Near-Miss Event Detection at Railway Level Crossings

Sina Aminmansour
School of Electrical Engineering and Computer Science
Queensland University of Technology
Queensland, Australia
Email: s.aminmansour@qut.edu.au

Frederic Maire
School of Electrical Engineering and Computer Science
Queensland University of Technology
Queensland, Australia
Email: f.maire@qut.edu.au

Christian Wullems
Centre of Accident Research and Road Safety - Queensland
Queensland University of Technology
Queensland, Australia
Email: c.wullems@qut.edu.au

***Abstract*—Recent modelling of socio-economic costs by the Australian railway industry in 2010 has estimated the cost of level crossing accidents to exceed AU$116 million annually. To better understand the causal factors of these accidents, a video analytics application is being developed to automatically detect *near-miss* incidents using forward facing videos from trains.**

As near-miss events occur more frequently than collisions, by detecting these occurrences there will be more safety data available for analysis. The application that is being developed will improve the objectivity of near-miss reporting by providing quantitative data about the position of vehicles at level crossings through the automatic analysis of video footage.

In this paper we present a novel method for detecting near-miss occurrences at railway level crossings from video data of trains. Our system detects and localizes vehicles at railway level crossings. It also detects the position of railways to calculate the distance of the detected vehicles to the railway centerline. The system logs the information about the position of the vehicles and railway centerline into a database for further analysis by the safety data recording and analysis system, to determine whether or not the event is a near-miss.

We present preliminary results of our system on a dataset of videos taken from a train that passed through 14 railway level crossings. We demonstrate the robustness of our system by showing the results of our system on day and night videos.

I. Introduction

Improving safety at railway level crossings continues to be a priority for Australian railway operators. Approximately a third of all rail-related fatalities during the ten-year period between 2000 and 2009 occurred as a result of collisions between road vehicles and trains at level crossings [1]. The cost to railways and society as a result of these collisions has been estimated to exceed AU$116 million annually, based on socio-economic cost modelling commissioned by the Australian Railway Industry Safety and Standards Board [2].

Australia has railways covering large expanses of land with as many as 23,000 level crossings [3]. Between 2000 and 2009, an average of 62 collisions per year occurred between road vehicles and trains at level crossings in Australia. From a data analysis perspective, the number of collisions is relatively low and is insignificant for meaningful quantitative data analysis due to statistical uncertainty. Precursor events such as *near-miss* occurrences at level crossings occur at rates that are orders of magnitude more frequent than collisions. Such occurrences are reportable under the national rail safety legislation and are defined in the classification guideline (OC-G1) as occurrences where *"the driver of a moving train takes emergency action, or would have if there was sufficient time, to avoid impact with a person, vehicle or other obstruction and no collision occurred. Emergency action includes continuous audible warning and/or brake application"* [4].

Near-miss occurrences are reported by train drivers, typically via radio to a network control officer who completes a form on behalf of the train driver, or via completion of a form at the end of the shift. Unlike collisions, near-misses are not usually investigated due to the large number of occurrences and resource limitations. Near-miss reporting suffers from a number of critical shortcomings that limit its usefulness for data analysis [5]. These include subjectivity around definitions of what can be considered a near-miss and inconsistencies in reporting.

The Cooperative Research Centre for Rail Innovation, an Australian government funded research initiative, is supporting a project with several Australian railways to improve the quality and objectivity of near-miss reporting. The safety data recording and analysis system being developed by the research team analyses various sources of data to determine whether critical thresholds defining a safe envelope of operations have been exceeded [6]. Data analysed includes parameters obtained from vision algorithms operating on high-definition forward facing video footage[1] and parameters from train-borne systems such as global positioning system receivers, automatic train protection systems and event recorders.

The vision algorithms and methods described in this paper are used to detect road vehicles and their distances from the nearest railway, allowing exceedances of two critical thresholds around near-miss occurrences to be detected. These thresholds are the fouling of the stop line, give-way line or the danger zone by a road vehicle at a level crossing while the train is on approach. The stop line is a continuous line marked across traffic lanes at all level crossings, located 3.5 meters from the nearest railway for level crossings with stop signs, or 3.0 meters from the signal or boom barriers at level crossings with active controls [7]. The give-way line is a broken line

[1]Video taken by a camera located in the driver cabin. This camera has a similar view as the human driver.

located 3.5 meters from the nearest railway at level crossings with give-way signs. The *danger zone* is the area bounded by 3 meters of the nearest railway each side of the track environment. The videos processed by our system are captured with a forward-facing camera installed inside the driver cabin of the train. This forward facing camera provides a view that we call the *cabin view* of the railway. In the rest of this paper we refer to the forward facing camera installed in a train as the *cabin camera*.

A *railway* is a permanent track composed of a line of parallel metal rails fixed to sleepers. The railway corresponds to the green region in Figure 5(c). The topological skeleton of the green region will be called the *railway centerline* (the green line in Figure 6c).

The detection of rails is more robust when the evidence for the left and right rails can be combined in a simple operation. This combination becomes trivial when the curves corresponding to the left and right rails are parallel. This is why we compute from the cabin view a virtual view that is similar to the image which a camera high in the air looking down to the railway would produce. For obvious reasons, we call this virtual view the *bird's eye view*. The bird's eye view is derived from the cabin view by applying an appropriate homography ϕ as illustrated in Figure 2.

In Section II, we discuss related work. Section III outlines our approach for detecting near-miss events. In Section IV we provide details of our railway detection algorithm. Sections V and VI describe the process of detecting and localizing vehicles at railway level crossings. Preliminary results are presented in Section VII.

