Enhanced Rail Component Detection and Consolidation for Rail Track Inspection

Hoang Trinh Norman Haas Ying Li Charles Otto Sharath Pankanti
IBM T. J. Watson Research Center
19 Skylikne Dr, Hawthorne, NY 10532

Abstract

For safety purposes, railroad tracks need to be inspected on a regular basis for physical defects or design non-compliances. Such track defects and non-compliances, if not detected in a timely manner, may eventually lead to grave consequences such as train derailments.

In this paper, we present a real-time automatic vision-based rail inspection system, with main focus on anchors - an important rail component type, and anchor-related rail defects, or exceptions. Our system robustly detects important rail components including ties, tie plates, anchors with high accuracy and efficiency. Detected objects are then consolidated across video frames and across camera views to map to physical rail objects, by combining the video data streams from all camera views with GPS information and speed information from the distance measuring instrument (DMI). After these rail components are detected and consolidated, further data integration and analysis is followed to detect sequence-level track defects, or exceptions. Quantitative analysis performed on a real online field test conducted on different track conditions demonstrates that our system achieves very promising performance in terms of rail component detection, anchor condition assessment, and compliance-level exception detection. We also show that our system outperforms another advanced rail inspection system in anchor detection.

1. Introduction and Related Work

According to recent safety statistics published by the Federal Railroad Administration (FRA), the total impact of damage caused by all reported derailment accidents in the US amount to hundreds of millions of dollars annually, more than 10% of which were due to track problems. For safety and efficiency reason, railroad track inspection for physical defects and irregularities is required to be performed in a regular basis in order to maintain a high standard of track condition.

Rail inspection generally includes a wide variety of specific tasks, ranging from locating and evaluating the condition of different rail components (tie plates, anchors, joint bars, etc) to monitoring rail surfaces, alignments and curvatures, to detecting sequence-level track design non-compliances. It is very error-prone, time-consuming and expensive for railroad companies to use human inspectors to perform those tasks manually, especially for long-term and large scale deployment. It is therefore of their great interest to enhance, or even replace the current manual inspection process by automatic rail inspection systems using advanced computer technologies.

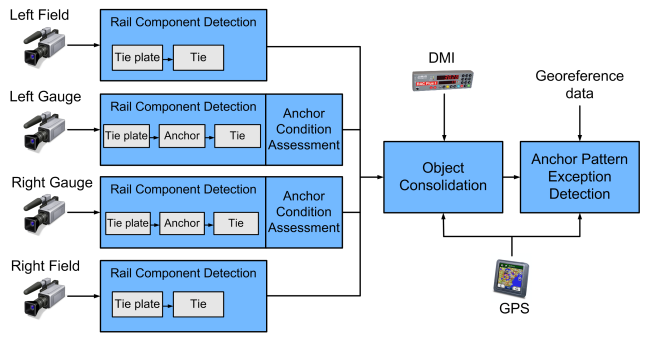


Figure 1. Overall architecture of our rail inspection system.

In this paper, we present a real-time automatic vision-based rail inspection system, with main focus on anchors - an important rail component type, as well as anchor-related rail exceptions. There are two types of anchor-related exceptions: tie-level exception and compliance-level exception. The tie-level exception mainly refers to shifted and spread anchors, while a compliance-level exception is identified when there are more than 15% of ties that have abnormal anchor patterns within a 100-foot track segment. Our system focuses on solving the following problems:

- Detect ties, tie plates and various types of rail anchors.
- Assess the condition of anchors (shift, spread, normal).
- Consolidate detected components across video frames

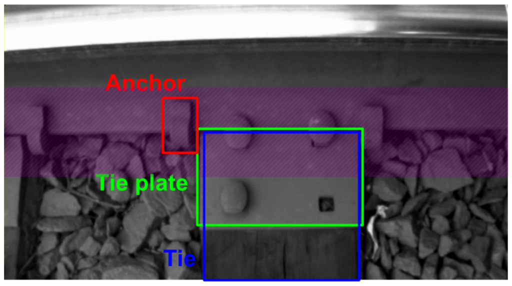


Figure 2. Rail components that we want to detect: Tie plate (in green), tie (in blue), anchor (in red). The purple rectangular region is ROI in which anchors are searched.

and across camera views to map to physical rail objects.

