account all available detection information from the beginning to end, therefore it tends to achieve a better prediction than the real-time algorithm, which operates in a more greedy fashion.

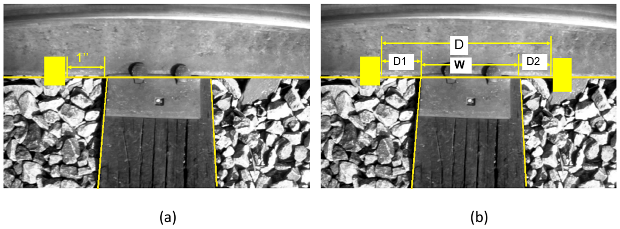
IV. Anchor-Related Exception Detection

There are two levels of anchor-related exceptions. At tie level, the exception refers to any shifted or spread anchors, as both conditions could indicate that the rail at that particular location is running (thus unstable). At compliance level, the exception refers to the scenario where there are more than 15% of ties having abnormal anchor patterns within a 100-foot track segment. Both of them are elaborated below.

A. Anchor Condition Assessment

Figure 10 shows two examples of shifted and spread anchors, respectively, along with the definition. Specifically, an anchor is considered shifted if it is more than 1 inch away from its associated tie horizontally (Fig. 10 (a)). On the other hand, a spread occurs when the horizontal distance between two anchors of the same tie is 4 inches more than the tie width (Fig. 10 (b)), *i.e.* $D1 + D2 = D - W \geq 4$ inches. Consequently, the spread condition of an anchor pair can be readily assessed once we obtain the shift value of each anchor.

Now that both shift and spread conditions are defined in inches, yet the anchor-to-tie distance can only be measured in pixels from the image, we need find a way to reliably convert distances from pixels to inches. Unfortunately, due to the wide angle fisheye distortion, the inch-to-pixel mapping is not uniform for all columns in the image. On the other hand, in the interest of easing the operational process of video inspection, we do not want to introduce complicated camera calibration. What we need is therefore, an approach which will associate the inch-to-pixel mapping with specific (horizontal) position within an image.



(a) (b)

Fig. 10. Example and definition of: (a) shifted anchor, and (b) spread anchor.

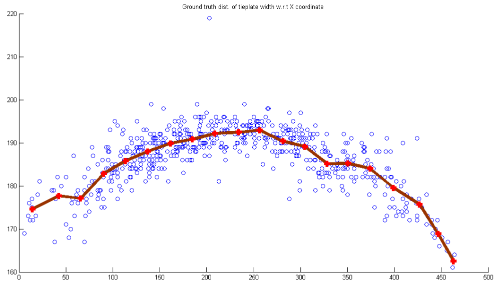


Fig. 11. Plot of tie plates' X-positions against their widths (in pixels), which is fitted by a curve (in dark red).

To accomplish this, we annotated roughly 3000 tie plates detected in videos from both gauge-view cameras, and plotted their X-positions against their width (measured in pixels) in Fig. 11. There we see that the tie plate's measured width varies with its horizontal position in the image due to the fisheye distortion. The closer the tie plate is to the image boundaries, the narrower it appears to be. To ease the mapping, we first quantize the tie plate X-coordinates into 20 different bins, then fit a conversion curve by performing an averaging operation for each bin, followed by a linear interpolation. The conversion curve is shown in Fig. 11 (in dark red), which roughly approximates a quadratic function. The red crosses indicate the average tie plate width of each bin. Now, considering that a physical tie plate's width is 7.5 inches, given an image position x, we can calculate its specific inch-to-pixel mapping as: $\frac{7.5}{y(x)}$, where $y(x)$ indicates the projected tie plate width given x, following the conversion curve.

Once the anchor-to-tie distance is obtained in pixels, we can calculate its specific inch-to-pixel mapping based on its X-coordinate, and subsequently convert such distance into inches. The anchor condition can then be properly assessed.

