Application of Computer Vision Techniques for Railroad Inspection using UAVs

Pooja P. Harekoppa

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Kevin B. Kochersberger
Amos L. Abbott
Michael Hsiao

July 21, 2016
Blacksburg, Virginia

Keywords: Computer Vision, Machine Learning, Railroad inspection, Unmanned Aerial Vehicle (UAV)

Application of Computer Vision Techniques for Railroad Inspection using UAVs

Pooja P. Harekoppa

ABSTRACT

The task of railroad inspection is a tedious one. It requires a lot of skilled experts and long hours of frequent on-field inspection. Automated ground equipment systems that have been developed to address this problem have the drawback of blocking the rail service during inspection process. As an alternative, using aerial imagery from a UAV, Computer Vision and Machine Learning based techniques were developed in this thesis to analyze two kinds of defects on the rail tracks. The defects targeted were missing spikes on tie plates and cracks on ties. In order to perform this inspection, the rail region was identified in the image and then the tie plate and tie regions on the track were detected. These steps were performed using morphological operations, filtering and intensity analysis. Once the tie plate was localized, the regions of interest on the plate were used to train a machine learning model to recognize missing spikes. Classification using SVM resulted in an accuracy of around 96% and varied greatly based on the tie plate illumination and ROI alignment for Lampasas and Chickasha subdivision datasets. Also, many other different classifiers were used for training and testing and an ensemble method with majority vote scheme was also explored for classification. The second category of learning model used was a multi-layered neural network. The major drawback of this method was, it required a lot of images for training. However, it performed better than feature based classifiers with availability of larger training dataset. As a second kind of defect, tie conditions were analyzed. From the localized tie region, the tie cracks were detected using thresholding and morphological operations. A machine learning classifier was developed to predict the condition of a tie based on training examples of images with extracted features. The multi-class classification accuracy obtained was around 83% and there were no misclassifications seen between two extreme classes of tie condition on the test data.

Application of Computer Vision Techniques for Railroad Inspection using UAVs

Pooja P. Harekoppa

GENERAL AUDIENCE ABSTRACT

Railroads have to be inspected from time to time for defects and deterioration. A large number of accidents are reported to be caused due to Railroad track defects. So the railroad inspections have to be done very frequently and are carried out by skilled experts in the area. The process can be very time consuming and exhaustive and could also be prone to human error. To automate this process ground vehicles have been designed to capture images and later process them to look for defects. But these ground vehicles need to be custom designed and could be very expensive and also operating them on track blocks the track for rail movement. To minimize the disruption to train services and to accommodate the increase in passenger and freight traffic on railroads, the time that can be allocated to access the rail infrastructure by foot patrols or by ground inspection vehicles needs to be minimized. Hence rail infrastructure owners are under pressure to find more effective means to perform the task. So a solution to this could be use of unmanned aerial vehicles (UAVs) to capture images of railroad tracks. This has the potential to be completely automated in the future with advanced technology to fly the UAVs on the track autonomously. But in this work the research is limited to identifying two kinds of defects on the tracks from the captured images. One is inspection of spikes which hold the tie plate to the ground and other is tie conditions for cracks. The way this is currently performed is by teaching the computer to identify spike region in the image from non-spike region and also crack region on the ties from non crack region. Once the computer has learned to identify this, new images captured from the UAV could be analyzed for the presence of these defects. The algorithms for locating the region of interest in the track images captured from UAV and then analyzing these regions for defects were developed as part of this thesis work and it was shown to give good results subject to quality of the captured image.

Dedication

I would like to dedicate this thesis to my parents and my grandmother who have raised me to be the person I am today. Thank you for the unconditional love, patience and support you have given me throughout this process. You have taught me that it is possible to achieve anything I set my mind to. Thank you for everything.

Acknowledgments

Firstly, I would like to express my sincere gratitude to my advisor Prof. Kevin Kochersberger for continuous support of my Graduate studies through assistantship for research, for his patience, motivation, and immense knowledge. He believed in me to do the research work I was given and provided his complete support to help me accomplish it through continuous funding and by giving me the liberty to pick tasks I was interested in. He played a key role in motivating me to do more for this thesis and I am really honored to have been part of Unmanned Systems lab under his guidance.

I would also like to thank Prof. Lynn Abbott for co-advising me, providing me with opportunities to work on during the first year of my Masters and also for being a great teacher. I loved attending all the Computer Vision classes in my first semester and it kept me motivated in the field and helped me explore different areas for research.

My sincere thanks also goes to Jennifer Player who provided me an opportunity to work with Bihrle Applied Research as a Graduate Research Assistant, and which brought me this work which I have accomplished for my thesis. Also my heartfelt thanks to Stanton Coffey for being a great mentor throughout this work and guiding me in the right direction, without his precious support it would not have been possible to accomplish this research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Rail transport provides a means for movement of passengers and goods using wheeled vehicles on railway tracks. In contrast to road vehicles, rail vehicles are directionally guided by the tracks on which they run. Rail transportation in United States mainly consists of freight shipments. Currently railroads provide limited service for passenger travel in United States, but in most other countries it is still a very vital mode of transportation. According to Federal Railroad Administration's (FRA) Office of Safety Analysis, 9,764 of the 30,195 train accidents in the United States from January 2000 to January 2010 occurred due to defects in the tracks [1]. Track defects can be classified into two categories based on what component of the track is involved; track geometry or track structure.

Rail inspection is the practice of examining rail tracks for flaws that could lead to accidents. Track defects are reported to be the second leading cause of accidents on railways in the United States according to the United States FRA Office of Safety Analysis [2] and the leading cause of these accidents are attributed to human error. Railroad engineering practices and FRA regulations require tracks to be inspected for physical defects at specified intervals, which may be as often as twice per week. Every year, North American railroad companies spend millions of dollars to inspect the rails for internal and external flaws.

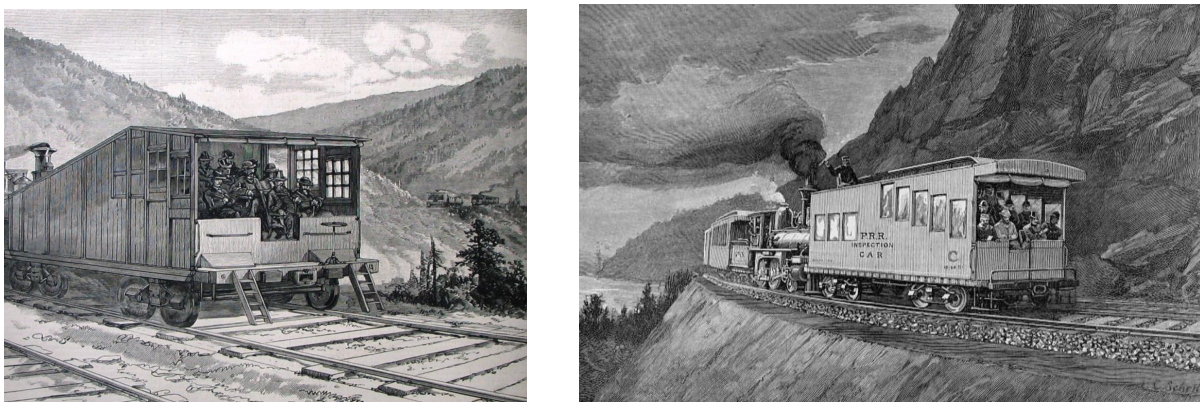

Figure 1.1: Inspection Cars depicted from 1882 and 1891 [3] [4]

In the early days the rail track inspections were done visually. The people on board an inspection car looking for defects on the track is depicted by images in figure 1.1. There was a need for better rail inspection method due to increase in railroad accidents from derailment caused by defects on track. Since then a lot of technologies have been developed for this inspection task. They range from using magnetic induction, ultrasound, radiography amongst others.

Figure 1.2: Track Geometry cars in Russia (left) [6] and New York, USA (right) [7]

The task of inspection covers a wide variety of areas such as detecting cracks on rail, gauge size, monitoring the condition of joint bars, spikes and anchors [8]. Some of these tasks re-

lated to the alignment of the track are automated using track geometry car which measures the required parameters for alignment by physical contact with the rails. Examples of such vehicles are shown in figure 1.2. But other aspects such as spiking patterns are visually inspected by track inspectors. To make these inspections automated, ground vehicles have been designed to capture images and later process them to look for defects using Computer Vision and Machine Learning technologies. One such vehicle is shown in figure 1.3. But these ground vehicles need to be custom designed and hence could be very expensive. Also, operating them on track blocks the track for rail movement. To minimize the disruption to train services and to accommodate the increase in passenger and freight traffic on railroads, the time that can be allocated to access the rail infrastructure by foot patrols or by ground inspection vehicles needs to be minimized. Hence rail infrastructure owners are under pressure to find more effective means to perform the task [5].

Figure 1.3: Camera setup of a hi-rail vehicle [8]

The main goal of this work is to use images of rail tracks captured from UAVs and then process these images using Computer Vision and Machine Learning techniques to detect anomalies in railroads. A UAV is an unmanned aerial vehicle and can operate under various levels of autonomy, it can be either remote controlled by a human operator or by computer providing partial or full autonomy. Figure 1.4 shows an example of a UAV with on board camera.

Figure 1.4: Unmanned Aerial Vehicle (UAV) [9]

The rail track components are shown in 1.5. Rails are the straight metal guides that serve as the medium for the train to travel along. Rail defects include physical obstructions on the track, cracks, over/under-spacing between the rails, extreme curvature. The ties, or the wooden or the concrete planks the rails sit on, are susceptible to defects such as cracks and improper alignment. The ballast, or the material the railroad sits on, is a mixture of different materials serving to dampen the force of the train on the rails and ties.

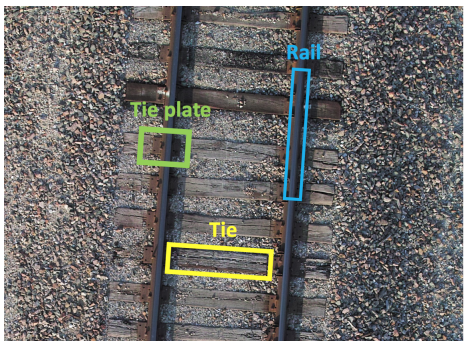

Figure 1.5: Rail track components

The last common component, the tie plate, connects the ties to the rail and can be missing spikes or can crack. As described, the railroad components are fairly complex and difficult to maintain. Hence developing an inspection algorithm is important, but a complete algorithm is difficult to develop. So a subset of defects are targeted for analysis by many systems.

For the purposes of this thesis two kinds of defect identification are targeted; detecting missing spikes on tie plates and cracks on ties. This work was done in collaboration with Bihrle Applied Research and the dataset for the research work was provided by them. An example image from the dataset is shown in figure 1.6. From an image such as this the required region on the rail for defect analysis has to be localized. Two regions that were inspected in this work are shown in figure 1.7. First image corresponds to the region containing the tie plate and second image shows the region containing the tie.

Once the rail and tie plate region are localized the search area for the spikes is limited, spikes will only be found in certain positions on the tie plate. These locations include a column next to the base of the rail and another column further from the rail as shown in the figure 1.7. The task of spike inspection is to identify missing spikes and this is currently done using machine learning models. This work explores both feature based classifier and a neural network for classification of region of interest into holes or spikes. The other region analyzed for defect is the tie. The crack regions on the tie are localized and characterizing features related to cracks are extracted from these images. The tie images are grouped into categories based on their condition and a machine learning model is trained to classify the input ties into one of these categories based on the extracted features.

Figure 1.6: Image captured from UAV for inspection

The complete system has the capability to detect the exact location of the two kinds of defect

Figure 1.7: Localized tie plate and tie images for defect analysis

from images taken from an UAV. This can help the railway department maintain history of the inspection and use it for studying the measure of deterioration of track conditions over time. Also, because data will be available digitally, comparative analysis of degradation over time is possible thereby enabling forecast for the future rail condition.

Figure 1.8: UAVs used for inspection; AR 180, Inspire 1, eXom [10] [13] [14]

The dataset used in this research work for defect analysis was provided by Bihrle Applied Research and are taken from Lampasas (TX), Chickasha (OK) and El Paso (NM) subdivisions. The UAV used for most of the data collection was a AR-180 by AirRobot [10]. It is designed to cover a large number of high performance applications. It comes under the 5 kg weight class excluding payload and has a flight time of 40 minutes. It can record color still images with 10 MP image size and video with resolution of 1920 x 1080/50p, 25p, 12 x Optical zoom. The other UAVs used for some of the dataset collection were Inspire 1 by DJI [11] and eXom by Sensefly [12]. Inspire 1 includes professional 4K camera for video

recording at 4K @ 24-30 fps or 1080p @ 24-60 fps and capture still images at 12 MP. eXom includes a camera capable of taking still images at 38 MP and HD video at 1280 x 720. Reference pictures of these UAVs are shown in figure 1.8.