- Detect anchor pattern compliance-level exceptions.

Recently, many machine-vision based automatic systems have been introduced to solve different tasks in rail inspection [10, 7, 1, 2, 3, 4, 5, 9], including our own work [6, 8]. Each of these systems is focused on a different subset of rail inspection problems. For example, [2] proposed an approach to detecting steel fasteners, e-clips and fast clips using SIFT features and a correlation-based matching approach. In [9] and [5], image processing techniques were presented to detect broken rail clips, which are very common on concrete ties. Several commercial systems include the one developed by MERMEC Group to detect track surface defect with high-speed line-scan cameras [7], or by RailVision for measuring rail wear, track gauge, curvature, rail cant and vegetation cover using an array of cameras and laser equipment [10].

Although anchor is a vital type of rail fastener component, very few existing systems have investigated that rail component type [4, 1]. Moreover, no quantitative analysis or performance report in anchor detection are available for such systems. In addition, the cart used in [4] traveled with a much slower speed than our hi-rail vehicle. Besides identifying the presence/absence of anchors, it is equally important to assess the shift and spread conditions of the anchors. A shifted anchor is certainly not functioning as it is supposed to, and therefore is equivalent to being absent. To our knowledge, our system is the only one to address and successfully solve the problem of anchor condition assessment. We also believe that our system is the first to present a solution for the problem of anchor pattern compliance exception detection.

2. Detecting Rail Components

In this section, we describe our work on detecting three vital rail components: tie plate, tie and anchor. Figure 2 shows an example of each type of components in one captured video frame.

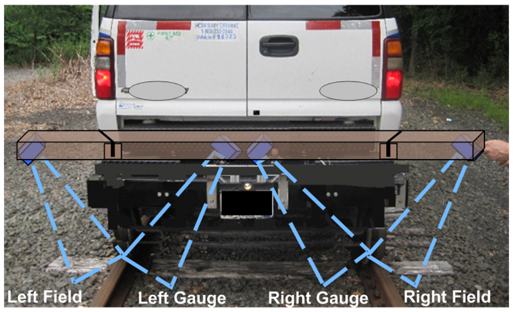


Figure 3. The four cameras (blue) mounted on a hi-rail vehicle and their fields of view. The captured videos have resolution of 640×400 and frame rate 20 FPS.

2.1. Background on Rail Components and Camera Setting

Tie plates are attached to the rail by steel spikes to prevent the rail from latitudinal movements. There are two tie plates associated to each tie, one for each rail; and each tie plate has two sides (gauge side and field side). **Ties** are used as a base to support and fix railroad tracks, and to transfer the loads from rails to the underlying ballasts and subgrades. **Anchors** wrap around the bottom and sides of the rail base, on both sides of a tie to prevent it from longitudinal movement. There are at most 4 anchors for one tie, two anchors on each side of the rail.

In order to capture lateral views of the gauge side and the field side of both rails, we use four cameras in total. (Figure 3) All four cameras are connected on the same FireWire bus, which controls the time-synchronization between cameras with high accuracy. The field of view of each camera are set to 24 inches to guarantee 50% overlap of images when traveling at 10 mph. For this camera setting, at each time point, each side of each tie plate is seen by only one camera, a tie will be seen by all four cameras, and the anchors will be seen only by the two gauge view cameras.

2.2. Tie Plate Detection

To detect tie plates, we applied the same approach presented in [6]. Specifically:

1. Use Hough transform to detect two dominant horizontal lines in the image, which correspond to two horizontal edges of the tie plate.

