Automatic Bridge Crack Detection – A Texture Analysis-Based Approach

Sukalpa Chanda[1], Guoping Bu[2], Hong Guan[2], Jun Jo[1], Umapada Pal[3], Yew-Chaye Loo[2], and Michael Blumenstein[1]

[1] School of Information and Communication Technology, Griffith University, Queensland 4222, Australia
[2] School of Engineering, Griffith University, Queensland 4222, Australia
[3] Computer Vision and Pattern Recognition Unit, Indian Statistical Institute, Kolkata-700108, India

Abstract. To date, identifying cracks in bridges and determining bridge conditions primarily involve manual labour. Bridge inspection by human experts has some drawbacks such as the inability to physically examine all parts of the bridge, sole dependency on the expert knowledge of the bridge inspector. Moreover it requires proper training of the human resource and overall it is not cost effective. This article proposes an automatic bridge inspection approach exploiting wavelet-based image features along with Support Vector Machines for automatic detection of cracks in bridge images. A two-stage approach is followed, where in the first stage a decision is made as whether an image should undergo a pre-processing step (depending on image characteristics), and later in the second stage, wavelet features are extracted from the image using a sliding window-based technique. We obtained an overall accuracy of 92.11% while conducting experiments even on noisy and complex bridge images.

Keywords: Crack Detection, Wavelet-based Crack Detection, Automatic Bridge Inspection, SVMs.

1 Introduction

Physically investigating bridge conditions sometimes becomes unfeasible due to several factors such as the physical surroundings of the bridge, lack of expert knowledge and human resources. Bridges for the purpose of for maintenance and repair requires timely decision-making. Many bridge authorities consult Bridge Management Systems (BMSs) to manage their routine inspection information and to decide on consequent maintenance services. With the advent of sophisticated devices and powerful computers, an effort to automatically conduct bridge inspection has been noted in the recent past. Unfortunately, the proposed methods were not fully capable of addressing the challenges in automatic crack detection. The main difficulties encountered in automatic crack detection methods are variable lighting conditions, random camera/view angles, and random resolution of bridge images. Moreover, we found that

N. El Gayar et al. (Eds.): ANNPR 2014, LNAI 8774, pp. 193–203, 2014.

automatic crack detection gets even harder where the background texture randomly changes and hence segmentation of background and foreground elements becomes very challenging. This article proposes a non-trivial method which addresses the above mentioned challenges efficiently. It relies on a two-stage approach. At first, upon initially analyzing the characteristics of the pixel values in 'R' , 'G' and 'B' channels, the image is identified as either a 'complex image' or a 'simple image'. If the image is identified as a 'complex image' then we need to execute a pre-processing step, otherwise the image is directly processed for feature extraction. Using a non-overlapping sliding window, texture analysis–based features are extracted from the image region beneath the sliding window. Later those features are passed to a Support Vector Machine (SVM) classifier to decide whether the region beneath the sliding window contains a crack or not.

The rest of the article is organized as follows. In Section 2, we discuss related published work, and in Section 3 we describe methodology including the data acquisition process and experimental framework. In Sections 4, the pre-processing step is discussed. Feature extraction and classification approaches are discussed in Section 5. In Section 6 we discuss our experimental results, and finally Section 7 puts forward the conclusions of our paper.