The following chapters in the document are structured as follows: Chapter 2 provides background information on the railroad specifics, relevant research work, well known Computer Vision and Machine Learning techniques made use of in course of the research. Chapter 3 describes the technique used for Rail Detection in the image captured by an UAV. It also describes the tie localization method developed in the research. The following Chapter 4 describes the processing done to isolate the tie plate region and the ROIs for spike detection and the machine learning and neural network classifiers used for hole vs spike classification. In Chapter 5, the processing techniques developed to inspect tie regions for cracks are described.

Chapter 2

Background

2.1 Related research work

Railroad defects lead to several accidents, fatalities, infrastructure cost throughout the year. As a result there is a continuous need for inspection of railroads. This process is generally done my skilled human inspectors to ensure a thorough inspection. Specialized cars have been designed to inspect track geometry while inspectors visually inspect structural faults, either on foot, or by riding in an inspection vehicle called the Hi-Rail vehicle at a low speed over the tracks. However this process can be very tedious and time consuming and also maintenance of the expanding network of rail tracks requires a considerable investment of time and effort. Hence there is a large scope for automated system for defect identification.

Fewer published works have explored the use of computer vision to detect rail defects in the past but the field is picking up interest due to the advent of high resolution cameras and introduction of advanced Computer Vision and Machine Learning techniques for recognition. Since the railroads have a fixed geometry there are very few unexpected objects in the imagery. Though the imagery could vary greatly depending on the illumination and shadows, objects can still be detected using smart techniques and hence Computer Vision techniques

are a natural choice for analyzing defects. The approach followed in this thesis was inspired by [32] to begin with. Interesting features are extracted from a region of interest and then a trained machine learning model is used for classification.

2.2 Computer Vision techniques

2.2.1 Thresholding

Thresholding is a way of image segmentation using which gray scale image is converted to a binary image. An example of image thresholding is shown in figure 2.1.

Figure 2.1: Thresholding [16]

Simple thresholding methods operate on one pixel at a time, if the value is greater than a threshold value, it is assigned one value (may be white), else it is assigned another value (may be black). This method uses a global threshold value. But it may not be a good idea to use a global threshold in all the cases since image could have different lighting conditions in different areas. Adaptive thresholding algorithm works by calculating the threshold for a small regions of the image. This results in different thresholds for different regions of the same image and it gives better results for images with varying illumination.

2.2.2 Histogram Equalization

Histogram equalization is used to increase the contrast in a image [33]. In a bright image, all the pixels would correspond to very high values. Histogram equalization spreads out the most frequent intensity values on the histogram which results in low contrast regions gaining high contrast. This is useful in images where both the foreground and background regions are either too dark or too bright. In many of the recognition applications all the training and testing set images are histogram equalized to make them all have the same lighting conditions

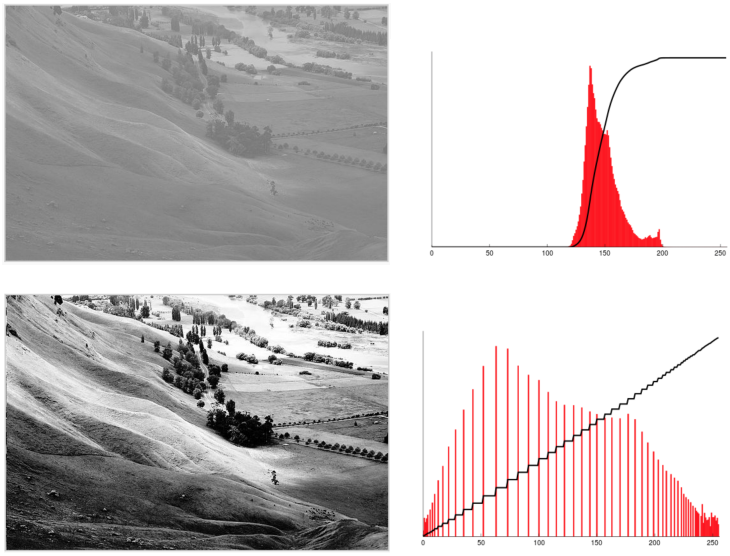

Figure 2.2: Histogram equalization [20]

The traditional histogram equalization considers global contrast of the image. In cases such as the tie plate image with one end of rail darker due to the shadow, this is not a good idea. An example of this is shown in 2.3. Image titled 'Global Histogram Equalization' shows the output image with global contrast adjustment. Though the background contrast has improved it, the right tie plate still does not have the same illumination as the left tie plate. To overcome this, a adaptive histogram equalization method called CLAHE (contrast Limited Adaptive Histogram Equalization) [33] is used. CLAHE divides the image into small

patches and applies histogram equalization on them independently. In order to avoid noise being amplified in the small patch region after histogram equalization, contrast limiting is applied. If any histogram bin for a patch is above a set contrast limit the corresponding pixels are clipped and distributed uniformly to the other bins in the patch. To overcome the tiling artifact between patches after equalization, bilinear interpolation is used. The result is shown in the image titled 'CLAHE' in 2.3. equalizeHist() function from OpenCV is used to obtain the global histogram equalization output and createCLAHE() and clahe.apply() are used to obtain the CLAHE output.

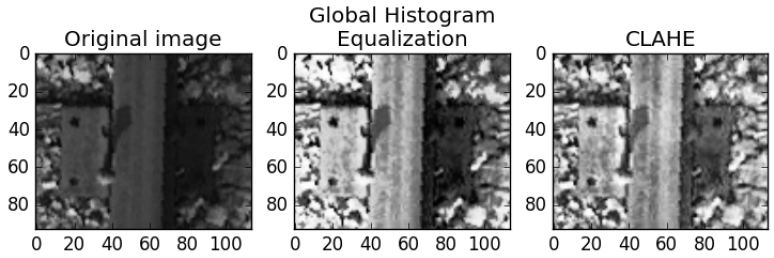

Figure 2.3: Global Histogram equalization vs CLAHE

2.2.3 Morphological Operations

Morphological operations apply set operations such as intesection, union, inclusion and complement to combine input image with a structural element [21].

The morphological operations used in this research work are: Erosion, Dilation, Opening, Closing. Erosion and Dilation form the basis for Opening and Closing operations. The applications of these operations are [22]: removing noise, isolating individual elements and joining disparate elements in an image and region thinning or thickening

The structural element is shifted over the image from top left to bottom right and depending on the matching of pixels in the structural element and the image based on a set operator, the output pixel value is decided. The two main components of the morphological operation

are the structural element and the set operation.

Figure 2.4: Morphological Erosion and Dilation [22]

Erosion Erosion is used for thinning the foreground region and it does it by eroding the boundary of the foreground region. For every pixel in the foreground region, the structural element is superimposed on it and the number of matching pixels is found. If every pixel in the image matches that of the structural element the corresponding pixel in the output image is left as it is, else it is changed to the background pixel value. The erosion result is shown in figure 2.4.

Dilation Dilation is the inverse of Erosion and it is used to grow the foreground region. The dilation result is shown in figure 2.4.

2.2.4 Sobel Operator

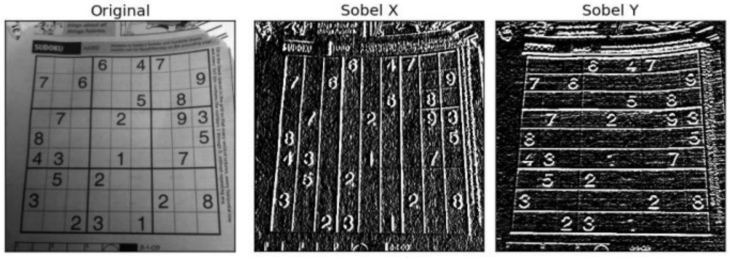

Figure 2.5: Sobel X and Sobel Y filter results [18]

Sobel Operator combines Gaussian smoothing and differentiation to compute an approximation of the gradient of an image intensity function. It provides two operators, Sobel X and Sobel Y depending on the direction of derivatives to be taken, vertical or horizontal respectively.

2.2.5 Gabor filters

Gabor filters are generally used in texture analysis, edge detection, feature extraction etc. Gabor filters are considered as special classes of bandpass filters where they allow a certain 'band' of frequencies and reject the others. When a Gabor filter is applied to an image, it gives the highest response at edges and at points where texture changes. In order to extract features from an image, in the discrete domain, two-dimensional Gabor filters are given by equation 2.1 and 2.2. B and C are normalizing factors to be determined. f defines the frequency being looked for in the texture. By varying θ, we can look for texture oriented in a particular direction. By varying σ, the support of the basis or the size of the image region being analyzed can be changed. [19].

$$G_c[i,j] = Be^{-\frac{(i^2+j^2)}{2\sigma^2}} \cos(2\pi f(i\cos\theta + j\sin\theta)) \tag{2.1}$$

$$G_s[i,j] = Ce^{-\frac{(i^2+j^2)}{2\sigma^2}} \sin(2\pi f(i\cos\theta + j\sin\theta)) \tag{2.2}$$

An example of gabor filter response is shown in figure 2.6. Response to orientations of 0°, 45°, 90° and 135° are shown on the right. The original character picture and the accumulated result of all four orientations are shown on the left [19].

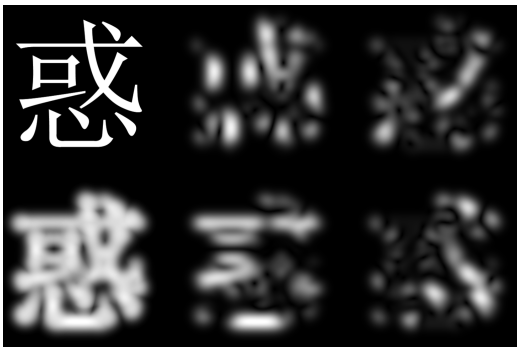

Figure 2.6: Demonstration of a Gabor filter applied to Chinese OCR [19]

2.3 Machine Learning

Machine learning is a method of data analysis that operates by building a model from an example training set of input observations.

Machine learning tasks are typically classified into three broad categories, depending on the nature of the learning and feedback.

- Supervised learning: Example inputs and their desired outputs are provided to the learning algorithm and the goal is to learn a general rule that maps inputs to outputs.
- Unsupervised learning: No labels are given to the learning algorithm, leaving it on its own to find structure in its input.
- Reinforcement learning: A computer program interacts with a dynamic environment in which it must perform a certain goal and the model is made to learn without explicitly telling it where the goal is.

In the following subsections three kinds of learning models are described.

2.3.1 SVM

A support vector machine is a supervised learning model. It constructs a hyperplane or set of hyperplanes in a high dimensional space, which can be used for classification or regression. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class, since in general larger margin leads to lower generalization error.

Figure 2.7 shows the Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called the support vectors.

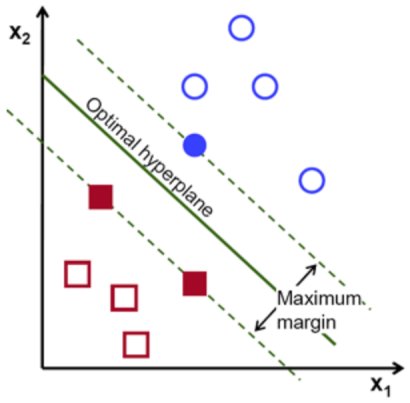

Figure 2.7: Maximum-margin hyperplane and margins for an SVM [25]

Consider training dataset, $(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)$ where each point $\vec{x}_i$ indicates a data point and y_i indicates the corresponding label. Each $\vec{x}_i$ is a feature vector used to describe the data. From these feature vectors, SVM tries to find the hyperplane that separates the points with different output labels. This hyperplane is estimated such that its distance from the nearest point $\vec{x}_i$ from either group is maximum. Hyperplane is defined by $\vec{w}\cdot\vec{x}+b=0$, where $\vec{w}$ is normal to the hyperplane and $\frac{b}{\|\vec{w}\|}$ is the perpendicular distance from the hyperplane to the origin [26].

In order to create nonlinear classifiers, kernel trick (originally proposed by Aizerman et al. [27]) can be applied to the maximum-margin hyperplanes. In the resulting algorithm,

every dot product is replaced by a nonlinear kernel function.

