Crack Detection using Edge Detection and Transfer Learning Models

MSc Research Project
MSc Data Analytics

Pooja Chopra
Student ID: X20145870

School of Computing
National College of Ireland

Supervisor: Dr. Bharathi Chakravarthi

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Pooja Chopra
Student ID:	X20145870
Programme:	MSc Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Dr. Bharathi Chakravarthi
Submission Due Date:	31/01/2022
Project Title:	Crack Detection using Edge Detection and Transfer Learning Models
Word Count:	5119
Page Count:	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	31st January 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Crack Detection using Edge Detection and Transfer Learning Models

Pooja Chopra
X20145870

Abstract

Cracks in bridges can cause severe loss of life, money and property. Early detection and continuous monitoring can avoid the collapsing of bridges. The manual inspection of cracks requires specialist experience, and it is a tedious task that requires plenty of time. This paper gives an alternative approach to monitoring cracks in the bridge. Images collected from the drone are used in this study for training the network for detecting cracks. The cracks in the bridge's deck, walls and pavements are detected individually and together using transfer learning algorithms such as xception, resnet50 and VGG19. These three models were compared for this task, and it was found that xception model outperformed these three models. 2 extra ReLu layers were added to xception model to increase its efficiency of the model. A few transformations were carried out on images, such as image segmentation, canny transformation and changing the image's contrast. These transformations increased the accuracy of models on deck and pavement.

1 Introduction

The bridges reduce travel time and are crucial for transportation over the rivers and travel between 2 islands. The cracks are one of the primary reasons behind the deterioration of the bridges. Cracks occur due to the weakness of the tension. Suzuki et al. (1990) Cracks are the source of the moisture that causes corrosion. They are extremely dangerous to vibrating objects, and cracks can also lead to material infiltration.

Cracks can appear in the early stages and years after the construction in the service period. ElSafty and Abdel-Mohti (2013) There are many causes of cracks in the early stages, such as temperature effects, drying shrinkage and plastic shrinkage. Cracks, especially in the deck, can lose stiffness, reduce durability, loss of functionality, and sequentially deterioration of structural safety. Sousa et al. (2014) Cracks can affect the strength of an arch bridge. Specifically, large cracks affect the bridge's strength to a greater extent.

The bridge inspection is crucial. As Bridges age, more cracks can appear. Due to this reason, inspections of bridges have become mandatory. Previously, the condition of the bridges was checked manually and by inspecting the bridge point by point to get the exact location of the crack. During inspection, bridges were closed during the inspection for safety reasons. These methods were time-consuming, costly and highly dependent on the specialist. In this paper, bridge crack's are detected using image processing.

This paper detects cracks in the deck, pavement, and walls. Figure 1 shows deck and pavement with cracks.

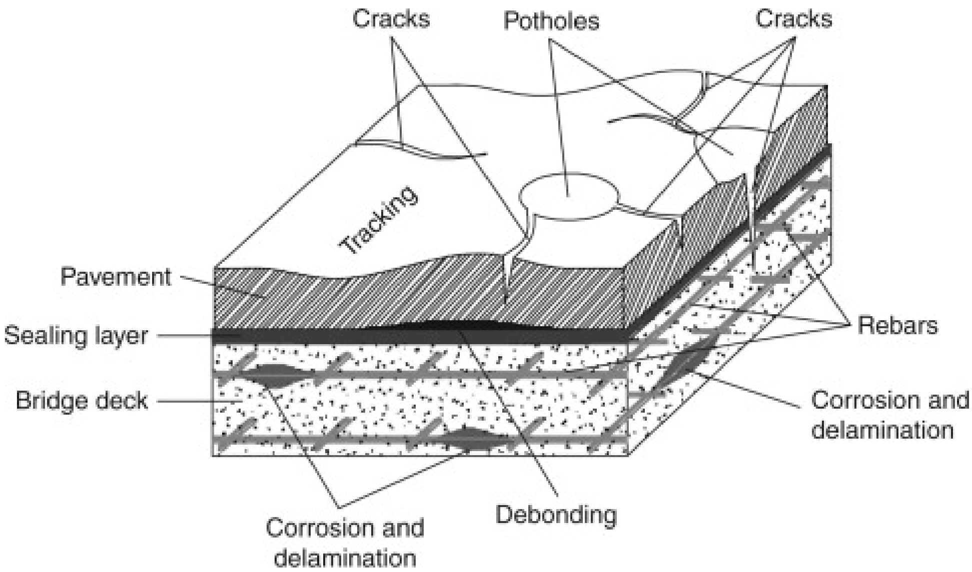

Figure 1: Deck and pavement in bridge

Research Question

- To what extend the edge detection method (Canny Transformation) and image segmentation increases the efficiency of the crack detection in bridges.

Null hypothesis

- Edge detection and image segmentation will significantly increase the efficiency of prediction models in different parts of bridge with cracks of different thickness.

Contribution to current literature

- In this study, two extra layers reLu layers were added to xception model to enhance the efficiency of the model.
- Different combinations of pre-processing steps are involved in this research for detecting the cracks in 3 different parts (Deck, pavement and walls) of the bridges.

In this paper, the next section is related work in which different researches are discussed in detail for crack detection and works related to different machine learning algorithms are discussed. Later in the methodology section, the methods used in this paper is covered. In the design part, the design of this research is discussed in detail. Next to that is the implementation of this research. Then the evaluation of the study is covered. The parameters that can be used for evaluation are discussed, and why that parameter is crucial is covered. Lastly, the research is concluded in the conclusion section.

2 Related Work

Crack detection in bridges is essential for avoiding structural damage and loss of life. This section covers existing works on crack detection and different image processing techniques. Multiple researchers worked on detecting the cracks, and they used various approaches. These methodologies are discussed in this section.