2 Related Work

This section describes some of the existing works in automated crack detection using image processing and pattern recognition techniques. Lee et al. [1] proposed an algorithm for automatic detection of cracks. Their proposed method consisted of crack detection and crack tracing using the difference between the intensity of a crack and its background. Ehrig et al. [2] introduced three different crack detection algorithms namely template matching, sheet filtering based on Hessian eigenvalues, and percolation based on the phenomenon of liquid permeation. Their study focused on determining the suitability of each for crack detection. Mohajeri and Manning [5] proposed a method to identify cracks in concrete using directional filter. They stated that the crack is longitudinal if there is a high concentration of object pixels in a narrow interval of x (transverse) coordinates, and it is transverse if there is a high number of object pixels in a narrow interval of y (longitudinal) coordinates. Tong et al. [6] developed a new method of crack image processing using a pre-processing step which separates crack pixels from the background of the image. Abdel-Qader et al. [3] compared edge-detection algorithms in the context of bridge crack detection using a threshold based approach. Jahanshahi and Masri [4] proposed a morphological operations and Otsu's thresholding based method for segmentation. The purpose of the segmentation process was to reduce unnecessary data in the original image. The appropriate structural element size (in pixels) for the morphological operation was set based on camera focal length, the distance from the object to the camera, camera sensor resolution and size, as well as crack thickness. Oh et al. [8] developed an automatic system for bridge inspection that used median filter in order to remove noise for effective crack detection. Later, morphological operations were applied to determine the

connections between crack segments. Lee et al. [9] developed a bridge inspection system that consisted of a robot transportation vehicle, a hydraulic transportation boom and a remote-controlled robot. The remote-controlled robot was used to acquire images of bridge slabs and girders. These images were then sent to an embedded computer for crack detection. Miyamoto et al. [10] developed an automatic crack recognition system to detect crack width on concrete surfaces, where the system could recognize the location and width of cracks. The crack width was computed using the information of difference in brightness in the cracked and non-cracked areas. Flash thermography was used for detecting cracks on concrete surfaces by Sham et al. [11]. In this article we perform a comparative analysis between two different forms of texture analysis features for the purpose of bridge crack detection.

3 Methodology

3.1 Image Acquisition and Dataset Details

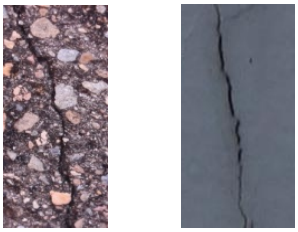
We used 50 different images of bridges with a resolution of 5616 × 3744 (21 megapixels), all with random lighting conditions. Based on certain image characteristics those images can be categorized distinctly into two types- "Normal" and "Complex". In "Normal" images we noticed a near consistent background all along the image with a high contrast between the foreground and the background. Whereas in "Complex" images we noted a rapid change in intensities in both foreground and background, or the background was fused with the foreground. We considered 1369 "window" regions (image patches/sub-images) of type 'crack' and 'non-crack' from those images and our experiment was done on these 1369 sub-images.

3.2 Method Overview

At first using a heuristic we automatically try to determine whether an image is of type "normal" or "complex". For "normal" images no further pre-processing is required but for "complex" images we had to perform certain pre-processing steps before features were extracted. In order to analyse the bridge image locally, we deployed a sliding window strategy. For better computational efficiency, a non-overlapping 30x40 pixel window is glided over the entire image and from the region beneath each window (we call them 'window regions') wavelet (and also Gabor filter) features were extracted. Such feature from each window region is classified into a 'crack' or 'non-crack' region by an SVM classifier. The size of the sliding window is set after an empirical analysis of the images. It is noted that the cracks in the images were approximately 25 pixels in width, so a 'crack' region is supposed to contain the crack with the background part, whereas the 'non-crack' region should have the background element only.

3.3 Challenges with "Complex" Image and Their Characteristics

During initial experiments we noticed that our features were able to perform well when the images consisted of a near consistent background all along the image with a high contrast between the foreground and the background, which we term as 'normal' images. However, our features did not perform well with 'complex' images, which had a rapid change in intensities in both foreground and background, or when the images were dull in nature (where the background was fused with the foreground). We noticed that for the 'normal' images, the values of the 'R', 'G' and 'B' channels for a pixel were very close to each other (low standard deviation for all 3 values) and the range of those values was quite extensive. However for 'complex' images, the values of the 'R', 'G' and 'B' channels for a pixel were quite different to each other (high standard deviation for all 3 values) and the range of those values was quite narrow. Using this information (heuristic) we can easily cluster all input images broadly into two groups - 'complex' and 'normal'. Example of a 'complex' and a 'normal' bridge image are shown in Figure 1.