Some common kernels are listed below:

- Polynomial (homogeneous): $k(\vec{x_i}, \vec{x_j}) = (\vec{x_i} \cdot \vec{x_j})^d$
- Polynomial (inhomogeneous): $k(\vec{x_i}, \vec{x_j}) = (\vec{x_i} \cdot \vec{x_j} + 1)^d$
- Gaussian radial basis function: $k(\vec{x_i}, \vec{x_j}) = \exp(-\gamma \|\vec{x_i} - \vec{x_j}\|^2)$, for $\gamma > 0$. Sometimes parameterized using $\gamma = 1/2\sigma^2$

2.3.2 k-NN

k-Nearest Neighbor (k-NN) is a non-parametric learning algorithm [23]. Non-parametric means that it does not make any assumptions on the underlying data distribution. In the real world most of the practical data does not have the typical theoretical assumptions made (ex: Gaussian mixtures, linearly separable etc.) and non-parametric algorithms like k-NN are very useful. k-NN does not use the training data points to do any generalization and hence training phase is fast, it trains by keeping all the training data to use it during testing stage. This is in contrast to other techniques like SVM where you can discard all non support vectors without any problem. The number 'k' decides how many neighbors influence the classification. This is usually a odd number if the number of classes is 2. The algorithm has different behavior based on k.

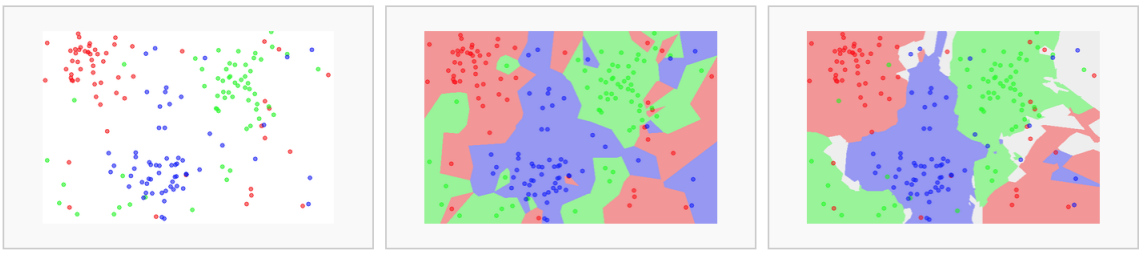

Figure 2.8: Input dataset, 1NN and 5NN classification map [24]

1NN and 5NN classification map for an input dataset is shown in figure 2.8.

2.3.3 Neural Network

Neural networks are one of the techniques which can be used for image recognition. Neural Network models are used to approximate functions that depend on a large number of input data. The universal approximation theorem [29] states that a multi-layer feed forward neural network can represent any arbitrary function.

An artificial neural network is an interconnected group of nodes. Each circular node in figure 2.9 represents a neuron and an arrow represents a connection from the output of one neuron to the input of another. The connections are associated with a set of adaptive weights which can be thought of as connection strengths between neurons, which are activated during training and prediction.

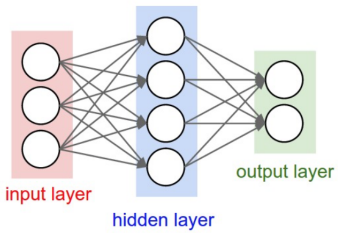

Figure 2.9: Two layer neural network [28]

Every image can be represented as a two-dimensional array, where every element of that array contains color information for one pixel. This is used as input to the neural network, to train it to recognize or classify them. Each output neuron corresponds to one image or image class. Training set for training neural network is a set of pairs of input (flattened RGB arrays), and output vectors (where corresponding image neuron is 1) and the network is trained using Backpropagation learning algorithm [30].

Chapter 3

Rail Detection

The dataset contains images captured from a camera on a UAV of railroads in different sub-divisions. For the purpose of this report Chickasha sub-division images are shown as examples in this chapter. These images are processed to detect the rail and once the rail region is localized the image is further processed to detect the adjoining tie plate and tie regions. These processing steps involve various Computer Vision techniques and are described in detail in the following sections.

3.1 Rail Localization and Image Alignment

Input example images captured from a UAV on the Chickasha subdivision are shown in figure 3.1. The resolution of these images is 4704 x 3136 pixels. From these images, in order to analyze any region of the rail, the first logical step would be to find the rails in the image. Once the rail region is identified the image is rotated to align the rail vertical in the image. This makes further processing simpler because of the linear nature of the railroad. From this image the horizontal tie regions are detected and then the adjacent tie plate region is localized. Tie region is analyzed for cracks and tie plate region is processed to find the region

of interest (ROI) for missing spike analysis. The analysis to detect spikes in the ROI is done based on a machine learning model trained to identify spikes vs holes.

Figure 3.1: Images from Chickasha Sub Division

Edges are frequently used to detect objects in computer vision since object boundaries often generate sharp changes in brightness [35]. The first step in the rail detection process is finding vertical edges. The vertical edges are obtained using Sobel Y operator. This operation is expected to produce the rail boundary in the output. The output of this step is shown in image titled 'Sobel Y result' in figure 3.2. But in order to isolate the rail region, the edges around the ballast and other objects in the image need to be removed. So some processing has to be done to isolate the rail region in the image. This can be done using a dilation operation. Before that, Sobel Y image is binarized using adaptive threshold function in OpenCV. The resulting image is then dilated so that the arbitrary ballast and other texture region is grouped together to be foreground and the rail region not having any edges inside of it remains as background after the dilation operation. The output of the dilation process is shown in the image titled 'Dilated' in figure 3.2.

Since the rail region is fairly isolated at this point, it can be used to find the alignment of the rails in the image. If the location of the rail at the top and the bottom of the image can be found, the angle of inclination can be calculated. Dilated image is used to find the x co-ordinates of the rails in the first row (x_1) and last row (x_2) of the image. This is done by finding the indices in the first row where the intensity value is minimum. The first index

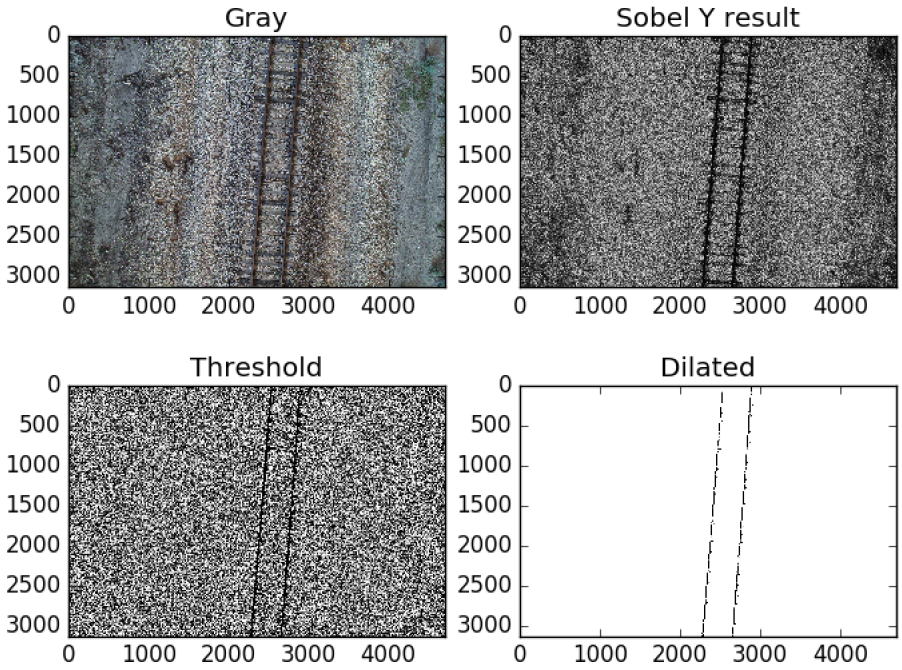

Figure 3.2: Processing done to isolate the rails

in the list gives the x co-ordinate of the first rail and the last index gives the x co-ordinate of the second rail in the image. Similarly the rail co-ordinates are found in the last row of the image. The x co-ordinate of the second rail in the first row is also found using the same approach (x_{12}). Using this information the inclination of the rail is found using the formula:

$$m = \frac{y_2 - y_1}{x_2 - x_1} \tag{3.1}$$

$$\theta = \arctan(m) \tag{3.2}$$

If $x_1 < x_2$: $angle = \theta - 90$; else $angle = 90 + \theta$

The original image is then rotated to align the rails vertical in the image. The midpoint between the rails in the first row of the image is used as the center of rotation for alignment.

It is found by the following expression:

$$center(c) = \frac{x_1 + x_{12}}{2} \tag{3.3}$$

In order to get the Rotation matrix for the required angle, getRotationMatrix2D(center, angle, scale) function from OpenCV is used. This function calculates the affine matrix for 2D rotation based on the following parameters.

Parameters are described as follows: center (c) denotes the center of the rotation in the source image, angle denotes the rotation angle in degrees, positive values mean counter-clockwise rotation (the co-ordinate origin is assumed to be the top-left corner), scale (s) denotes Isotropic scale factor which is chosen as 1 in this case.

The output is an affine transformation, 2x3 floating-point matrix given by equation (3.4).

$$\begin{bmatrix} \alpha & \beta & (1-\alpha) * c.x - \beta * c.y \\ -\beta & \alpha & \beta * c.x + (1-\alpha) * c.y \end{bmatrix} \tag{3.4}$$

where,

$$\alpha = \mathtt{s} \cdot \cos(\theta), \beta = \mathtt{s} \cdot \sin(\theta) \tag{3.5}$$

For purposes of aligning the rails vertically, the center of the rail in the first row is used as center of rotation with scale as 1, and the output image dimension same as the input image dimension. The aligned images are shown in figure 3.3. Note that the dilated image is now inverted to make the rail region appear as foreground.

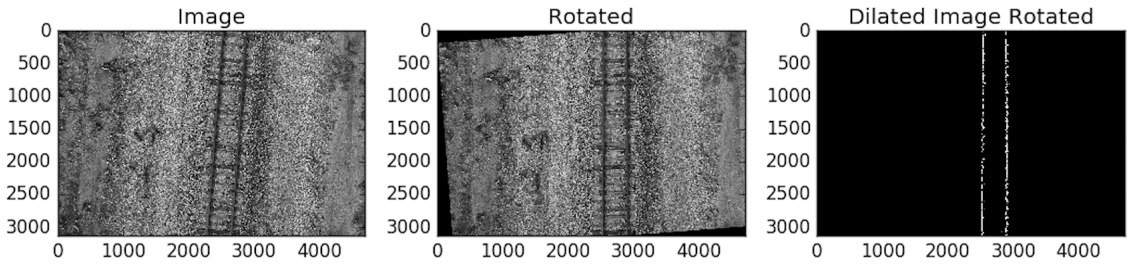

Figure 3.3: Processing done to isolate the rails

3.2 Tie localization

Once the rails are aligned correctly, the horizontal edges are found using Sobel X operator. The reasoning behind using Sobel X operator is it highlights the horizontal boundary of the tie region. The output of this step is shown in image titled 'Sobel X' in figure 3.4. This image is then inverted (bitwise not) and the resulting image is shown in image titled 'Inverted Sobel X' in figure 3.4. This image is then dilated to expand the foreground region overlapping the rail further. This is shown in the image titled 'Dilated' in figure 3.4. To remove the region outside of the rails with the intensity range of mid gray, the output image of the dilation is converted to a binary image using a global threshold. Adaptive thresholding is not considered in this case since the threshold has be computed based on the entire image and not using the sub region. The threshold value chosen for the operation is 80 and the result is shown in image titled 'Threshold' in figure 3.4. To expand the rail and the tie foreground region further, the image is dilated. The dilated image is shown in image titled 'Dilated Threshold' in figure 3.4. As seen in the image, the resulting output still has few black regions on the tie. Morphological closing operation is then applied on this image to close the black holes on the tie. The output is shown in image titled 'Closed' in figure 3.4.