2.1 Pre-Processing and feature extraction

Chanda et al. (2014) focused on the challenges faced when different bridges' images are taken together. They categorized images into two types(complex and normal). Normal images are images in which foreground and background contrast are high. Complex images are the images in which there is a rapid change in the intensities in the background and foreground. Firstly, the images are checked to test whether they are complex or normal. Then, six pre-processing steps were followed on complex images, and no processing steps were included on normal images. Steps involve Converting RGB to HSV and then checking the range of hue and saturation values. If a pixel Saturation value is less than or equal to 0.2 and the Hue value is greater than or equal to 0.9, then set values of H=0.6, S=1.0, V=1.0. Later, the converted images were converted into RGB. The resultant image was converted into greyscale. The greyscale image contrast was enhanced by using a histogram equalizer. Lastly, filtering was applied to the greyscale values. For feature extraction, Gabor Filter and wavelet transformation were done. Finally, they applied SVM to classify the cracked and non-cracked images. This research is helpful in the prediction model obtained in this research that can be applied to any bridge as the training dataset contain multiple bridges. Abdel-Qader et al. (2003) transformed the images using four methods, namely, Sobel Edge Detector, Fourier Transform, Fast Haar Transform and Canny Edge Detector. The results of all these transformations were checked, and they found that the Haar transformation gives the best result among these four transformations. This research contains detailed information about multiple transformations take can be done on an image.

Hoang (2018) proposed Min-Max Gray Level Discrimination (M2GLD). It is an intensity adjustment method. Researchers worked on the integration of the Otsu method and M2GLD. They also detected the shapes of cracks. Cracks properties such as area, length, width, parameter and orientation of the cracks were found out. They used median filters for reducing the noise and enhance the contrast using a 3 X 3 graphic window. Researchers used Sobel filters and Otus threshold for detecting the edges of the cracks. One has to tune two parameters, margin parameter and adjustment ratio, for implementing this method on any dataset. However, the researcher provided a novel and useful approach to detect cracks, and they also predicted the properties of the crack that can be used for the detection of cracks. Li and Duan (2019) Tried to find out the width of the cracks in the cracked images. They used various pre-processing steps to remove the noise and enhance the contrast of the images. In the pre-processing stage, the images and enhanced the contrast in the images to get clear cracks. Pre-processing steps involved converting the image into greyscale then reducing the noise using median filters. The contrast was enhanced by using the 3 x 3 graphics window. After these pre-processing steps, edges were detected using improved edge extraction of Sobel operator, local adaptive Otus threshold segmentation. Finally, isolated noise areas were removed. After all these steps, cracks were detected, and also the authors measured the width of the cracks.

The width of the cracks predicted in this research is very useful, and width can be used in identifying the cracks that can harm the bridge severely. Normally, cracks of greater width are the harmful ones that require immediate repair.

Yang and Geng (2018) proposed a 2 steps method. Firstly, images were pre-processed by enhancing the crack's linearity and then an adaptive threshold was used. Secondly, researchers used pixel percolation processing. Their method efficiently detected pavement cracks and edges were preserved while pre-processing the images. This research is good as the edges were preserved while pre-processing the images. . Suzuki et al. (1990) discussed different methods for the detection of the cracks. Firstly, image processing for crack detection was discussed. However, they mentioned a few shortcomings of this method, such as shadows and noise being considered the crack. Secondly, the targeting approach was discussed. The target algorithm is used for checking the size of the crack, and the accuracy of the size is quiet. However, it's difficult to track the exact location of the crack. Then lastly, recent techniques that combine multiple images of a scene was discussed. Techniques such as Curvelets, wavelets and Contourlets transformation are used for combining different images of a scene and these techniques are called Digital Image Correlation(DIC). DIC address the drawbacks of the image processing and targeting approach.

This subsection involved multiple kinds of research based on feature extraction. The next subsection focuses on new approaches that do not require pre-processing steps and feature extractions.

2.2 Novel Neural Network techniques

Li et al. (2020) Skip-Squeeze-and-Excitation Networks (SSENets) was proposed for crack detection. It consists of Atrous Spatial Pyramid Pooling (ASPP) module and Skip-Squeeze-Excitation (SSE) module. ASPP helped extract information from the images, and depthwise the network helped in computations. The skipping method was used to increase the correlation between deeper networks and shallow networks. The researcher used astrous convolution for detecting crack tropology in place of the pooling layer. The SSE design had skip-connection logic. 97.77% accuracy was obtained using this method, outperforming other models implemented in this research. This research is crucial as it gave good accuracy without the involvement of any pre-processing steps. Li and Zhao (2019) A novel approach spatially tuned robust multi-feature (STRUM) was developed to detect cracks. In this method, there is no need for tuning threshold parameters. The algorithm utilizes the curve fitting to detect a crack in noisy images. The features are computed that were spatially tunned in old methods. This research is nice as it worked well with the noisy image without feature extraction.

Tian et al. (2021) used the YOLOv3(You Only Look Once version 3) algorithm for detecting the cracks. They used GIOU_Loss, MSE and CIOU_loss for improving the accuracy of the model. CIOU_loss helped in getting results more fastly while maintaining the accuracy of the models. Traditionally YOLOv3 depends on positioning loss, bounding box loss and classification loss. In this study, researchers used CIOU_Loss, MSE and GIOU_Loss and these losses, specifically CIOU_loss, gave better results than the traditional approach. This research gave a new novel approach with good results.

Cha et al. (2017) used Convolutional Neural Networks(CNN) without any feature extraction to detect the crack in civil infrastructure. They used 40000 images to train the network that had a pixel value of 256 X 256-pixel resolution that got the accuracy of 98% percent by doing so. They tested their method on 55 images of a high resolution of 256*256 pixels. They tested their result with models made on feature detected images, and it gave good results. Li and Zhao (2019) utilized a deep convolutional neural network (CNN) inspired by Alexnet. To avoid tunning and pre-processing steps, researchers developed this method. They trained the model using 60000 images. They tested the result using different learning rates. A learning rate of 0.01 gave the best accuracy of 99.06%. These 2 research Cha et al. (2017), Li and Zhao (2019) gave good result. However, the CNN training takes more time compared to other models.