II. Related Work

Many rail safety applications rely on Computer Vision and Video Analytics to extract safety critical information from videos. These applications include detecting obstacles in front of trains [8][9], assisting train drivers [10], detecting turnouts [11] [12], or detecting events at level crossings [13] [14].

Our aim is to localize vehicles with respect to the railway from cabin views. To achieve this objective, we first localize the rails in the images. This is a fundamental task in vision based rail safety applications. The localization of railways was considered in previous works, but the problem is not considered solved satisfactorily yet. Most of proposed methods are sensitive to the environmental conditions. Factors like lighting conditions ranging from sunshine to night-time, changing appearance of metallic rails in varying weather conditions and shadows make railway detection a challenging problem.

All the published algorithms for localizing railways are based on either processing a cabin view, or processing a bird's eye view. Section IV-A sketches how a bird's eye view is computed. In the following sub-sections we review some of previous works on railway detection.

A. Cabin View Based Methods

In [10], the extraction of the rails is done by applying template matching with pre-defined rail patterns on the near-range region in front of the train. For the far-range region, rail patterns are computed at runtime according to the position of the detected rail segments in the near-range region. Unfortunately the system has many parameters that need to be tuned. The authors have tested the system with more than 200,000 parameter configurations for selecting a good configuration for the system.

In [15], a rail extraction method based on a sliding window approach was proposed . The system starts four search windows from the bottom of a cabin view, and in each window it computes the gradient image and looks for maxima of the gradient magnitude as the edges of the rails. Two of the four search windows are positioned on the left rail and the other two are positioned on the right rail. As rails converge to a vanishing point in a cabin view, the width of the sliding windows decreases when the search continues from the bottom to top of the image. This system is sensitive to lighting changes.

B. Bird's Eye View Based Methods

In [8], an obstacle detection algorithm is proposed for maintenance trains. To localize the railway, the authors apply a perspective transformation to the cabin view to get a bird's eye view. They then search this image for parallel straight line segments that define the railway. Their algorithm scans the image with a search window from the bottom to the top of the image. Although the authors use an adaptive method to select a global threshold value for the Canny edge detector, their system is sensitive to changes in lighting conditions.

A railway detection algorithm for autonomous trains is introduced in [9] to detect obstacles in front of a train and also provide information about a region of interest in front of the train for other sensors. This system is based on lane detection techniques developed for driver assistant systems. The authors apply an edge detector in the bird's eye view. The edge image is then divided vertically into 10 regions, and for each region vertical edges are selected. Railways are extracted by combining the information from different regions and information about the inter-rail distance. The RANSAC algorithm is then used for finding the railway centerline. This approach is also sensitive to the choice of threshold parameter value for the computation of the edge image. Moreover, the approach does not exploit information about the location of the rails in previous frames.

A rail and turnout detection algorithm is proposed in [11]. Rail detection is done by determining local maxima and minima in the gradient image of the gray level cabin view. The left and right rails are then detected by fitting a line with the RANSAC algorithm on the bird's eye view. Again, this approach suffers from the problem of having to select a threshold for the edge image that works for a wide range of lighting conditions.

A near-miss event detection algorithm is introduced in [13]. This approach also relies on computing a bird's eye view. In this paper, the authors avoid the problem of having to select a threshold for an edge detector by using the Line Segment Detector (LSD) algorithm [16]. LSD is a linear-time line segment detector that gives accurate results, a controlled number of false detections, and requires no parameter tuning. LSD works well with natural images. The rails are detected by creating a histogram of the vertical projection of the points belonging to the LSD segments. Although the use of LSD

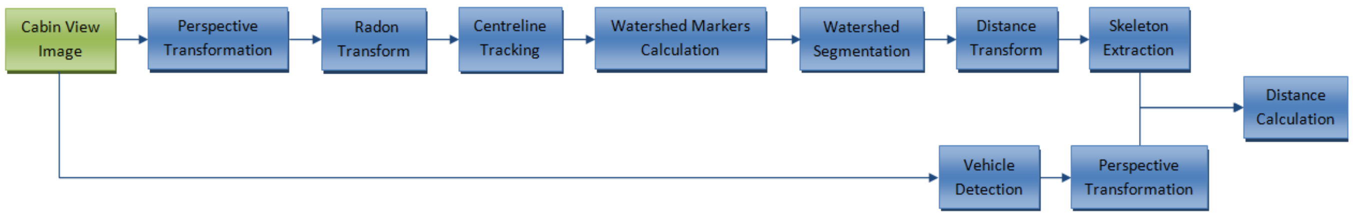

Fig. 1. The sequence of steps involved in the detection of near-miss events from the video data captured in train cabins.

avoids the threshold selection problem, LSD has its own limitations. It does not detect segments in blurred regions of images. Another drawback of the approach presented in [13] is that a histogram of vertical projections loses its power of discrimination when the rails are tilted at an angle, because the projections of the rail points become spread over several bins.

C. *Proposed Method*

Although a few systems have been described for the localization of railways from cabin views, to the best of our knowledge, there is no publicly available software. And most of the published algorithms are too sensitive to their threshold parameters. These algorithms tackle the problem of localizing railways without taking into account information from previous frames. Another issue is that most of the algorithms cannot work in low light conditions. To address these shortcomings, we introduce in this paper a new approach for localizing railways by combining the Radon Transform [17], Watershed [18], and Distance Transform [19] algorithms. Moreover, we exploit fully our knowledge of the geometry of the railways, including the inter-rail distance and the maximum curvature of the rails. To make the detection algorithm more robust, we combine the evidence of the left and right rails in a standard Bayesian framework with belief update to track the railway from frame to frame.