Next step is to identify the plate region next to the rail on the tie. Initially the plates adjacent to the two rails were considered together. But since in some cases the tie may not be perfectly horizontal, handling the tie plates adjacent to the two rails separately gives

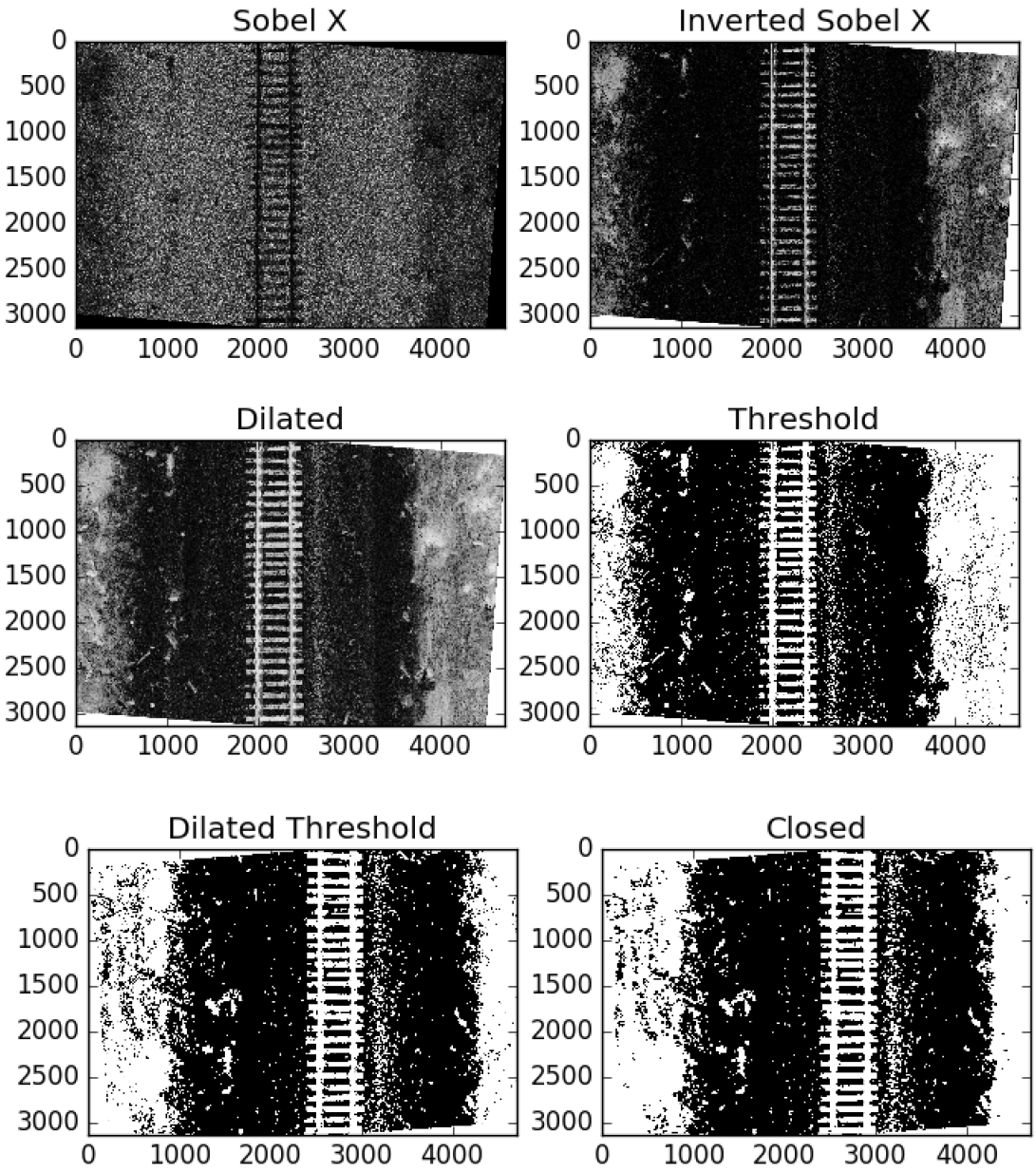

Figure 3.4: Processing done to isolate the rails with tie

better results.

After the rail is detected and the tie is made to appear as foreground, the next step is to identify where the tie regions are in the image so that the sub-region can be used for detecting

defects on the ties. For this, the cropped region shown in figure 3.5 is used. In order to find the start and end of the tie region in terms of row numbers, the average pixel of the row is calculated which is shown by the white plot along Y-axis in the image titled 'Sum' in figure 3.5. This plot is overlaid on the cropped rail image with maroon and blue colors so the plot is clearly visible against its background. This vector of average pixel values is then binarized to either 0 or 255 with a threshold value of 150. This is shown by the white plot along Y-axis in the image titled 'Threshold' in figure 3.5. This column vector is then used to obtain the change in each row with respect to the row after it. And the rows at which this value is maximum are obtained. This is shown by the plot in image titled 'Difference' in figure 3.5.

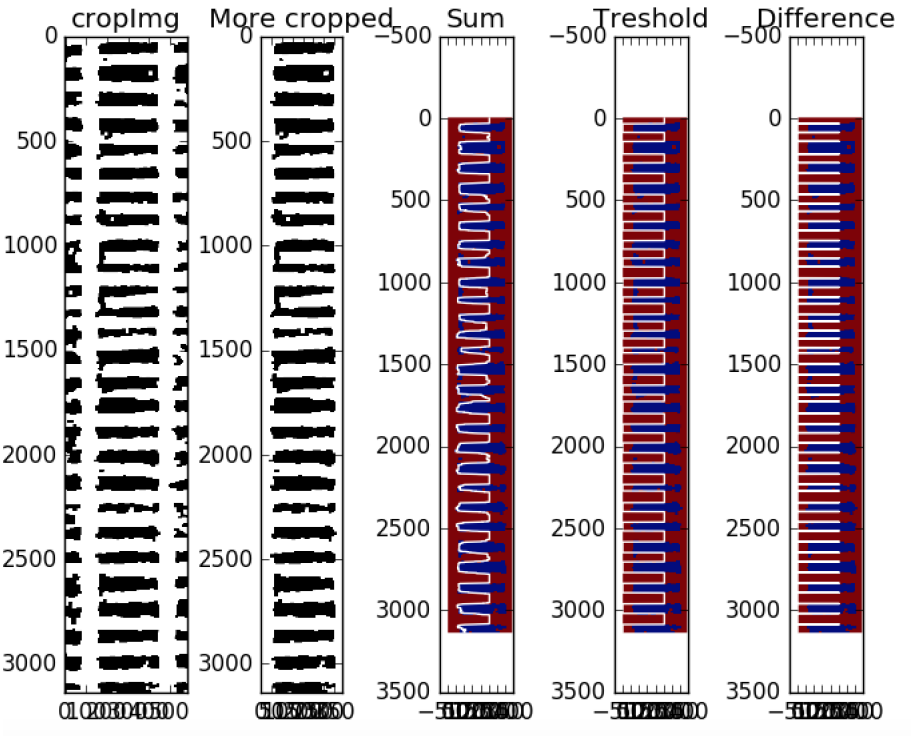

Figure 3.5: Processing done to isolate the ties

Now, the tie region can be successfully extracted from the input image using the boundary found in the above steps. From the resulting images, tie plate and tie regions are cropped

to analyze for missing spikes and tie cracks respectively. This is explained in detail in the following chapters.

3.3 Assumptions

The above method works only with images of railroad where the rails are almost aligned to the image boundary. The approach would not work for cases with curved rail tracks since the assumption made for using Sobel X and Sobel Y operators would not hold good. The algorithm can be extended to the images with curved rails by considering sub-regions of the image separately where the rail is close to being linear.

Chapter 4

Tie Plate Inspection

In order to properly inspect the spikes on the tie plate, the tie plate boundary has to be found and the region of interest (ROI) has to be localized. Once the tie plate boundary is extracted, the ROI is found relative to it.

The requirement here is, given an image of the tie region, number of spikes present in the tie plate need to be analyzed. The steps to be performed for this are listed below:

1. Localize the tie plate region
2. Localize the Region of Interests (ROIs)
3. Extract features from these ROIs
4. Classify the ROIs as spike or hole

Once the region of interest is localized, possible solutions considered for missing spike analysis are described below:

1. Feature extraction and classification: In this method, features are extracted from the ROI and a machine learning model is trained to classify the ROI as a hole or a spike based on this. This is inspired from the work in [32].

2. Template matching: This approach is applicable for images with high correlation. In this project one of the prominent cues helping the identification of a region of interest as a hole or a spike is the shadow of the object cast by the sun. Since different images are taken at different time of the day, the shadow appears at different end of the object of interest (spike or hole). Hence template matching would not be a good option for identification of hole or spike. Experiments were done to prove that during this work.

3. Neural Network: Neural network could be used to classify objects as hole or spike. But the main limitation of neural network is it requires huge amount of data. Since the ROI data is quite diverse, neural network was expected to produce better results for hole/spike classification.

These classification approaches are described later in the chapter. In the next section the approach for localizing the ROI is explained.

4.1 ROI localization

Assuming we have the tie plate region localized the next step is to find candidate regions for hole or a spike. For the tie plate images used in the dataset discussed above, each plate typically has 4 regions with hole or spike and following methods were initially explored to find the ROIs:

1. Using edges: The input image is converted to gray scale, and then the image is thresholded to increase the contrast in the image. The resulting image is then used to detect edges and these edges are then used to find the ROIs.

2. Using contours in binary image: The input image is converted to gray scale, and then the image is thresholded to binarize the image into foreground and background pixels. Contours are obtained from the resulting image using findContours() function in OpenCV.

The edges and contours obtained from processing are shown in the below figure 4.1.

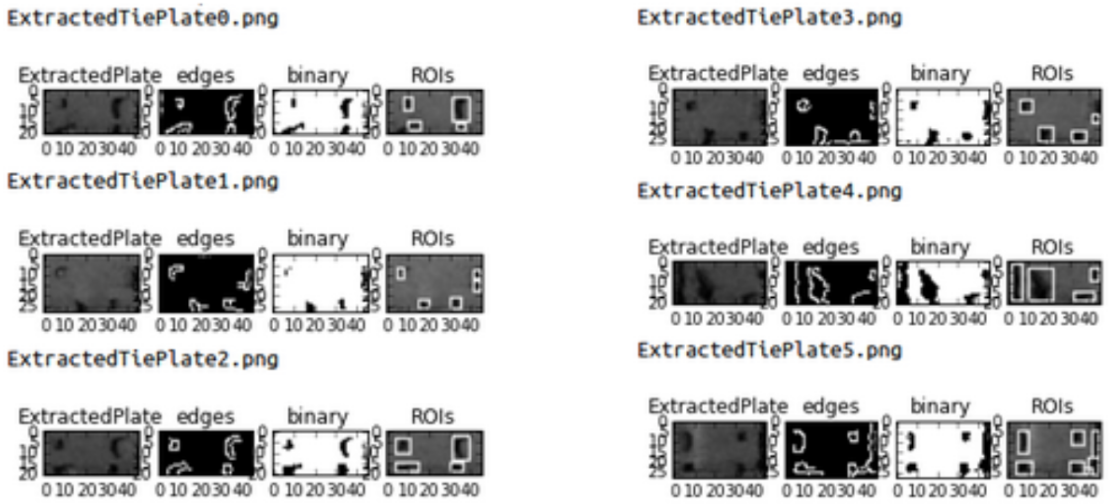

Figure 4.1: Edges and contours found on the tie plate, success (left) and failure (right) cases

4.1.1 Contour based approach

The resulting ROIs found using the contour based approach described above are shown with a white bounding box in the figure 4.1. Area constraints were applied to the contours to eliminate large and small regions which do not represent the ROIs.

This approach has failure cases because of imperfectly localized tie plate and stray pixels in binary image. These failure cases are shown in figure 4.1.

In the next method, image was divided into four quadrants for localizing the ROI. The divided quadrants are shown in figure 4.2. It helped overcome two problems faced with using the entire image; it helps identify missing ROIs and also eliminate false detections.

There were still few cases where the earlier approach of using the entire image seemed to work better. In the example shown in figure 4.3, the earlier approach is able to identify the

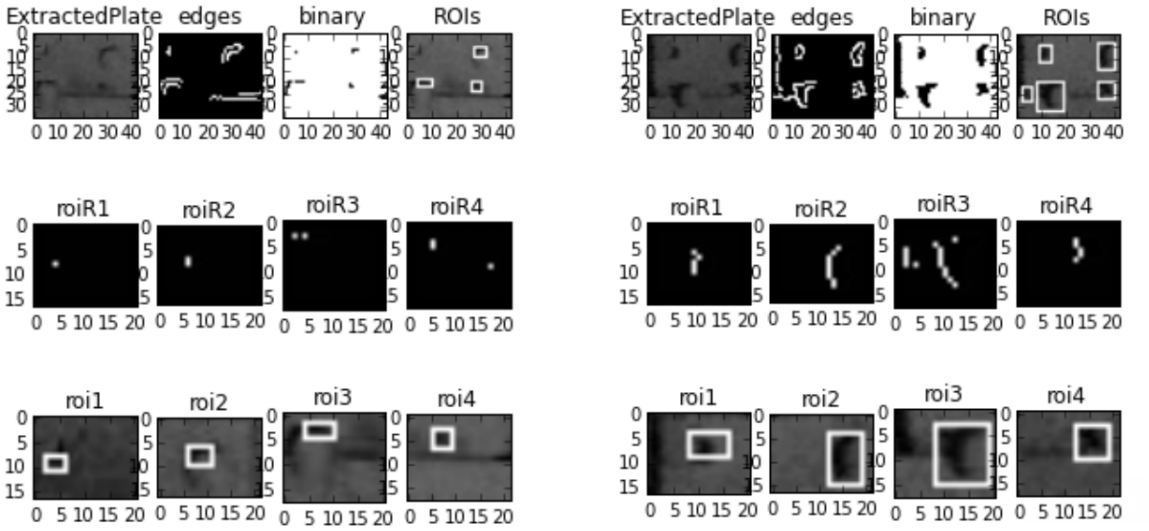

Figure 4.2: Quadrant based approach

four regions correctly but the quadrant based approach fails. To overcome this, the quadrant based approach was used only if four regions were not detected from using the entire image.