Wu et al. (2021) proposed a new method multi-scale deep learning method (MS-DPDL) for detecting cracks in concrete. They trained the model on one dataset and tested the result in the new dataset. The researcher compared multiple deep learning techniques with multi-scale deep learning methods, and the method outperformed the other methods. Deep pixel distribution is used for background subtraction. This method was used to train on the model, and then it was transferred to new videos. It was worked well in both the videos. Research gave high accuracy and transferability.

This subsection covered various new approaches and algorithms for crack detection without much pre-processing. Normally image processing takes a lot of time. A few papers on crack detection focused on the computational timing of training models. The next section consists of multiple kinds of research on computational timing.

2.3 Approaches with less computation timing

Yamaguchi and Hashimoto (2010) Focuses on the computation time as the size of the digital image is 10 MP, and computation takes a lot of time. The authors used percolation-based image processing. In which termination and skip added methods were used to reduce computation time. This research is useful as the speed and time of processing is crucial. Zhang et al. (2020) In research, long short-term memory (LSTM) and 1 Dimension convolution neural network(1D-CNN) was used in the frequency domain to predict the cracks in the concrete bridge. In the pre-processing, the authors transformed the images in the frequency domain. LSTM improved the performance of their model. The model implementation was fast. They used thousands of images and trained the data on large-scale images with good accuracy and computation timing. One of the major contributions of this study is the computation time. The time required for 1D CNN and LSTM is comparatively low compared to other methods. This research is good with respect to computational timing.

2.4 Summary & how this research add value to current literature

Multiple methods and ways of crack detection were used previously. In this study, crack detection is done with edge detection and transfer learning models. Edge detection and pre-processing steps involve contrast enhancement, threshold, and edge detection is canny transformation. Xception, Vgg19 and resnet50 transfer learning models are used for predicting cracks in the bridge. Additional two layers were added to the xception model that gave good accuracy.

3 Methodology

This section covers the methodology used in this study, starting from raw data to the obtained model. Below are the steps involved in this study.

3.1 Data Collection and equipment used

The Raw data was collected from the Kaggle website.[1] The dataset contains 72 walls, 54 bridge decks, and 104 pavement images. The images were taken from a 16MP Nikon digital camera. The dataset contains holes, shadows, surface roughness, edges and background debris.

3.2 Data Pre-Processing

Firstly, Data was imbalanced for all three parts of the bridge(Deck, walls and pavement). The datasets were downsampled to increase the F1 score. The edges of the cracks were detected using the canny transformations, segmentation and transfer learning models.

Image Segmentation was done using a simple threshold and adaptive mean threshold. Bradley and Roth (2007) Image threshold is a method of predicting pixel as dark or light. Adaptive threshold helps in checking variation of illumination. There are two types of adaptive threshold, i.e. mean threshold and gaussian threshold. In the Mean threshold, the mean of the neighbouring threshold is subtracted with a constant C. While in the Gaussian threshold, the Gaussian-weighted sum of neighbouring values is subtracted from the constant C. In a Simple threshold, a pixel threshold greater than the threshold is given maximum value. Else, it is given 0.

Canny (1986) Canny edge detection is a multi-stage algorithm for the detection of the edges. It is used to extract structural information from an image. This process reduced the amount of data. Canny transformation is one of the transformations that was used in research Abdel-Qader et al. (2003).

3.2.1 Edge Detection using Canny transformation in pavements and decks

The below steps were performed on the raw image to obtain better results. The figure 2 summarizes these steps

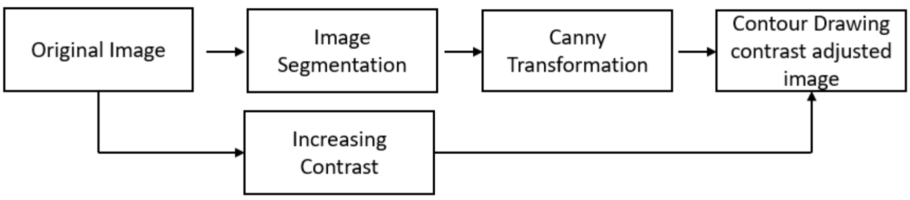

Figure 2: Operations on Deck and pavement images

[1] https://www.kaggle.com/aniruddhsharma/structural-defects-network-concrete-crack-images

Operations on Images: Figure 4 shows steps of Pre-processing on deck and pavement images

1. **Image Segmentation:** Changed RGB images to greyscale for applying threshold function. The images were tested for different threshold functions present in python. From figure 3, the binary threshold was found to be most appropriate for the datasets of deck and pavement.

2. **Edge Detection using Canny transformation**

3. **Increasing Contrast**: To enhance the crack in images, the contrast was set to 1.5 and brightness to 0. As we can see from figure 4, increasing contrast made images clearer.

