UTS UNIVERSITY OF TECHNOLOGY SYDNEY

Crack Detection and Classification using Digital Image Processing

by Thi Hong Nhung Nguyen

Thesis submitted in fulfilment of the requirements for
the degree of

Doctor of Philosophy

under the supervision of

- Prof. Stuart Perry, Principal supervisor
- A/Prof. Donald Bone, Co-supervisor
- A/Prof. Thuy Thi Nguyen, External supervisor
- A/Prof. Le Thanh Ha, External supervisor

University of Technology Sydney
Faculty of Engineering and IT

Dec 2022

Certificate of Original Authorship

I, Thi Hong Nhung Nguyen declare that this thesis, is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the School of Electrical and Data Engineering/Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of the requirements for a degree at any other academic institution except as fully acknowledged within the text. This thesis is the result of a Collaborative Doctoral Research Degree program with The VNU University of Engineering and Technology (VNU-UET).

This research is supported by the Australian Government Research Training Program.

Signed: Production Note: Signature removed prior to publication.

Date: 22/12/2022

Crack Detection and Classification using Digital Image Processing

by

Thi Hong Nhung Nguyen

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

Abstract

Crack detection and segmentation, and the processes of crack classification and crack index calculation which they support, are essential tools in road survey and maintenance applications. The outputs of crack detection and segmentation are used to define the severity levels of the cracking of road surfaces. Based on the crack severity levels, necessary repair and maintainence can be decided.

There are many image processing methods that may be applied to this domain, including traditional methods using low-level features such as edges, and modern methods using machine learning algorithms such as deep learning which are able to abstract high-level features. However, many challenges associated with the use of digital image processing to detect and segment cracks in images are still not solved. Some of these challenges and limitations include: (1) crack images are often noisy, have low-resolution, and contain many artifacts; (2) the associated road crack datasets are imbalanced, with only a small proportion of the data repesenting crack information; (3) three dimensional (3D) data such as crack point clouds are informative to analyze and monitor the development of crack but current acquisition methods for this data produce low-density point clouds and this problem needs to be addressed to make these data sets more useful; (4) the automated calculation of crack indices is frustrated by the lack of robust, standardised methods for automatically identifying cracks and measuring crack parameters such as crack length and crack width.

To solve the above limitations, this thesis focuses on three main contributions:

The first contribution is the proposal of a new architecture for crack detection and segmentation. This method improves the ability to segment the crack from noisy and imbalanced road crack datasets. A combination of crack detection at the region (or sample) level and crack segmentation at the pixel level is shown to increase the accuracy of crack segmentation.

In the second contribution, a novel method of crack point cloud upsampling is proposed. By combining the point clouds and their corresponding 2D images in a model based on a GAN (Generative Adversarial Network) framework, the proposed method aims to generate high-resolution point clouds from low-resolution point clouds and matched 2D images. The high-resolution point clouds can be used to improve the classification of crack point clouds and support crack monitoring.

The final contribution is a method to calculate crack parameters such as crack length and crack width from segmented cracks. This contribution proposes an approach for evaluating the crack length results based on the traditional metric Precision Recall Curve (PRC). The new approach is suitable for a range of narrow features such as crack lines.

This thesis shows the impressive power of using digital image processing and machine learning for crack analysis in both 2D or 3D crack data.

Acknowledgements

Throughout the writing of this dissertation I have received a great deal of support and assistance.

I would like to acknowledge the Joint Technology and Innovation Research Centre - a partnership between University of Technology Sydney and Vietnam National University, and the Faculty of Engineering and Information Technology – UTS who granted me this Ph.D scholarship.

I am deeply grateful to my supervisors, Associate Professor Stuart Perry, Associate Professor Donald Bone, Associate Professor Thi Thuy Nguyen, and Associate Professor Thanh Ha Le, whose expertise was invaluable in formulating the research questions and methodology. Your insightful feedback pushed me to sharpen my thinking and brought my work to a higher level.

I would like to thank Associate Professor Min Xu and Dr. Hoang Dinh in SEDE - UTS for their valuable guidance throughout my studies. I would like to thank Dr. Nhu Thuc Nguyen who helped me with data collection.

I would like to thank Mr. Ninh Hong Quang Nguyen for his support to my family in Australia. I would like to thank Ms. Leonie Smyth for helping me with English skills.

Finally, my deep and sincere gratitude to my family for their continuous and unparalleled love, help and support. I would like to thank my husband, Mr. Quang Duy Nguyen, and my sons for their patience and encouragement. You are always there for me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context and Motivation

Road survey and maintenance are essential tasks for governments all over the world. These tasks are high cost and take time in not only developing countries like Vietnam, but also in developed countries such as Australia. So, decreasing the time and cost for road assessment is a high-priority mission. One of the most effective ways to do that is using modern techniques such as advanced digital image processing to do road surveys automatically.

The development of digital image processing (DIP), computer vision, and machine learning has supported many aspects of human life including civil engineering, where road conservation acts as a crucial area of support for society. DIP supports road surveys with detection, segmentation, and evaluation of problems with road surfaces such as rutting, holes, and cracking.

In a technical report [1], Shahin and Kohn illustrated the different observable distress types in asphalt surfaces as shown in figure 1.1 [1]. Figure 1.1 shows a variety of different observable distress in asphalt surfaces.

In road surveys and inspection, the two indexes: pavement serviceability index (PSI) and Pavement Condition Index (PCI), are important metrics for assessing the condition of a road [1]. To calculate the PSI and PCI, three main indexes must be calculated and

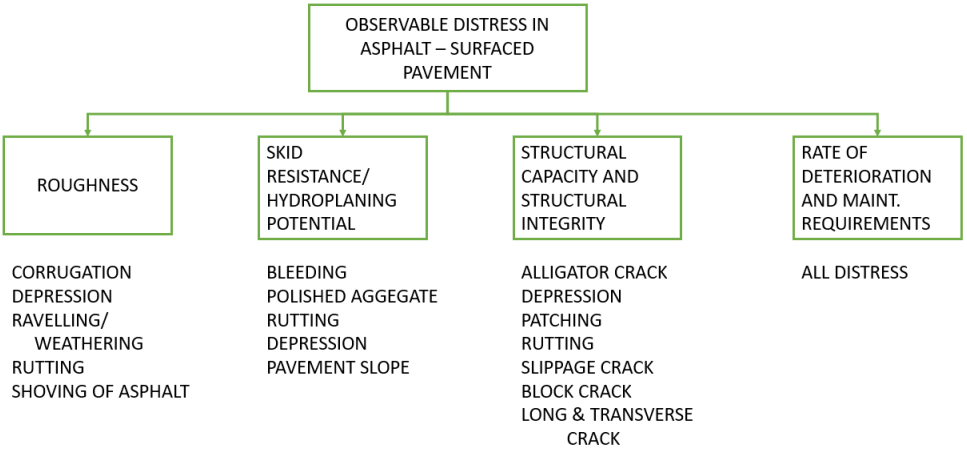


FIGURE 1.1: Different types of observable distress in asphalt surfaces [1]. All crack types belong to the "structural capacity and structural integrity" distress class.

evaluated: International Roughness Index (IRI), rutting index and cracking index. IRI is the most common roughness index that is obtained from measuring the road condition. IRI can be considered as a longitudinal (along the direction of traffic travel) slice of the road showing elevation as it varies with longitudinal distance along the road. Rutting index measures the damage to the road caused by wheel tracks. Cracking index shows the damage caused by separation of parts of the road surface.

Among these kinds of distress, cracks are one of the most serious with many morphologies such as alligator cracks, longitudinal and transverse cracks, slippage and block cracks. Cracks are further classified as "structural capacity" and "structural integrity" distress. The crack indexes contribute an important component of the overall pavement condition indexes.

Of the three indexes, IRI and rutting index are calculated automatically in real systems but cracking index is manually or semi-automatically computed [11–14]. So, automated methods for the calculation of the cracking index are still an active area of research.

Algorithms for crack detection and segmentation from digital images could potentially serve a role in automating crack index calculation, but this is an ongoing and challenging problem for researchers in measuring road damage. In image based approaches, an image of the road surface is often captured by a camera mounted on a specially equipped vehicle [13, 14]. However, in practice, crack indexes are currently computed manually by technicians.

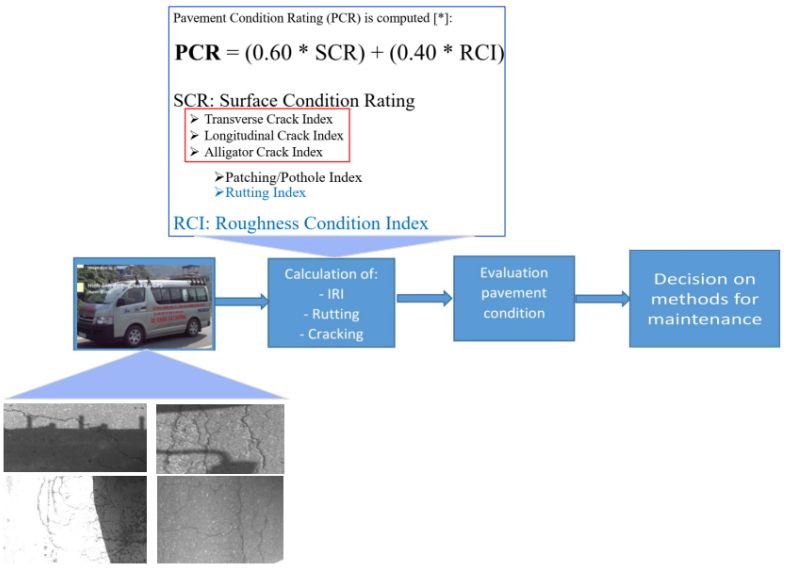


FIGURE 1.2: The data collection system for road maintenance. A special car collects road images from attached cameras [2]. The second step is evaluating road damage indexes based on road images. Finally, a method to repair the road is decided based on the surface damage.

Figure 1.2 shows the main steps in road maintenance that involve the capture of images of the cracked pavement. A specially equipped car captures the road images while it is driving on the road. The speed of the car (in general about 70 to 80 kph [15]) may lead to low quality images being captured. In the second step, road indexes such as IRI, rutting, cracking are calculated. Pavement condition is also assessed by the pavement condition rating (PCR), which can be calculated from the surface condition rating (SCR) and roughness condition index (RCI) [10] using equation 1.1. The two final steps in this simple system are evaluating the road conditions and proposing a method for road maintenance.

$$PCR = (0.60 * SCR) + (0.40 * RCI) \tag{1.1}$$

As can be seen from equation 1.1, the severity level and the total distress in each type of distress must be calculated. Among road distress types, crack damage is very important and its assessment is challenging for civil engineering and data researchers. This dissertation aims to support crack surveying and monitoring using digital data, and digital data processing of crack data. The proposed methods in this dissertation support crack surveys

by estimating the severity levels of all types of crack based on image data. To do that, firstly, the crack must be detected, segmented, and classified based on images of the road. Then, crack width and length will be calculated and classified based on standard practice for qualifying cracks [16–18] and manuals for crack assessment and repair [10].

In recent years, three dimensional (3D) data such as point clouds have been used widely in many applications of digital image processing. Point cloud data relating to cracks are used in construction surveys for managing the development of cracks on construction surfaces, in health monitoring, and in assessing the condition of construction works [19, 20]. However, high-density point clouds or 3D data are expensive and sometimes impractical to collect, so collecting low-density point clouds and then up-sampling them offers a practical way to acquire this data. This dissertation proposes a new method using combination of crack point clouds and 2D crack images for crack point cloud up-sampling.

There are some challenges for this research including:

- *Removing noise.* Road images are complex and contain many unwanted objects and display diversity due to being captured in many different conditions of light and weather.
- *Definition of cracks.* There is no clear definition of what constitutes a crack on the road or in an image. So it is complicated to separate false cracks from real ones.
- *Making ground truth.* The quality of ground truth images has a strong effect on the final result of detection and segmentation systems. Ground truth images should be made by road experts.
- *Variable spatial resolution.* Consistent spatial resolution must be used for calculating the crack width and length. For calculating the real size of crack features in images, having constant consistent spatial resolution across the data set would simplify the algorithms and make them more robust.
- *Low-resolution, noisy and imbalanced crack datasets.* Crack images are often noisy and low-resolution because they are captured under many different conditions. Crack datasets often consist of imbalanced data due to the number of crack pixels being smaller than the number of background pixels in images collected from the real world.

- *Difficulties in determining the reason for the crack.* Different reasons may cause similar crack patterns. For this reason, determining the reasons for various cracks can be difficult.

- *Combining crack images and crack point clouds.* Using crack images to augment crack point clouds is another challenge. The two sets of data are quite different and identifying the best method of combining the images and the point clouds into a single useful data structure for detection and classification is a complex task.

Research Objectives

This research aims to develop methods for road crack detection, segmentation and classification based on image data using digital image processing. This dissertation also proposes a method for up-sampling crack point clouds. High-resolution point clouds can help to increase the detection accuracy of cracks in point clouds. High-density crack point clouds are used for many purposes in crack surveying. For the research in this dissertation, it is intended that the proposed methods could also be applied to the detection and segmentation of cracks on other structures such as buildings, tunnels and bridges and that the method could be used as the crack detection component of an automated structure maintenance system. The main objectives are:

- *Objective 1:* Proposing a method based on CNN and DIP for crack detection and segmentation at the pixel level. This method is expected to solve the problem of noisy and low-resolution crack road images.