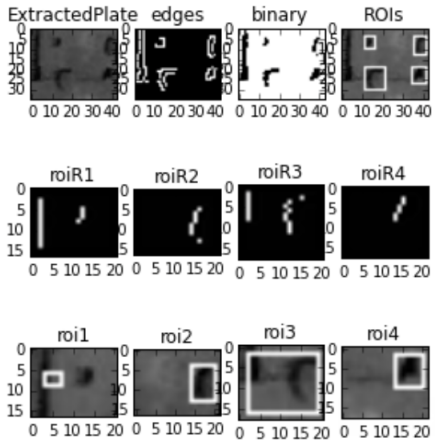

Figure 4.3: Failure case of quadrant based approach

4.1.2 Filter Responses

Gabor filter responses were explored as additional features to use for classification. But the Gabor filter responses provided a new approach for detecting ROIs more accurately. When the Gabor filters are applied on the input image cumulatively, the final response was seen to highlight only the shadow region on the tie plate i.e., essentially the region corresponding to hole or a spike. The responses obtained from using Gabor filters with 45 degrees shift are shown in figure 4.4.

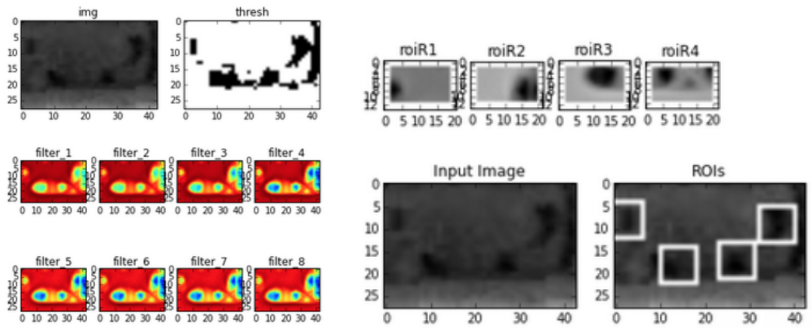

Figure 4.4: Filter response approach

This approach was then tested to identify the ROIs on multiple images and the results on one such image sample are shown in figure 4.4.

Steps performed in this case are listed here:

a. Input image is converted to gray scale and then binarized.

b. 16 Gabor filter responses are cumulatively applied on this binary image.

c. This image is then divided into 4 regions.

d. In each region the minimum pixel value and its corresponding location is found.

e. ROIs are designated as the region around this value.

New improvements were made by Stanton at Bihrle, corporate sponsors of this project, to localize the tie plate more accurately. This data set includes the relative positions of ROIs based on geometry. The positions of hole or spike found on the tie plate are shown with red dots in the first image in figure 4.5.

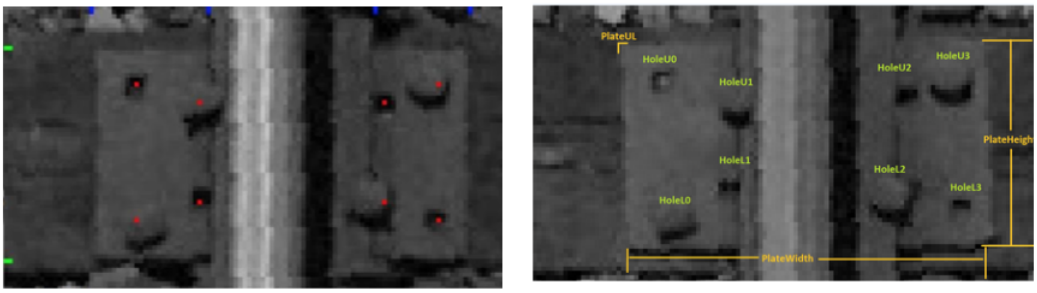

Figure 4.5: ROI localization based on known geometry [36]

An alternate approach was developed to extract the tie plate boundary and is described in the following section.

4.2 Tie plate Extraction

This section covers the tie plate extraction technique developed to accurately find the tie plate boundary. The idea here was to first extract the exact location of the rail, then extract the vertical boundary first and use the cropped result to find the horizontal boundary. The input image to this step is shown in the first subplot in figure 4.6. As one can see in this image, the tie plate is hard to differentiate from the background, especially on the side covered by shadow. In order to have a better input for further processing, the contrast in the image is enhanced using adaptive histogram equalization method called CLAHE [33]. The output is shown in the second subplot in figure 4.6.

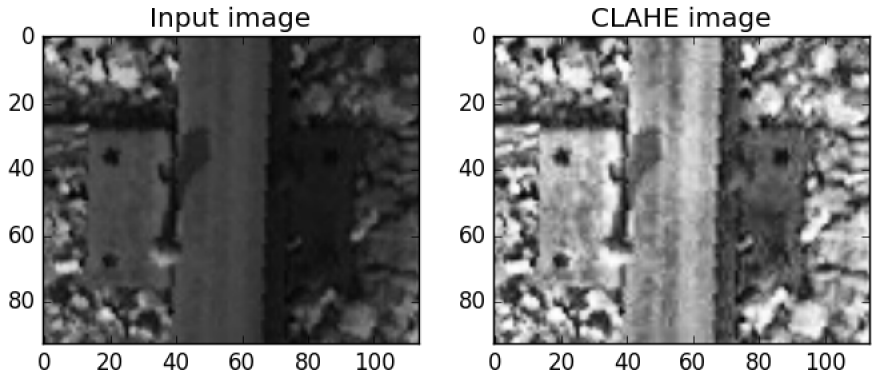

Figure 4.6: Rail Detection

4.2.1 Rail Detection

From the previous stages of processing it is known that the rail is almost at the center of the image in the cropped image shown in figure 4.6. But for accurate localization of tie plate, the rail base is localized again using image intensity based approach.

For rail detection, it can be seen from the images that since the rail is almost perfectly aligned with the vertical image boundary, the columns corresponding to the the rail have almost same intensity in every row for a given column. This information is exploited in the rail detection. So in order to find the rail head boundary, the average intensity of each column is calculated and this is shown in the green plot in first subplot of figure 4.7 under the title 'Rail Detection'. As seen from the plot this does not give much information on where the rail boundary could be. So the next idea was to find the standard deviation of intensity values in each of the column, this is shown by red plot in the same subplot. This does not help since the tie plate region has pixel intensity in the same range in the columns.

The presence of ballast in the region outside of tie is taken into account during the column operation since including rows outside of tie plate provides a good distinguishing factor for the columns with rail region. In order to get a better estimate of the rail boundary region, only the first few rows and the last few rows in the image are considered. The plot of

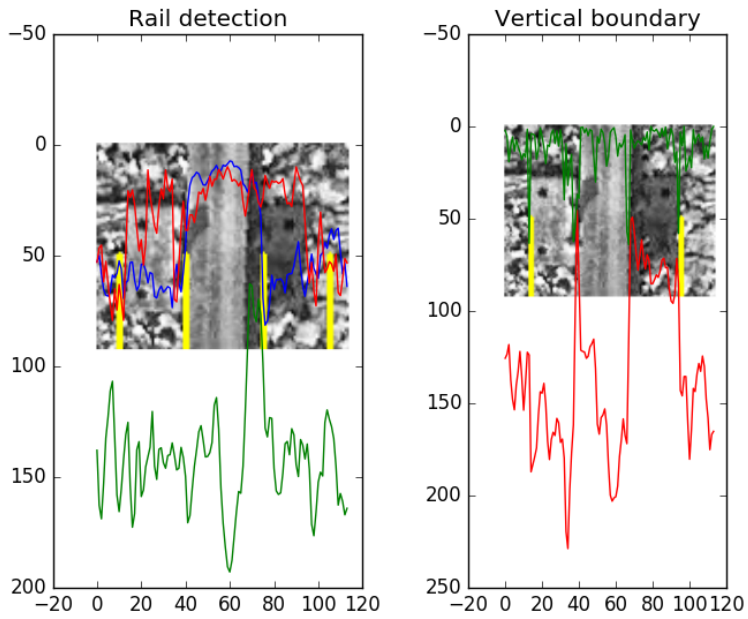

Figure 4.7: Rail Detection

standard deviation of intensity values in the columns of this new sub region of the image are shown in blue plot under subplot titled 'Rail Detection' in figure 4.7.

From this row vector, the rail boundary is found as the column in the image where the standard deviation goes above a set threshold value (40). This position is found for both sides of the row center and is shown by the two yellow vertical lines in the center of the image in the subplot. Since we know the tie plate is always adjacent to the rail head, the initial estimate of the vertical tie boundary is found from a fixed offset distance from the rail boundary and this is shown by the two outer yellow lines in the subplot. The fixed offset distance is set as estimate of the maximum plate width across all images. This is currently considered as 30 pixels.

4.2.2 Vertical tie plate boundary

With this initial estimate of the vertical tie boundary, the columns which fall inside this in the image are looked up for the final estimate of the boundary in the next steps. So the columns starting from the initial estimate to the 12 columns towards the center are considered as probable candidates for boundary. Steps performed next are shown in the second subplot titled 'Vertical boundary' in figure 4.7. The mean value of the columns in the rows around the center of the image are shown in the red plot. As the ties can vary in intensity based on coloring, illumination and shadow cast by the rail, the intensity value as it is may not give reliable information in all cases. So the change in mean intensity value from one column to its next is found and is shown as the green plot in the image. As expected, this plot has peaks around the vertical tie plate boundary due to change in material. The next step is to find the column where this difference is the maximum when moving from 12 pixels inside of the initial estimate of the tie plate boundary towards the end. This is indicated by vertical yellow lines in the subplot.

4.2.3 Horizontal tie plate boundary

Having figured out what works best for the vertical tie plate boundary detection, a similar approach was used to find the horizontal tie plate boundary. The image was first cropped with vertical boundary found in the previous processing stage. Then the first few columns and last few columns are used for processing since these columns contain ballast data and the tie plate boundary can be easily distinguished from it. The change in mean intensity value from one row to its next is found and is shown as the green plot in the the subplot of figure 4.8. The peaks of this plot are found in range of rows considered as probably candidates for the boundary. This region is selected based on the relative geometry from the center of the image and image boundary. The estimated boundary is shown in the horizontal yellow lines in the plot.

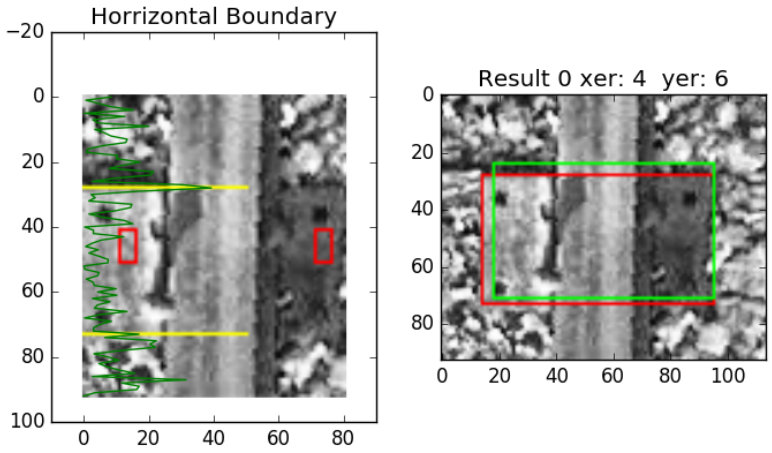

Figure 4.8: Rail Detection

The second subplot in figure 4.8 shows the estimated boundary for the tie plate. Green bounding box shows the initial estimated bounding box and red bounding box shows the bounding box estimated in the above algorithm. As you can notice for this example, the estimate of this algorithm is a better estimate of the boundary than the initial reference.