In Section VII, we show experimentally that our system is capable of localizing railways from cabin views. Morevover, we show that without parameter changes, our system can localize rails both during daytime and nighttime.

III. Near-Miss Detection

In order to automatically detect near-miss events from cabin views, we detect vehicles at railway level crossings and localize the vehicles with respect to the railway. We can determine whether or not the road vehicle fouls the stop line, give-way line or danger zone while the train is approaching. If an event is flagged as a near-miss, analysts will review the relevant video sequence.

At railway level crossings, vehicles are normally seen sidewise. To detect the vehicles, we use the classifier described in [20]. This classifier is based on Histograms of Oriented Gradients (HOG). The software code is publicly available. After detecting the position of the vehicles in the cabin view, the position of the vehicles are transferred to the bird's eye view for estimating the ground distance of the detected vehicles with respect to the railway. Sections V and VI give some details about the detection of the vehicles and estimation of the ground distance. The architecture of our system is sketched in Figure 1. Section IV-A explains how the cabin view is transformed to a bird's eye view.

IV. Railway Detection

As illustrated in Figure 1, the process of detecting railways starts with passing a cabin view to a pipeline of modules represented in the first row of this figure. The process starts with transforming a cabin view to a bird's eye view by applying the homography to the cabin view (Figure. 2). The result of the transformation is passed to a function that uses the Radon transform [17] for computing a set of candidate positions for the railway centerline in a search window (Figure. 3). The height of this search window is small enough that the rails will look straight in this window. The evidence for the candidate positions is then combined with the position of the railway centerline in the previous frame to find the most likely position of the railway centerline in the current frame and its tilt angle.

The Watershed algorithm [18] is used in our system for extracting the (possibly curved) railway region. Our system uses the tilt angle of the bottom railway centerline, and also knowledge about the geometric structure of the railways for positioning the Watershed seeds as illustrated in Figures 4 and 5.

After extracting the railway region, the Distance Transform algorithm [19] is applied to extract the railway skeleton in the bird's eye view (see Figure 6). The width of the detected railway region is monitored to flag any inconsistency. Algorithm 1 summarizes the different steps taken to localize the railway.

A. *Homography between Cabin View and Bird's Eye View*

From projective geometry [21], we know that given two images (taken by two different cameras) of a quadrilateral lying on a plane in 3D, there exists a unique perspective transformation (homography) ϕ, mapping the image coordinates of the quadrilateral vertices in the first image (cabin view) to the image coordinates of the quadrilateral vertices in the second image (bird's eye view). Figure 7 demonstrates the use of bird's eye views for localizing and calculating the distance of vehicles to a railway centerline.

B. *Railway Centerline Candidates*

To find the railway centerline in our system, we initialize a search window at the bottom of a bird's eye view, and apply the Radon Transform [17] to detect the strong lines in this search window. The variation of the Radon transform that we use projects binary edge image along a number of

Algorithm 1 Railway Centerline Localization

```
1: Inputs: $I_{cv}$ // A cabin view
2:         $\phi$ // Homography between cabin view and bird's eye view
3:         $h$ // Horizon line
4:         $R_p$ // Position of the railway centerline in the previous frame
5:         $r$ // Minimum radius of the railway curve in pixels
6:         $w$ // Railway width in pixels
7: Output: $L_c$ // Railway centerline position (list of points)
8: $I_h \leftarrow$ Region of $I_{cv}$ under the horizon line $h$
9: $I_{bev} \leftarrow$ Bird's eye view transformation of $I_h$ with $\phi$
10: $I_{roi} \leftarrow$ Search region at the bottom of $I_{bev}$
11: $L_c \leftarrow$ Radon Transform ($I_{roi}$) // Candidate rail lines
12: $R_c \leftarrow$ Tracking ($L_c$ , $R_p$) // Railway centerline position in $I_{roi}$
13: $a \leftarrow$ Angle ($R_c$) // Tilt angle of the railway centerline
14: $W_m \leftarrow$ Watershed Markers ($R_c$, $a$, $r$, $w$ )
15: $R_r \leftarrow$ Watershed ($W_m$, $I_{bev}$) // Detection of railway region
16: $D_r \leftarrow$ Distance Transform ($R_r$)
17: $L_c \leftarrow$ Skeleton ($D_r$) // Railway centerline position
```

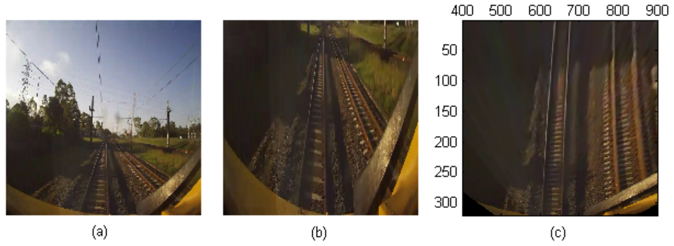

Fig. 2. Subplot (a) shows a cabin view. Subplot (b) shows the region of the cabin view under the horizon line. This region is used for transforming the cabin view to a bird's eye view. Subplot (c) shows a bird's eye view, resulting by applying the homography to the cabin view.

radial directions. The Radon transform is better suited than the probabilistic Hough transform for our application as we are not concerned with gaps between line segments. At the cost of a slight computation cost increase, the Radon transform generates fewer artifacts than the Hough transform. In the same way as for the Hough transform, the strong lines in the original image correspond to peaks in the accumulator matrix (See Figure 3b). We restrict the computation of the accumulator matrix to the angle intervals corresponding to lines oriented ± 45 degrees from the vertical.