- *Objective 2:* Proposing a method for up-sampling low-resolution crack point clouds. This method uses a combination of images and point clouds in a Generative Adversarial Network.

- *Objective 3:* Proposing an algorithm for evaluating the crack width, crack length and crack orientation.

1.2 Contributions

The literature review in Chapter 2 identifies some gaps in the existing literature. The first gap is that recent research has made little progress in the segmentation of cracks at the pixel level, especially in noisy images. The second gap is the lack of methods for crack characteristic recognition, and the lack of methods for crack classification and for the inference of reasons for the cracks. The third gap is there are no significant contributions in crack point cloud classification. This research dissertation tries to fill the above gaps with following original contributions:

- A two-stage model that is based on convolutional neural networks is proposed for crack segmentation at the pixel level.
- A method for up-sampling crack point clouds by using a combination of crack images and crack point clouds using a GAN framework that contributes to improving crack point cloud classification.
- A new method for crack pattern calculation is proposed by developing digital image processing and using the segmented results.
- Data sets are collected to demonstrate the methods and made available to the research community.

1.3 Publications

Publications arising from this work:

Nguyen, Nhung Hong Thi, et al. "Combination of Images and Point Clouds in a Generative Adversarial Network for Upsampling Crack Point Clouds." IEEE Access 10 (2022): 67198-67209.

Perry, S., Cruz, L. A. D. S., Dumic, E., Nguyen, N. H. T., Pinheiro, A., Alexiou, E. (2021, October). "Comparison of Remote Subjective Assessment Strategies in the Context of

the JPEG Pleno Point Cloud Activity.“ In 2021 IEEE 23rd International Workshop on Multimedia Signal Processing (MMSP) (pp. 1-6). IEEE.

Nguyen, N. H. T., Perry, S., Bone, D., Le, H. T., Nguyen, T. T. (2021). “Two-stage convolutional neural network for road crack detection and segmentation.” Expert Systems with Applications, 115718.

Nguyen, N. T. H., Le, T. H., Perry, S., Nguyen, T. T. (2018, December). “Pavement crack detection using convolutional neural network.” In Proceedings of the Ninth International Symposium on Information and Communication Technology (pp. 251-256).

1.4 Dissertation Outline

Chapter 1 Introduction and Background presents the necessity of crack detection, segmentation and classification for real-world applications. The definitions of crack indexes, and their role in road survey and maintenance systems are also described in this chapter.

Chapter 2 Literature Review shows the recent work on crack detection, segmentation, crack classification and crack point cloud processing. Advantages and disadvantages are considered for all of the papers reviewed. Both traditional methods based on low-level digital image processing and modern methods based on high-level digital image processing using machine learning algorithms are considered in this chapter. From these reviews, research gaps in the area of crack image processing and its application are identified.

Chapter 3 Crack Detection and Segmentation using a Two-stage Convolutional Neural Network Architecture details a proposed method for using CNN for crack detection and segmentation at the sample level and the pixel level. A new model that contains two stages of CNN is proposed for crack segmentation at the pixel level and to solve the problem of imbalanced crack data.

Chapter 4 Merging 2D and 3D Data for Crack Detection illustrates a new method for upsampling crack point clouds by using a combination of 2D crack images and crack point clouds. An architecture based on a Generative Adversarial Network (GAN) is applied to upsample the low-resolution crack point clouds.

Chapter 5 Analysis and Classification of Crack Characteristics details a new method for calculating the crack characteristics such as crack length, crack width, and crack orientation. From these indexes, cracks may be classified based on crack width and crack orientation. This chapter contains a proposed method to evaluate the crack length calculation results.

Chapter 6 Conclusions and Future Work is the last chapter in this dissertation and includes conclusions, future work and the application of the research for real road survey systems.

Chapter 2

Literature Review

2.1 Background in Digital Image Processing

An image can be defined by a 2D function, *f(x,y)*, where x and y are spatial coordinates. The pair (x,y) defines a position of a point or a pixel in an image, and the value of f is the intensity (or colour) level of the pixel (x,y). The number of pixels and the number of intensity levels are finite and discrete. In this way images are defined as digital data. In the book "Digital Image Processing" [21], Gonzalez defines the field as follows: "The field of Digital Image Processing refers to processing digital images by means of a digital computer".

There is a controversial problem among researchers about the boundary between image processing and other related areas such as image analysis and computer vision. A common opinion about the differences is that image processing has images as both input and output. However, Gonzalez et al. believed the above idea is "a limiting and somewhat artificial boundary" [21]. This dissertation is in agreement with Gonzalez et al.

A clear boundary between image processing and computer vision is hard to define. However, computerized processes can be classified in three levels: low-level, mid-level, and high-level processes. The low-level processes have inputs and outputs that are images and include some basic functions like contrast enhancement, noise reduction, and image sharpening. Mid-level processes take images as inputs, and have outputs that are image

features such as edges and contours. High-level processing is similar to image analysis and contains advanced tasks such as object detection and recognition.

In this dissertation, the above three levels of computerized processes are used for various purposes from image pre-processing to image analysis and computer vision, including tasks such as reducing noise in images, image feature extraction and object detection and segmentation. The phrase "Digital Image Processing" as used in this dissertation, includes processes where the inputs and outputs are images, and processes where inputs are images and outputs are image features or image feature recognition. The main role in this work of these processes is in image analysis and recognition employing modern approaches such as machine learning.

2.2 Background in Machine Learning

Machine learning is defined as concerning "the question of how to construct computer programs that automatically improve with experience" [22]. When running a machine learning algorithm, an expected result can be represented by a function *y(x)*, where x is the input data and y is the output vector that is "encoded in the same way as the target vectors" [23]. In this dissertation, for the detection and segmentation, x takes the form of crack road data (images or point clouds), and y is a $[1 \times 2]$ vector. The two values of the vector y indicate the two classes of the input data that should be recognized, crack and non-crack.

Broadly, machine learning algorithms can be classified into three types: supervised learning, unsupervised learning, and reinforcement learning. In supervised learning, human experts supervise the algorithms by feeding them the data with correct labels. In contrast with supervised learning, unsupervised learning is training without label data. Reinforcement learning [24] observes the interaction between a given situation and the environment to decide an action that can maximize a reward or minimize a risk. This dissertation mainly uses supervised learning, where the label or ground truth data are created by civil engineering experts.

Many different machine learning algorithms are implemented in this dissertation. However, the main contributions are based on deep learning algorithms and use Convolutional Neural Networks (CNNs). The following section is a brief introduction to CNNs and how they are used for object detection problems.

Background in convolutional neural networks

CNNs have established themselves as a powerful class of models for image recognition problems [25]. Before CNNs became widely used in classification problems, fully connected neural networks were used. however, the disadvantage of traditional fully connected networks is that there are a lot of weights that must be trained because in a fully connected layer each neuron is connected to every neuron in the previous layer, and each connection has its own weight. This results in long times for the training process, requiring a large amount of data and powerful hardware. This problem can be solved by using a convolutional network. There are two main aspects of a CNN that allow for effective training: firstly, the CNN uses grouped local connections by convolution with kernels; secondly, the CNN shares weights in all junctions.

The CNNs use the convolution function. The convolution function is used for the *Extraction of features* [26]. An input image is considered as a matrix $\mathrm{W_{m,n}}$ of size $M \times N$. This input image is convolved with a kernel $\mathrm{k_{p,q}}$ of size $P \times Q$. In the proposed architecture in Chapter 3, a square input image of size 96×96 pixels and a kernel of size 3×3 are used. This operation can be written as the equation 2.1, where $c_{i,j}$ is the (i,j)th element of the convolutional layer.

$$c_{i,j} = f(\sum_{p=m}^{M} \sum_{q=n}^{N} w_{i,j} \cdot k_{i+p,j+q} + b_{m,n}) \tag{2.1}$$

where f in equation 2.1 is a transfer function, $\sum_{p=m}^{M} \sum_{q=n}^{N} w_{i,j} \cdot k_{i+p,j+q}$ is the convolution operator of the input and the kernel, and $b_{m,n}$ is the bias input to the layer.

For training a CNN, a set of data D must be given:

$$D = \{x^{(i)}, y^{(i)}\} \tag{2.2}$$

where: i ranges from 1 to m, where m is the number of samples in total and $x^{(i)}$ is the ith sample and $y^{(i)}$ is the label of $x^{(i)}$. The variable $y^{(i)}$ may be equal to 0 or 1 and $x^{(i)}$ is denoted as negative or positive samples, respectively. We also denote $z_j^{(i)}$ as the output of unit j in the last layer for $x^{(i)}$. The probability that the label $y^{(i)}$ is j can be calculated by:

$$p(y^{(i)} = j|z_j^{(i)}) = \frac{e^{z_j^{(i)}}}{\sum_{l=1}^{k} e^{z_j^{(i)}}} \tag{2.3}$$

The response of the cost function is:

$$J = -\frac{1}{m}[\sum_{i=1}^{m} \sum_{j=1}^{k} 1\{y^{(i)} = j\} \log \frac{e^{z_j^{(i)}}}{\sum_{l=1}^{k} e^{z_j^{(i)}}}] \tag{2.4}$$

where $1\{.\}$ stands for the indicator function.

2.3 Cracking Phenomenon Understanding

Road condition is evaluated based on several indexes [1], where cracking index is an important example showing the damage caused by the separation of parts of the road surface in the form of cracks.

A crack on the road is defined in a report from the American Association of State Highway and Transportation Officials (AASHTO) [17] as follows: "A crack is a discontinuity in the pavement surface with minimum dimensions of 1 mm width and 25 mm length". The *PIARC Technical Committee C4.2 Road/Vehicle Interaction*, offer another definition of crack as "A crack is a discontinuity in the road surface that has a minimum length, width and depth". In Miller's technical report [10] most crack types presented are described including the crack direction factor. These definitions indicate that cracks on road surfaces contain four main characteristics: length, width, depth and orientation.

In a paper about crack detection and classification [3], Nguyen et al. describe cracks in road images as follows:

- Darker than its surrounding (because of the crack form, many light rays cannot be reflected from the crack to the camera),
- Continuous (the crack can be thinner than aggregates but it is always longer than them),
- Dominant orientation (over the entirety of the crack or on each segment of the crack there is a dominant orientation of the crack)

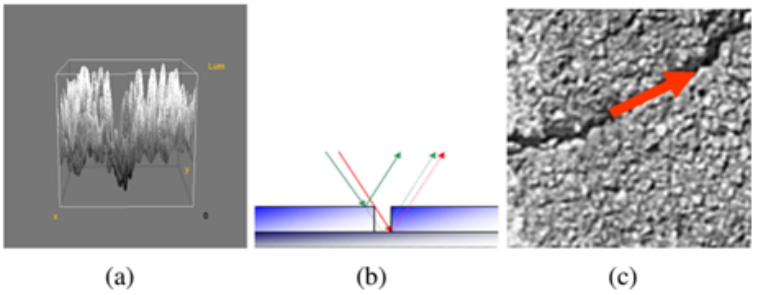


FIGURE 2.1: Crack Analysis on road images [3]. Sub-figure (a) shows a plot of the intensity of a crack image in three dimensions (*x, y, intensity*). Sub-figure (b) shows two different illuminations when light approaches the edge of the crack and the bottom of the crack. Sub-figure (c) indicates the direction of the crack.

Figure 2.1 illustrates some particular properties of cracks in road images. Figure 2.1(a) shows the intensity of a road image in three-dimensions, figure 2.1(b) is the illumination model and figure 2.1(c) is the dominant orientation of crack.

Oliveira, in his Ph.D dissertation [15], defined some features of a crack in images based on direction, width and length of cracks. Cracks should:

- Present a linear development along a given direction
- Exhibit a certain width, for instance, equal or greater than 2 mm
- Present a significant length

In experiments, Oliveira also used the width minimum of 2 mm and the length minimum of 25 mm as advised in a report from AASHTO [16].

In this work, a definition of a crack in pavement images is proposed. This definition can be applied for images in which the spatial resolution is such that 1 pixel is equivalent to 1 square millimeter.

A crack can be defined as:

- **Width:** At least 1 pixel;
- **Length:** Unconnected cracks or branch cracks have a length of at least 25 pixels;
- **Intensity:** A crack is lower (darker) than its surrounding by at least 5 gray levels for a 256 gray level - 8 bits image.

For maintenance purposes, an important consideration is whether or not a crack is connected to the network of cracks on the road or isolated. If an isolated crack is less than 300 millimeters in length (or 300 pixels in images), it will not be recorded [10].

Road crack detection and segmentation are essential processes in road maintenance. There are many image processing methods, including traditional and modern methods, for addressing this issue. Traditional methods use edge detection or some other low- or mid-level processes for crack detection, but these approaches are sensitive to many types of noise and unwanted objects on the road. For the purpose of increasing accuracy, pre-processing image methods are required before applying these techniques. Recently, some systems have been published that utilize machine learning techniques, such as deep learning, to detect cracks in images and these methods have achieved high performance. However, some of them are very complicated, some others use data that does not rely on a standardised or consistent collection process, and some methods still need pre-processing.