2. Find the two vertical edges of the tie plate as follows.

    - For the image region between the two detected horizontal lines, compute its edge map using the Sobel operator, then sum up the edge magnitude for each column.

    - For each column, sum up all magnitudes within a window that is centered on it. The window size

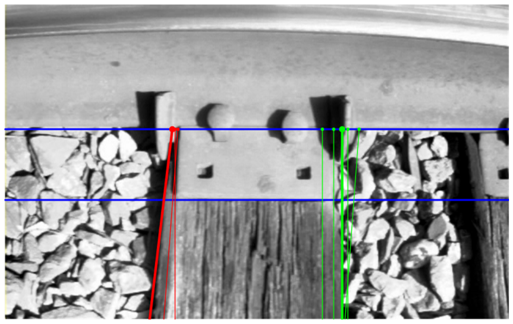


Figure 4. Candidate left vertical edges (in red) and right vertical edges (in green) of a tie, and the two selected edges (the two thickest lines).

approximately equals the width of a tie plate (assumed to be fixed).

- Find the two minimums in the above plot, corresponding to the tie plates left and right vertical edges.

2.3. Tie Detection

After tie plates are detected, we implement a simple and robust approach for tie detection. The upper horizontal edge of the tie is aligned with the upper edge of the tie plate. The lower horizontal edge of the tie usually is aligned with the bottom boundary of the image. The remaining task is to identified two near-vertical edges of the tie:

1. Use Hough transform to detect near-vertical lines in close proximity of the vertical edges of detected tie plates.
2. For each detected vertical line, compute the mean intensity difference between its left image region and its right image region. The intuition is that the tie surface is uniformly texture, and so is the ballast surface on the two sides of the tie, however the texture of the tie and the ballast are very different.
3. Select the two lines with max distance computed in step 2, corresponding to the left and right edges of the tie (Figure 4).

Note that if tie plate is not detected in a frame, the search area for the vertical lines becomes the whole image, in which case tie detection may suffer from higher false positive rate. We then represent each detected tie as a rectangular bounding box. Although the polygon formed by the four detected lines are not always rectangular, it can be closely approximated by a rectangle.

2.4. Anchor Detection

Detecting anchors is a crucial step towards detecting anchor defects (shift, spread) and anchor pattern compliance exceptions, which are potential causes of derailment.

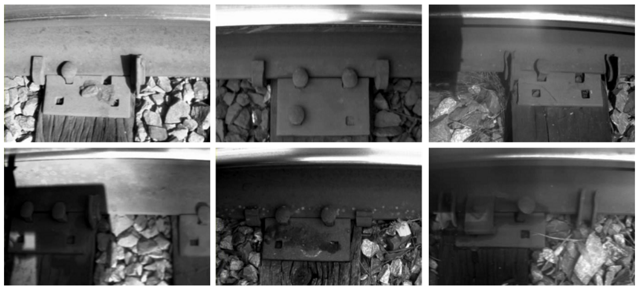


Figure 5. Anchor appearance has a high variability coming from diversities in anchor type, size, shape, camera view point, occlusion and light condition.

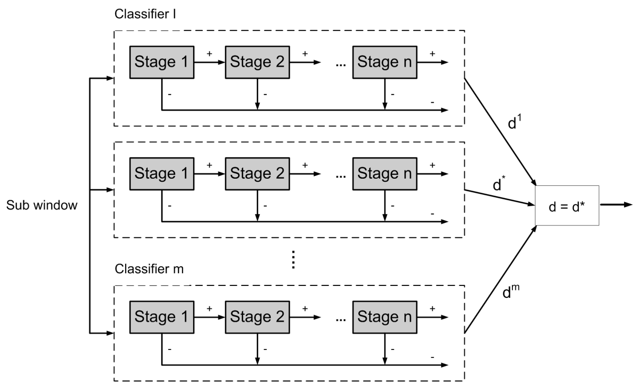


Figure 6. Our model-switching mechanism to combine multiple cascade classifiers. At each time point, we only return the detection results d^* from the classifier with the highest number of detections among multiple cascade classifiers.