For detecting the railway centerline in the search window, we compute a new score matrix from the Radon score matrix by combining the evidence of the left and right rails. The new score matrix is calculated as follows:

$$\text{Score}(i, \theta) = \text{Radon}(i - g, \theta) + \text{Radon}(i + g, \theta) \quad (1)$$

In Equation (1), i is a bin index, and g is half of the railway width in pixels. We keep the best lines according to the new score matrix as the candidate positions of the railway centerline. Figure 3 illustrates how the railway centerline is found using the Radon Transform.

C. Tracking the Railway From Frame to Frame

Because of the train's lateral motion and the varying rail curvature, the rails do not stay in the same position in the image.

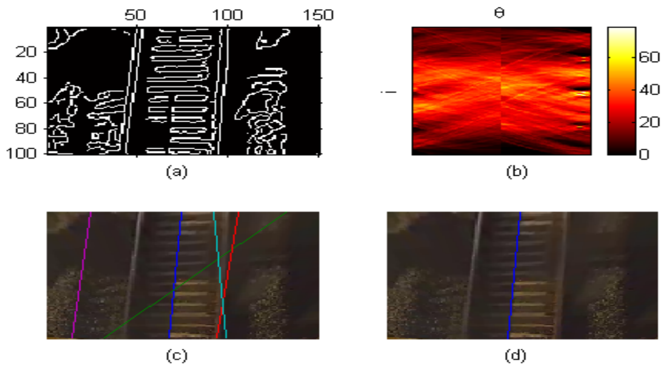

Fig. 3. Subplot (a) shows the edge points detected by a Canny edge detector applied locally. Subplot (b) shows the Radon Transform accumulator matrix. Subplot (c) shows the best 5 railway centerlines given the evidence of the current frame. Subplot (d) shows the most likely position of the railway centerline, taking into account the location of the railway centerline on the previous frame.

The position vector x of the railway centerline is predicted using the following equation:

$$P(R_t = x) = \sum_{y} P(R_t = x | R_{t-1} = y)\, P(R_{t-1} = y) \quad (2)$$

where y is the position vector of the railway centerline in the previous frame.

We approximate $P(R_t = x | R_{t-1} = y)$ with

$$\exp(-\lambda(|T_x - T_y| + |B_x - B_y|) \quad (3)$$

where T_x and B_x are the horizontal coordinates of the top and bottom intersection points of the line (characterized by the vector x) with the bounding rectangle of the search window. Similarly, the variables T_y and B_y correspond to the line characterized by the vector y. We have experimentally found that if λ takes a value around 0.25, the tracking becomes robust to all lateral motions we have observed.

D. Railway Segmentation

The Watershed algorithm is used in our system for extracting possibly curved railway regions in the bird's eye view. We used the non-parametric marker-based implementation of the Watershed algorithm described in [18]. To define the markers for the region that corresponds to the railway in a bird's eye view, we exploit the geometric information that we have about the construction design of the railways in Australia. We know that the distance between two rails is 1067mm, and the minimum curvature radius for railways is 125 meters. The distance between two rails in a bird's eye view in our system is 50 pixels. By having this information the minimum railway curve radius in pixel can be calculated in a bird's eye view. In our system the minimum curvature radius is 5857 pixels.

Thanks to the information about the minimum radius of a railway curve in pixels and also thanks to the tilt angle of the railway centerline, we can position the Watershed markers on a birds eye view for detecting railway regions. Figure 4 demonstrates how Watershed markers are calculated according to the geometric information that we have about the railways.

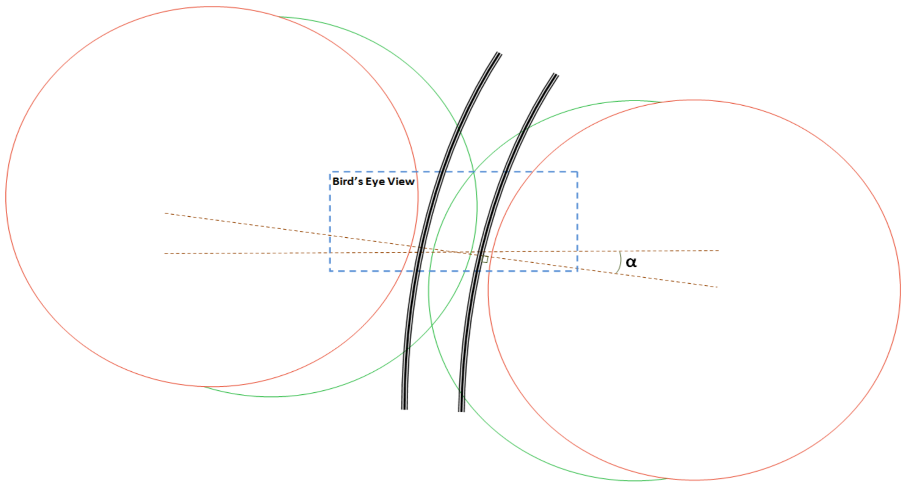

Fig. 4. Watershed markers on the bird's eye views are positioned after calculating the position of four circles that characterize the outside and inside regions of the railway. The two red disks are guaranteed to be outside the railway. The intersection of the two green disks is guaranteed to be inside the railway. The radius of these disks matches the minimum curvature of the railway. The angle α is the yaw angle of the railway.