4. **Adding contour to enhanced contrasted image**

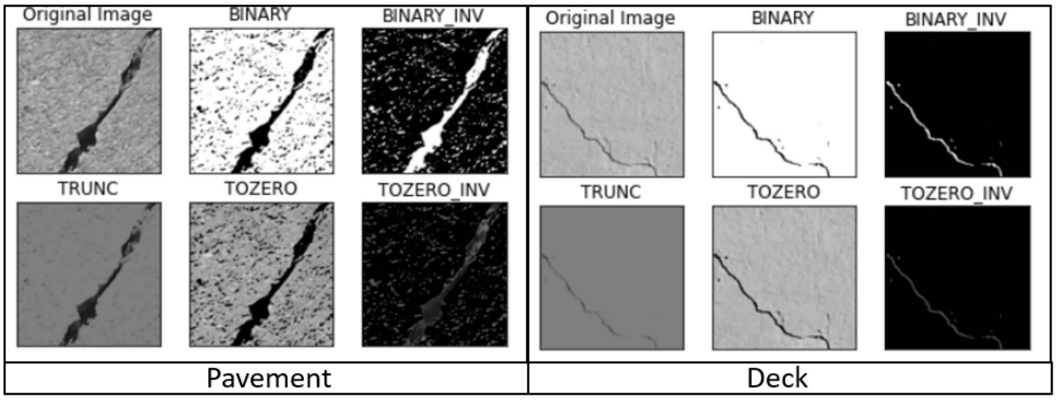

Figure 3: Threshold functions on pavement and walls

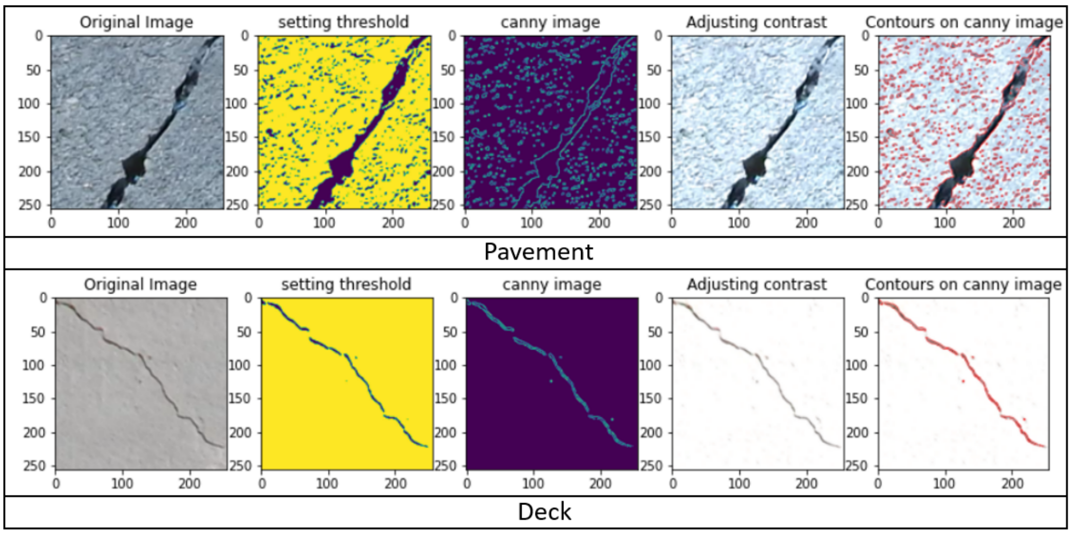

Figure 4: Pre-processing of pavements and deck images

3.2.2 Edge Detection using Canny transformation in walls

The images of walls are lighter, and the cracks are also thinner. The below steps were performed on the raw image to obtain better results. The figure 5 summarize the pre-processing steps involved in the wall's crack detection.

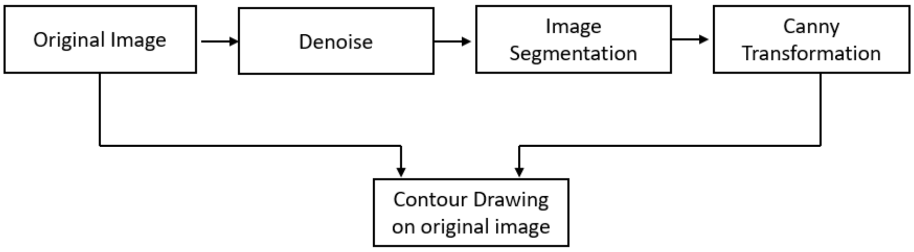

Figure 5: Operations wall images

Operations on Images

1. **Denoising**:Median Blur filter was used to remove the noise from the images.
2. **Edge Detection using Canny transformation**
   - The images were tested for different threshold functions present in python. From figure 6, Adaptive Gaussian threshold found to be most appropriate for walls dataset. Hence, Adaptive Gaussian threshold was applied on the images.
   - Using canny transformation to detect edges.
3. **Adding contour on original image** Final image is shown in figure 7

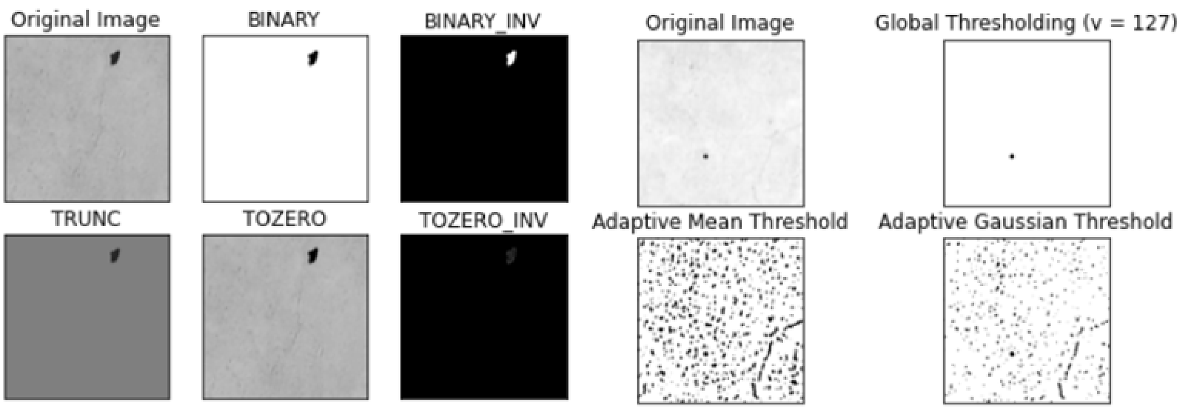

Figure 6: Threshold functions on wall image

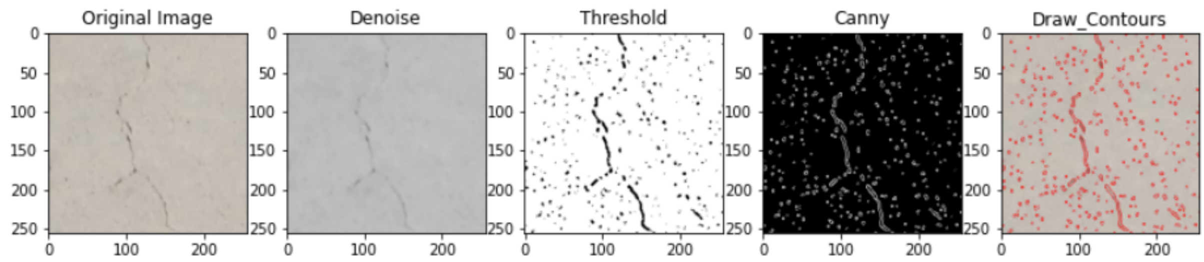

Figure 7: Pre-processing on wall image

3.3 Scenarios/Case studies run

1. Crack detection on deck dataset with and without edge detection using Xception Model.
2. Crack detection on pavement dataset with and without edge detection using Xception Model.
3. Crack detection on walls dataset with and without edge detection using Xception Model.
4. Crack detection on a combined dataset with no-prepossessing using Xception, VGG19 and Resnet Model.
5. Crack detection on a combined dataset with edge detection using Xception, VGG19 and Resnet Model.