Despite the considerable work that has been done on this problem, the accuracy of current crack analysis methods in determining the required crack parameters such as crack width, length and direction, is often inadequate. In fact, because there is no clear definition of what constitutes a crack, current methods sometimes erroneously identify other image features as cracks. The lack of high quality ground truth data presents a particular impediment to overcoming the limitations of current methods.

The following sections will review the current state of crack detection, segmentation and classification using 2D data such as images and 3D data such as point clouds. Crack processing methods can use varying techniques from traditional digital processing to advanced neural networks with machine learning algorithms. Cracks on the road can be classified depending on severity levels or the distress types of the cracks.

2.4 Crack Processing using Traditional Digital Image Processing

In a traditional road maintenance system, cracks are detected and evaluated directly by road experts or road technicians under the supervision of experts. However, this manual method takes a lot of time and labor. Hence, automated or semi-automated methods that use images are expected to save time and improve performance in evaluating the crack indexes and road condition. There are many works that examine the automatic detection of cracks on road surfaces [7, 13, 14, 27–30]. These works indicate that the main reason why the crack detection issue is so difficult is due to the images containing a lot of noise and unwanted artifacts, for example shadows, dust and paint lines. In addition, in real world road survey systems, images are taken by a camera that is attached to a vehicle moving at high speed and under many different conditions [13]. So, the image quality varies and crack detection remains challenging. Besides that, much of the state of the art work on the road crack problem are restricted to crack detection and classification and cannot calculate crack indexes.

Methods for crack detection, segmentation and classification can be divided into two groups: those based on traditional Digital Image Processing (DIP), and those based on neural networks and machine learning. The traditional methods based on techniques such as edge detection, Gabor filters, and image features extraction will be reviewed in the below sub-sections.

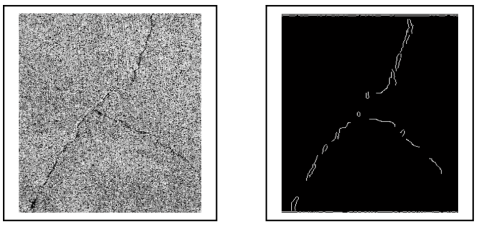


FIGURE 2.2: Improved Canny method result [4]. The left image is original image, the right image is the result from the Canny method. The technique is successful in removing black-white dot noise in the image, but does not preserve the continuity of cracks.

2.4.1 Edge detection

Edge detection methods are traditional DIP methods used in crack detection and classification. Edge detection is based on the first and the second order derivatives of an image [31]. Some methods that improve on the Canny approach [32], were proposed for crack detection [4, 33]. In the proposal of Huili Zhao et al. [33], they pre-process the road images by increasing the contrast of each image then apply the Canny edge detection method. The proposal of Zhao detects the cracks and is better at removing dot noise compared with the Canny method. However, this method cannot remove some other objects such as zebra crossings or other signage on the pavement. In Agaian's method [4], an extension method of the Canny approach was suggested using a fusion of two edge detection test images called the AC test images. The images can be processed by a "Modified Canny kernel" or a "Modified Canny kernel and Modified gradient Canny". This technique is applied to highlight the branch edge. The drawback of this method when applied to road images is that the crack results are not continuous as it should be as shown in fig 2.2.

Traditional Segmentation. In a review paper about crack segmentation [34], the authors survey seven different edge detection methods for crack detection. These methods are applied to 30 images and the results indicated that dynamic optimization is the best approach. The experiments in this paper illustrate that segmentation cannot distinguish the true crack from some similar features such as those from paint. Edge detection can also be based on the wavelet transform [35–37]. Cuhadar et al. utilized the wavelet transform for road condition determination [38]. This work was applied to IRI data and other types of pavement condition data and used the wavelet transform as a segmentation method.

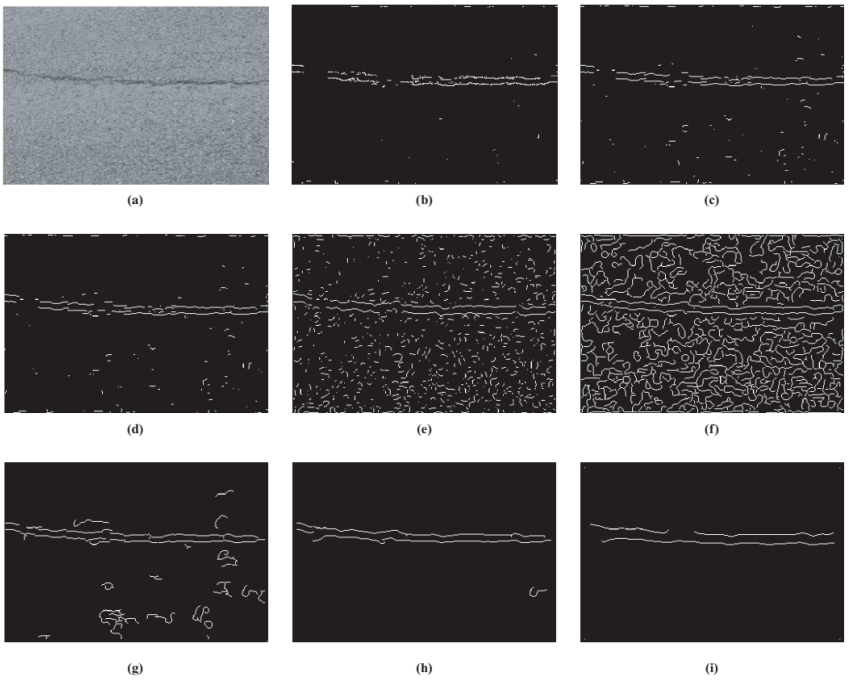


FIGURE 2.3: Edge detection experimentation results (pavement image 1): (a) original image, (b) Robert edges, (c) Sobel edges, (d) Prewitt edges, (e) LOG edges, (f) Canny edges, (g) à trous algorithm–based edges at scaling 21, (h) à trous algorithm–based edges at scaling 22, and (i) à trous algorithm–based edges at scaling 23 [5].

But this work only removes noise in data and must be supported by the Geographic Information System (GIS). Wavelet transform based methods were used as an edge detection for pavement images [5]. Wang et al. [5] used Mallat's algorithm [39] and the à trous algorithm for multi-resolution analysis and their works were successful in detecting longitudinal cracks and potholes. However, this work was not applied to a wide range of road conditions. Besides that, the à trous algorithm requires excessive computation and memory. In closing, edge detection is not a powerful method because there is a lot of noise in road imagees that could be detected as cracks. Figure 2.3 shows a comparison of some traditional filters for edge detection and the à trous algorithm at different scales. The results indicate that the à trous algorithm removes more noise than traditional filters. However the results depend a lot on the scaling.

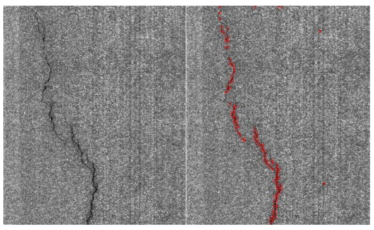


FIGURE 2.4: Gabor filter applied to a road image. The left image is the original image, the right image is result from the Gabor fiter [6]. The detected cracks seem bigger than the real cracks.

2.4.2 Gabor filters

Gabor filters perform image filtering based on Gabor functions [40]. These filters are also used for feature image extraction [41, 42]. Gabor filters are effective for texture segmentation [43] and road images can be thought of as texture images, so Gabor filters are used for pavement crack detection and segmentation [6, 44]. In a proposal for crack detection [6], Salman used a bank of Gabor filters, that contain filters with multiple orientations. The number of filters used in a Gabor filter bank controls the quality of the output. The higher the number of orientations, the higher the accuracy but this also increases the computational time and false positive rate. The advantage of this method is that most of the crack pixels are detected and so the crack line is segmented. However, this method is also sensitive to noise. Figure 2.4 shows the result of the Gabor method and it can be seen that some weak cracks are not detected.

2.4.3 Adaptive thresholding

Adaptive thresholding is another common technique used in many computer vision and graphics applications [45]. The adaptive thresholding technique works by comparing a pixel to the average of nearby pixels and ignores low gradient changes. In this proposal [45], Bradley et al. utilized a tool called an "integral image", that is also used in a proposal for face detection [46], for decreasing the number of operations when calculating the sum of a rectangular region in an image. Adaptive thresholding is a powerful technique for image processing that compensates for spatial changes in illumination and removes noise. In another crack detection paper [47], Fan et al. used a deep CNN model to detect the

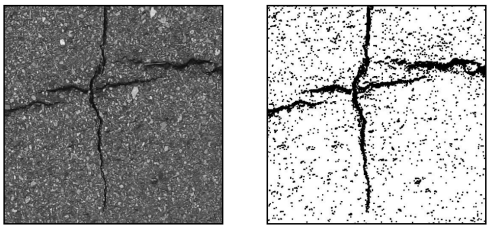


FIGURE 2.5: The result of the NDHM method [7]. The left is the original image, the right shows the NDHM result. The NDHM method is sensitive to noise as shown by the many black dots detected.

area that contains the crack, then applied an adaptive thresholding method to segment the cracks from areas where cracks were detected in the image. This method is simple and fast. Adaptive thresholding also achieves high accuracy on images with strong cracks. However, this tool cannot remove dot-like noise and road image data often contain a lot of dot-like noise.

2.4.4 Crack detection based on image features

Image features such as the Neighboring Difference Histogram Model (NDHM) [7] can be applied to crack detection. The NDHM approach executes the weighted difference between the number of the potential crack pixels with the number of surrounding pixels. The probability of a pixel being a crack pixel increases if the number of crack pixels surrounding it increases. Li et al. [7], compare the proposed method and some other classical thresholding methods such as Otsu [48] and Kapur [49] and show that the NDHM method achieves superior results. This method can bring out almost all of the crack pixels but it also detects noise and shows dots and salt and pepper noise in the output as illustrated in the result in figure 2.5

In a proposal from Hu et al. [8], Local Binary Patterns (LBP) feature classification is also applied for pavement crack detection. In this work, there are 35 features of an image which are extracted and grouped into six sub-classes. The input image is divided into 3x3 blocks. Then, every block is mapped with an LBP feature and divided into a specific pattern. After that, a pixel will be defined as a crack or not based on the patterns. A comparison of the LBP feature with the Canny detection method is shown in the top row

of fig 2.6. However, in difficult cases such as the paint in the bottom row of fig 2.6, both the LBP features and the Canny filter detect the paint as a crack.

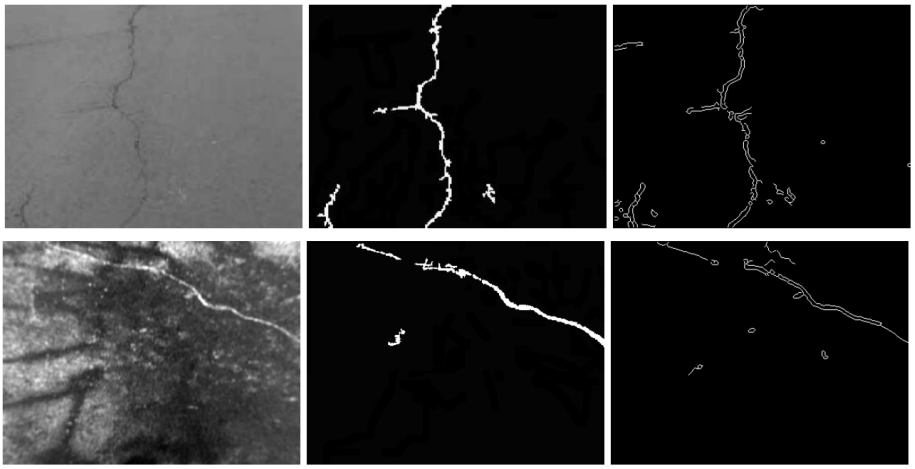


FIGURE 2.6: Crack detection based on LBP features [8]. The left column shows the original images, the center column shows the result from Hu, and the right columns shows the results of the Canny method. The original image in the second row contains paint; the method from Hu and the Canny method detect the paint as a crack.

2.4.5 Geography features

Geography features have been used for cracks detection in many kinds of materials [50] and for crack detection on roads [9]. The proposal of Nguyen et al. [9], consider a crack to be a line-like structure with a Gaussian cross-section intensity profile as shown in fig 2.7. After enhancing crack images using a filter based on a Gaussian function, the crack center-line is extracted. The final step is removing fake cracks such as unwanted edges. This method [9] needs some pre-processing and post-processing steps before it is able to reliably identify true cracks.

The combination of 2-dimensional images and 3-dimensional data processing techniques is used to show the detail of cracks in a proposal for enhancing crack detection by Median [51]. The method of Median works by extracting the geometric information of the road image. Median et al. achieve a high balanced accuracy of 96% and 93% for transverse cracks and longitudinal cracks respectively. In some older proposals [15, 30] a method that is based on crack characteristics, for example, the width of crack, the curve of the crack and the length of crack, was applied to identify whether a pixel is a crack pixel or not.

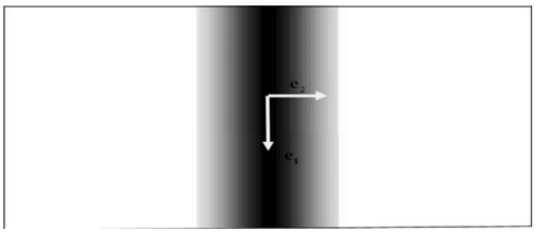


FIGURE 2.7: Geometric crack as a Gaussian function [9]. A crack intensity can be modelled as a Gaussian function with the darkest value (lowest intensity) in the middle of the crack.