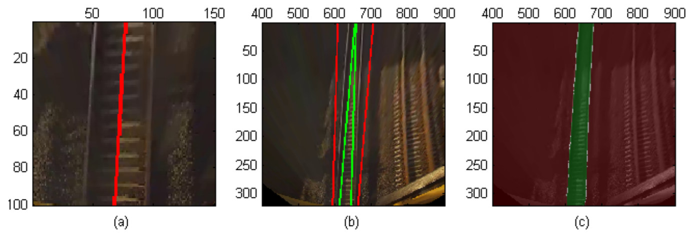

Fig. 5. Subplot (a) shows the result of the detection of the railway centerline with the Radon transform algorithm on the bottom region of the bird's eye view. Subplot (b) shows the Watershed markers. Subplot (c) shows the result of the segmentation after applying the Watershed algorithm.

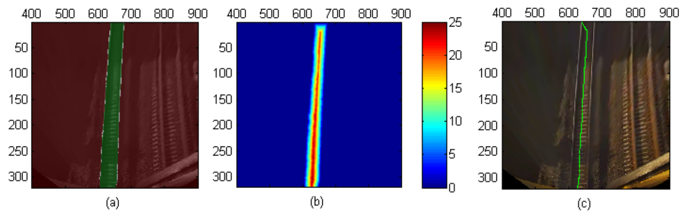

Fig. 6. Subplot (a) shows the result of the segmentation with the Watershed algorithm. Subplot (b) shows the result of the Distance Transform applied to the detected railway region. Subplot (c) shows the skeleton of the railway region.

The red disks in Figure 4 correspond to the exterior markers. The intersection of the green disks is used for the interior markers. Figure 5 demonstrates the application of the Watershed algorithm with the calculated position of the markers on a bird's eye view.

E. Railway Centerline Computation

The extracted region of the railway with the Watershed algorithm can be represented as a binary image. By applying the Distance Transform [19] to this binary image, the (possibly curved) railway centerline can be extracted. Given a binary image where the foreground pixels are set to one and the background pixels are set to zero, the Distance Transform estimates the distance of each foreground pixel to the background.

Figure 6b shows the application of the Distance Transform algorithm on a detected railway region. The railway centerline pixels have the largest values. We scan each row to find the local maxima of the Distance Transform to derive the railway centerline. This process is illustrated in Figure 6c.

V. Vehicle Detection

There are many applications that rely on vehicle detection algorithms including driver assistant systems and autonomous vehicles. A survey of vehicle detection algorithms can be found in [22]. We integrated the overall winner of the PASCAL VOC challenge [23] in our system to detect vehicles in the cabin views. We use the publicly available object detection implementation of Pedro Felzenszwalb [20] in our application with the models of vehicles trained on the VOC 2010 dataset. This vehicle detector accepts as input a cabin view, and returns a set of bounding boxes wrapping the detected vehicles in the image. Figure 8 illustrates how the detector works on a cabin view at a railway level crossing.

In order to estimate the ground distance between vehicles and the railway centerline from a cabin view, we compute the distance of the bottom of the bounding boxes that are returned from the vehicle detector algorithm to the railway centerline in the birds eye view representation of the scene. The bounding boxes are projected to the bird's eye view by applying the perspective transformation ϕ that maps the bottom segment of each bounding box (hopefully on the ground) to its corresponding position on the bird's eye view. Figure 7 demonstrates this process.

As our system relies on the vehicle detector algorithm for detecting near-misses, a failure in detection of a vehicle in a cabin view can cause the system to fail to detect a potential near-miss event.

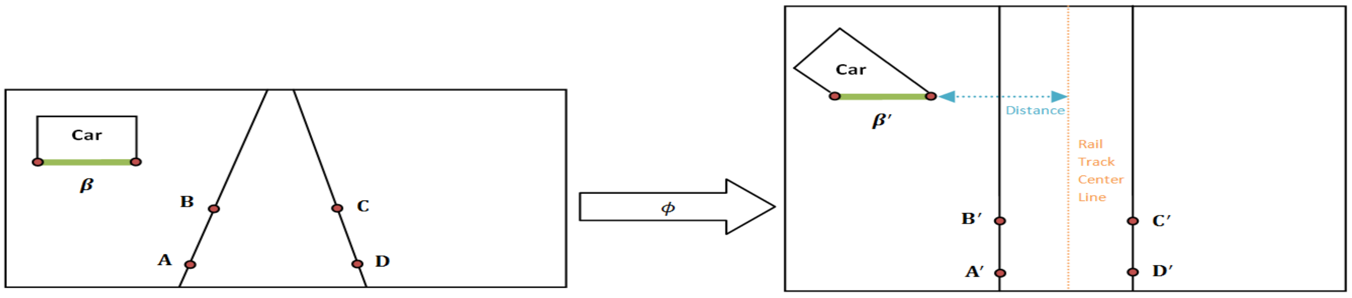

Fig. 7. The homography ϕ maps the points A, B, C and D to A', B', C' and D' respectively. The homography ϕ is used to transform each point on the cabin view to its corresponding position on the bird's eye view. We transform the bottom edge of each detected vehicle (β) in a cabin view to its corresponding position (β') in the bird's eye view for calculating the distance of the vehicle to the railway centerline.