3.4 Model Building

Transfer learning models(Xception, VGG19 and Resnet50) were applied on deck, wall and pavement datasets, and then models were applied on a combined dataset containing the three parts(deck, pavement and wall).

3.5 Evalution

Models will be tested using below parameters

- Accuracy
- f1 score
- Precision
- Recall
- Loss

4 Design Specification

Transfer learning models were implemented for different images in this study. In this study xception model, VGG 19 and Resnet50 models are used.

4.1 Xception Model

Chollet (2017) Xception model is an intermediate step in-between depthwise separable convolution and regular convolution operation. The model is inspired by the Inception model. The xception model was built on the same number of layers as the inception model. Still, it outperformed the inception model because of the efficient use of parameters present in the model. Figure 8 contain the architecture of xception model. It has 71 layers and can be split into three parts entry flow, middle flow and exit flow.

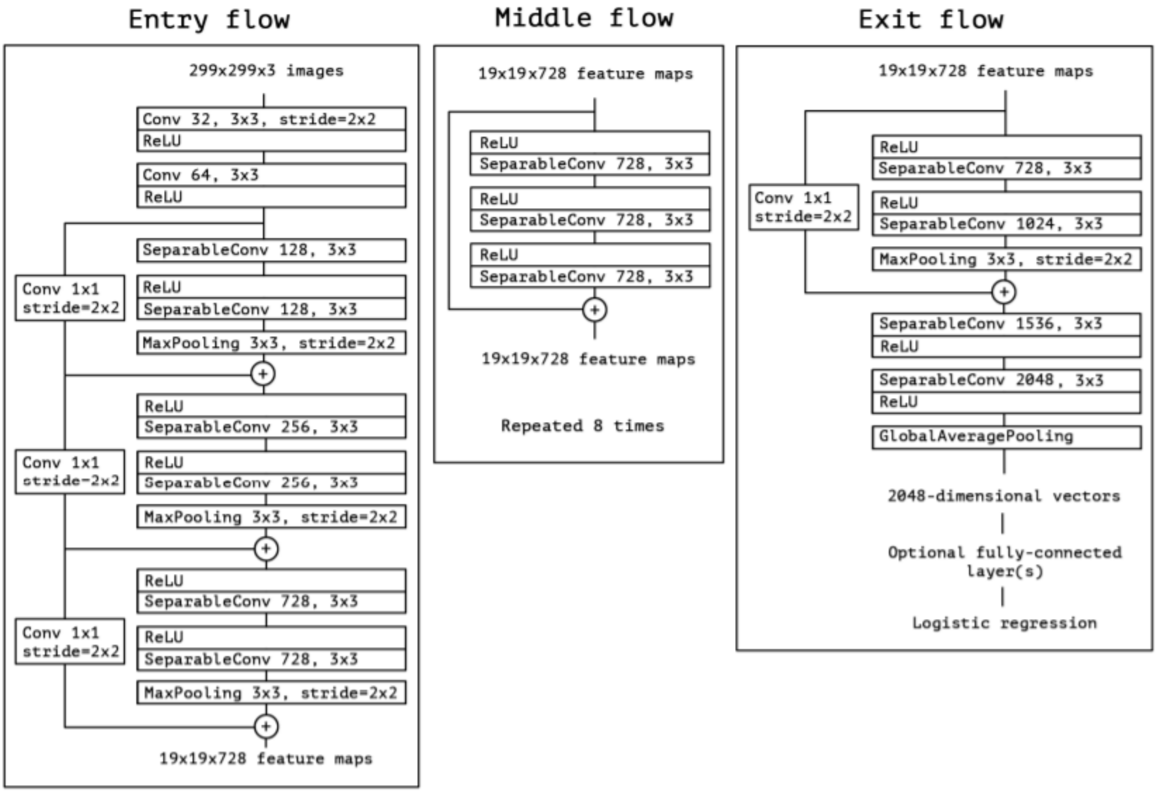

Figure 8: Xception arhichecture

4.1.1 Changes in Xception Model implemented in this study

In this study, two extra dense layers with activation function ReLu were added to increase the accuracy of the model. Figure 9 shows the model implemented in this research.

ReLu Activation Function: ReLu stands for Rectified Linear Unit. It is a non-linear activation function. This activation is also called the ramp function.

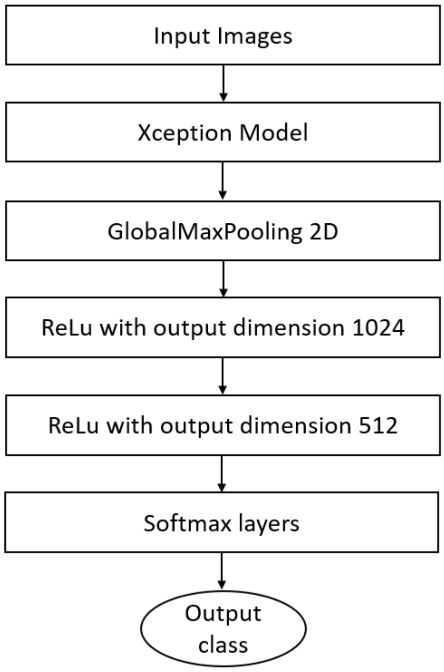

Figure 9: Flow chart of xception model used in this study

4.2 VGG19

Simonyan and Zisserman (2014) VGG19 was made by increasing the depth of the architecture using a small convolution filter of size (3X3). The authors trained their network on ImageNet. Figure 10 shows the configuration of VGG19.