We implement a learning-based anchor detector based on the Adaboost discriminative classifier. We observe that a long track segment may include multiple subsegments, each with a different type of anchor. Training only one single cascade classifier for all subclasses of anchors would decrease its discriminative power, due to high in-class variability of anchors. As opposed to the standard Adaboost algorithm [11] that used a single cascade classifier, we employ multiple cascade classifiers, somewhat similar to that introduced in [12], as depicted in Figure 6. Specifically, we train multiple binary classifiers, each corresponds to a subclass of anchors. For detection, we employ a model-switching mechanism as follows. We keep all classifiers running simultaneously, but at any time point we only return the detection results from one selected classifier - the one with the highest number of detections in the last 50 frames. For each frame, we apply a sliding window detection approach within a ROI, which is defined to be the horizontal image stripe covering the region around the lower edge of the rail, where anchors should be installed. The width of the stripe equals the image width. (Figure 2)

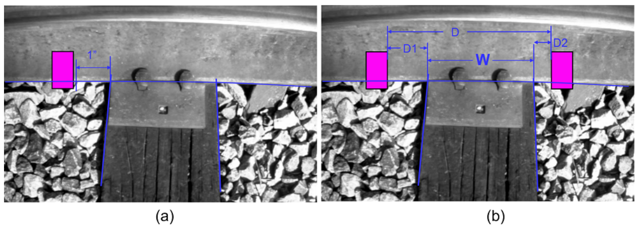


Figure 7. Definition of anchor shift (a) and spread (b).

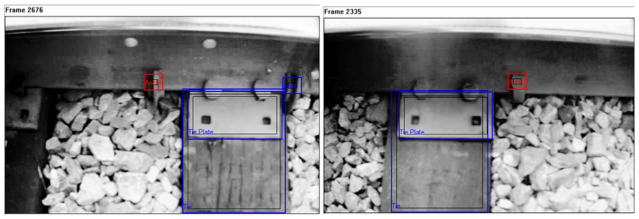


Figure 8. Some examples of shifted anchors detected by our system.

2.5. Anchor Condition Assessment

After anchors are detected and located, the next step is to assess their conditions. An anchor is considered shifted if it is more than 1 inch away from its associated tie horizontally (Figure 7 (a)). A spread occurs when the horizontal distance between two anchors of the same tie is 4 inches more than the tie width (Figure 7 (b)), i.e.: $D1 + D2 = D - W \geq 4$ inches. Therefore a spread of an anchor pair is automatically obtained by computing the shift values of each anchor in the pair. Both shifts and spreads are considered track defects, since they are strong evidence that the rail at that location is running (unstable).

Pixel-Inch Calibration

Since shifts and spreads are defined in inches, while the anchor-tie distance is computed in pixels, we have to be able to reliably convert distances from pixels to inches in order to accurately detect shifts and spreads. Due to wide angle fisheye distortion, the pixel to inch mapping is not uniform for all columns in the image. We take advantage of the fact that the width of the tie plates are fixed at 7.5 inches. We annotate the bounding boxes for roughly $3,000$ tie plates for each gauge-view cameras. Since we're mainly interested in the horizontal distance, we plot the tie plates' width values in pixels with respect to the X coordinate of their locations. We quantize the X coordinates to 20 different bins. The conversion curve is fitted by performing bin averaging for each of the 20 bins and followed by linear interpolation. The resulted curve allows us to map an arbitrary image pixel to inches, given its column index. As illustrated in Figure 9, the pixel-inch mapping function roughly approximates a quadratic function.