In brief, these image features are not effective in crack detection because road images are often obtained with uncontrolled illumination and taken in different conditions.

In summary, methods based on traditional image processing explore the change in intensity of the crack pixels and the surrounding pixels or background. These methods are mostly simple and use less hardware memory. However, this kind of approach is sensitive to noise and artifacts in road images. The following sub-section reviews methods that use neural networks in machine learning for object and road crack detection and segmentation.

2.5 Crack Processing based on Machine Learning with Convolutional Neural Networks

2.5.1 Crack detection

Deep learning is a family of machine learning methods based on multiple layers of artificial neural networks. Neural networks in machine learning are widely used in crack detection and segmentation, and these models have many advantages over traditional machine learning models [28, 29, 50, 52, 53]. Deep learning models can learn features in images automatically, while traditional machine learning approaches need image features that are designed by users. Deep learning can also learn subjective defects which are hard to train —like minor product labeling errors. Recently, deep learning has become a powerful method that is used for detection and segmentation problems. There are 12 techniques that were examined for crack detection, six of which are based on neural networks including unsupervised and supervised methods [15]. Oliveira used two level of

crack detection: pixel-based crack detection and block-based crack detection. The accuracy of block-based crack detection is high, around 90%. However, this approach requires complicated pre-processing of the data before training and testing.

Deep learning methods based on CNN for detection are used in various publications for crack detection [28, 29, 51, 54].

In a paper from the International Conference on Image Processing (ICIP) [28], a CNN was applied to detect images that contain cracks. The colour images used in the study were collected by smart phones. So, the data in those experiments are different from data collected by practical systems: black - white images collected by a special, fast moving camera. Zhang defines a positive sample as a patch whose center is within $f = 5$ pixels of the crack centroid, and a negative sample is a patch that contains no cracks. In this paper, the authors compare the receiver operating characteristic (ROC) curve when using a Support Vector Machine - SVM [55], a Boosting algorithm [56] and the proposed model. The result shows that the proposal obtains the highest AUC (Area Under the receiver operating characteristic (ROC) Curve), of 0.845. The proposed model achieved a final F_1 score of 89%, which is not particularly high for the crack detection problem and the cracks detected in the results are bigger than the real cracks. In addition, Zhang's model is trained with a large number of kernels (48 kernels in every CNN layer), so the CNNs may take a long time for training.

Maeda et al. in their method [54] surveyed different kinds of deep learning and convolutional networks that were used for detecting road damage and cracks in images. A new data set was collected by a smart phone that is attached to a moving car. There are eight types of road damage in this data set, including five kinds of crack, rutting, white line blur and crosswalk blur. Maeda et al. [54] compares the performance of a GPU server and a smartphone when they are used for crack detection. The experiments achieve a precision and recall of around 75% with an inference time of 1.5 seconds on a smartphone. Although the final results are not particularly high, this work is the first work involving experiments on crack detection using a smartphone.

Fan et al. in another proposal for crack detection [29], applied a CNN model to two public road crack datasets. Two crack datasets without pre-processing: CFD [57] and AigleRN

[58], were used by Fan et al. [29] to solve the problem of imbalanced data, with the ratio of positive pixels to negative pixels being 1:65 and 1:98.5 in the CFD dataset and the AigleRN dataset respectively. The training phase uses crack images with labels, and the experiments show that training ground truth with thinner crack labels leads to thinner output cracks [29]. The proposed architecture contains nine layers in the following order: two CNN layers that have 16 kernels in each layer, a max-pooling layer, two CNN layers where each layer has 32 kernels, and three fully-connected layers. The kernel size in each CNN layer is 3×3, and the kernel size in each max-pooling layer is 2×2. This kernel size is commonly used in crack detection problems [2, 29]. The final score is high, more than 90%; however, Fan's method uses a multi-label classification model, so it needs post-processing to discriminate between cracks and non-cracks. Besides that, the small number of training images (72 images of the CFD data set and 24 images of the AigleRN data set) risks creating an unstable model. Zhang et al. proposed a CNN model for 3D pavement crack data [59]. This model reaches a high accuracy with data in good condition, but there are some samples which fail. These samples have weak cracks and input images with noise such as shadows, paint or other crack-like patterns. In addition, the small sample size used by Fan [29] cannot apply to low spatial resolution images that contain extensive noise and artifacts like the 2StagesCrack dataset [60]. Moreover, a high F1-score was achieved by Fan with an imbalanced training set with twice the number of negative samples compared to positive samples. This approach can lead to a high number of false positives [61].

Another model based on CNN, Zhang et al. [28], developed a deep learning method for crack detection, however the output crack detection probability map shows that the probability of actual crack pixels and some surrounding pixels are the same. Hence the detected crack is bigger than the real crack. There is a similar situation in a proposal from Nguyen et al., Nguyen et al. [2], where the proposed method also outputs cracks much larger than the ground truth.

The *YOLO v2* model [52] has been applied to automated road crack detection and classification. *YOLO v2* is based on the original YOLO deep learning method for real-time object detection [62]. The YOLO method works by dividing the input image into small boxes and predicting the coordinates of bounding boxes and class probabilities for these boxes. The *YOLO v2* method achieved an F_1 score of 0.87 overall for crack and other damage

detection on roads. However, this work focused just on detecting the area that contained the cracks and not on pixel-level segmentation. A model called RetinaNet [63], based on deep learning, has been proposed for road damage detection. RetinaNet can use different neural networks as the backbone for learning feature maps. A disadvantage of RetinaNet is that this model detects some artifacts like paint, and line shadows as cracks. A two-module model was proposed to detect cracks rapidly [64]. The first module extracts cracks from road images at the same size as the original image, then patches that contain cracks are cropped, and the second module detects the cracks from the cropped patches. This proposal obtained a high precision and recall at 0.9774 and 0.9521, respectively. However, Park's model failed for images that contain both cracks and road markings, or images that have cracks on the border.

Summarising, CNN architectures for crack detection have some common characteristics as below:

- Input samples are positive samples and negative samples. Positive samples contain crack pixels in the centre, and negative samples have no cracks.
- CNN architectures for crack detection often contain three to five convolutional layers. The number of kernels in each convolutional layer can be unchanged or increase from the first layer to the final layer.
- *Max-pooling* layers are used to decrease the number of parameters.
- *Softmax* is utilized in the last layer, which is a fully connected layer with two neurons, this layer classifies the input image into crack (positive) or non-crack (negative) classes.

The above works for crack detection can be considered as crack detection at the *sample level*. The accuracy of experiments is evaluated at the image region classification level, not the pixel classification level. Therefore, some segmentation methods are proposed to detect the crack at the smallest unit of the image, the pixel. In the next section, some works that are used for crack detection at the pixel level or crack segmentation will be reviewed.

2.5.2 Crack segmentation

This section reviews some papers using CNNs for general object segmentation first. Then, some works that are used for crack segmentation are surveyed.

Region-Based Semantic Segmentation (*R-CNN*) is proposed in a paper published in 2014 [65]. In this work, Girshick et al. presented a model that can detect and segment an image based on regions with CNN features, which was denoted as R-CNN (Region-Convolutional Neural Networks). R-CNN contains three modules: the first module generates approximately 2000 category independent region proposals; the second module extracts a feature vector from each region by applying a large CNN; the third module classifies the regions by a linear algorithm such as a Support Vector Machine (SVM) [66]. R-CNN creates bounding boxes for all proposed regions, so that this network can segment an image into different parts based on proposed regions. This model gets a mean Average Precision (mAP) of 53.3% on VOC2012 (The PASCAL Visual Object Classes Challenge 2012). There are some drawbacks of R-CNN that are evident in Girshick's proposal [67]. In the training phase, R-CNN requires a large amount of memory and is slow, while in the testing phase, object detection is slow.

Fast R-CNN is an improved proposal of R-CNN [67]. The main idea of Fast R-CNN is the sharing of computation in performing a convolutional network forward pass for each object proposal. In the Fast R-CNN architecture, there are two new points. First, it uses regions of interest (RoI) as the input to the pooling layer when extracting a feature vector from the feature map. Second, the output of Fast R-CNN contains two sibling layers: one for estimated object classes and the other for refining bounding box position for the detected object. The training time and testing time of Fast R-CNN are decreased compared with R-CNN and SPPNet [68]. However, selective search computing should be used to speed up the training time.

Mask R-CNN is introduced in a paper of He et al. [69] and is an extension of Faster R-CNN for the purpose of segmentation at the pixel level [70]. In Mask R-CNN, a mask is added in parallel with the existing branch for bounding box recognition. Furthermore, He et al. introduces key elements of Mask R-CNN, including a pixel-to-pixel alignment method called RoIAlign. RoIAlign removes the harsh quantization of the RoI pooling

layer and properly aligns the extracted features with the input. Mask R-CNN can exactly locate the pixels of each object instead of just bounding boxes. Experiments show this approach can achieve higher mAP than earlier R-CNN versions. However, the mAP of Mask R-CNN for small objects is still low, which means that this model is not suitable for segmenting small objects.

Fully Convolutional Networks (FCN) was introduced for segmantic segmentation [71] by Long et. al. FCN transfers knowledge from VGG16 [72] to perform segmentation. In the FCN architecture, a 1x1 convolutional layer is used to convert the fully connected layers in VGG16 into fully convolutional layers that enable a classification net to output a heatmap in low resolution. These low resolution semantic feature maps are the input of the upsampling part, which is done using transposed convolutions. The upsampling process is further refined through every stage. FCN can take inputs of arbitrary size and generate the output in the corresponding size. In addition, the semantic segmentation can extract the spatial information of an object and this method achieved a mean Intersection over Union (IoU) higher than other nets for segmentation such as R-CNN. Even so, FCN is a bit complicated and takes a long time in training because of the large number of kernels in convolutional layers.

U-Net was originally proposed for biomedical image segmentation [73], and it is widely used for other image segmentation applications. The U-Net architecture used a segmentation model that includes two parts: a contracting part for computing features and an expanding part for spatially localizing patterns in the image. While the contracting part uses max-pooling layers to decrease the number of parameters and decrease the image size, the expanding part uses up-pooling layers to increase image size. Hence, the output image size is equal to the input image size. In U-Net architecture, the authors use *concat* layers that concatenate feature maps in the same level of two parts and the concat layers improve the localization accuracy of objects. This network contains 23 convolutional layers in total. *Overlap Title* and *separation of touching objects* are the main strategies of Ronneberger's proposal. While the *Overlap Title* strategy lets the U-Net model predict part by part the whole image, the remainder technique forces the network to learn the small separation borders between touching cells. The U-Net model achieved a high IoU result, at 0.9203 on the ISBI cell tracking challenge.

SegNet is a deep convolutional encoder-decoder architecture for image segmentation [74]. This proposal has been used to predict indoor scenes and outdoor road scenes. The encoder network consists of 13 convolutional layers like the first 13 layers in the VGG16 network [72] and produces a set of feature maps. For each of the 13 encoders, there is a corresponding decoder which up-samples the feature map. There is a special technique that SegNet only stores *max-pooling indices*, therefore the number of parameters is decreased. This work shows the performance of SegNet is equal to that of FCN. In addition, SegNet uses less memory than FCN. SegNet is effective in boundary delineation and experiments show that SegNet outperforms all the other methods for most objects.

XNet is another CNN that is used for segmentation of medical X-Ray image data [75]. XNet is based on an encoder-decoder that is commonly used in image segmentation. Although this net used small data-sets for training, just 108 images of 10 kinds of body part, the results show an overall accuracy of 92% and AUC of 0.98. The architecture of the X-Net encoder includes a smaller serial down-sampling component compared with many other segmentation networks. Furthermore, Bullock et al. use two encoder-decoder modules for improving features extraction and avoiding overly-reducing the image resolution. X-Net gains more than 90% in all evaluating indexes such as F1-Score, and AUC when applied to three categories of the X-ray images. In this paper, the authors also compare XNet with SegNet. The results show that XNet gives a higher F_1 score in every category and XNet uses a small number of parameters.

DeepCrack is a new model that utilized a deep hierarchical neural network for crack segmentation [76]. In this paper, Liu et al. propose a deep hierarchical CNN for crack segmentation at the pixel level. DeepCrack uses the first 13 layers which correspond to the first 13 layers in VGG-16, but the fully connected layers and fifth pooling layer are removed to achieve meaningful side-output with different scales and decrease the memory requirements and computation time. A guide filter that is based on *guided feathering* [77] achieves the final refined prediction and noise removal in the low level prediction. The data set used in this paper includes more than 500 images, some of them are downloaded from the Internet, and some others are captured by the authors. So, these images are have differences in spatial resolution and characteristics, which is a problem for this data set in

real world applications. The DeepCrack model achieves the mean IoU of 85.9 and an F_1 score of 86.5%.

In summary, the above image segmentation architectures often share some common characteristics:

- The methods contain a backbone in the network architecture. This backbone often includes two symmetrical parts: encoding and decoding or contracting and expanding;
- Some works use existing networks such as VGG16 [72] as the encoding part, and propose a new architecture for decoding part;
- The methods use concatenation layers;
- The methods use the convolutional 1×1 layer for channel-wise pooling, often called a projection layer or feature map pooling.