B. Compliance-level Exception Detection

To ensure safety, railroad companies require different anchor patterns for different rail types. For instance, for jointed rail there should be 8 boxed ties per 39-foot rail segment, while for continuously welded rail, only alternate boxed ties are required. By boxed tie, we mean a tie with all 4 anchors in good condition, as illustrated in Fig. 1 (a). To obtain these pattern requirements for real railroad tracks, we used the geo-reference data provided by the partnering railroad company. Specifically, for each geo-location indexed by milepost and footage, or GPS latitude and longitude, it tells us the required anchor pattern for that particular spot. We then use this information to detect the compliance-level exceptions for any 100-foot track segment.

In particular, we first count the total number of boxed ties contained within a 100-foot segment which is measured based on GPS information, and denote it by C. Then we compare the tie count C with the required number of boxed ties R enforced by the targeted anchor pattern. If C is smaller than 85% of R, *i.e.* $(R-C)/R >= 15\%$, then a compliance-level exception is declared.

Note that since we process the video frame by frame, we need to constantly check if we have covered a 100-foot track segment. If yes, we perform the number comparison and identify any exception; otherwise, we update the counter C and read in the next tie. A confidence score will also be measured for every detected exception based on the confidence of anchor detection. The complete algorithm is illustrated in Fig. 12. Note that we continuously perform this inspection for every possible 100-foot segment, thus when a new tie moves into this 100-foot segment window, the earliest tie will move out and the inspection will be kicked off again.

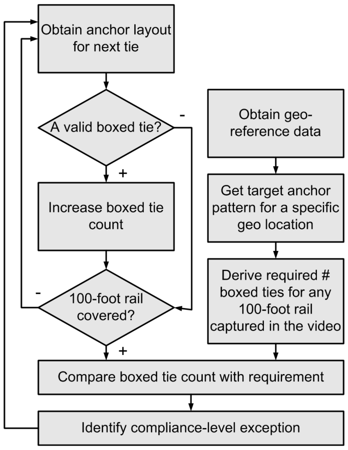
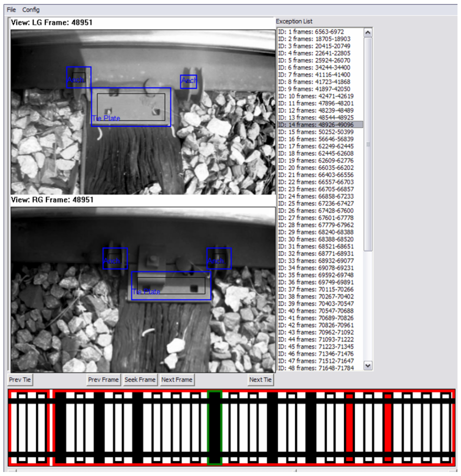

Obtain anchor layout for next tie
A valid boxed tie?
-
+
Increase boxed tie count
100-foot rail covered?
-
+
Obtain geo-reference data
Get target anchor pattern for a specific geo location
Derive required # boxed ties for any 100-foot rail captured in the video
Compare boxed tie count with requirement
Identify compliance-level exception

Fig. 12. Flow chart of detecting compliance-level anchor pattern exception.