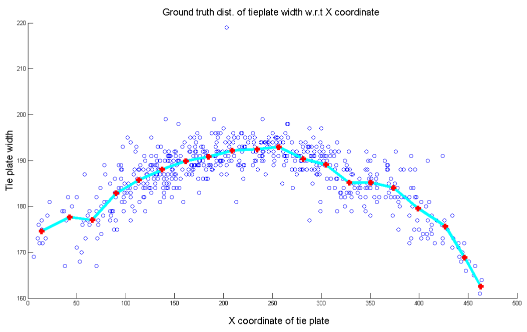
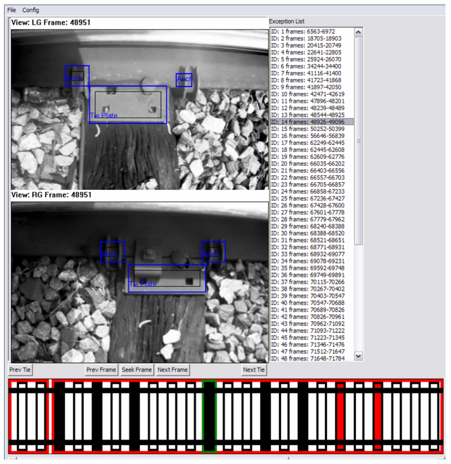


Figure 9. Pixel-inch mapping function is constructed by fitting a curve to a set of manually labeled data points.



```
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
```

Figure 10. Our exception visualization tool shows the consolidated results: A black tie indicates a tie with complete boxed anchors (4 anchors), a red tie indicates a tie with incomplete boxed anchors (1-3 anchors), and a white tie has 0 anchors.

3. Compliance-level Exception Detection

3.1. Rail Component Consolidation

To evaluate a rail segment's compliance with the designed pattern, we must account for the same physical components being detected in multiple video frames in a given view and also in different views. For a given frame, we associate any detected anchors with the tie plate nearest to them, and store the presence of the anchor along with the tie to anchor distance as part of the tie plate data structure. We then perform cross frame and cross view object matching using the tie plate position data. Figure 10 illustrates a schema of a rail segment of interest, in which the anchor information of each physical tie in the rail segment is available and visualized. The schema is generated by the exception visualization tool that we built as part of our system.

Cross-frame Matching

Our cameras' 24" field of view and video capture rate of 20 fps produce a frame to frame overlap of .63 when traveling at 10 mph. With this degree of overlap objects will typically be visible in two frames when traveling at speed, and while the vehicle is still accelerating components can be detected in many consecutive frames. We account for multiple detections of objects using a simple tracking scheme. When an object is detected for the first time we calculate an expected position for it in subsequent frames based on the vehicle's speed (obtained from DMI data), and the known field of view of the cameras. In the next frame we compare any new detected objects with the expected position from the previous frame, and decide whether or not it is the same object by thresholding the distance between the expected and detected positions. Please note that we receive vehicle speed measurements every second, and must interpolate between them to get a speed for an individual frame. When we match a new tie plate detection to a previous detection, we combine the information regarding anchors associated with the tie plates from both frames.

Cross-view Matching

As we mentioned in Section 2.1, at any time point a tie can be seen by as many as four camera views. When the expected position of a tie plate moves out of the camera's field of view, we attempt to match the tie plate with tie plates from other views that have also finished passing the field of view. Pairs of tie plates are matched by comparing their positions in frames where both tie plates were present. Tie plates are matched iteratively in an attempt to build up a set of detections in all four views representing one complete tie. If a complete set of tie plate detections cannot be found, and some tie plates' expected position moves further outside the field of view we assume tie plate detection failed for the missing view(s) and report the largest set of tie plates found as a complete tie with missing data. If anchors were found at all four positions on the combined tie, we say the tie has boxed anchors and can maintain a count of ties with boxed anchors to evaluate compliance with railroad safety rules. More details will follow in Section 3.2.

3.2. Compliance Exception Detection

Different rail types have different standard/required rail anchor patterns, as shown in Figure 11. For any rail track geographical location, we need to obtain the geo-reference data which contain the required anchor pattern for that specific geo-location, which is indexed by milepost and footage, or GPS latitude and longitude. Based on such data, for any given 100-foot track that is captured in the video, we will know exactly what is its target anchor pattern by matching the GPS data. To detect compliance exception, we first count the total number of boxed ties for every 100-foot rail track (denoted by C). A boxed tie is a tie with all 4 anchors in normal (not shifted) condition. We then compare the tie count with the required number (denoted by R). If the count is smaller than 85% of the requirement, i.e. $(R - C)/R >= 15\%$, then a compliance-level exception is declared.