While a disadvantage of the crack detection methods is that those methods just detect samples or small areas that contain cracks so the crack detected is much bigger than the real crack [2, 28], the drawback of crack segmentation methods is that road images are often of variable resolution and contain a lot of artifacts, in particular, noise that is hard to remove. Therefore, some new approaches have used two stages for crack detection and crack segmentation in one architecture. The next section will focus on reviewing the new proposals that used a two-stage architecture.

2.5.3 Two-stage architectures for crack detection and segmentation

A two-stage model that contains a first stage based on a CNN and a second stage based on adaptive threshold was used in a work for crack detection [47]. In the first stage, an input image is classified into positive (contains cracks) or negative (contains no cracks) regions using a seven-layer CNN model, the second stage uses an adaptive threshold method to post-process the results from the first step. This method achieves high accuracy at 98.70%, however, some colour samples with a lot of noisy pixels cannot be properly segmented. In

TABLE 2.1: Comparing two-stage architectures used for detection and segmentation

Methods	Stage 1	Stage 2
Fan's method [47]	Deep learning using CNN	DIP technique using: - Adaptive thresholding
Nguyen's method [78]	DIP techniques using: - Morphological transform - Contrast enhancement	Deep learning using CNN
Park's method [64]	Deep learning using CNN	Deep learning using CNN
Ni's method [79]	Deep learning using CNN (GoogLeNet)	Deep learning using CNN (ResNet)

contrast with the method of Fan *et. al* [47], Nguyen *et. al* [78] uses some DIP techniques in the first stage and CNN in the second stage. Nguyen's method uses several morphological transforms and contrast enhancement methods, however it is hard to choose a threshold in each DIP task to make the decision on whether a pixel belongs to a crack or not.

A two-stage model where each stage is based on a CNN is proposed in a paper for patch-based crack detection [64] and in a proposal for measuring thin-crack width [79]. The patch-based model [64] obtains a high F_1 score, however, this proposal is not strong enough to detect cracks from some artifacts that look like cracks, such as road markings. The method for crack measuring [79] uses two models that are based on CNNs, where each CNN model detects cracks at different scales. The first stage uses the GoogLeNet model [80] for detecting crack samples at the size of 224×224 pixels, and the second stage uses the ResNet architecture [81] for detecting crack samples at the size of 32×32 pixels. There was no report on CNN performance from Ni et al. [79], however, the GoogLeNet and SegNet models are huge architectures, and have a long testing time and need powerful hardware.

These existing two-stage architectures often use one stage of traditional DIP and one stage of deep learning, or two stages of deep learning. Table 2.1 shows a comparison of some methods that use two stages for crack detection and segmentation.

In summary, image detection and segmentation based on CNN methods achieve a high accuracy in crack road problems. However, some post-processing methods should be used for calculating the crack length, crack width and crack orientation on the road images.

The next section reviews some works that solve the problem of evaluating the crack characteristics and cracks classification.

2.6 Related Work for Crack Width and Crack Length Calculation

2.6.1 Crack length calculation

Crack length is defined as "the length measured along the crack path using all available data points between crack termini" [17]. Few researchers have tried to define the crack length. Ni *et. al* used the Zhang–Suen algorithm [82] to obtained the skeletons of cracks in their paper [79]. To solve the problem that the Zhang–Suen algorithm results are not single-pixel width skeletons, the authors used post-processing to obtain single-pixel skeletons. This post-processing method deleted pixels in some specific cases to avoid non-single pixels, so this technique also missed some valid pixels. Ni et al. also showed that their method suffered a high error when applied to cracks that are thinner than 5 pixels, and this problem may be due to the use of Otsu's method [48] to binarise the segmented cracks.

2.6.2 Crack width calculation

Crack width is defined as the average gap in millimeters (or inches) between the two edges of a crack measured at points along the crack [17]. There has not been much research on crack width calculation. In a Ph.D. dissertation for crack detection and characterization [15], the author uses a skeleton technique, then measures the branch skeletons to estimate the crack width. A drawback of Oliveira's method is the detected crack width only depends on the width of cracks that respond to the skeleton filter. Ni et al. defined the perpendicular lines with the skeleton, then the crack width is defined by the distance between two intersection points of the crack edge and the perpendicular line [79]. This method has two disadvantages: first, it depends on the edge detection results, and edge detection methods are sensitive to noise; second, if there are two close cracks, it is hard to

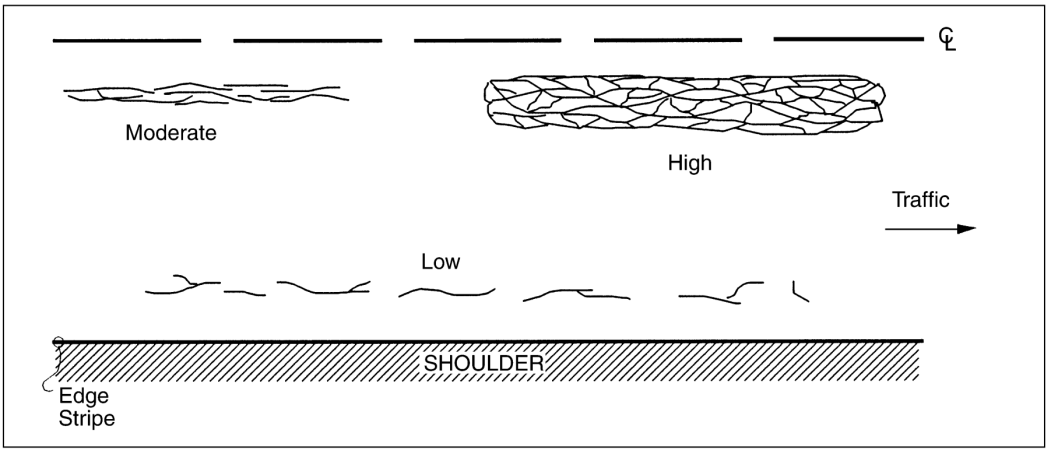


FIGURE 2.8: Fatigue crack examples [10]. The high level severity is indicated when the cracks are highly connected. This type of crack is aligned with the direction of movement of the traffic.

define which edge belongs to each crack. The proposed method to calculate crack width in this dissertation addresses these problems.

2.7 Crack classification

2.7.1 Crack classification based on types of crack

In a manual for the long-term pavement performance program [10], a crack is identified as a common type of damage on road surfaces, including asphalt or concrete surfaces. Cracks can be divided into 6 types [10] as below:

Fatigue Cracks

Fatigue cracks are also called alligator cracks, because the cracks are connected and show a shape similar to alligator skin. Traffic loading is the cause of alligator cracks. So, this crack type often occurs where the wheel paths are used repeatedly by heavy trucks. Fatigue cracks can be measured by the number of cracks per square meter.

Figure 2.8 shows some examples of fatigue cracks. It can be seen that the more connections between the cracks, the higher the severity of the distress.

Block cracks

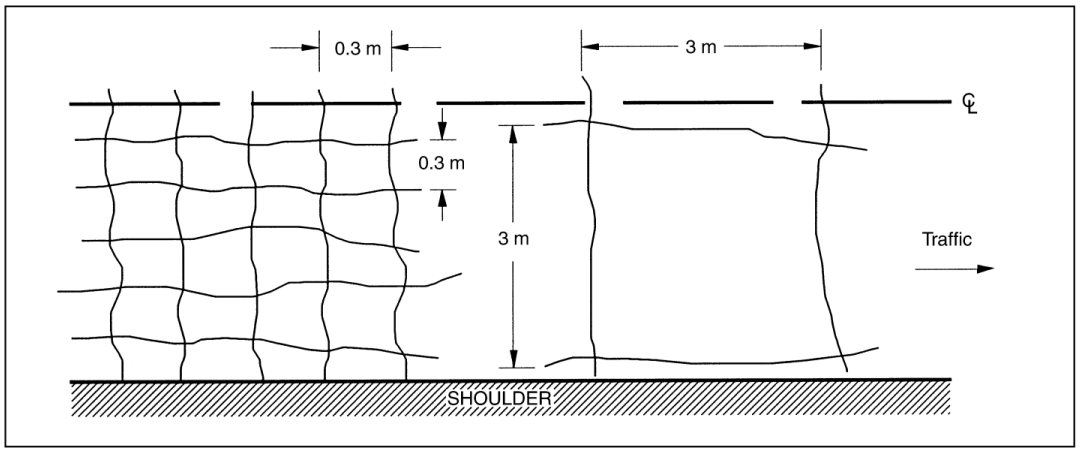


FIGURE 2.9: Block Cracks [10] are shown as cracks developing in the longitudinal and transverse direction and tend to create like-square blocks. The size of block is as small as $0.3m \times 0.3m$ or as big as $3m \times 3m$

Block cracks divide the road surface into rectangular pieces, where each rectangle can have a size from 0.1 to 10 square meters. The "fatigue" phenomenon on roads is the main reason causing block cracks. Figure 2.9 shows two examples of block cracks. The left cracks are $0.3m \times 0.3m$ block cracks and the right cracks are $3m \times 3m$ block cracks. The smaller block sizes of crack indicate a higher level of distress.

Edge cracks

Edge cracks are defined as crescent shaped cracks that occur from the edge of the road to the shoulder of the road. These cracks can be measured by recording the length of pavement edge affected. Figure 2.10 shows a typical example of edge cracks. Material fatigue of the road surface may be the source of the edge cracks.

Longitudinal cracks

Longitudinal cracks are parallel with the road center line. For distinguishing longitudinal cracks from other types of cracks, the longitudinal cracks must be within the lane of the road. Figure 2.11 shows two types of longitudinal cracks: wheel path and non-wheel path.

Reflection cracks

Reflection cracks occur on the surface above the joints of concrete pavements. Figure 2.12 illustrates an example of reflection cracks. The "reflection" damage from the layer under

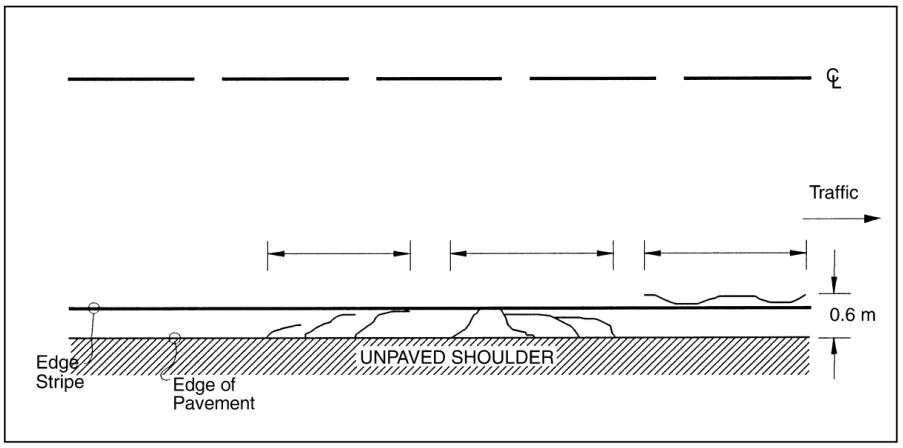


FIGURE 2.10: Edge cracks [10] are cracks near the unpaved shoulder. Edge cracks connect the edge stripe and the edge of pavement.

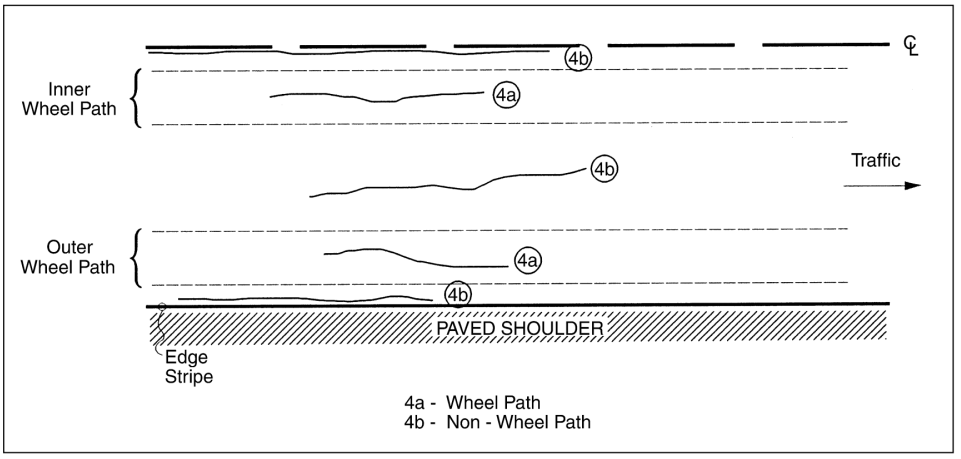


FIGURE 2.11: Longitudinal Cracks [10] are oriented along the direction of traffic flow. This type of crack occurs on every part of the road surface, from edge to the middle.

the road surface causes the reflection cracks, so the reflection crack belongs to the joint of the underlying layer.

Transverse Cracks

Transverse cracks are predominantly perpendicular to road center-line. The width of transverse cracks may be different along the cracks as shown in figure 2.13. So, the crack width must be defined in different parts of transverse cracks and indicated as the minimum, maximum or average of crack width.