(a) (b)

Fig. 1. (a) A 'complex' bridge image (b) A 'normal' bridge image

3.4 Motivation Behind Pre-processing Step

After an image is categorized as a 'complex' image, we needed to further process it so that the crack mark became more prominent with respect to its surroundings. After undergoing a series of colour space conversions and filtering of values in various colour space channels, we could obtain an equivalent grey scale image of the complex image. Further on we noticed that if we could process this grey scale equivalent image of the 'complex' images by a contrast stretching algorithm then the same features becomes effective. So we took a two stage approach to deal with this process. In the first stage using our heuristics we decided whether an image is of type 'complex' or type 'normal'. If 'complex' then we process the whole image using our pre-processing methods and then forward it to the feature extraction process. If an image is 'normal' then we do not process it with any pre-processing method and directly start extracting features from it. Since at the current time we could not acquire a large number of bridge images, instead of dividing our entire corpus into training and test subsets we implemented a five-fold cross-validation scheme. The features extracted from all 'crack' and 'non-crack' windows were put together and then we divided all feature vectors into five sets; we used 4 sets for training and the remaining one for

testing. The process is repeated 4 more times so that each of the remaining 4 sets in the last training set is used for testing. We also noticed that if we implemented a five-fold cross-validation scheme involving feature vectors from both 'complex' and 'normal' images simultaneously then the accuracy diminishes. We investigated further and found that mostly feature vectors from 'complex' images were being incorrectly classified. Even through tuning values of the SVM parameters the situation did not change. Only after removing all feature vectors that belonged to 'complex' images, the accuracy improved. However, upon implementing a five-fold cross-validation scheme for feature vectors obtained exclusively from 'complex' images, we obtained almost similar accuracy as we achieved on 'normal' images. It is worth mentioning here that the best optimized parameters for feature vectors from two different types of images were quite different.

4 Pre-processing

We executed our pre-processing step on only those images that in our first stage were identified as 'complex' images. The images were in RGB format, but we transformed that to a HSV colour space. The reason behind this is in HSV space the image intensity can be separated from the colour information. Also this transformation for 'complex' type images provided us robustness against lighting changes, and shadows. In the HSV colour space 'Hue' defines the colour component and ranges between 0-1.0 (another scale is 0-360 degree), 'Saturation' describes how white the colour is; whereas the 'value' defines the lightness component in a pixel (0 means white and 1 means completely black). During our initial experiments we noticed that highlighting the crack can be achieved by analysing the Hue and Saturation channel value in the image, and then manipulating them to our desired values. If in a pixel the Hue value is >=0.9 and the corresponding Saturation value is <=0.2 then we change them to Hue=0.6, Saturation=1.0 and Value (intensity/brightness) =0.1; otherwise we set saturation to 0.2 and keep the rest of the two channel values intact. With the first option we are ensuring that the crack pixels are labelled as a proper blue colour with a dark shade (see figure 2b; in the Hue axis 0.6 resembles blue and a Saturation of 1 ensures that the pixel can be visually perceived as the true blue, the low intensity value ensures darkness with respect to the surroundings). With the second option we try to ensure that rest of the pixels become a more grey-like shade by selecting a low saturation value. From the final output image in figure 2(f) it is clearly evident that using our pre-processing step we can easily convert a 'complex' image type as shown in figure 2(a) to appear like a 'normal' image. If we compare figure 1(b) with figure 2(f), it is evident that they look visually similar.

Our pre-processing steps can be outlined as follows:

(i) RGB to HSV colour space transformation;
(ii) Check the range of Hue and Saturation values of a pixel and set the values of all H,S,V channels accordingly.
(iii) Conversion to RGB.
(iv) Then convert the RGB to Grey scale.

(v) Enhance contrast of the grey scale image by applying histogram equalization technique on the grey scale image.
(vi) Perform final filtering on grey scale values (fix all grey scale values above a certain threshold to one particular high grey scale value) to get the desired output image.