We have developed an exception visualization tool to visualize or validate each detected exception at both tie and compliance level. As shown in Fig. 13, images from both gauge-view cameras are shown side by side, along with the list of detected compliance-level exceptions on the right. The object detections for each frame are overlayed on top of the images, along with indications of shifted or spread anchors, if any. The bottom portion of the GUI displays the detected ties within the vicinity of the current frame, where a black tie indicates a boxed tie with all 4 anchors, a red tie indicates a tie with 1-3 anchors, and a white tie with 0 anchor.



```
ID: 1 frames: 6563-6972
ID: 2 frames: 18705-18903
ID: 3 frames: 20415-20749
ID: 4 frames: 22641-22805
ID: 5 frames: 25924-26070
ID: 6 frames: 34244-34400
ID: 7 frames: 41116-41400
ID: 8 frames: 41723-41868
ID: 9 frames: 41897-42050
ID: 10 frames: 42471-42619
ID: 11 frames: 47896-48201
ID: 12 frames: 48239-48489
ID: 13 frames: 48544-48925
ID: 14 frames: 48926-49096
ID: 15 frames: 50252-50399
ID: 16 frames: 56646-56839
ID: 17 frames: 62249-62445
ID: 18 frames: 62445-62608
ID: 19 frames: 62609-62776
ID: 20 frames: 66035-66202
ID: 21 frames: 66403-66556
ID: 22 frames: 66557-66703
ID: 23 frames: 66705-66857
ID: 24 frames: 66858-67233
ID: 25 frames: 67236-67427
ID: 26 frames: 67428-67600
ID: 27 frames: 67601-67778
ID: 28 frames: 67779-67962
ID: 29 frames: 68240-68388
ID: 30 frames: 68388-68520
ID: 31 frames: 68521-68651
ID: 32 frames: 68771-68931
ID: 33 frames: 68932-69077
ID: 34 frames: 69078-69231
ID: 35 frames: 69592-69748
ID: 36 frames: 69749-69891
ID: 37 frames: 70115-70266
ID: 38 frames: 70267-70402
ID: 39 frames: 70403-70547
ID: 40 frames: 70547-70688
ID: 41 frames: 70689-70826
ID: 42 frames: 70826-70961
ID: 43 frames: 70962-71092
ID: 44 frames: 71093-71222
ID: 45 frames: 71223-71345
ID: 46 frames: 71346-71476
ID: 47 frames: 71512-71647
ID: 48 frames: 71648-71784
```

Fig. 13. Exception visualization tool.

V. Performance Evaluation

A. Data Collection

Our partnering company has dedicated a track inspection vehicle (as shown in Fig. 4) to this project from the very beginning. Since then, we have installed our cameras, computer, GPS and DMI in the vehicle, and made trips to some nearby tracks to adjust the imaging setup and test the video capture tool. Many issues were discovered and resolved during this test phase including motion blur, under- and over-exposed images, shadow effect, as well as the challenge of high data volume. Once all these problems were addressed, we made additional trips to main line tracks which generally experience the highest traffic densities. For each capture session, the hi-rail vehicle was run between 0.4 km to 2.5 km at up to 16 km/h, then it was backed up to let us capture another session. We were able to capture videos while driving in reverse.

We have thus collected video data under different weather conditions (bright sun, partly cloudy and overcast), at different times of day (morning, noon and afternoon) and on different days, as well as with different track alignments (tangent/straight and curved). The collected data also imaged a large variety of fastener types: regular tie plates, mountain tie plates (extra large), different spiking and anchor patterns, and anchors in different conditions. Consequently, we were able to test our system using a very rich and diverse data set.

Towards the end of project, we conducted a 3-day real-time field test by running an inspection on mainline tracks selected by the partnering company. Specifically, the test consisted of inspecting 3 segments of 1.6-km track sections for presence and absence of tie plates, ties, anchors, anchor shifts and spreads, as well as compliance-level anchor pattern exceptions. To fully test the capability of the system, these tests were conducted on different days and at different times of the day, with changing lighting conditions (cloud cover and sun angle). In total, this field test covered 8820 ties, 35280 anchors and 17640 tie plates. Note that since a tie plate is visible in both gauge and field view, it is equivalent to 2 tie plate objects that the system shall detect. Performance evaluation on this field test data set will be mainly reported here.

B. Rail Component Detection

Let us first define the concept of **correct match** between an annotation and a detection. Specifically, denote their bounding boxes as A_{bb} and D_{bb}, respectively, we require a correct match to meet three criteria as stated in Table I.

TABLE I
THREE CRITERIA TO DEFINE A CORRECT MATCH.