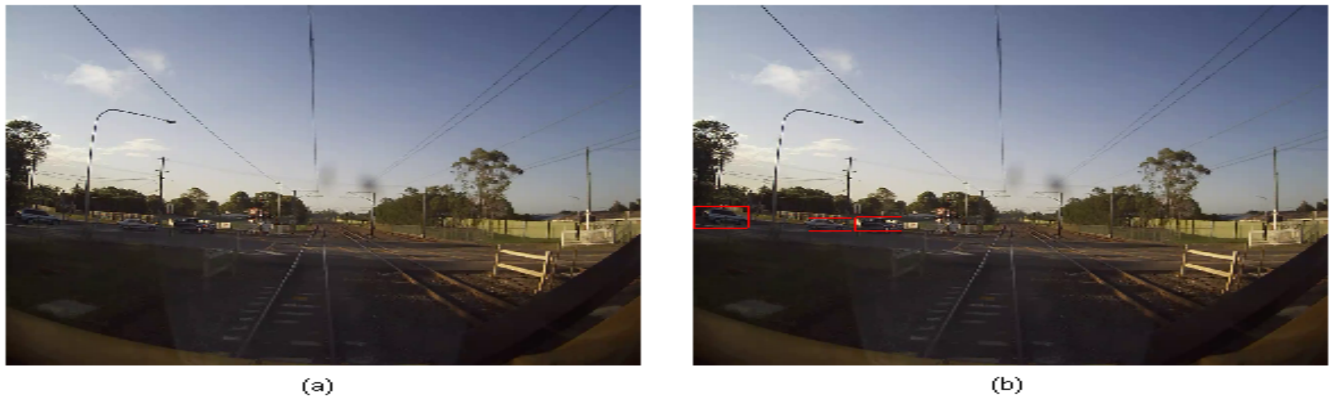

Fig. 8. Results of applying the vehicle detector algorithm to a cabin view at a railway level crossing. Subplot(a) shows a cabin view. Subplot(b) shows the position of the 3 cars that were detected by the vehicle detector.

VI. Distance Calculation

The output of the railway detection algorithm is the list of pixels in the bird's eye view corresponding to the railway centerline. The output of the object detector (applied to a cabin view) is a list of bounding boxes of vehicles. As Figure 7 illustrates, we estimate the distance on the ground between the vehicle and the railway centerline by computing the distance of the bottom edge of the vehicle bounding box to the railway centerline. The elevated parts of the vehicle are distorted non-linearly by the homography ϕ, and therefore cannot be used in a straightforward fashion for distance estimation.

Given that the distance between the railway sleepers is 685mm, and the distance between the rails is 1067mm, it is easy to convert a distance in the bird's eye view (in pixels) to a ground distance (in meters). After calibrating the camera, we found that 1 meter on the ground corresponds to 46.8 pixels in the bird's eye view.

After calculating the distance of the vehicles to the railway centerline in meters for each frame, the distance of each vehicle is then saved into a database for further analysis of the captured event to decide whether a near-miss event occurred or not. Figure 9 illustrates the calculation of the distance of a vehicle to the railway centerline.

VII. Experimental Results

We have developed our system in a Linux environment and written our code in C++ and Matlab. Our dataset was supplied to us by the Cooperative Research Centre for Rail Innovation. The dataset contains 14 videos of 14 railway level crossings captured by a cabin camera. Ten videos were taken during daytime and four videos during nighttime. Figure 10 shows some sample video frames of our dataset. For evaluating our railway detector we randomly selected 200 frames from each daytime video and 250 frames from each nighttime video. In total we have used 3000 frames for our experiments.

To validate our railway detector we have developed an automated test that checks the consistency of the segmentation of the railway in each frame. As the width of the railway is constant in a bird's eye view, the distance of each point on the railway centerline to the rails should be about half the size of the railway width. We retrieve the distance to the rails of each point on the railway centerline using the Distance Transform result. For each point on the railway centerline if its distance value is significantly different from its expected value (half the inter-rail distance) we mark this point as inconsistent. For each frame we calculate the number of points on the railway centerline marked as consistent, and report this value as the consistency rate for that frame. Our automated test calculated the average consistency rate of all the day and night time video frames used for our experiments. Table I shows the results for all the day and night time frames. Table II shows the

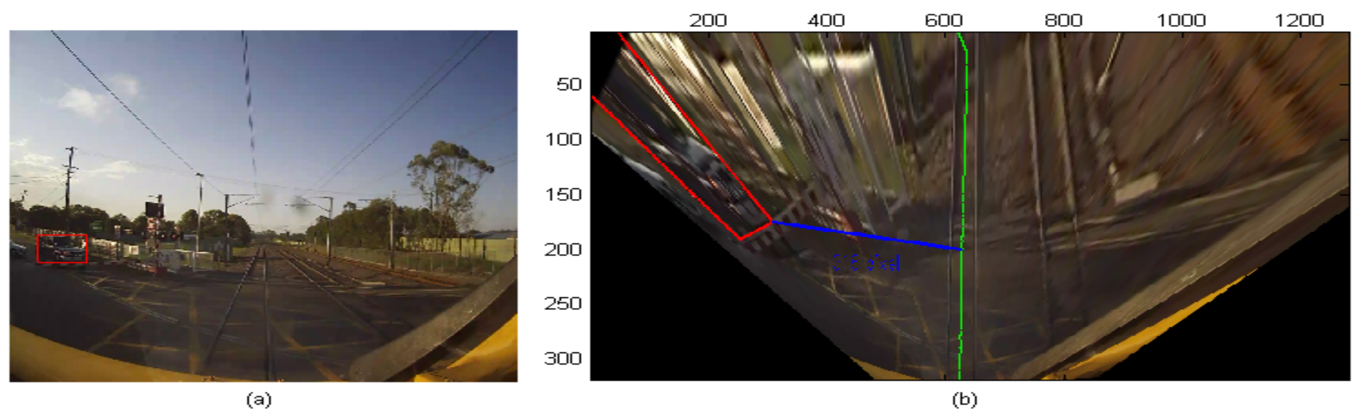

Fig. 9. Subplot (a) shows a cabin view where a vehicle was detected and marked with a red bounding box. Subplot (b) shows the warped bounding box of the detected vehicle in the bird's eye view. The detected railway centerline is coloured in green. The distance between the detected vehicle and the railway centerline is shown in blue. This distance is 319 pixels, which corresponds to 6.8 meters on the ground.