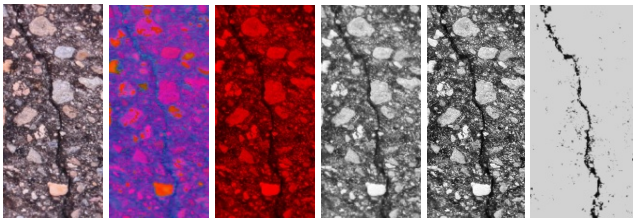


Fig. 2. (a) Extreme left - an original input image, (b)-(e) same image in various intermediate stages; (f) extreme right - corresponding final output image after pre-processing

5 Feature Extraction and Classification

We were keen to study and perform a comparative analysis between different texture analysis-based methods for the purpose of crack detection. All those different features were extracted from the sliding window that glided over the image. We experimented with two different texture-analysis based features, which included Gabor filter features and Daubchies Wavelet features. Wavelet features outperformed the Gabor filter features in our experiments. Since description on all those features are easily available we are not describing them further vividly but only providing a short description of the Gabor and wavelet feature.

5.1 Gabor Filter

The Gabor filters are band-pass filters which essentially do texture analysis. The response of these filters is the product of a Gaussian envelope function multiplied with a complex oscillation [17].

The Gabor filter response image with respect to a crack region and a non-crack region is shown in figure 3. Details about Gabor filter can be found in [18].

5.2 Wavelet Features

The wavelet transform is a useful technique used to analyze non-stationary signals in time-frequency domain. Daubechies wavelets are a family of orthogonal wavelets defining a discrete wavelet transform. This consists of 4 scaling function coefficients and 4 wavelet function coefficients. The four scaling function coefficients used in our experiments were as follows:

$$\frac{1+\sqrt{3}}{4\sqrt{2}}, \frac{3+\sqrt{3}}{4\sqrt{2}}, \frac{3-\sqrt{3}}{4\sqrt{2}}, \frac{1-\sqrt{3}}{4\sqrt{2}}$$

The sliding window is glided all over the grey scale image. The image region beneath each window is copied. We extracted features after size normalizing each such grey scale image regions obtained from the sliding windows to 32 *x* 32 dimensions. The wavelet response image with respect to a crack region and a non-crack region is shown in figure 4. Details of the feature can be found in [14].

5.3 Classifier Details

In our experiments, we have used Support Vector Machines (SVMs) for classification. In our experiments, we noted that the Gaussian kernel SVM outperformed other non-linear SVM Kernels; hence we are reporting our classification results based on the Gaussian kernel only. The best Kernel parameters were selected for each class type by means of a series of validation experiments. The best optimized results were obtained when (1/2σ2) in the Gaussian kernel was set to values such as 80.00 (while dealing with 'normal' images) and 9.00 (while dealing with 'complex' images) with the penalty multiplier value set to1.

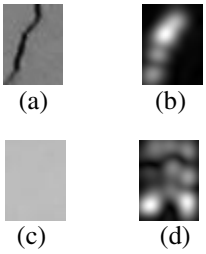


Fig. 3. (a) Top left - an original input image with crack, (b) Corresponding Gabor response of crack image.(c) Bottom left - an original input image without crack, (d) Corresponding Gabor response of non-crack image.

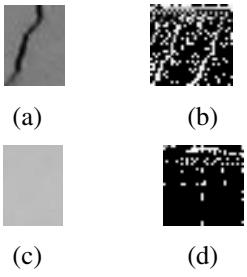


Fig. 4. (a) Top left - an original input image with crack, (b) Corresponding Wavelet response of crack image.(c) Bottom left - an original input image without crack, (d) Corresponding Wavelet response of non-crack image.