1. The intersection ratio $\frac{A_{bb} \cap D_{bb}}{A_{bb}} \geq 60\%$,
2. The non-intersection ratio $\frac{D_{bb} - (A_{bb} \cap D_{bb})}{A_{bb}} \leq 30\%$,
3. The entity types of the two objects match with each other.

The detection performance is measured by *precision* and *recall*, where precision is the fraction of detected objects that are correct, while recall is the fraction of matched objects that are detected.

1) Tie Plate Detection: The low in-class variability of tie plates make tie plate detection the most reliable part of our system. As tabulated in Table II, we have achieved very good performance with 100% recall and 99.3% precision rates, on the field test data. The F_1 score, which is defined as the harmonic mean of precision and recall (*i.e.* $F_1 = 2 \times (\frac{precision \times recall}{precision + recall})$), is also calculated.

TABLE II
SYSTEM PERFORMANCE EVALUATION ON RAIL COMPONENT DETECTION USING REAL-TIME FIELD TEST DATA.

Component	Groundtruth	Precision	Recall	F_1 Score
Tie plate	35280	99.3%	100%	99.6%
Tie	8820	88.2%	82.3%	85.1%
Anchor	35280	96.5%	96.7%	96.6%

We would also like to report a comparative performance study on tie plate detection, in terms of using and not using the global optimization approach as proposed in Section III-D. Since the railroad tracks that we covered in the field test were in very good condition, the benefit of applying component optimization is not obvious. Consequently, we have chosen a different test data set which includes challenging issues such as heavy occlusion (due to debris) and constantly-present shadows (see Fig. 14 for some examples). To study the performance, we annotated tie plates on 6000 video frames (including all 4 camera views).

TABLE III
COMPARATIVE PERFORMANCE EVALUATION OF TIE PLATE DETECTION USING TEST DATA WITH HEAVY OCCLUSIONS AND SHADOWS.

Approach	Precision	Recall	F_1 Score
Batch Algorithm	**84%**	**92%**	**88%**
Real-time Algorithm	79%	92%	85%
Single-view detection [15]	83%	84%	84%

Table III tabulates the performances of three different approaches, where we see that both of our batch and real-time optimization algorithms have outperformed our previous approach that only relies on visual cues from a single camera view [15]. Moreover, the batch algorithm has achieved the best performance, with 84% precision and 92% recall rates.

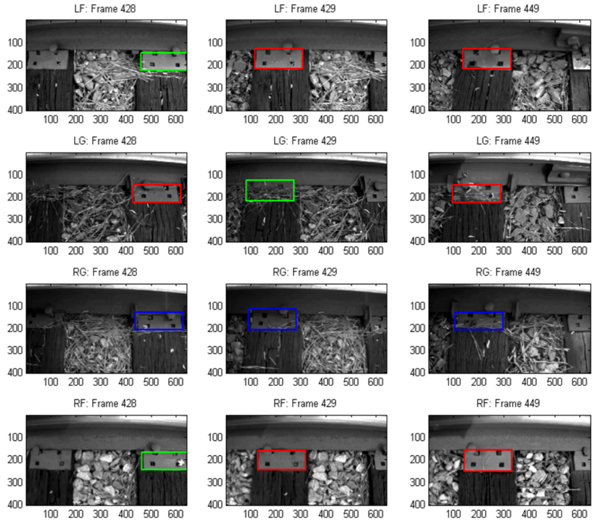


Fig. 14. Selected frames showing all 4 camera views, with the original detections in red, corrected detections in blue and inserted detections in green.

Fig. 14 illustrates some detection examples which show that our optimization algorithm has successfully inserted missing detections (LF of frame 428 and LG of frame 429) and corrected wrong detections (RG of all three frames). As we can see, due to the heavy debris, sometimes the tie plates are completely occluded in certain camera views (*e.g.* the LG view of frame 429), consequently, it is literally impossible to detect them based on the visual cues of a single camera view. In contrast, by leveraging the temporal and spatial constraints of the object *w.r.t.* nearby detections, our approach can effectively predict the correct object location in spite of insufficient visual information.