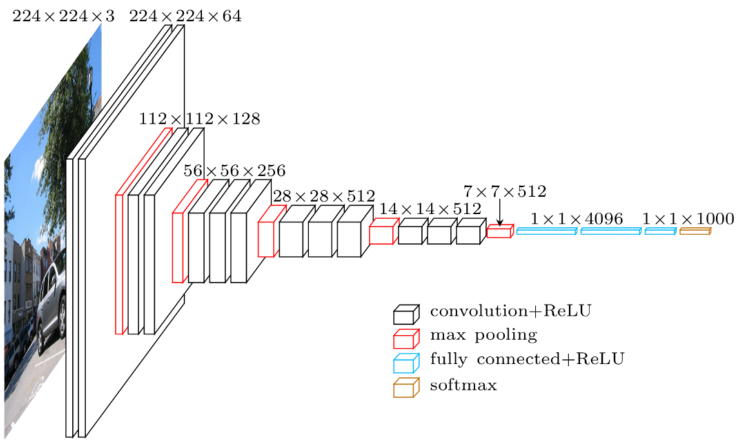

Figure 10: VGG configuration

4.3 Resnet50

ResNet50 has 1 max pool layer, 48 convolution layers and 1 average pool layer. Residual neural networks jump over some layers by skipping connections and shortcuts.

Usually, triple or double-layer skips are present in a network that has Relu and batch normalization. Figure 11 shows the architecture of Resnet50

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 11: Resnet50 Architecture

5 Implementation

The code for this study was written in python language using colab. The data was stored in google drive. The stored data were transformed using threshold function, canny transformation using the cv2 library of python as described in the methodology section. The results of the transformation can be seen in figure 7 and fig:4. The models were developed using the Transfer learning model(xception model, VGG19 and Resnet). The effect of edge detection was checked on the different types of datasets. All different datasets were combined to check whether all images could be clubbed together in a single model or not.

6 Evaluation

Metrics such as f1 score, confusion matrix precision, recall and accuracy are considered for the evaluation. Below is the confusion matrix in Figure 12
where,
TP = True Positive, TN = True Negative, FP = False Positive, FN = False Negative,

Below are formulas of matrics that will be using the case studies.

$$Precision = \frac{True\,Positive}{True\,Positive + False\,Positive}$$

$$Sensitivity/Recall = \frac{True\,Positive}{True\,Positive + False\,Negative}$$

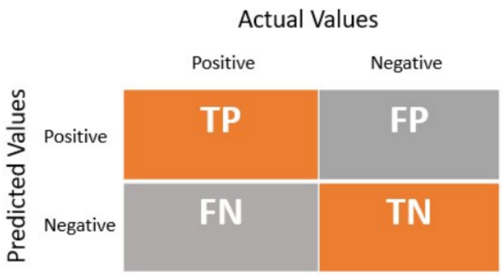

Figure 12: Confusion Matrix

$$F1score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

$$Accuracy = \frac{True\,Positive + TrueNegative}{True\,Positive + True\,Negative + False\,Positive + False\,Negative}$$

Below are case studies Conducted in this study

6.1 Case Study 1: Crack detection on deck dataset with edge detection using Xception Model

Crack detection was done using the xception model with and without edge detection. Table 1 is the comparison between the models with and out edge detection. We can see from the table that the model with edge detection is performing better.

Table 2 classification report of the xception model implementation after edge detection on deck dataset. From the confusion in figure 13, we can say that 370 cracked images out of 391 were correctly predicted by the model.

Table 1: Comparision of metrics obtained Xception Model of deck dataset

Matrics	**Xception**	
	Without pre-processing	**With pre-processing**
Accuarcy	0.7198	0.9506
Precision	0.8307	0.9537
Recall	0.7237	0.9801
F1 score	0.7735	0.9667
Loss	0.7054	0.2950

Table 2: Classification report of deck dataset after edge detection

	Precision	**Recall**	**f1-score**	**Support**
0	0.95	0.95	0.95	419
1	0.95	0.95	0.95	391
accuracy			0.95	810
macro avg	0.95	0.95	0.95	810
weighted avg	0.95	0.95	0.95	810

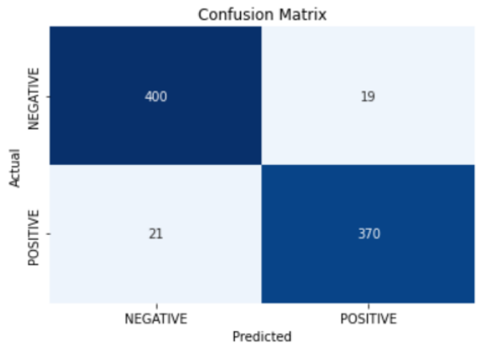

Figure 13: Confusion Matrix of deck dataset after edge detection

6.2 Case Study 2: Crack detection on pavement dataset with and without edge detection using Xception Model

Crack detection was done using the xception model with and without edge detection. Table 3 is the comparison between the models with and out edge detection. We can see from the table that the model with edge detection is performing better.

Table 4 classification report of the xception model implementation after edge detection on pavement dataset. From the confusion matrix in figure 14, we can say that 480 cracked images out of 509 were correctly predicted by the model.