6 Results and Discussion

We did some analysis on our experimental results to provide more insight to our proposed method. As we have mentioned earlier, when five-fold cross validation was implemented separately on feature vectors from 'complex' and 'normal' images we obtained higher accuracy compared to when we implemented five-fold cross validation on feature vectors from both window image types together. Here we are reporting accuracy only on Wavelet features. This fact is depicted in Table 1. We can see that we obtained 87.06% accuracy when feature vectors from 'complex' and 'normal' images were considered together. Similarly while considering feature vectors from only 'normal' images we obtained 93.26% (873 correctly classified considering 936 samples from 'normal' images during five-fold cross validation) whereas while considering feature vectors from only 'complex' images (388 correctly classified considering 433 samples during cross validation) we obtained 89.60%. Thus the average accuracy of our systems becomes 92.11%. In Table 2 we try to inspect the performance of our system by training it using feature vectors exclusively obtained out of one particular image type ('complex'/ 'normal') and testing it on the other image type. In Table 3 we provide a comparative analysis of two different features and followed by an error analysis in Sub-section 6.3.

Table 1. Five-fold cross validation accuracy

Image Type	Accuracy	
Complex Image and Normal Image	87.00%	
Only Normal Image	93.26%	92.11%
Only Complex Image	89.60%	

6.1 Effect on Performance Due to Complex (Normal) Images in Training (Test) Set

We have mentioned earlier that our experiments involved two different types of images: 'complex' and 'normal'. We were interested to see what happens when we only train our classifier with feature vectors of "crack" and "non-crack" image regions obtained from all 'complex' ('normal') regions and test them with "crack" and "non - crack" image regions obtained from all 'normal' ('complex') images. Since during our earlier experiments we obtained the highest accuracy while using wavelet features, we are reporting this experiment with the wavelet feature only. From the results we can conclude that 'normal' images turned out to be much better as a training set and provides us a more generalized learning model.

Table 2. Effect of training image types on accuracy

Train set type	Test Set type	Accuracy
Complex Image	Normal Image	78.95% (739 out of 936)
Normal Image	Complex Image	87.06% (377 out of 433)

6.2 Comparison between Texture Analysis Based Features

Here in Table 3 we provide a comparison between the two different feature extraction methods. Note here we are reporting the accuracy while dealing with all feature vectors simultaneously irrespective of the image type (complex/normal). We obtained highest accuracy with Wavelet features.

Table 3. Comparison between two different features

Gabor Filter	74%
Wavelet	87%

6.3 Error Analysis

Upon analyzing the errors we noticed that most of the time 'window regions' with a blurred appearance were misclassified to the wrong class. This happened to 'window regions' obtained from both 'complex' and 'normal' image types where the foreground element was not prominent compared to the background element in the images and that they tend to fuse with each other. Nevertheless, it is worth mentioning here that in such images our contrast stretching algorithm did not perform well, which is one of the reasons behind not recognizing the cracks. An example of such an image is shown in figure 5. It should be noted that the region marked within the rectangular area highlights a crack mark, which is almost invisible there; however the crack mark is more visible in regions above the rectangular area.

Fig. 5. An invisible crack mark within the rectangular region

7 Conclusions and Future Works

In this paper, we have investigated the problem of automatic bridge crack detection in bridge images. Two different features (Gabor filter and Wavelet) were evaluated for

this purpose. The novel issue with our proposed system is that we have obtained encouraging accuracy even while dealing with complex bridge image types with heterogeneous background and foreground characteristics. However, the present preprocessing is based on a threshold approach, which prevents it to be equally efficient under all kind of images. In future we shall look for a more robust technique to accomplish our objective. Further future work includes autonomous image data acquisition using devices such as robots or UAVs. Obtaining an image at a specific position in high precision is not a trivial task, when using an autonomous device. Various sensors, such as optical, acoustic and magnetic sensors, may aid in this task. Multiple sensors, based on individual specialties, are commonly used in order to complement limitations imposed by certain sensors and thus enrich the perception of single sensors. However, it is challenging to integrate the heterogeneous types of sensory information and produce useful results. A pilot study of the likelihood-based data fusion system has been implemented for robot positioning [15] [16]. This system integrates a Light Detection And Range (Lidar), a vision sensor (a webcam) and an Inertial Measurement Unit (IMU). The implementation outcomes showed promising results [15].