TABLE I. AVERAGE CONSISTENCY RATE OF THE RAILWAY DETECTOR

Video	Total Number of Frames	Average Consistency Rate
Daytime videos	2000	95.72%
Nighttime videos	1000	92.33%

TABLE II. AVERAGE CONSISTENCY RATE OF SAMPLE VIDEOS TAKEN IN QUEENSLAND, AUSTRALIA.

Suburb	Day/Night Time	Average Consistency Rate For 200 Frames
Antigua	Night	93.23%
Bundaberg	Day	98.66%
Bundaberg	Night	90.14%
Grahams Creek	Day	99.02%
Mooloolah	Day	98.02%

TABLE III. SAMPLE CALCULATION OF THE DISTANCE OF VEHICLES TO THE RAILWAY CENTERLINE BY OUR SYSTEM AND A HUMAN.

Vehicle Number	Distance (Our System)	Distance (Human)	Pixel Difference	Ground Difference
1	468	469	1	2.1cm
2	327	331	4	8.5cm
3	334	334	0	0cm
4	247	248	1	2.1cm
5	480	481	1	2.1cm
6	334	337	3	6.4cm
7	259	259	0	0cm
8	501	497	4	8.5cm
9	337	333	4	8.5cm
10	244	241	3	6.4cm

consistency rate for a representative sample of our videos.

The environmental conditions of the sample of frames that we used include low light condition, drizzle, rain with droplets on the windscreen, windscreen wiper in operation, glare, dirty windscreen and shadows (see Figure 10). For our experiments, each frame was checked manually by a person to ascertain whether the railways were segmented correctly. The automated consistency test was also run.

To evaluate our distance calculation algorithm, we have tested our algorithm with a video of level crossings that have cars present at the level crossing. The system detected cars 40 times in the video. For each detected car in each frame it calculated the distance of the car to the railway centerline and reported this value in pixels. To evaluate the system we asked a person to also compute the distance of each detected vehicle to the railway centerline in a bird's eye view. The person computed the distance of the 40 detected vehicles to the railway centerline and reported the value for each detected vehicle. We have compared the distance of each detected vehicle computed by our system and the distance computed by a person. We calculated the difference of the reported values. For the 40 detected vehicles in the video the difference was 2.3 pixels that is around 4.9cm on the ground. Table III shows some samples of the calculation of the distance of each vehicle to the railway centerline reported by the system and a person.

ACKNOWLEDGMENT

This research is funded by the Cooperative Research Centre for Rail Innovation (established and supported under the Australian Governments Cooperative Research Centres program). Project No. R2.119.

VIII. CONCLUSION

In this paper, we have introduced a vision based system for detecting near-miss occurrences at railway level crossing. Our system is built around a railway detector and a vehicle detector. Our experiments show that the system is capable of detecting and localizing vehicles at level crossings with respect to the railway centerline. We have shown that our system is able to work in wide variety of conditions from daytime to nighttime, without having to change any parameter settings. Our system is capable of self diagnosis by computing a consistency measure of the detected railway region. The system can fail in rainy conditions because our segmentation algorithm get confused by the motion of the windscreen wipers. This is an issue that we plan to address in future work.

REFERENCES

[1] Independent Transport Safety Regulator, "Level crossing accidents in australia," Transport Safety Bulletin, 2011.

[2] R. Tooth and M. Balmford, "Railway level crossing incident costing model," Railway Industry Safety and Standards Board (RISSB), Tech. Rep., 2010.