Table 3: Comparision of metrics obtained from Xception Model on pavement dataset

Matrics	**Xception**	
	Without pre-processing	**With pre-processing**
Accuarcy	0.8008	0.9502
Precision	0.8512	0.9559
Recall	0.8104	0.9485
F1 score	0.8303	0.9522
Loss	0.4629	0.2383

Table 4: Classification report pavement dataset after edge detection

	Precision	**Recall**	**f1-score**	**Support**
0	0.95	0.96	0.95	535
1	0.95	0.94	0.95	509
accuracy			0.95	1044
macro avg	0.95	0.95	0.95	1044
weighted avg	0.95	0.95	0.95	1044

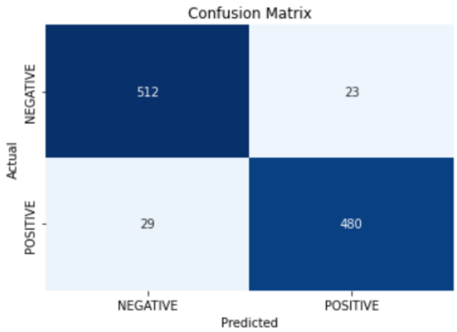

Figure 14: Confusion Matrix of pavement dataset after edge detection

6.3 Case Study 3: Crack detection on walls dataset using xception model with different scenarios

Table 5 contains the comparison between different Scenarios on walls dataset. Table 6 has classification report of the xception model implementation after edge detection on wall dataset. From the confusion matrix in figure 15, edge detection didnt gave good result for walls dataset.

Table 5: Comparision of metrics obtained Xception Model of walls dataset

Matrics	**Xception**	
	Without pre-processing	**With pre-processing**
Accuarcy	0.7450	0.6457
Precision	0.8475	0.6822
Recall	0.7918	0.6176
F1 score	0.8187	0.6483
Loss	0.8242	0.6914

Table 6: Classification report of xception model after edge detection

	Precision	**Recall**	**f1-score**	**Support**
0	0.70	0.55	0.61	790
1	0.61	0.75	0.67	751
accuracy		0.65	1541	
macro avg	0.65	0.65	0.64	1541
weighted avg	0.65	0.65	0.64	1541

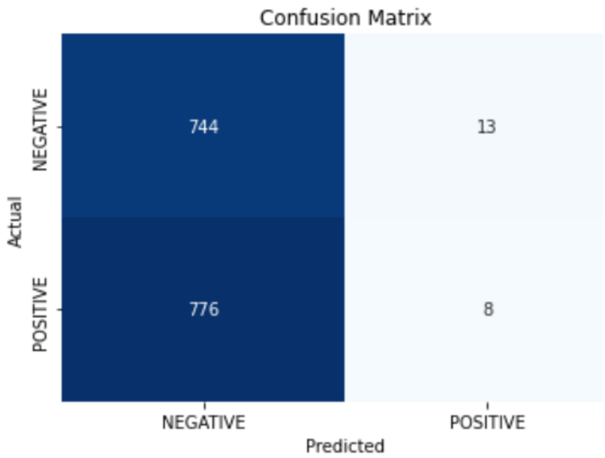

Figure 15: Confusion Matrix of deck dataset after edge detection

6.4 Case Study 4: Crack detection on the combined dataset without pre-processing using Xception, VGG19 and Resnet

Different models were implemented on the combined dataset to find out if we could combine all three datasets or not for the prediction. Table 7 contains the result of the experiment. Xception model outperformed the VGG19 and resnet50.

Table 7: Comparision of Different Models on combined dataset

Matrics	Models		
	With Xception	**With VGG19**	**With Resnet**
Accuarcy	0.7481	0.6338	0.7472
Precision	0.8040	0.6160	0.6819
Recall	0.7094	0.4814	0.6776
F1 score	0.7537	0.5405	0.6797
Loss	0.5393	0.6271	0.5071

6.5 Case Study 5: Crack detection on the combined dataset with edge detection using Xception, VGG19 and Resnet

Different models were implemented on the combined pre-processed dataset to find out if we could combine all three datasets or not for the prediction. Table 8 contains the result of the experiment. Xception model outperformed the VGG19 and resnet50.

Table 8: Comparision of Different Models on combined dataset

Matrics	Models		
	With Xception	**With VGG19**	**With Resnet**
Accuarcy	0.8480	0.6771	0.8179
Precision	0.8721	0.6510	0.7930
Recall	0.8508	0.5925	0.7075
F1 score	0.8613	0.6204	0.7478
Loss	0.3461	0.5377	0.3862

Table 9: Classification report pavement dataset after edge detection

	Precision	**Recall**	**f1-score**	**Support**
0	0.82	0.89	0.85	1690
1	0.88	0.81	0.84	1704
accuracy		0.85	3394	
macro avg	0.85	0.85	0.85	3394
weighted avg	0.85	0.85	0.85	3394

Table 9 contains classification report of xception model and figure 16 shows the confusion matrix of xception model. Figure 17 contains the accuracy and loss of epoch of xception model.