Practicing measuring road cracks

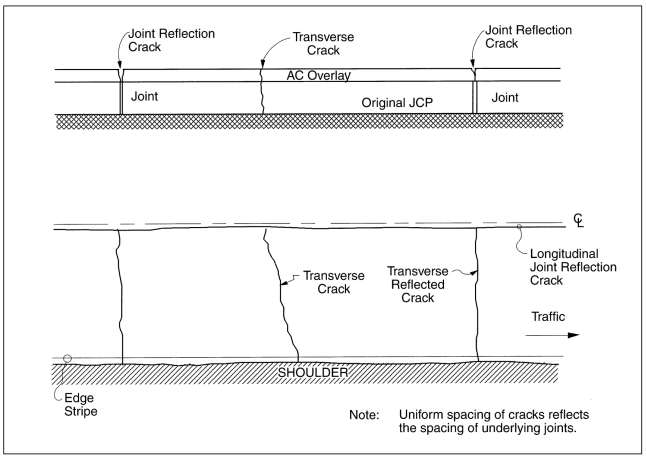


FIGURE 2.12: Reflection Cracks [10] occurs when damage starts from the layer under the surface layer and is reflected by the road surface. Reflection cracks look like transverse cracks.

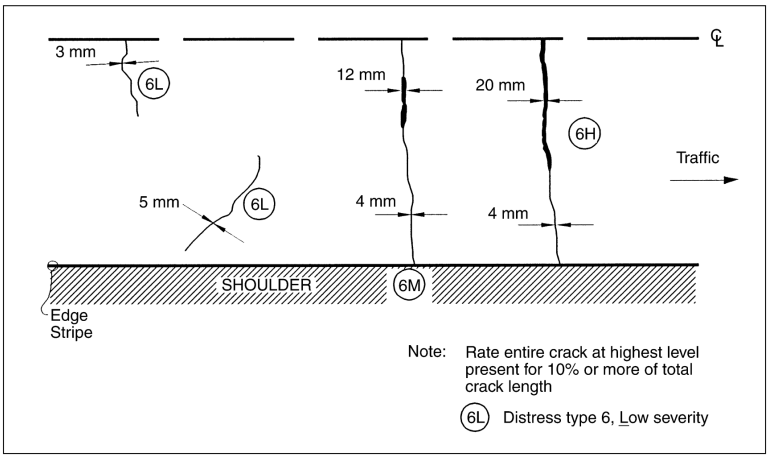


FIGURE 2.13: Transverse Cracks [10] can be evaluated by the width of cracks. Transverse cracks have width from as small a value as 3 mm to as large a value as 20 mm.

Each type of crack can be measured in a different way. However, some common tasks should be done for measuring crack characteristics:

- Using a tape to define the length of a crack;
- Using a special ruler to estimate the width of a crack;
- Calculating the total number of cracks within a square meter.

Table 2.2 shows a summary of types of crack and the classification based on the crack severity levels and the parameters of the crack that must be recorded and calculated.

TABLE 2.2: Types of cracks and severity levels

Type of crack	Severity levels			Parameter must be collected
	Low	Moderate	High	
Fatigue crack	- None or only a few connecting cracks	- Interconnected crack forming a complete pattern	- Severely Interconnected crack forming a complete pattern	- Crack severity levels in square meters
Block crack	- Crack width <= 6 mm	- Crack width from 6 mm to 19 mm	- Crack width >= 19 mm	- Record crack width - Crack severity levels in square meters
Edge crack	- Crack with no breakup or loss of material	- Breakup and loss material up to 10%	- Breakup and loss material more than 10%	- Record length of pavement edge affected
Longitudinal crack	- Crack width <=6 mm	- Crack width from 6 mm to 19 mm	- Crack width more than 19 mm	- Record the length of longitudinal crack
Reflection crack	- Crack width <= 6 mm	- Crack width from 6 mm to 19 mm	- Crack width more than 19 mm	- Record the length and width of crack
Transverse crack	- Crack width less than 6 mm	- Crack width from 6 mm to 19 mm	- Crack width more than 19 mm	- Record the length and width of crack - Record the crack in square meters

In summary, characteristics of cracks are used for crack classification. There are three main features of a crack should be collected, calculated, and evaluated as below:

- Crack width
- Crack length
- Crack orientation

2.7.2 Crack classification based on the severity levels and distress types

Crack classification based on the severity levels is researched in a Ph.D dissertation on crack detection and characteristics [15]. In this work, Oliveira detects the cracks based on the block level first. Then, the author calculates "connected component crack blocks" (*cccb*) and the skeleton of the *cccb*. Finally, the cracks are divided into three types: large, medium and small, based on the width of the cracks. Henrique also exploits the standard deviations of the column and row of the cccb and uses these deviations as features for classifying the cracks as longitudinal cracks, transversal cracks or miscellaneous cracks. The results show Henrique's method detects cracks in his data set at 100% accuracy. The disadvantage of Henrique's method is that it cannot distinguish the cracks at large and medium severity level. On the other hand, crack classification based on the direction of

cracks suffers from a large number of false positives if the cracks are thin with a width smaller than 2 mm.

A *Dual-scale deep learning* model is proposed by Ni et al. [79] to detect cracks and calculate the width of the cracks. In this paper, the authors use two existing models, GoogLeNet [80] and ResNet [81], to detect cracks at two different scales. First, GoogLeNet is used to detect the large scale samples that contain cracks. Then, the ResNet model localizes the cracks at the small scale. The data set in this case study is collected from a self-setup camera. Width of the cracks in this data set is thin, from appropriately 0.2 mm to 2 mm. The experimental results show that Li's method gets a high accuracy of 99.8%, precision of 93.5%, and recall of 92.9%. Although this paper does not focus on crack classification, their method can be used for crack width calculation and is valuable for estimating crack severity levels.

There is currently no widely accepted solution to the crack classification problem. Some published papers classify the cracks in two ways: the first way is based on the severity levels, and the second way is based on the distress types. In both cases, the crack characteristics must be calculated, but the two approaches do not yield consistent results.

This dissertation will develop methods for calculating the cracks length and cracks width. From these features, almost all types of crack can be classified.

2.8 Crack Point Clouds Processing Related Work

Significant advances have been made in the upsampling of point cloud data in recent years [83–85]. There are several methods for upsampling point clouds, including both traditional methods and modern methods based on CNNs.

2.8.1 Traditional methods

Traditional methods such as interpolation between input points, have some disadvantages, arising principally from the fact that point clouds often do not have any spatial order or a regular structure [83]. In a method for the extraction of break-lines and ground points

from lidar point clouds [86], experiments showed that interpolation errors are mainly distributed around the break-lines. This strongly suggests that interpolation methods may not be effective for crack point cloud processing. Some previous experiments [83, 87] also indicated that methods based on neural networks demonstrated superior performance compared with traditional methods. Hence, we focus on these advanced methods using CNNs for upsampling point clouds.

2.8.2 Convolutional Neural Networks for point cloud upsampling

Methods based on CNNs have been used by many researchers for upsampling point clouds [83–85, 88–90]. These methods often have two parts. The first part extracts features from point clouds, and the second part reconstructs points from feature expansion and optimizes the output by comparison with ground truth point clouds.

PU-Net [83] uses a network architecture that has four components; patch extraction, point features embedding, feature expansion, and coordinate reconstruction. The advantage is this model can learn both local features and global features of a point cloud. PU-GAN [84] is a proposal for upsampling point clouds using a Generative Adversarial Network (GAN) and demonstrates improvements in the quality of the resulting point clouds compared to earlier approaches. A method called PCSR used Adversarial Residual Graph Networks for upsampling point clouds [85]. Their experiments show that the residual blocks are effective for obtaining better performance and stable training. However, this approach has some disadvantages. These models cannot fill large holes or missing parts, and are also not effective for very small structures. The method we propose, based on CNN and the combination of point clouds and images improves the quality of up-sampled point clouds and solves the problem of filling in missing information.

2.8.3 Combining 2D images and 3D data

There are existing point cloud processing methods using a combination of 2D images and 3D data. Based on the level of features used for combination, these works can be divided into two main types: Low-level feature combination and high-level feature combination.

Low-level feature combination

In methods using feature combination, the low-level features collected by simple transforms are combined with point clouds or 3D data. Image features such as grayscale pixels and entropy values have been combined with depth information from point clouds for upsampling point clouds [91, 92]. In these works, the depth values for locations in the point cloud are assigned to corresponding pixels of an image, and then the upsampled point clouds are created by interpolation using the local entropy of pixels around the current pixel being processed. However, the simple combination of the depth value of each point and the grayscale image data may lead to aliasing errors due to the inconsistencies in the directions of gradients in the depth data.

High-level features combination

High-level image features can be extracted by CNNs and used to enrich point clouds. In a proposal for LiDAR Point Cloud Segmentation [93], an effective fusion method of RGB data and LiDAR was developed to combine features from colour images with features from point clouds to segment the 2.5D point clouds. Yang et al. [94] combine features from images and 3D data for object reconstruction. They used images for reconstructing 3D objects, demonstrating that features from images could be concatenated with features from 3D data. The combination of different kinds of features such as image features and point cloud features can be considered as a form of *transfer learning* [95].

This dissertation proposes two methods of point cloud and image combination. One method uses low-level image features and the other uses high-level image features.

2.8.4 Crack 3D data detection and segmentation

Crack point clouds are used for crack detection and segmentation [59, 96–100]. However, the effectiveness of these methods depends on the quality of the point clouds. To enhance the point cloud quality and improve the accuracy of the crack detection and segmentation, an upsampling crack point cloud approach is necessary. The proposed method aims to upsample crack point clouds, which is also a significant contribution to improving crack detection and segmentation on crack point clouds.

The following chapters describe methods for cracks detection, segmentation, classification, and upsampling of crack point clouds that increase the accuracy of crack detection in point clouds. These contributions fill the research gaps that have been shown in the literature review.

Chapter 3

Crack Detection and Segmentation using a Two-stage Convolutional Neural Network Architecture

3.1 Introduction

The cracking index quantifies the damage caused by separation of parts of the road surface in the form of cracks. Road crack detection is the process of inspecting and identifying cracks on a road surface for road condition evaluation and maintenance. Road crack detection can be performed manually (relying on human vision and cognition) or automatically by machine vision. Human inspection requires an expert's knowledge, and is laborious and time consuming. Methods for automatic crack detection from road images have been developed to improve processing speed and obtain performance better than that of humans [15]. This is a challenging task in computer vision and image processing that has been the subject of research for decades [1, 13, 14, 28, 76]. This chapter will present an advanced machine learning approach for automatic road crack detection and segmentation for road condition evaluation.

A key aspect to the calculation of the cracking index is obtaining accurate estimates of the length and width of cracks, which is in turn enabled by precise pixel-level segmentation of

cracks. However the actual calculation of the cracking index has subjective components not easily automated and hence is ongoing work and is not within the scope of this thesis. This chapter focuses on accurate pixel-level segmentation of cracks using advanced machine learning methods to support the eventual automatic calculation of the cracking index. There is a potential for significant improvement in crack evaluation by moving from manual measurement of the crack dimensions and density to semi-automatic calculation of crack positions and structure using digital image processing and machine learning. Semi-automatic detection methods are more efficient for road surveys than human inspection [14, 15].

Some existing methods only focus on either detection at the region level or segmentation at the pixel level, and try to optimize only the detection or segmentation performance. However, by concentrating only on detection or segmentation alone, these methods do not see the problem holistically, which results in a reduction in overall performance. In particular, it is difficult to achieve a significant result in the case of challenging data such as noisy images with weak crack features, and imbalanced data. By developing a novel method, addressing both detection and segmentation in a single framework, the proposed solution addresses this gap in the prior art and shows substantially improved performance compared to previous approaches.

There are many proposed approaches to detect and segment cracks in road images. Traditional digital image processing (DIP) methods such as the Canny edge detector [32], and methods based on Gabor filters [40] exploit the change in intensity between pixels to define the edge, so a crack is considered as a feature which responds to edge detection filters. However, these algorithms are sensitive to many small details in road images and hence don't provide effective noise rejection. In addition, the optimal parameters required to allow these filters to trade-off the removal of noise against weak crack preservation change from image to image.

On the other hand, machine learning approaches, especially models based on neural networks, are used widely for object detection. Convolutional Neural Networks (CNN) are one of the most powerful recognition methods. Some work uses CNNs for detection of cracks [28, 29] while some use CNNs for pixel-wise segmentation of cracks in images [76].

In the training phase, images used for training are small sections of the overall image, wherein the positive inputs contain cracks while negative inputs do not contain cracks. After training, the model is used for crack detection. The training phase image sections used for detection systems are often inputs that are small sections of images, wherein the positive inputs contain cracks while negative inputs contain non-crack images.

The output of a detection model would be a binary decision as to whether a crack was or was not in the input image section. Nguyen et al. proposed a CNN model for crack detection Nguyen et al. [2]. The advantages of this CNN architecture is its ability to remove almost all noise and artifacts in the original image at a relatively large size of 750×1900 pixels, while detecting all image patches that contain cracks. However, a disadvantage of this model is that the detected cracks are not localised as precisely as in the ground truth. Another CNN model for crack detection from Zhang et al. Zhang et al. [28] achieves a high score but suffers from the problem that the detected cracks are larger than the ground truth cracks. Liu et al. in their paper Liu et al. [76] show that the handling of thin cracks of several pixels in width and broken, intermittent cracks differs from the handling of wider crack regions, so post-processing is necessary to improve the results.