References

1. Lee, J.H., Lee, J.M., Park, J.W., Moon, Y.S.: Efficient algorithms for automatic detection of cracks on a concrete bridge. In: The 23rd International Technical Conference on Circuits, Systems, Computers and Communications, Shimonoseki, Japan, pp. 1213–1216 (2008)
2. Ehrig, K., Goebbels, J., Meinel, D., Paetsch, O., Prohaska, S., Zobel, V.: Comparison of Crack Detection Methods for Analysing Damage Processes in Concrete with Computed Tomography. In: International Symposium on Digital Industrial Radiology and Computed Tomography, Berlin, Germany, P2 (2011)
3. Abdel-Qader, I., Abudayyeh, O., Kelly, M.E.: Analysis of edge-detection techniques for crack identification in bridges. Journal of Computing in Civil Engineering, American Society of Civil Engineers 17(4), 255–263 (2003)
4. Jahanshahi, M.R., Masri, S.F.: A novel crack detection approach for condition assessment of structures. In: ASCE International Workshop on Computing in Civil Engineering, Miami, Florida, pp. 388–395 (2011)
5. Mohajeri, M.H., Manning, P.J.: ARIA: An operating system of pavement distress diagnosis by image processing. Transportation Research Record (1311), 120–130 (1991)
6. Tong, X., Guo, J., Ling, Y., Yin, Z.: A new image-based method for concrete bridge bottom crack detection. In: Image Analysis and Signal Processing, Hubei, China, pp. 568–571 (2011)
7. Yamaguchi, T., Hashimoto, S.: Practical image measurement of crack width for real concrete structure. Electronics and Communications, Japan 92(10), 605–614 (2009)
8. Oh, J.K., Jang, G., Lee, J.H., Yi, B.J., Moon, Y.S., Lee, J.S., Choi, Y.: Bridge inspection robot system with machine vision. Automation in Construction 18(7), 929–941 (2009)
9. Lee, J.B., Shin, D.H., Seo, W.J., Jung, J.D., Lee, Y.J.: Intelligent bridge inspection using remote controlled robot and image processing technique. In: International Symposium on Automation and Robotics in Construction (ISARC), Seoul, Korea, pp. 1426–1431 (2011)

10. Miyamoto, M., Konno, M.A., Bruhwiler, E.: Automatic Crack Recognition System for Concrete Structures Using Image Processing Approach. Asian Journal of Information Technology 6, 553–561 (2007)
11. Sham, F.C., Chen, N., Long, L.: Surface crack detection by flash thermography on concrete surface. Insight - Non-Destructive Testing and Condition Monitoring 50(5), 240–243 (2008)
12. Burges, C.: A Tutorial on support Vector machines for pattern recognition. Data Mining and Knowledge Discover 2, 2 (1998)
13. Vapnik, V.: The nature of statistical learning theory. Springer (1995)
14. Daubechies, I.: The wavelet transform, time-frequency localization and signal analysis. IEEE Transactions on Information Theory 36(5), 961–1005 (1990)
15. Jo, J., Tsunoda, Y.: A Data Fusion Model based on ROI and Likelihood for the Integration of Multiple Sensor Data. Accepted and will appear in the Proceedings of the 2nd International Conference on Robot Intelligence Technology and Applications 2013. Springer, Germany (2013)
16. Jo, J., Tsunoda, Y., Sullivan, T., Lennon, M., Jo, T., Chun, Y.: BINS: Blackboard-based Intelligent Navigation System for Multiple Sensory Data Integration. In: The 17th International Conference on Image Processing, Computer Vision, & Pattern Recognition, Nevada, USA (2013)
17. http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/TRAPP1/filter.html
18. http://mplab.ucsd.edu/tutorials/gabor.pdf