4.3 Future work

Another important approach could be to remove the shadow region using the entropy filter method [34] and use the resulting image for further processing.

4.4 ROI Classification

Once the ROI is localized the next step is to look for the presence of spike in these regions. The tie plates used in the processing so far have 4 ROIs in each plate. Each of the ROI either has a spike or a hole. So a machine learning model was trained for classifying the ROI into one of the two categories. The ROIs boundaries are shown in the image in figure 4.9. The binary values of the ROI1 in terms of 0s and 1s are also shown in the figure.

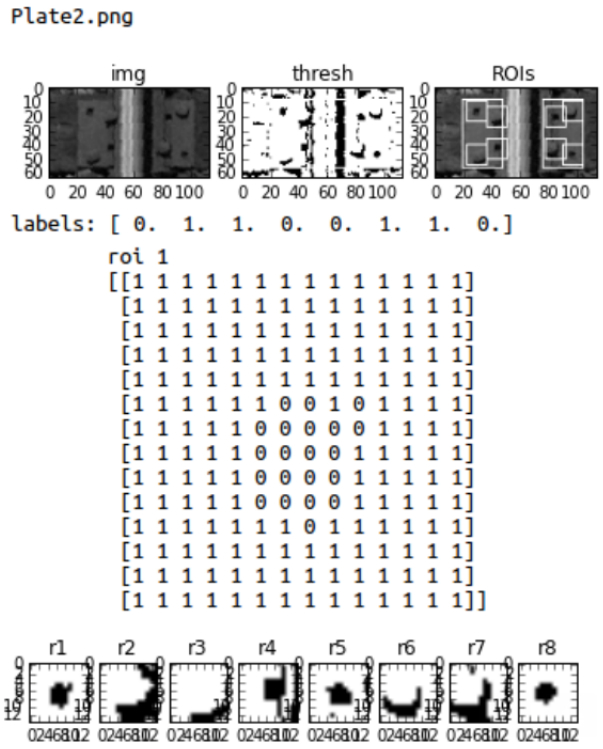

Figure 4.9: ROI

4.4.1 Feature based approach

The inspiration for the feature based approach was the work in [32]. A subset of features described in that work were considered for the initial hole vs spike classification. The features

used are listed below:

a. Feature 1:

$$\frac{\text{Number of foreground pixels}}{\text{Total number of pixels}} \tag{4.1}$$

b. Feature 2:

$$\frac{\text{Distance between center of mass to image center}}{\text{Half of the diagonal distance in the image}} \tag{4.2}$$

c. Feature 3:

$$\frac{\text{x offset of the center of mass from x value at the image center}}{\text{Half of the width of the image}} \tag{4.3}$$

d. Feature 4:

$$\frac{\text{y offset of the center of mass from y value at the image center}}{\text{Half of the height of the image}} \tag{4.4}$$

e. Feature 5:

$$\frac{\text{Minimum value of sum of pixel values in all columns}}{\text{Height of the image}} \tag{4.5}$$

f. Feature 6:

$$\frac{\text{Minimum value of sum of pixel values in all rows}}{\text{Width of the image}} \tag{4.6}$$

These features resulted in a classification accuracy of around 88% with SVM.

Gabor filter response as features:

Gabor filter responses [37] were used to extract the features described above. The filter responses for each of the ROI are shown in figure 4.10. This method gave a classification accuracy of around 89%.

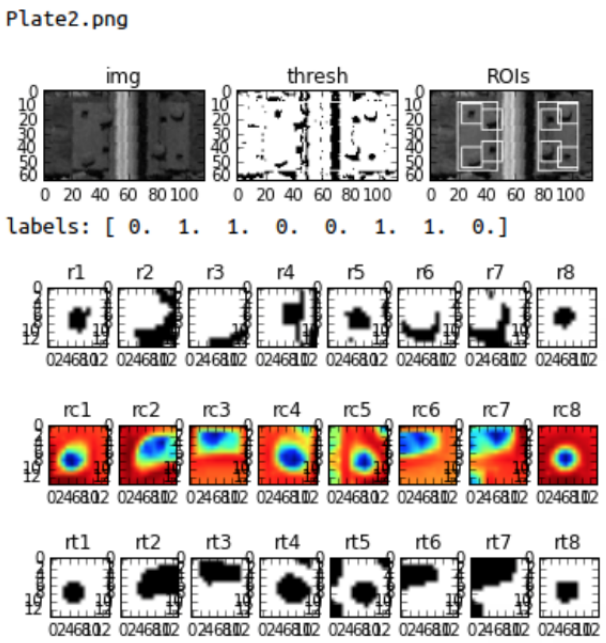

Figure 4.10: Gabor features extracted from ROIs as features

Pixel values of binary image as features:

Going ahead, pixel values of binary image were tried out as features to the classifier. The steps performed are listed below:

a. Input image is converted to gray scale.

b. Use adaptive thresholding to binarize the gray scale image

c. Select the 8 ROIs based on known geometry (coordinates picked from the input file)

d. Divide all pixel values by 255 so the binary image has 0s and 1s

e. Use pixel values of binary image as features

This gave an accuracy of 93.5%. A point to note here is this was achieved with the default parameter settings for SVM in the scikit library [38].

Using additional features:

Seeing binary pixel values give higher accuracy for classification, some of the features which were tried earlier were added to the feature vector and the results were analyzed. The following feature list seemed to give the best accuracy for classification:

a. Pixel values of binary image

b. Old Feature 5:

$$\frac{\text{Minimum value of sum of foreground values in all columns}}{\text{height of the image}} \tag{4.7}$$

c. Old Feature 6:

$$\frac{\text{Minimum value of sum of foreground values in all rows}}{\text{width of the image}} \tag{4.8}$$

This gave an accuracy of 94.24%

Fine tuning:

The SVM classifier parameters were then find tuned to increase the cross validation accuracy. The parameter settings were first left at default selection. SVM classifier gave the best accuracy with polynomial kernel and higher value for the penalty parameter C. Default kernel used by SVM was RBF(Radial Basis function). Accuracy improved to around 95% for hole vs spike classification compared to 92% seen earlier with the default parameters.

There was a lot of fine-tuning done even in the thresholding part for Chickasha dataset to extract better features. Depending on the average pixel intensity on the plate, shadow region is detected and then these plates are pre-processed with an averaging filter. Also different parameters were selected for the adaptive thresholding in this case. With these changes a huge improvement was seen with using Lampasas trained model for testing Chickasha dataset. The results improved from around 73% to around 84%.

The results of training on Lampasas and testing on Lampasas and Chickasha datasets are shown in table 4.1. As expected the classification accuracy for the Lampasas trained model drops when tested with Chickasha dataset. Also as expected the accuracy of classifying ROIs in the sunny side of the tie plate in Chickasha dataset is more than that with all ROIs. But the accuracy with sunny side ROIs is still lesser compared to Lampasas. A point to note here is though the sunny side of the Chickasha dataset has the same illumination as the Lampasas dataset the shadow cast by spikes on the sunny side of plate in Chickasha dataset is more elongated when compared to Lampasas.

Table 4.1: Spike vs Hole Classification accuracy with SVM trained on Lampasas dataset

Classification accuracy with SVM trained on Lampasas dataset					
Test Dataset	No. of train images	No. of test images	Cross Val Accuracy	Cross Val Error	Test Accuracy
Lampasas	3200	6400	96.03%	±1.68%	96.25%
Chickasha (all ROIs)	3200	2451	96.13%	±2.35%	84.37%
Chickasha (sunny side ROIs)	3200	1390	96.13%	±2.35%	86.97%

Ensemble of Classifiers:

An ensemble based majoring voting method was also explored for classification. The different classifiers used as part of the ensemble are SVM with polynomial kernel, Random Forest, Logistic Regression, Decision Tree, k-Nearest Neighbor and SVM with RBF kernel. Results obtained with the Lampasas dataset are shown in table 4.3.

Table 4.2: Spike vs Hole Classification accuracy with SVM trained on Chickasha dataset

Classification accuracy with SVM					
Test Dataset	No. of train images	No. of test images	Cross Val Accuracy	Cross Val Error	Test Accuracy
Chickasha	2000	1100	93.65%	±2.29%	93.36%
Chickasha	3200	3200	93.75%	±1.31%	93.05%
Chickasha	4000	5528	93.72%	±1.82%	92.31%

Table 4.3: Spike vs Hole Classification accuracy on Lampasas dataset

Classification accuracy of different classifiers					
Classifier	No. of train images	No. of test images	Cross Val Accuracy	Cross Val Error	Test Accuracy
SVM Polynomial	3200	6400	96.03%	±1.68%	96.25%
Random Forest	3200	6400	95.09%	±1.18%	95.14%
Logistic Regression	3200	6400	92.13%	±1.92%	90.89%
Decision Tree	3200	6400	92.38%	±1.85%	90.32%
k-NN	3200	6400	96.47%	±1.29%	95.51%
SVM RBF	3200	6400	92.91%	±1.39%	91.68%
Ensemble	3200	6400	96.06%	±1.44%	96.52%

4.4.2 Neural Network based approach

A multi-layered neural network (two convolutional layers and one fully connected layer) was also used as a classifier for the hole vs spike classification process. The accuracy obtained

using that is shown in table 4.4 for Lampasas dataset and 4.5 for Chickasha dataset.

Table 4.4: Spike vs Hole Classification accuracy with NN trained on Lampasas dataset

Classification accuracy with NN trained on Lampasas dataset			
Test Dataset	No. of train images	No. of test images	Test Accuracy
Lampasas	7200	1800	96.16%
Chickasha	9000	2449	82.60%

As expected, the results improved with increase in dataset size used for training the model. This can be seen in the results in table 4.5.

Table 4.5: Spike vs Hole Classification accuracy with NN trained on Chickasha dataset

Classification accuracy with NN trained on Chickasha dataset			
Test Dataset	No. of train images	No. of test images	Test Accuracy
Chickasha	2480	620	93.22%
Chickasha	5312	1328	94.65%
Chickasha	5820	3880	95.20%

Chapter 5

Tie Condition Inspection

The second kind of defect analyzed in this work is the cracks on ties. An example of a clean tie without cracks and a tie with cracks are shown in figure 5.1.

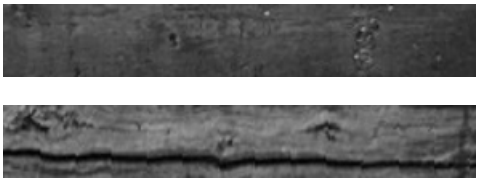

Figure 5.1: Clean tie and a tie with crack

From the example in figure 5.1 it might seem that the task of crack detection could be as simple as applying a thresholding operation on the tie image. Since the crack region has a darker intensity compared to the rest of the tie region, thresholding and inverting the image would make the crack region appear as foreground. But the main challenge is to detect the cracks in presence of ballast. Since the ballast data is quite random, thresholding operation will not be able to differentiate it from the cracks. Examples of this are shown in figure 5.2. The algorithm also needs to ensure that in presence of ballast covering the tie, rest of the non-crack region does not appear as foreground as shown in the two images in figure 5.2.

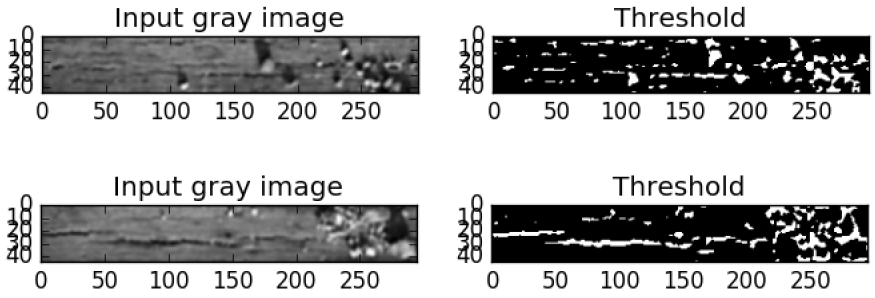

Figure 5.2: Examples of thresholding on challenging cases for tie crack detection

5.1 Detecting cracks

As described in the previous section, the first obvious step towards detecting cracks is thresholding the image. An adaptive thresholding method was used to ensure crack regions under different illumination appear as foreground after the processing. The results of this operation are shown in images in figure 5.3.