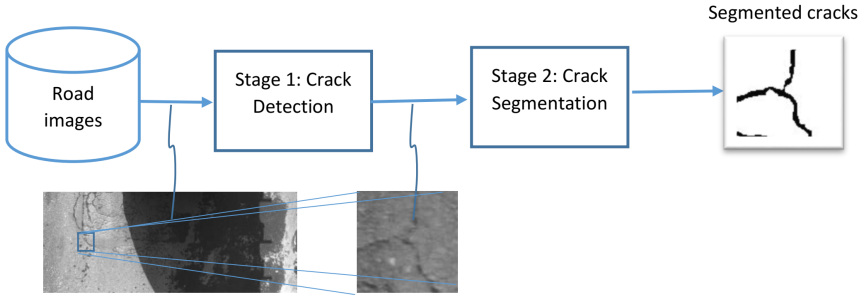


FIGURE 3.1: The proposed framework for road crack detection and segmentation. The inputs are original images with a big size of 750×1900. The output of the first stage is small samples containing cracks. The second stage segments the crack samples and produces binary images containing only cracks.

Figure 3.1 shows the proposed framework for road crack detection and segmentation. The image acquisition can be done by a camera attached to a car, a special purpose drone, or a smart-phone. Following this, samples for training and testing are created with the support of road experts. Two separate stages based on convolutional neural networks are trained from the samples, the first stage for detection and the second stage for segmentation of cracks. Previous work only focused on either detection or segmentation. In this chapter

the two steps are done in one framework based on a machine learning approach. The proposed method shows a strong performance improvement when applied to unbalanced datasets such as crack datasets, where the number of crack pixels is substantially smaller than the number of non-crack pixels. The main contributions of this chapter are:

- A new two-stage architecture based on CNNs that is effective for noisy, low-resolution images, and imbalanced datasets. The experiments show that the performance of this model is superior to models that use one stage for detection or segmentation exclusively.
- The model achieves superior performance compared to combining prior art detection and segmentation methods to create a two-stage approach, demonstrating that the specifically tailored two-stage model takes full advantage of a two-stage detection and segmentation paradigm.
- A new dataset of challenging road images containing cracks has been collected. The images are carefully labeled by experts for training and testing the proposed model. The dataset will be made available to the research community.

3.2 Two-stage Convolutional Neural Networks for Crack Detection and Segmentation

In the crack detection problem, in the training phase, each positive input sample contains a crack line that goes through the center of the sample [2, 28, 29, 54]. In the testing phase, some samples that contain cracks near their center may also be detected as positive samples. So the final detected cracks using regression contain the true cracks along with some of the surrounding pixels. Because of this, the size of training samples is very important. The smaller the input region samples are, the better the localisation of the detected crack is compared to the actual crack. However, for low-resolution images, the region sample must be large enough to distinguish clearly between positive samples and negative samples, especially for weak cracks and noise-like cracks.

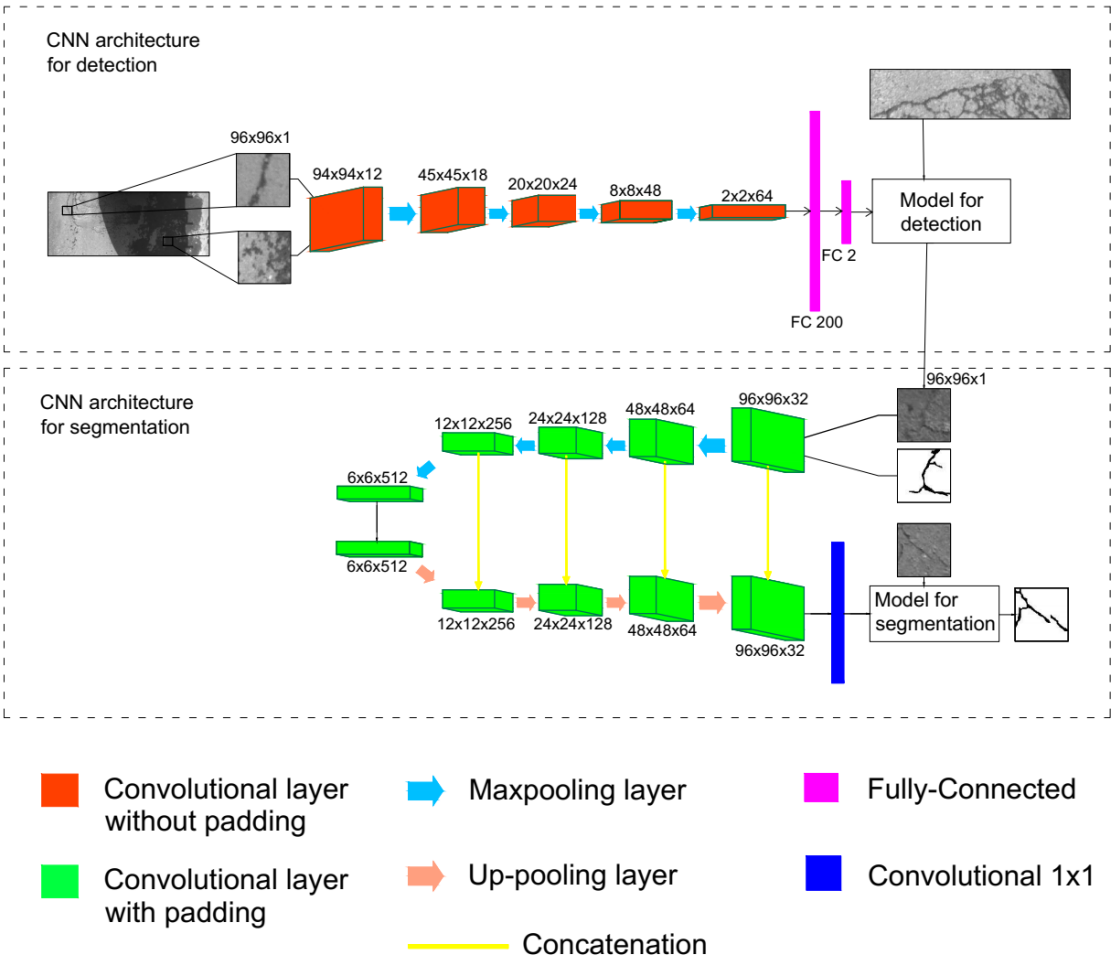


FIGURE 3.2: The proposed two-stage model for detection and segmentation. The first stage is used for detection, and the second stage is used for segmentation. The output of the first stage is the input of the second stage. The main layers of the architecture are convolutional layers, down and up pooling layer, fully connected layer, and activation function.

The following section will present a proposed model that combines two stages of detection and segmentation. The first stage uses a CNN as a detection model that is trained on image patches to seek all regions that contain cracks, while also removing most of the noise and artifacts. The second stage involves segmentation of the crack at the pixel level in small patches instead of the whole original image. As a result, the combined model has the advantages of both the detection and segmentation approaches.

Figure 3.2 shows the proposed two-stage architecture for crack detection and segmentation at the pixel level.

3.2.1 CNN architecture for crack detection

Each *convolutional layer* represents a feature map of the image such as shape, edge or intensity. For enhancing useful features and emphasising weak crack features, a number of neural network layers must be used. In this work, a five-layer CNN model to extract the features of the crack samples is used. The number of layers is chosen empirically. It is shown that for the particular crack detection problem being studied in this work, a five-layer CNN architecture achieves the highest score. The number of kernels in the first, second, third, fourth and fifth CNN layer are 12, 18, 24, 32, and 64, respectively. A one-pixel stride is applied for all CNN layers. This helps to compute all of the features, even when the cracks are small. The detection-stage CNN layers did not use padding, which means samples used by the CNN are not given additional padding when convolved with the kernel. This technique reduces the size of each layer and hence the number of parameters. After each CNN layer, a max-pooling layer with kernel size of 2×2 is added. A max pooling function selects the maximum value in each 2×2 block, also reducing the size of the layers of the network. In this way, the max pooling layer helps to decrease the number of weights and avoids over-fitting.

Fully connected (FC) layer. In CNNs, FC layers are given at the end of model for collecting and integrating all features that have been learned in earlier layers. The number of neurons in the final layer is equal to number of classes of objects. In this problem, images need to be classified into two classes, positive (crack) and negative (non-crack), so two neurons are used in the second FC layer.

Integrating features is achieved by two FC layers. While the previous CNN layers extract the crack features, the FC layers are used for collating all learned features. The first FC layer contains 200 neurons for flattening the features and arranging them into a vector. The feature vector collates all the feature information and high-weight elements from the prior convolutional layers. The second FC layer includes two neurons corresponding to the two classes to be classified: crack and non-crack.

Probability of samples being either a crack or non-crack region sample must be estimated and this is the final step to classify an image patch as crack or non-crack. In this work, the activation function is the *softmax function* in the last layer. The output of this function

is the probability that a given region sample contains a crack. The range of this value is from 0 to 1, representing the certainty of the region containing no-cracks or at least one crack, respectively.

Pooling layer. The models that contain pooling layers scan the input image faster than non-pooling models [101] and avoid over-fitting [102]. In this proposed method, a max-pooling layer is used after every convolutional layer. A max pooling uses a simple and fast algorithm. Adding a pooling layer after a convolutional layer decreases training time, reduce the number of parameters and controls overfitting [103].

Prevention of overfitting in this proposed model is achieved using a pooling technique. After each convolutional layer, a *max-pooling* is performed to decrease the number of weights. The image augmentation techniques are used such as shifting, zooming, flipping, and rotation of the image patches to diversify the training data, which also contributes to preventing overfitting.

Activation function. In convolutional layers and the first fully connected layer, the ReLU function [104] is used as the activation function. ReLU is widely used in CNN classification models and this function speeds up the training time [28, 102]. Softmax is utilized in the last layer, which is a fully connected layer with two neurons, this layer classifies the input image into crack (positive) or non-crack (negative) classes.

3.2.2 CNN architecture for crack segmentation

The proposed convolutional neural network architecture for segmentation (Figure 3.2) comprises two parts: a contraction or encoding part and an expansion or decoding part. Encoder-decoder models are used widely in image segmentation [73–75]. These models are also used for semantic segmentation of cracks in road image data.

The contracting part is designed to compute the features of objects in images. This part utilizes a model that contains five layers of convolutional neural networks for extracting image features. The contracting uses an architecture that increments the number of filters in each layer in a manner consistent with previous work on segmentation and object detection [2, 72, 73]. As the network is built for segmentation, this task aims to detect

and classify an object not only at the block level but also at the pixel level, so the number of kernels is not as small as the number of kernels in previous work [2]. In addition, in contrast with U-Net, which contains a very large number of filters in each CNN layer, the number of kernels is decreased in each successive layer by a factor of two, such that there are 32, 64, 128, 256, 512 kernels in sequence from the first layer to the fifth layer. The padding is used in all convolutional layers in the segmentation stage. This means that all samples are given additional padding when convolved with the kernel, so size of output segmented images is kept the same as the input images. A difference in the number of elements between the detection and segmentation stages is due to the use of padding in the segmentation stage that is not used in the detection stage. A max-pooling layer is added after each convolutional layer to decrease the number of weights and the sample size.

The expanding part is the inverse of the contraction process. While the contracting part is used for extracting features, the expanding part is used for spatially localizing patterns in images. A symmetric structure of decoder and encoder is presented in this architecture. The expanding part expands the input into a larger image as the data passes through each layer. So, the expanding part can work as a *deconvolutional network*, which acts in some sense as the inverse of the *convolutional network*. In addition, the *up-sampling* is applied with a size of 2×2 to restore the size to that of the input samples. After the fifth layer, the output image is the same size as the input.

Convolution 1×1 can be used to increase or decrease the number of channels while maintaining the size of the image. In this architecture, a convolution layer with a 1×1 kernel that acts as a *sigmoidal activation function* is used. This layer acts to process the feature maps to generate a segmentation map and thus categorize every pixel of the input image.

The concatenation layer is used for concatenating two layers together along one axis. The aim of concatenating layers is to enhance the shuffling of information across many layers of the network. In this model, there are four concatenation layers are used, each concatenation layer concatenates a pair of convolutional and deconvolutional layers into a single layer that contains both feature maps of the two input layers.

3.3 Crack Segmentation Results and Evaluation

3.3.1 Crack datasets preparation

For the purpose of comparing and assessing the proposed method, the following benchmark and self-created datasets are used.

DeepCrack dataset. This dataset contains 537 images for training and testing each with a size of 544×384 pixels [76]. Ground truth images at the pixel level are available. This dataset is used in the detection and segmentation phases for assessing the proposed method.

CrackIT dataset. The CrackIT dataset [15] was collected in Portugal and Canada. This dataset has no ground truth at pixel level, so this dataset is initially used in the detection phase. Some previous detection methods are implemented as well as the proposed detection method to this dataset and the 2StagesCrack dataset and compare the results of these methods. Ground truth at the pixel level for this dataset is created in order to further evaluate the performance of the proposed approach. A tool from the Matlab commercial software package [105] is used to allow experts to manually create the ground truth in the same manner as ground truth for the 2StagesCrack dataset as described below.