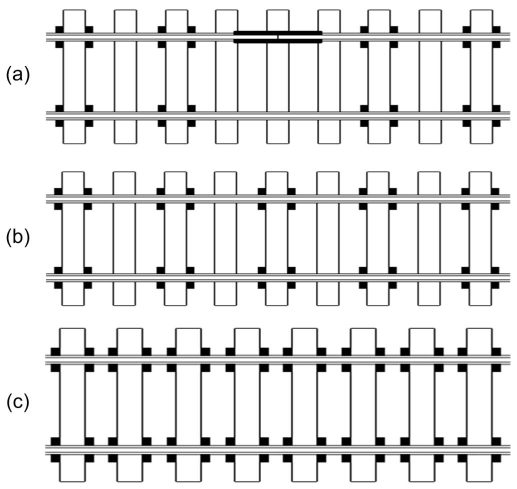
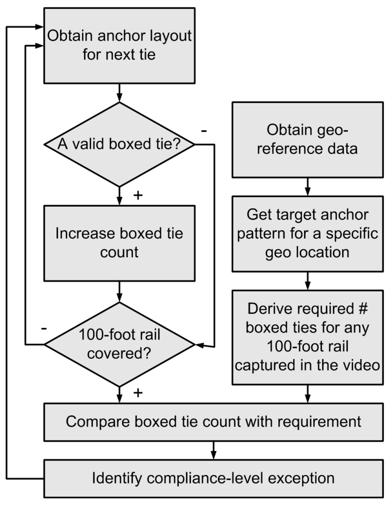


Figure 11. (a) Anchor pattern for jointed rail, (b) for continuously welded rail (CWR), (c) for rail sections with high curvatures or at railroad crossings.


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
Figure 12. Flow chart of detecting compliance-level anchor exception.

Note that we have to constantly check if we have covered a 100-foot track. If yes, we compare the boxed tie count with the required number, otherwise we read in the anchor layout of the next tie. In our current implementation, once an exception is detected, we slide the inspection window by the whole 100-foot, otherwise, we slide it by 1 tie. A confidence score will also be calculated for each exception based on the confidence of anchor detection.

The complete algorithm for is illustrated in Figure 12.

4. Experimental Results

To evaluate the performance of our system, we install our complete system (including hardware and software) on a high-rail vehicle. We conduct a real online field test by

Component	Precision	Recall
Tie plate	99.3%	100%
Tie	88.2%	82.3%
Anchor	96.5%	96.7%

Table 1. Our system's performance on rail component detection.

running the vehicle on real railroad tracks, covering different track segments, different days, different times of day, different weather conditions. The vehicle runs at average speed of 10 mph. The test consists of inspecting three segments of one-mile track for presence/absence of tie plates, ties, anchors, anchor shifts and spreads, and anchor pattern exceptions.

For quantitative analysis, we have recorded track video data during the online test. After the test, we manually annotated ground truth of tie plates, ties and anchors on those collected videos. All the experimental results below are reported based on directly comparing the system results with the ground truth.

A detection is considered correct if the overlapping region between the detection bounding box and a ground truth bounding box of the same component $\geq 60\%$ of the ground truth bounding box.

4.1. Rail Component Detection

Tie Plate Detection

The low in-class variability of tie plates make tie plate detection the most reliable part of our system. For our test, we achieve very good performance in tie plate detection, with 100% recall and about 99.3% precision. (Table 1)

Tie Detection

Tie detection is mostly needed for anchor condition assessment: both shift and spread are defined by the distance between an anchor and its associated tie. Our tie detector is not as reliable as the tie plate detector, achieving 82.3% recall and about 88.2% precision. (Table 1)

Anchor Detection As shown in Table 2, our anchor detector using multiple cascade classifiers with model-switching mechanism achieves better precision and recall than using a single classifier. Both approaches outperform the edge-based anchor detector presented in [6].