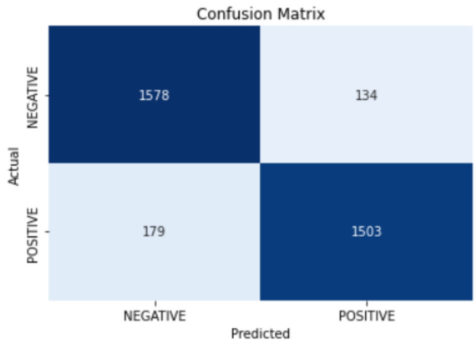

Figure 16: Confusion Matrix of combined dataset after edge detection

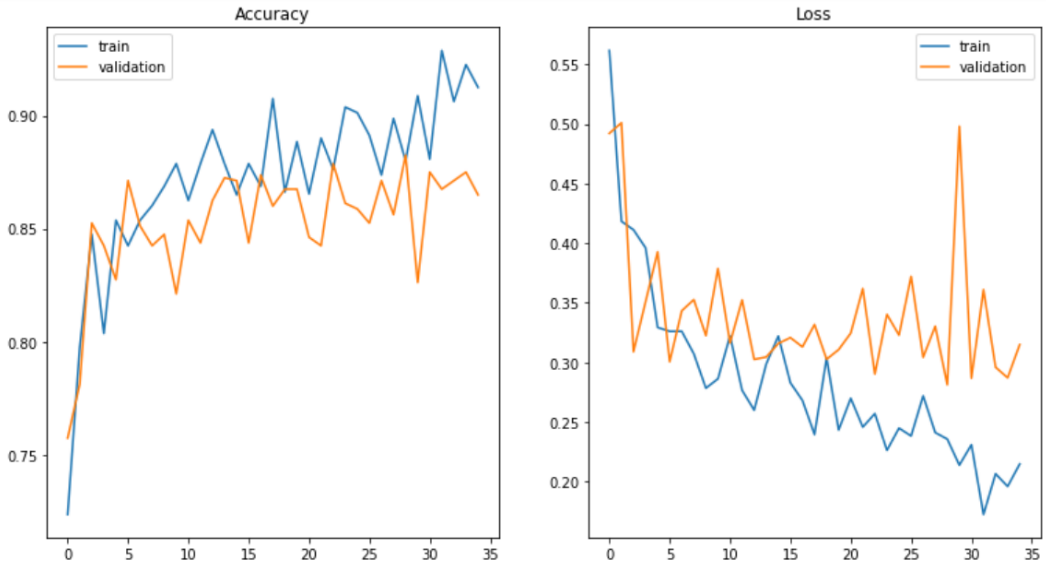

Figure 17: Accuracy and Loss of combined Dataset

6.6 Discussion

The study was started by comparing the models obtained with and without edge detection on the deck dataset. Extra dense layers were added to xception model for better results. The study was conducted on the walls and deck dataset as well. It was found that all the datasets that model with edge detection give better predictions in pavement and deck. Walls predictions after the edge detection were not good. From 75% accuracy was decreased to 65% accuracy after edge detection. Later, all the datasets were combined, and it was found that the models of individual datasets deck, pavement, and walls. The xception model gives good accuracy and maximum True Positive.

7 Conclusion and Future Work

The pre-processing steps helped increase the accuracy and f1-score of the models to a greater extent on pavement and deck. The bridge's three parts pavement, deck and walls were analyzed in this study using edge detection and xception model. The walls dataset didn't give good results after edge detection. In the future, different models that don't require preprocessing, such as SSENets and YOLOv3 edge detection, can be implemented, and the results can be compared with models with pre-processing steps. Along with that rough surface of the pavements can be smoother further to better predict the cracks.

References

Abdel-Qader, I., Abudayyeh, O. and Kelly, M. (2003). Analysis of edge-detection techniques for crack identification in bridges, *Journal of Computing in Civil Engineering* **17**: 255–263.

Bradley, D. and Roth, G. (2007). Adaptive thresholding using the integral image, *J. Graphics Tools* **12**: 13–21.

Canny, J. (1986). A computational approach to edge detection, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **PAMI-8**: 679 – 698.

Cha, Y.-J., Choi, W. and Büyüköztürk, O. (2017). Deep learning-based crack damage detection using convolutional neural networks, *Comput. Aided Civ. Infrastructure Eng.* **32**: 361–378.

Chanda, S., Bu, G., Guan, H., Jo, J., Pal, U., Loo, Y.-C. and Blumenstein, M. (2014). Automatic bridge crack detection – a texture analysis-based approach, pp. 193–203.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions, pp. 1800–1807.

ElSafty, A. and Abdel-Mohti, A. (2013). Investigation of likelihood of cracking in reinforced concrete bridge decks, *International Journal of Concrete Structures and Materials* .

Hoang, N.-D. (2018). Detection of surface crack in building structure using image processing technique with an improved otsu method for image thresholding, *Advances in Civil Engineering* **2018**.

Li, H., Xu, H., Tian, X., Wang, Y., Cai, H., Cui, K. and Chen, X. (2020). Bridge crack detection based on ssenets, *Applied Sciences* **10**: 4230.

Li, S. and Zhao, X. (2019). Image-based concrete crack detection using convolutional neural network and exhaustive search technique, *Advances in Civil Engineering* **2019**: 1–12.

Li, Y. and Duan, P. (2019). Research on the innovation of protecting intangible cultural heritage in the "internet plus" era, *Procedia Computer Science* **154**: 20–25. Proceedings of the 9th International Conference of Information and Communication Technology [ICICT-2019] Nanning, Guangxi, China January 11-13, 2019.
URL: *https://www.sciencedirect.com/science/article/pii/S1877050919307744*

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition, *arXiv 1409.1556* .

Sousa, C., Thomaz, E., Sousa, H., Roman, H., Morton, B., Silva, U., Correa, M., Pfeffermann, B., Lourenco, P., Vicente, R., Sousa, R. and Portugal (2014). *Defects in Masonry Walls Guidance on Cracking: Identification, Prevention and Repair.*

Suzuki, K., Ohno, Y., Praparntanatorn, S. and Tamura, H. (1990). Mechanism of steel corrosion in cracked concrete. corrosion of reinforcement in concrete. papers presented at the third international symposium on "corrosion of reinforcement in concrete construction", belfry hotel, wishaw, warwickshire, may 21-24, 1990.

Tian, J., Zhang, Y., Yao, Y., Yao, X., Shi, C. and Xu, Y. (2021). Road crack detection algorithm based on yolov3, *2021 International Conference on Intelligent Transportation, Big Data Smart City (ICITBS)*, pp. 39–42.

Wu, X., Ma, J., Sun, Y., Zhao, C. and Basu, A. (2021). Multi-scale deep pixel distribution learning for concrete crack detection, *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 6577–6583.

Yamaguchi, T. and Hashimoto, S. (2010). Fast crack detection method for large-size concrete surface images using percolation-based image processing, *Machine Vision and Applications* **21**: 797–809.

Yang, C. and Geng, M. (2018). The crack detection algorithm of pavement image based on edge information, *6th International Conference on Computer-Aided Design, Manufacturing, Modeling and Simulation (CDMMS 2018)*, Vol. 1967 of *American Institute of Physics Conference Series*, p. 040023.

Zhang, Q., Barri, K., Babanajad, S. K. and Alavi, A. H. (2020). Real-time detection of cracks on concrete bridge decks using deep learning in the frequency domain, *Engineering* .
URL: *https://www.sciencedirect.com/science/article/pii/S2095809920303301*