2) Tie Detection: Tie detection is mostly needed for anchor condition assessment as both shift and spread conditions are defined based on anchor-to-tie distance. Referring to Table II, we see that the tie detector is not as reliable as the tie plate detector, achieving 82.3% recall and 88.2% precision rates.

TABLE IV
COMPARATIVE PERFORMANCE EVALUATION OF ANCHOR DETECTION USING REAL-TIME FIELD TEST DATA.

Approach	Precision	Recall
Multiple classifiers with model-switching	96.5%	96.7%
Standard single Adaboost classifier	95.6%	93%
Edge-based approach [10]	83.3%	91%

3) Anchor Detection: We have applied three different approaches for anchor detection, specifically, the proposed one using multiple cascade classifiers with the model-switching mechanism, the standard single Adaboost classifier and the edge-based anchor detector proposed in [10]. Their performances are reported in Table IV. As we can see, the anchor detector with multiple classifiers has achieved the best result with both precision and recall being above 96%.

From gap analysis, we observe that anchor false positives are mainly caused by sharp shadows or debris on the rail, while false negatives (*i.e.* missed detections) are mainly caused by either extreme lighting conditions (*e.g.* too dark or too bright), or occlusions. Fig. 15 shows some examples of false positives and false negatives.

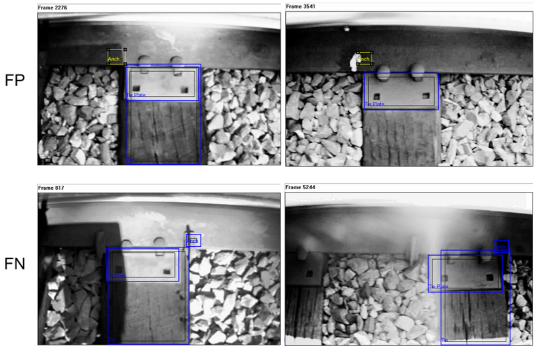


Fig. 15. Examples of anchor false positives (top row) and anchor false negatives (bottom row).

C. Anchor Condition Assessment

As described in Section IV-A, an anchor is considered shifted if it is more than $\tau = 1$ inch away from the associated tie. Nevertheless in practice, τ could take various slightly different values. Applying different values would result in detecting anchor shifts of different severity. Consequently, it would be helpful if we grant users the flexibility of deciding τ's value, based on their own individual needs.

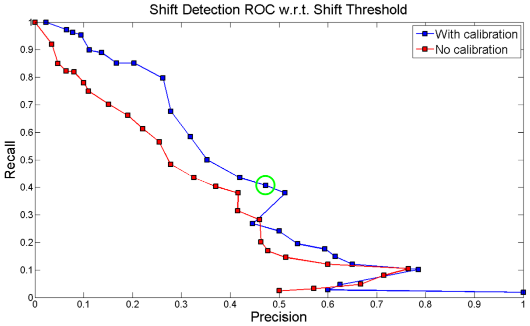


Fig. 16. Anchor shift detection ROC *w.r.t.* the shift threshold τ, with and without applying the inch-to-pixel calibration. The curves are plotted with τ ranging from 0 to 2.5 inches.

For this reason, we have evaluated the anchor shift detection performance with different values of τ, based on which we generated an ROC (Receiver Operating Characteristic) curve by computing an ROC point for each specific value. Fig. 16 plots two such ROC curves, with one applying the proposed/customized inch-to-pixel calibration and the other not (or rather, a fixed inch-to-pixel mapping). The value of τ ranges from 0 to 2.5 inches in this case. For each ROC point, we use F_1 score to indicate its overall performance. The point that has the best F_1 score is highlighted by a circle in the figure. As we see, this point has a precision of 47%, a recall of 40% and an F_1 score of 44%. The corresponding τ used for this point is 1.3 inches.