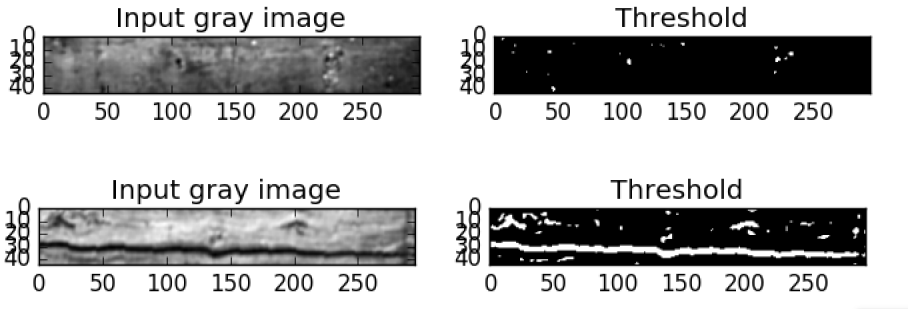

Figure 5.3: Examples of thresholding

The next logical step is to remove the non-crack region appearing as foreground in these images. In order to do this, morphological opening operation with a wide rectangular kernel was used. The idea behind using a opening operation and not an erosion operation is, along with rest of the foreground, erosion would essentially shrink the foreground region overlapping the cracks as well. Since the crack height and width have to be preserved, the

right operation would be morphological opening. A wide rectangular kernel is chosen for the operation since cracks are mostly horizontal in these images. This was the initial approach for crack detection, though it performed well for most cases, it failed to identify cracks which are oblique. To overcome this problem, the morphological closing operation is performed on multiple rotated versions of the images and then the resulting image is rotated back and a superimposed result from these various rotations is chosen as the output. The result of this is shown in image titled 'Superimposed output' in figure 5.4.

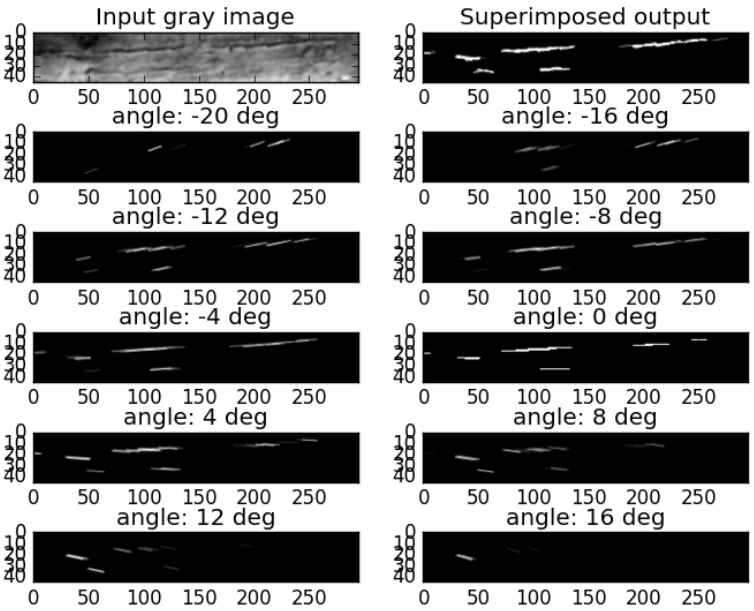

Figure 5.4: Result obtained by applying morphological opening at different rotations of the image

This step of opening operation applied with a wide kernel also removes non crack regions in the foreground to a certain extent. But the ballast data is still an issue and has to be handled separately. Initial idea was to remove the ballast region from the output by considering a Sobel Y operation on the input image instead of thresholding to detect cracks. But that does not work quite well since even for images with ballast the Sobel Y operation gives response

for some of the horizontal edges formed around ballast. The next idea was to use Sobel X operation result and remove the corresponding foreground regions from the Superimposed rotated dilation result. The exact steps performed for this technique are shown in figure 5.5. The Sobel X response is thresholded and then dilated to expand the foreground region. From this output, the contours are found for the foreground region and the corresponding region is set to background in the previously found superimposed dilation result. The contours obtained are shown in figure 5.5.

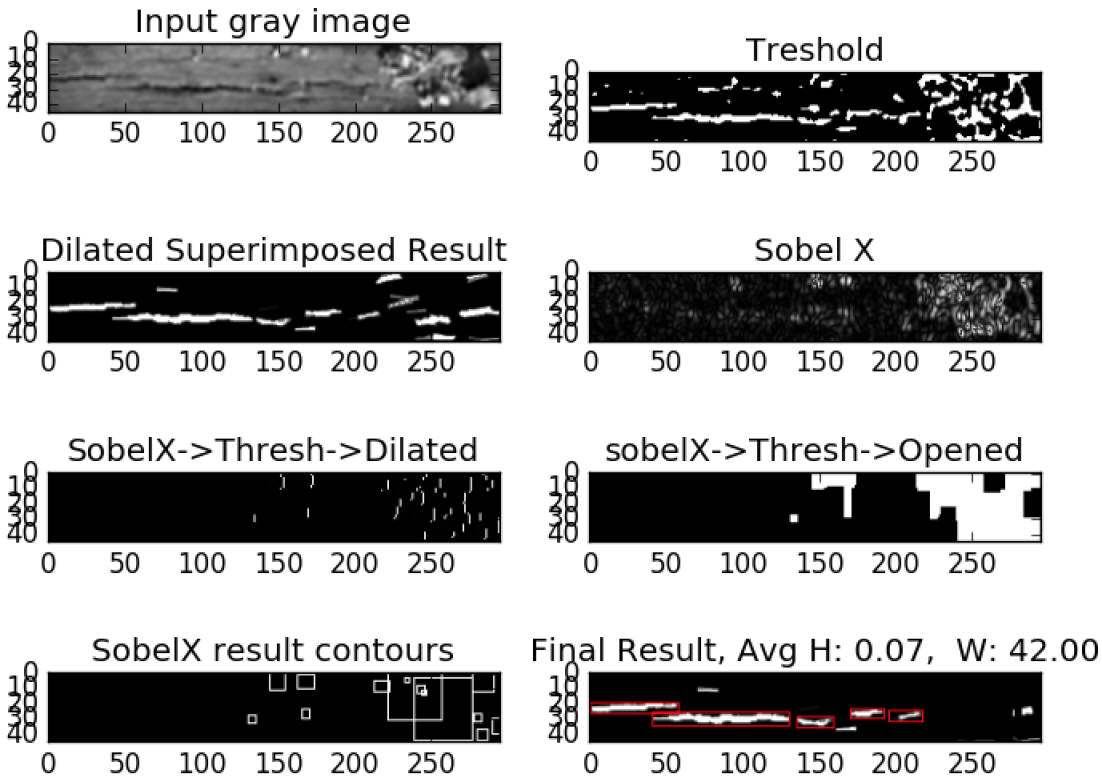

Figure 5.5: Crack detection results

5.2 Quantitative analysis

The contours around the foreground region from the steps described above are used to extract features for classifying images of ties into one of three categories. First category is for clean ties without any crack (label 0), second category is for ties with small cracks (label 1) and third category is for ties with severe deep/long cracks (label 2). Example of these three categories are shown in figure 5.6.

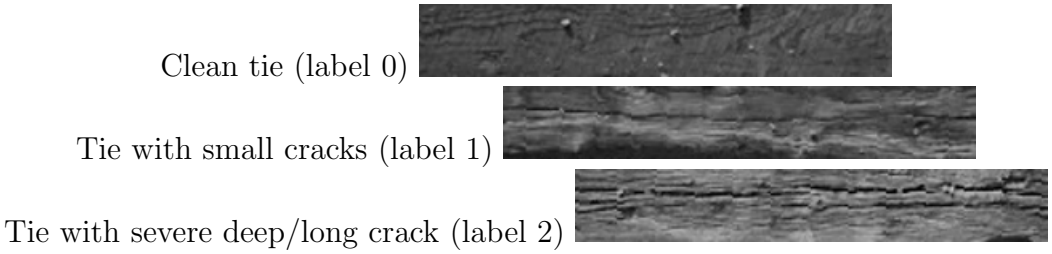

Figure 5.6: Ties grouped into three categories

Features used for training are listed below:

a. Average width of the contours.

b. Average height of the foreground region in the contours.

c. Standard deviation of the width of the contours.

d. Standard deviation of the height of the foreground region in the contours.

e. Number of contours.

f. Maximum width of the contour in the image.

The tie images were labeled and different classifiers were used for training. The results obtained are summarized in table 5.1.

Table 5.1: Classification of ties into one of three categories with 400 training examples

Tie condition assessment accuracy					
Classifier	No. of train images	No. of test images	Cross Val Accuracy	Cross Val Error	Test Accuracy
SVM	400	100	70.22%	±4.11%	73%
Random Forest	400	100	75.45%	±6.43%	75%
Logistic Regression	400	100	75.97%	±4.41%	83%
Decision Tree	400	100	72.98%	±6.36%	65%
KNN	400	100	73.20%	±3.33%	76%
Ensemble	400	100	77.46%	±3.51%	81%

Table 5.2: Classification of ties into one of three categories with 700 training examples

Tie condition assessment accuracy					
Classifier	No. of train images	No. of test images	Cross Val Accuracy	Cross Val Error	Test Accuracy
SVM	700	200	75.14%	±2.82%	77%
Random Forest	700	200	77.27%	±3.27%	77%
Logistic Regression	700	200	78.86%	±5.27%	83.5%
Decision Tree	700	200	73.55%	±5.76%	75%
KNN	700	200	76.56%	±2.24%	78%
Ensemble	700	200	79.86%	±3.96%	82%

As seen in the results, increase in training data seemed to result in better accuracy and lower cross validation error. Though the resulting accuracy was not very high, the classifier did not make any wrong predictions between categories 0 and 2 in both training sizes.

5.3 Failure cases

There are cases where the algorithm is still not intelligent enough to differentiate arbitrary objects in the image from cracks. Once such example is shown in figure 5.7. In this example the shadow cast by a foreign object is falsely detected as crack and is predicted as category 1 (ground truth label for this is 0) during test classification.

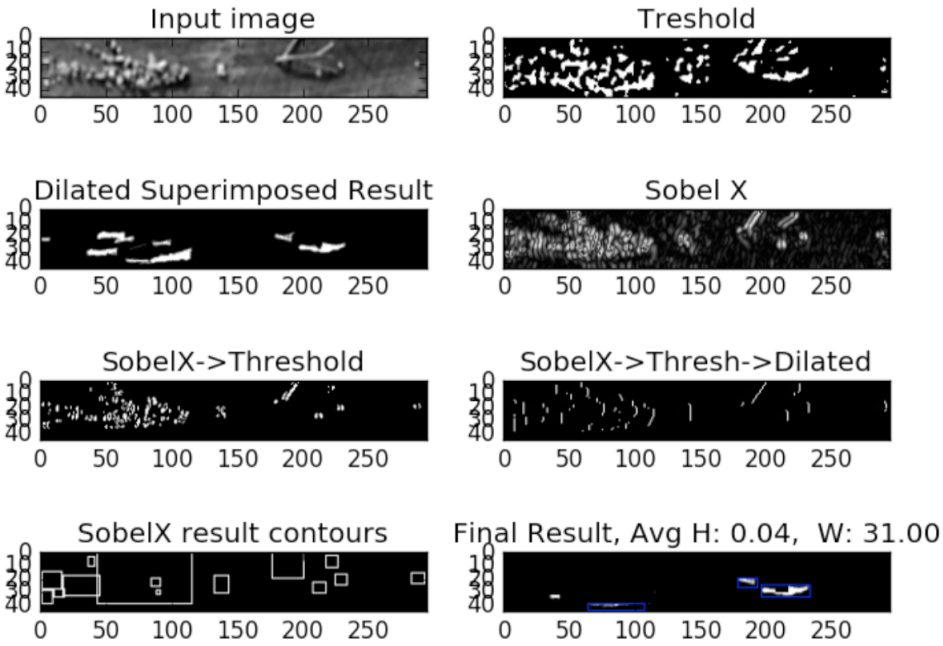

Figure 5.7: Crack detection result with foreign objects

Another case of misclassification is seen with images which have a darker tie area. In these images crack region does not stand out and does not appear as foreground region with the current settings for adaptive thresholding. An example of such a result is shown in figure 5.8. CLAHE was seen to improve the contrast but it also results in a lot of false detection for cracks, hence was not considered as a solution.

A good recommendation of future work would be to develop smarter crack detection process for feature extraction. And another area of improvement could be finding more features for better classification.