Anchor false positives are mainly caused by sharp shadows or debris on the rail. False negatives (missed detections) are mainly caused by extreme lighting conditions (too dark or too bright), or occlusions.

4.2. Anchor Condition Assessment

Once an anchor and its associated tie are detected, we can easily calculate the distance between them in pixels. We then convert this distance to inches using the Pixel-inch mapping function, described in Section 2.5). We declare the anchor as being normal or shifted based on this distance in

Alg.	Precision	Recall
Multiple classifiers	96.5%	96.7%
Single classifier	95.6%	93%
Edge-based approach [6]	83.3%	91%

Table 2. Comparative evaluation of our anchor detector using multiple cascade classifiers and model-switching with other approaches.

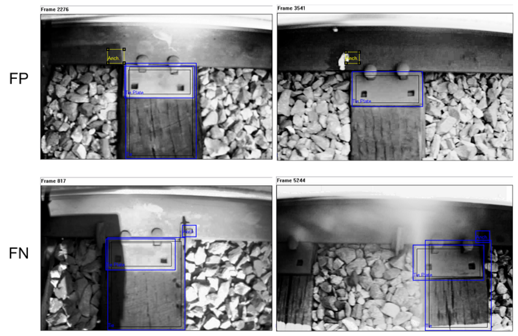


Figure 13. Examples of anchor false positives (top row) and anchor false negatives (bottom row).

inches. Although in general an anchor is considered shifted if it is more than $\tau = 1$ inch away from the associated tie, this number τ may not be fixed. In fact, the choice of τ reflects the acceptable threshold for the severity of the shifts, and in practice the user should have the flexibility to decide which threshold to use, based on their own judgment. For this reason, we decide to report the shift detection performance at different choices of τ. An ROC curve can easily be generated by computing an ROC point for each value of τ. In Figure 14, we plot two ROC curves, with and without pixel-inch calibration.

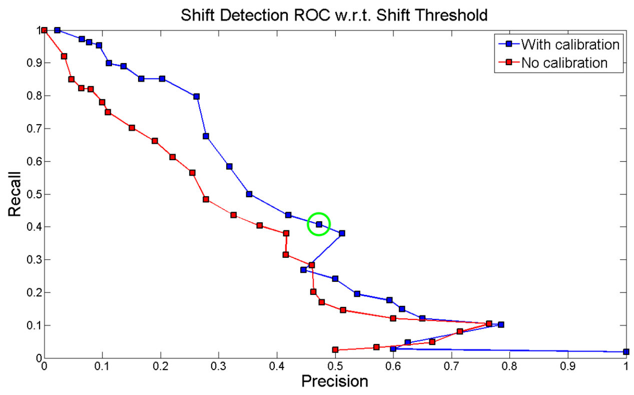


Figure 14. Shift detection ROC with respect to the shift threshold τ, with and without pixel-inch calibration. The curves are plotted with τ from 0 to 2.5 inches.

We plot the ROC curve with τ from 0 to 2.5 inches. The ROC point with best F measure corresponds to $\tau = 1.3$ inches, with precision 0.47, recall 0.4, and F-measure 0.44.

The ROC curve shows that our shift detection still requires further improvement. The shift detection performance was mostly affected by the inaccuracy of the bounding boxes of detected ties and anchors. In order to achieve a good shift detector, it would require a high level of accuracy in both tie and anchor bounding box localization.

We could not report the performance on spread detection, since we did not find any ground truth case of spread in the track segments that we sampled for annotation.