While this figure shows that by applying the customized inch-to-pixel mapping, the shift detection performance has been improved, we also admit that our current shift detection performance requires further improvement. From gap analysis, we realize that the low precision and recall rates are mainly caused by imprecise localization of ties and anchors. To achieve a reliable shift detector, we need to keep improving the localization accuracy.

As for the spread anchor detection, neither the real-time field test data nor our offline test data contain any instances of this condition; consequently, we can not report the performance here. Nevertheless, as we illustrated earlier, the spread condition of an anchor pair can be readily assessed once we obtain the shift value of each anchor. Consequently, its detection performance can be theoretically derived from that of anchor shift detection.

D. Compliance-level Anchor Pattern Exception Detection

Since a compliance-level anchor pattern exception severely affects the railroad safety, failure to detect it can potentially lead to grave consequences. On the other hand, if the system produces too many false positives, it would be labor-intensive for railroad companies to send crews to visually inspect all reported track segments. Consequently, the partnering company is requesting an anchor pattern exception detector with a high detection rate and a low false positive rate. More specifically, a 95% detection rate with not more than 1 false positive per inspection hour.

Because the 3-mile track of the real-time field test did not contain any anchor pattern exceptions, we performed such an evaluation on an 1-hour video captured from a different track. In this video, there were 3 true anchor pattern exceptions, and our system was able to detect all of them, thus achieving a 100% detection rate. On the other hand, it also generated 3 false positives during the 1 hour period. While this is still not up to the bar set by the partnering company, it is nevertheless quite encouraging. Note that the numbers of true positives and false positives are theoretically independent of each other, thus the absolute number of false positives is more of our concern.

VI. Conclusion

This paper describes our recent engagement with a railroad company to develop a real-time automatic vision-based rail inspection system. Specifically, the system is able to robustly detect important rail components with high accuracy and efficiency based on visual, location, DMI and contextual information. We have further discussed anchor exception detection at both tie and compliance levels. Quantitative analysis performed on a large video data set captured with different track and lighting conditions has demonstrated very encouraging performance.

The main challenge for us in the near future is to handle scenarios where heavy shadows and light overexposure exist in the videos. Also, we believe that our current tie detection approach needs to be further improved, and the global component optimization approach needs to be evaluated on other rail objects besides the tie plate. Thirdly, we shall conduct more extensive testing covering longer railroad tracks with varying defect conditions. Inspection for other rail objects such as spikes, spike holes and joint bar need to be developed as well. Finally, we need enhance our algorithms with potentially modified imaging system to accommodate a faster and more desirable inspection speed (e.g. 40 km/h). Controlled illumination of the rail infrastructure will also be explored to avoid ambient lighting artifacts.

Finally, we would like to point out that: 1) the object detection and optimization approaches that we proposed here can be either readily applied or with minor tunings to other rail fastening systems; 2) while the vision algorithms that we developed are finely tuned towards rail track structure and components, the encouraging results we achieved have demonstrated the applicability of machine vision technology to real applications in the general transportation domain (such as the advanced driver assistance system (ADAS)).

REFERENCES

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

[2] G. Rail Equipment Company, "AURORA: The future is finall here," *http://www.georgetownrail.com/aurora.php*, Accessed in November 2010.

[3] M. Group, "Track surface inspection system - TSIS," *http://www.mermecgroup.com/diagnostics/track-inspection/62/1/track-surface-inspection.php*, Accessed in November 2010.

[4] RailVision, "Trackvue," *http://www.rail-vision.co.uk/trackvue*, Accessed in November 2010.

[5] H. Hsieh, N. Chen, and C. Liao, "Visual recognition system of elastic rail clips for mass rapid transit systems," *Proceedings of the ASME/IEEE Joint Rail Conference and Internal Combustion Engine Spring Technical Conference*, 2007.

[6] M. Singh, S. Singh, J. Jaiswal, and J. Hempshall, "Autonomous rail track inspection using vision based system," *IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety*, 2006.