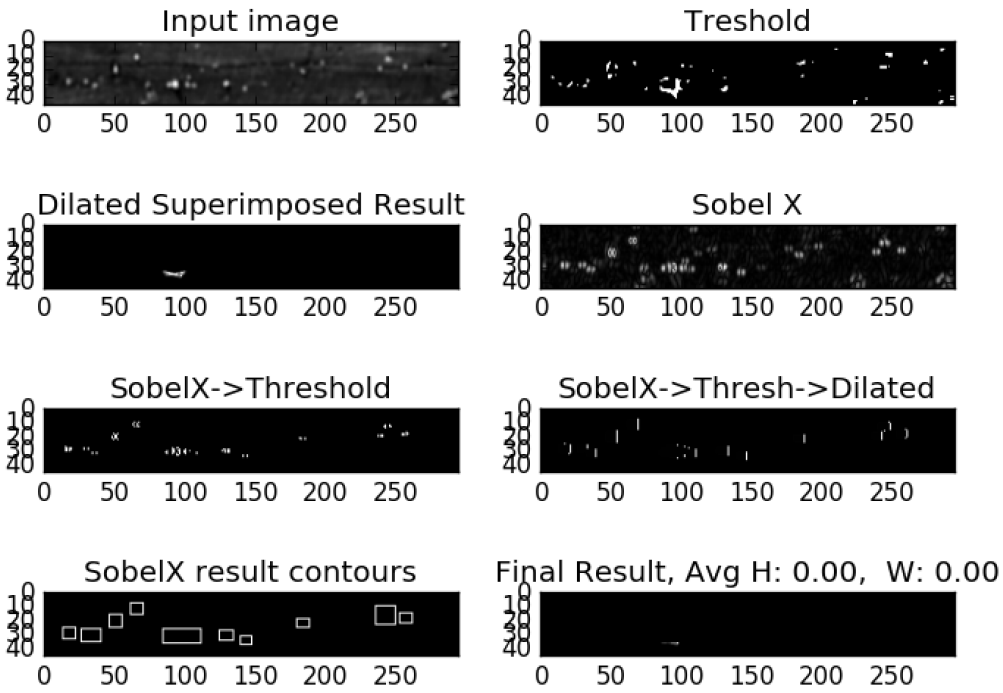

Figure 5.8: Crack detection result on darker images

Chapter 6

Conclusion and Recommendations

6.1 Recommendations

This section lists the possible areas for future work.

6.1.1 Variations in rail alignment

An area for improvement is to make the rail detection algorithm more robust to cases with curved rails. Currently the rail detection and tie localization algorithms are based on the assumption that rails are straight and do not curve. This assumption is exploited in using approaches such as Sobel X and Sobel Y filter operations to find the rails and ties.

With regard to the ROI classification for missing spike detection, currently the image resolution plays a bottleneck in recognizing certain regions as spikes or holes even for the ground truth labeling. Currently these regions are ignored for classification since these regions are humanly impossible to tell as to which category they belong.

6.1.2 Using Rail Following techniques

An important thing to note in this research work is the UAV used to capture images of the railroad is manually controlled by a pilot using a remote. Instead an autonomous rail following algorithm [39] could be used onboard an UAV for collecting images for post process inspection. The linear nature of the railroad is exploited in this approach for navigation.

6.1.3 Different datasets

Images of railroads from various subdivisions were available from Bihrle for this research [36]. Images from El Paso SubDivision are shown in figure 6.1. These images have a slightly lower resolution than the Lampasas and Chickasha dataset used in this work. Also in some images there is a bit of motion blur. But overall the dataset looks pretty clean and the sun is at a good position (i.e. sometime between 11am and 1pm) and both ends of the rail do not have any shadow.

Figure 6.1: Images from El Paso Sub Division [36]

Images from Clifford SubDivision are shown in figure 6.2. This dataset has fairly good resolution and not much motion blur. This dataset is similar to the Chickasha dataset in that the sun is at a severe angle (i.e. very early morning or very late in the evening). So the spikes cast long shadows and one half of the tie plate is completely in shadow.

Second set of images from Clifford SubDivision are shown in figure 6.3. These images probably do not have the resolution needed to do a good job of classification. However, this has a new tie plate geometry with only two spikes on either half of the tie plate.

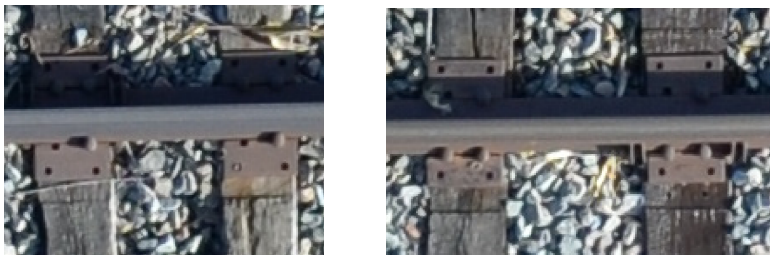

Figure 6.2: Images from Clifford Sub Division [36]

Figure 6.3: Image from Clifford Sub Division with two spike region on each tie [36]

Images from Decatur SubDivision are shown in figure 6.4. These images have great resolution but the problem again with these is the shadow of the rail covering part or all of a tie plate half.

Figure 6.4: Images from Decatur Sub Division [36]

6.2 Conclusion

Track defects are one of the main causes for train derailment. And this work was targeted to analyze tracks for two kinds of defects from track images captured by an UAV; spiking

pattern on tie plates and cracks on ties. The main challenge in identifying spikes on the tie plate was the motion blur, low illumination and shadow cast by the rail. In the spike vs hole classification process, the unidentifiable regions for ground truth labeling were removed for training and testing of the classifier and the accuracy obtained with the remaining set was around 96%. For the tie condition analysis, the cracks were detected in the tie images and the ties were grouped into one of the three categories depending on the severity of the crack. The classifier was trained to predict the condition of the tie in the test set based on features from the contours found around the cracks in the image. This gave an accuracy of 83% for 3-way classification but there were no misclassifications seen between categories 0 and 2 throughout the test dataset. Though, these were small subset of the track defects to be analyzed, UAVs, Computer Vision and Machine Learning put together certainly has a great potential for targeting other kinds of defect analysis which is possible to be done visually.

Bibliography

[1] Shah, Mubarak. "Automated Visual Inspection/Detection of Railroad Track." Final Report, Computer Vision Lab, University Of Central Florida (2010).

[2] Rail inspection, From Wikipedia, the free encyclopedia. Available: `https://en.wikipedia.org/wiki/Rail_inspection`

[3] Rail inspection, "An Inspection Car on the Pennsylvania Railroad", a wood engraving drawn by Charles Graham and published in "Harper's Weekly", November 1882.

[4] Rail inspection, "An Inspection Car En Route", a wood engraving drawn by F. Cresson Schell and published in "Harper's Weekly", November 1891.

[5] Vision Systems Design, Vision helps spot failures on the rail, January 2015. Available: `http://www.vision-systems.com/articles/print/volume-20/issue-1/features/vision-helps-spot-failures-on-the-rail.html`

[6] Track Geometry Car, Photo by Alexey Anisimov on 13 September 2006 somewhere near Pugachevsk, Russia.

[7] NYC Subway Plasser Theurer Track Geometry Car TGC3 at Jay Street-Boro Hall in Brooklyn, New York, Photo by Adam E. Moreira.

[8] Li, Ying, et al. "Component-based track inspection using machine-vision technology." Proceedings of the 1st ACM International Conference on Multimedia Retrieval. ACM, 2011.

[9] DJI Phantom 2 Vision+ V3 hovering over Weissfluhjoch, Photo by Lino Schmid.

[10] AirRobot AR180, Micro Unmanned Aerial System, System Description, November 2013. Available: https://www.unmannedsystemssource.com/wp-content/uploads/2014/05/WP-AirRobot-AR180C-em.pdf

[11] Inspire 1, DJI, Product page, Available: http://www.dji.com/product/inspire-1/camera#sub-feature

[12] eXom, senseFly, Product page, Available: https://www.sensefly.com/fileadmin/user_upload/sensefly/documents/brochures/eXom-brochure-en.pdf

[13] Inspire 1, DJI, Photo available: http://quadcopterhq.com/dji-inspire-1-review/

[14] eXom, senseFly, Photo available: http://www.uhurufly.com/#a-different-view

[15] Sawadisavi, Steven. Machine-vision inspection of railroad track. Diss. University of Illinois at Urbana-Champaign, 2009.

[16] Basic Thresholding Operations, OpenCV 2.4.13.0 documentation, OpenCV Tutorials, Available: http://docs.opencv.org/2.4/doc/tutorials/imgproc/threshold/threshold.html

[17] Image Thresholding, OpenCV 3.1.0-dev documentation, OpenCV-Python Tutorials. Available: http://docs.opencv.org/master/d7/d4d/tutorial_py_thresholding.html

[18] Image Gradients, OpenCV 3.0.0-dev documentation, OpenCV-Python Tutorials. Available: http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_gradients/py_gradients.html

[19] Use of a Gabor filter for Chinese OCR. Available: https://en.wikipedia.org/wiki/Gabor_filter

[20] Histogram Equalization & Specification, ECE 468/CS 519: Digital Image Processing, Hawkes Bay Image describing Histogram Equalization. Available: http://web.engr.oregonstate.edu/~sinisa/courses/OSU/ECE468/lectures/ECE468_4.pdf

[21] 2003 R. Fisher, S. Perkins, A. Walker and E. Wolfart. Available: http://homepages.inf.ed.ac.uk/rbf/HIPR2/morops.htm

[22] Eroding and Dilating, OpenCV 2.4.13.0 documentation, OpenCV Tutorials. Available: http://docs.opencv.org/2.4/doc/tutorials/imgproc/erosion_dilatation/erosion_dilatation.html

[23] 2010 Saravanan Thirumuruganathan, A Detailed Introduction to K-Nearest Neighbor (KNN) Algorithm. Available: https://saravananthirumuruganathan.wordpress.com/2010/05/17/a-detailed-introduction-to-k-nearest-neighbor-knn-algorithm/

[24] E. M. Mirkes, KNN and Potential Energy: applet. University of Leicester, 2011. Available: https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[25] Introduction to Support Vector Machines, OpenCV 2.4.13.0 documentation, OpenCV Tutorials, Machine Learning. Available: http://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html

[26] Fletcher, Tristan. "Support vector machines explained." Online]. http://sutikno. blog. undip. ac. id/files/2011/11/SVM-Explained. pdf.[Accessed 06 06 2013] (2009).

[27] Aizerman, Mark A.; Braverman, Emmanuel M. and Rozonoer, Lev I. (1964). "Theoretical foundations of the potential function method in pattern recognition learning". Automation and Remote Control 25: 821–837.

[28] Neural Network Part 1: Setting up the Architecture, Stanford University CS231n Convolutional Neural Networks for Visual Recognition course notes by Andrej Karpathy, Available: http://cs231n.github.io/neural-networks-1/

[29] Hornik, Kurt, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators." Neural networks 2.5 (1989): 359-366.

[30] IMAGE RECOGNITION WITH NEURAL NETWORKS HOWTO, Neuroph, v2.6. Available: `http://neuroph.sourceforge.net/image_recognition.html`

[31] Geometric Image Transformations, OpenCV 3.0.0-dev documentation, OpenCV API Reference. Available: `http://docs.opencv.org/3.0-beta/modules/imgproc/doc/geometric_transformations.html`

[32] Li, Ying, and Sharath Pankanti. "Anomalous tie plate detection for railroad inspection." Pattern Recognition (ICPR), 2012 21st International Conference on. IEEE, 2012

[33] 2016, Histograms - 2: Histogram Equalization, OpenCV 3.1.0-dev. Available: `http://docs.opencv.org/master/d5/daf/tutorial_py_histogram_equalization.html`

[34] scikit-image, Image Processing in Python documentation. Available: `http://scikit-image.org/docs/dev/auto_examples/plot_entropy.html`

[35] Forsyth, D.A. and Ponce, J. (2003). Computer Vision: A Modern Approach. Prentice Hall, Upper Saddle River, New Jersey

[36] Stanton Coffey, Bihrle Applied Research Inc.

[37] 2014, Krishna Murthy, Gabor Filters: A practical overview, Computer Vision Tutorials. Available: `https://cvtuts.wordpress.com/2014/04/27/gabor-filters-a-practical-overview/`

[38] Support Vector Machines, scikit-learn version 0.17.1 documentation, Available: `http://scikit-learn.org/stable/modules/svm.html`

[39] E. M. Smith, "A Collection of Computer Vision Algorithms Capable of Detecting Linear Infrastructure for the Purpose of UAV Control", Masters thesis, Mechanical Engineering, Virginia Polytechnic Institute and State University, 2016.