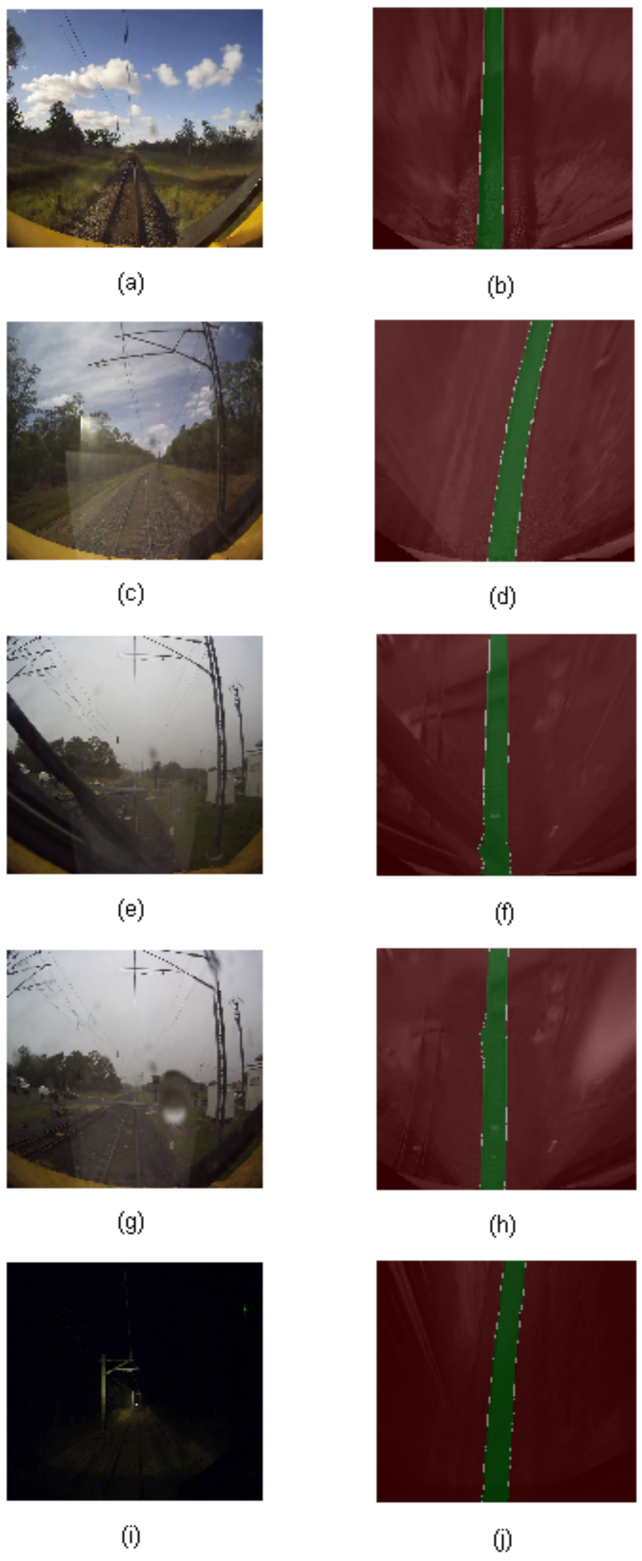

Fig. 10. The computer vision system can handle a wide range of weather conditions without any parameter adjustment. The images in the left column are cabin views, and the images in the right column are the corresponding birds eye views with the railway segmented in green. In Image (a), shadows cross the railway. In Image (c), the railway is curved, and sun glare is visible on the windscreen. In Image (e), the windscreen wiper is in operation because of the rain. In Image (g), droplets run off the windscreen. In Image (i), the light is very low.

[3] Railway Industry Safety and Standards Board, "Level crossing stock-take," 2009.

[4] Rail Safety Regulator's Panel, "Guideline for the top event classification of notifiable occurrences: Occurrence classification - guideline one (oc-g1)," Rail Safety Regulator's Panel, Tech. Rep., 2008.

[5] C. Wullems, G. Dell, and Y. Toft, "Improving the railway's understanding of accident causation through an integrated approach to human factors analysis and technical safety data recording," in *Proceedings of the 5th International Conference on Applied Human Factors and Ergonomics*, 2014.

[6] C. Wullems, Y. Toft, and G. Dell, "Improving level crossing safety through enhanced data recording and reporting: the crc for rail innovations baseline rail level crossing video project," *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of rail and rapid transit*, vol. 227, no. 5, pp. 554–559, 2013.

[7] Standards Australia, "As 1742.7-2007 manual of uniform traffic control devices part 7: Railway crossings," Tech. Rep., 2007.

[8] F. Maire and A. Bigdeli, "Obstacle-free range determination for rail track maintenance vehicles," in *2010 11th International Conference on Control Automation Robotics & Vision (ICARCV)*. IEEE, 2010, pp. 2172–2178.

[9] M. Gschwandtner, W. Pree, and A. Uhl, "Track detection for autonomous trains," in *Advances in Visual Computing*. Springer, 2010, pp. 19–28.

[10] B. T. Nassu and M. Ukai, "Rail extraction for driver support in railways," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*. IEEE, 2011, pp. 83–88.

[11] J. Corsino Espino, B. Stanciulescu, and P. Forin, "Rail and turnout detection using gradient information and template matching," in *Intelligent Rail Transportation (ICIRT), 2013 IEEE International Conference on*. IEEE, 2013, pp. 233–238.

[12] J. Wohlfeil, "Vision based rail track and switch recognition for self-localization of trains in a rail network," in *Intelligent Vehicles Symposium (IV)*. IEEE, 2011, pp. 1025–1030.

[13] S. Aminmansour, F. Maire, and C. Wullems, "Video analytics for the detection of near-miss incidents on approach to railway level crossings," in *Proceedings of 2014 Joint Rail Conference*. American Society of Mechanical Engineering, 2014.

[14] P. Ranganathan and E. Olson, "Automated safety inspection of grade crossings," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 2149–2154.

[15] J. C. Espino and B. Stanciulescu, "Rail extraction technique using gradient information and a priori shape model," in *15th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2012, pp. 1132–1136.

[16] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "Lsd: A fast line segment detector with a false detection control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 722–732, 2010.

[17] S. Deans, *The Radon Transform and Some of Its Applications*, ser. Dover Books on Mathematics Series. Dover Publications, 2007.

[18] F. Meyer, "Color image segmentation," in *Image Processing and its Applications, 1992., International Conference on*. IET, 1992, pp. 303–306.

[19] P. Felzenszwalb and D. Huttenlocher, "Distance transforms of sampled functions," Cornell University, Tech. Rep., 2004.

[20] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[21] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge Univ Press, 2000.

[22] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 5, pp. 694–711, 2006.

[23] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Jun. 2010.