[7] P. Babenko, "Visual inspection of railroad tracks," *http://server.cs.ucf.edu/ vision/papers/theses/Babenko_Pavel.pdf*, 2009.

[8] J. Edwards, J. Hart, S. Sawadisavi, E. Resendiz, C. Barkan, and N. Ahuja, "Advancements in railroad track inspection using machine-vision technology," *AREMA Conference Proceedings on American Railway and Maintenance of Way Association*, 2009.

[9] S. Sawadisavi, J. Edwards, E. Resendiz, J. Hart, C. Barkan, and N. Ahuja, "Machine-vision inspection of railroad track," *AREMA Conference Proceedings on American Railway and Maintenance of Way Association*, 2008.

[10] Y. Li, C. Otto, N. Haas, Y. Fujiki, and S. Pankanti, "Component-based track inspection using machine-vision technology," *ACM International Conference on Multimedia Retrieval*, 2011.

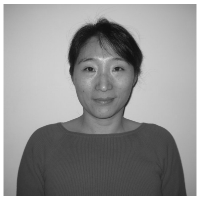
[11] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *CVPR*, 2001.

[12] Z. Zhang, L. Zhu, and S. Li, "Real-time multi-view face detection," in *International Conference on Automatic Face and Gesture Recognition*, 2002.

[13] C. Desai, D. Ramanan, and C. Fowlkes, "Discriminative models for multi-class object layout," in *International Conference on Computer Vision*, 2009.

[14] D. Hoiem, A. Efros, and M. Hebert, "Putting objects in perspective," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

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



Dr. Ying Li has been a Research Staff Member at IBM T.J. Watson Research Center since 2003. She was with the Exploratory Computer Vision Group before she joined Service research in 2011. Dr. Li's research interests include audiovisual content analysis and management, computer vision, business analytics and service management. Dr. Li has authored or co-authored 30 patents and around 50 peer-reviewed conference and journal papers, as well as 5 book and book chapters on various multimedia and computer vision related topics. Dr. Li received her M.S. and Ph.D. degrees from University of Southern California in 2001 and 2003, respectively. She is a Senior Member of IEEE.

Dr. Hoang Trinh received his Ph.D. degree in Computer Science from the Toyota Technological Institute at Chicago with main research focus on Computer Vision and Machine Learning. He subsequently worked at IBM T.J. Watson Research Center as a postdoc researcher. Dr. Trinh has recently joined UtopiaCompression as a Senior R&D scientist, working on several innovative research projects funded by ARMY and ONR. Dr. Trinh has substantial knowledge and expertise in object detection, segmentation and tracking, scene geometry reconstruction, and human activity recognition.

Norman Haas received his M.S. in Computer Science from Stanford University in 1978. He has worked at SRI International and Symantec. He worked at the IBM T.J. Watson Research Center from 1984 until 2013 on a variety of projects involving object recognition for retail checkout, automotive vision, and railroad track inspection. He is now retired.

Charles Otto is currently pursuing his Ph.D in Computer Science at Michigan State University. He has interned at IBM T.J. Watson Research Center for 3 years, and was involved in various projects in Computer Vision including work on object recognition for retail self-checkout, automotive vision, and railroad track inspection.

Dr. Sharath Pankanti is a Research Staff Member in Software Research Department at the T.J. Watson Research Center. He received Ph.D. degree in Computer Science from the Michigan State University. He is manager of the Exploratory Computer Vision Group where he has led a number of safety, productivity, and security focused projects. He is coauthor of more than 80 inventions and more than 125 technical papers. Dr. Pankanti has co-edited the first comprehensive book on biometrics, "Biometrics: Personal Identification" Kluwer, 1999 and co-authored, "A Guide to Biometrics", Springer 2004 which is being used in many undergraduate and graduate biometrics curricula. He is a member of ACM and fellow of IEEE.