4.3. Anchor Pattern Compliance Exception Detection

Detecting compliance exceptions for railroad tracks is expected to achieve a high detection rate and low false positive rate. A compliance exception negatively affects the rail safety at the sequence level, thus a failure to detect any single one of those can potentially leads to grave consequences. On the other hand, verifying a compliance exception requires a lot of time and resource for a railroad company, since it involves visually scanning a 100-foot track segment. With a high false positive rate, it would be very challenging for human inspectors to scan through all the reported exceptions to find true ones. From our own investigation with railroad companies, the desired false positive rate for compliance exception detection is 1 false positive per 1 hour of inspection at 95% detection rate.

Since there are no true exception in the 3-mile track we used in earlier tests, we performed this test on an 1-hour video captured from a different rail track at a different time. For this one-hour video, there are 3 genuine compliance exceptions. Our system detected all of them, achieving **100 %** detection rate, while generating **3 false positives per hour**. This is a very promising result compared to the desired performance from railroad companies.

5. Conclusion

We described an automatic vision-based rail inspection system, with main focus on anchors - an important rail component type. Our system can perform in real time at the vehicle speed of 10 mph, at frame rate 20 fps. It robustly detects important rail components such ties, tie plates, anchors with high accuracy and efficiency. The condition of detected anchors are also evaluated to detect shifts and spreads. Detected objects are then consolidated across video frames and across camera views to map to physical rail objects, by fusing the video data with GPS and DMI information. After these rail components are detected and consolidated, further data integration and analysis is followed to detect anchor pattern non-compliances, or exceptions. Quantitative analysis performed on a large video dataset captured on different track conditions demonstrates that our system achieves very promising performance in terms of anchor condition assessment, and compliance-level exception detection. To our knowledge, our system is the first to address and solve these two problem in rail inspection. We also show that our system outperforms another advanced rail inspection system [6] in component detection.

The main challenges for us in the near future is to handle cases of heavy local shadows, as well as light overexposure. We also believe that our current tie detection approach can be further improved. Although our experiments are focused on anchors - a particular type of rail component, our next plan is to extend our system to handle other rail component types such as spikes and joint bar bolts.

References

[1] AURORA. http://www.georgetownrail.com/aurora.php. 290

[2] P. Babenko. Visual inspection of railroad tracks. In *PhD Thesis*, 2009. 290

[3] A. Berry, B. Nejikovsky, X. Gilbert, and A. Jajaddini. High speed video inspection of joint bars using advanced image collection and processing techniques. In *Proc. of World Congress on Railway Research*, 2008. 290

[4] J. Edwards, J. Hart, S. Sawadisavi, E. Resendiz, C. Barkan, and N. Ahuja. Advancements in railroad track inspection using machine-vision technology. In *AREMA Conference Proceedings on American Railway and Maintenance of Way Association*, 2009. 290

[5] H. Hsieh, N. Chen, and C. Liao. Visual recognition system of elastic rail clips for mass rapid transit systems. In *Proceedings of the ASME/IEEE Joint Rail Conference and Internal Combustion Engine Spring Technical Conference*, 2007. 290

[6] Y. Li, C. Otto, N. Haas, Y. Fujiki, and S. Pankanti. Component-based track inspection using machine-vision technology. In *ICMR*, 2011. 290, 294, 295

[7] MermecGroup. http://www.mermecgroup.com/diagnostics/. 290

[8] S. Pankanti, L. Brown, J. Connell, A. Datta, Q. Fan, R. Feris, N. Haas, Y. Li, N. Ratha, and H. Trinh. Practical computer vision: Example techniques and challenges. In *IBM Journal of Research and Development*, 2011. 290

[9] M. Singh, S. Singh, J. Jaiswal, and J. Hempshall. Autonomous rail track inspection using vision based system. In *IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety*, 2006. 290

[10] Trackvue. http://www.rail-vision.co.uk/trackvue. 290

[11] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, 2001. 291

[12] Z. Zhang, L. Zhu, and S. Li. Real-time multi-view face detection. In *International Conference on Automatic Face and Gesture Recognition*, 2002. 291