2StagesCrack dataset. This dataset was collected in Vietnam and the images were taken from a monochrome camera attached to a trailer, driven across a variety of roads [2]. The images were collected under a variety of weather conditions, and in many cases the road was not clean, so the dataset contains noise. Moreover, the monochrome camera used is low quality and attached to a high speed car, so the captured images are low resolution. In this work, ground truth crack samples are created by annotating the images at the pixel level and the resultant ground truth augmented dataset is denoted as the **2StagesCrack** dataset. The ground truth annotation for the dataset is supervised by road experts. My contribution to the data collection process was pre-processing data such as cropping the original image to the samples used for the training and testing tasks. I also contributed by creating the ground truth for this dataset.

These experiments found that, in the appropriate space, the larger size of samples, the higher the detection accuracy. However, the smaller size of the sample led to a more exact regression of the crack line. In addition, the size of the sample should be sufficient to allow the system to learn to distinguish between negative and positive patches. Because of all the above reasons and based on the characteristics of the different data sets used, it was observed that a sample size of 96×96 was the best size for crack detection.

The *2StagesCrack* dataset is used in the experiments for the segmentation phase, and contains a total of 4000 samples, including 2000 original images of size 96×96 pixels and 2000 associated ground truth images. The "Image Segmenter" tool in the "Image Processing and Computer Vision" toolbox in the Matlab software [105] is used to create binary images for training. Each original image is processed to create ground truth with the "Draw Freehand" function in the Matlab Segmenter tool to outline the boundary of the cracks. The image is then binarised such that the crack pixels appear in black and the background pixels are in white.

Figure 3.3 shows examples of samples belonging to the *2StagesCrack* dataset. The first row shows negative samples, that contain no cracks and are used in the detection stage. The second row shows the positives samples, that contain some cracks and are used in the detection stage and segmentation stage as original images. The third row shows the associated ground truth images of the samples in the second row. In the two last rows, the first and second images show connected cracks and cracks with strong intensity. The first and the second images contain connected cracks. In addition, the cracks in the second row are thin in some regions. The third image shows a large crack suffering a degree of blur which may be due to the movement of the sensor vehicle when the image was captured. The fourth example is a discontinuous, thin and unclear crack, possibly affected by dirt on the road. The last example is a sample of a single, weak crack.

Figure 3.4 shows examples of samples belonging to the CrackIT dataset. The first row shows negative samples, the second row shows positive samples, the third row shows the original ground truth [15], and the last row shows the ground truth created for the experiments in this chapter. The original ground truth contained only a one-pixel representation

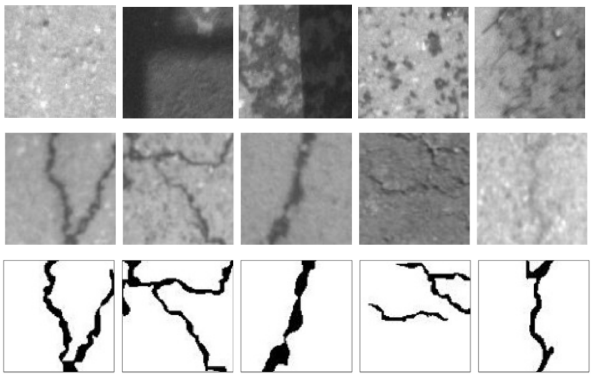


FIGURE 3.3: Examples of samples and ground truth for the 2StagesCrack dataset. The first row are negative samples containing many kinds of noise and artifacts. The second row are positive samples containing diverse cracks, from weak to strong cracks. The third row is the ground truth of the second row samples.

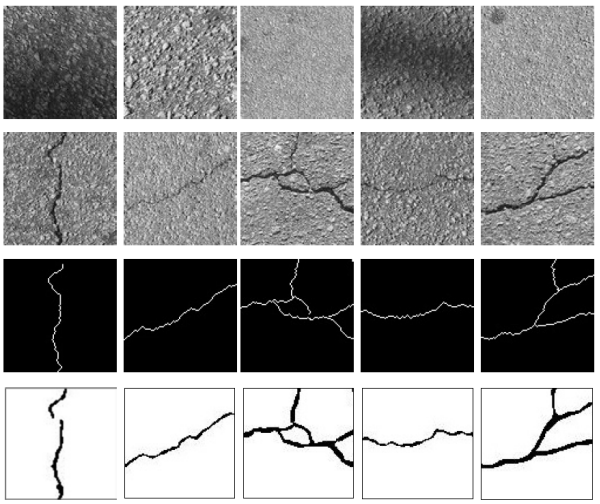


FIGURE 3.4: Examples of images and ground truth for the CrackIT dataset. The two first rows are negative and positive samples, respectively. The two last rows are ground truth indicating the center of each crack with a width of 1 pixel (the third row) and the entire width of the crack (the last row).

of the centerline of the crack. The ground truth images are enhanced to cover all pixels belonging to the crack in a manner that can be used for pixel-level segmentation.

Table 3.1 shows a comparison of the key features of the three datasets used in this work: CrackIT, Deepcrack, and 2StagesCrack. Each of these crack datasets are imbalanced. The proportion of pixels representing cracks in the whole dataset is less than 1% in the CrackIT dataset [15], 3.54% in the DeepCrack dataset [76], and around 0.3% in the 2StagesCrack dataset. With the proposed strategy of detection at the sample level and segmentation at the pixel level, the imbalance present in the original images is mitigated, with the

Table 3.1: Comparison of datasets.

CharacteristicsDatasets	**CrackIT**	**Deepcrack**	**2StagesCrack**
Size of image (pixels)	600 ×800	544 ×384	750 ×1900
Spatial resolution	1 pixel 1 square mm	No information	4 pixels area 1 square mm
Noise and artifacts	Shadow, paint	No information	A lot of shadows from trees and cars; paint, water streaks, wheel streaks, waste material on road
Percentages of crack pixels	Less than 1%	3.54 %	0.31%
Percentages of crack pixels in segmentation data	8.3%	16 %	6%
Other characteristics	Road and sidewalk images	Various types of image such as roads, building	Road images
	Strong intensity cracks	Strong intensity cracks	Various level of cracks, from weak to strong intensity; including cracks under shadows
	Thin crack	Large crack	From thin to large cracks
	Collected in Portugal	No information	Collected in Vietnam

proportion of pixels representing cracks in the segmentation stage becoming 8.3%, 16% and 6% in the CrackIT, Deepcrack, and 2StagesCrack dataset, respectively. Table 3.1 also illustrates that the 2StagesCrack dataset is a large scale dataset, at low resolution and contains a variety of noise and artifacts.

To increase the number of samples and diversify the images, data augmentation techniques are applied to the images before the training phase such as rotations (rotation range is 5 degree), shifts (shift range is 0.2), zoom (zoom range is 0.2), and flipping to create additional data samples.

All experiments in this chapter are implemented on a PC with a single CPU Intel core i5 processor and processor speed of 2.81 GHz, 8GB of RAM, running the Windows 10 operating system.

3.3.2 Methods for evaluation the crack detection and segmentation

The commonly used metrics for evaluation of detection and segmentation methods are Precision *Pr*, Recall *Re* and F1-score *F1*. These are defined as below:

$$Pr = \frac{TP}{TP + FP} \tag{3.1}$$

$$Re = \frac{TP}{TP + FN} \tag{3.2}$$

$$F_1 = \frac{2 \cdot Pr \cdot Re}{Pr + Re} \tag{3.3}$$

where *TP*, *FP*, *FN* are the numbers of True Positive, False Positive, and False Negative samples, respectively. In the detection stage, a positive sample is a region of the image that contains at least one crack, and a negative sample is a region of the image that contains no cracks. In the segmentation phase, a positive sample is a crack pixel, and a negative sample is a background pixel.

In testing, with N patches (N×96×96 pixels) images in total from which n patches (n×96×96 pixels) among the N are detected as crack areas, it follows that (N-n) are non-crack areas. The second stage segments the crack pixels from the results of the detection phase, so the F_1 for the (N-n) non-crack samples is $F_{1\text{-Det}}$, and the F_1 of the n crack patches is $F_{1\text{-Det}}$*$F_{1\text{-Seg}}$. For the experiments in this chapter, a stride of one pixel in the detection stage is used, so the number of detected patches is equal to the number of input pixels to the segmentation stage. Since the degrees of freedom of $F_{1\text{-Det}}$ and $F_{1\text{-Seg}}$ are the same, they can be combined to create a final F_1-score. The final F_1-score for crack segmentation at the pixel level in the whole image can be calculated as per equation (3.4):

$$F_{1\text{-final}} = F_{1\text{-Det}} * \frac{F_{1\text{-Seg}} * n + (N - n)}{N} \tag{3.4}$$

where $F_{1\text{-Det}}$ is the F_1 score of the first stage of detection and $F_{1\text{-Seg}}$ is the F_1 score of the second stage of segmentation.

Road crack datasets are imbalanced data, where the number of true crack pixels is very small compared with the number of true background pixels. F_1 score compensates for this effect and is hence a reasonable method for evaluating the performance of the proposed method.

3.3.3 Crack detection and segmentation results from the two-stage model

Table 3.2 shows the *Precision* and *Recall* results of various models for the detection stage. A SVM (Support Vector Machine) is implemented with a linear-kernel using HoG (Histogram of Oriented Gradients) features for crack detection. The results of the proposed model with some other traditional machine learning methods and some existing models are compared. The proposed architecture used a smaller number of parameters than prior convolutional neural networks as shown in the last column of Table 3.2. The results show that the proposed model achieves the highest F_1 score, at 92%. It also uses a significantly lower number of parameters than the other neural network based methods.

TABLE 3.2: Precision and Recall of different models in the Detection stage

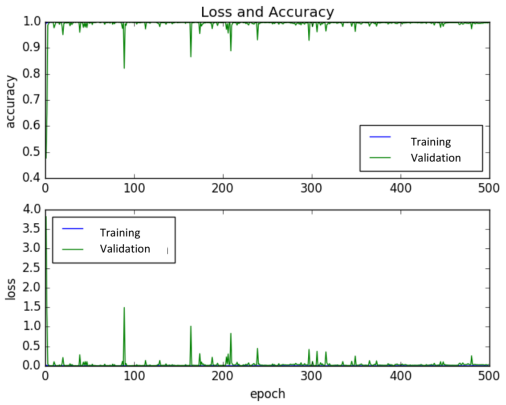
Method	CrackIT dataset			DeepCrack dataset			2StagesCrack dataset			Total
	Pr	Re	F1-score	Pr	Re	F1-score	Pr	Re	F1-score	parameters
Support Vector Machine	0.89	0.57	0.70	0.87	0.59	0.70	0.90	0.55	0.68	N/A
Decision Tree	0.82	0.59	0.68	0.80	0.65	0.72	**0.94**	0.51	0.66	N/A
Random Forest	**0.99**	0.54	0.69	**0.99**	0.50	0.66	0.98	0.52	0.68	N/A
Fan's method [29]	0.64	0.70	0.67	0.70	0.75	0.72	0.99	0.57	0.72	924,562
Zhang's method [28]	0.74	0.97	0.83	0.83	0.86	0.84	0.85	0.82	0.77	205,466
Proposed method	0.92	**0.89**	**0.90**	0.93	**0.90**	**0.91**	0.92	**0.91**	**0.92**	**58,404**

Table 3.3 shows the results of the proposed segmentation stage and four other models on the two datasets that have pixel-level ground truth available. Firstly, two classical DIP methods are used on the images: Gabor filter and Adaptive Thresholding. K-Nearest Neighbour (K-NN), one of the most fundamental and straightforward methods [106], is used in the experiments for crack segmentation and compared with the proposed method. After that, two CNN models for segmentation, U-Net and SegNet were implemented. The results indicate that the proposed method has significantly improved segmentation performance in both datasets, as measured by the F_1 score. A comparison of the number of parameters between the three models based on CNN is shown. It is clear that the proposed model for segmentation needs a much smaller number of parameters, while still achieving better performance.

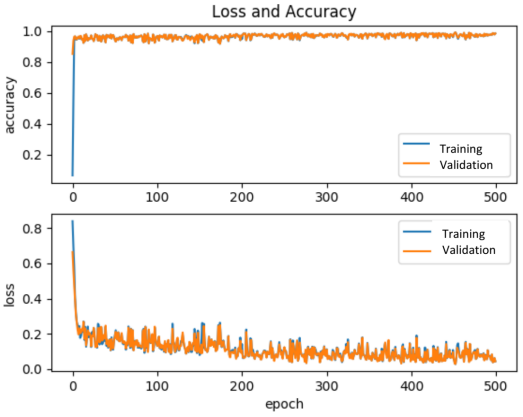
Figure 3.5 shows the accuracy and loss of the proposed architecture in training for detection and segmentation for the 2StagesCrack dataset. Figure 3.5a and Figure 3.5b show the training, validation accuracy and loss of the detection model and segmentation model, respectively. These graphs indicate that the proposed models have high accuracy and low

TABLE 3.3: Precision and Recall of different models in the Segmentation stage

Method	CrackIT dataset			DeepCrack dataset			2StagesCrack dataset			Total parameters
	Pr	Re	F1-score	Pr	Re	F1-score	Pr	Re	F1-score	
Gabor Filter	0.54	0.64	0.59	0.55	0.85	0.67	0.61	0.23	0.34	N/A
Adaptive Thresholding	0.17	0.82	0.28	0.25	0.90	0.40	0.53	0.31	0.38	N/A
K- Nearest Neighbour	0.52	0.48	0.49	0.66	0.50	0.57	0.69	0.28	0.40	N/A
SegNet	0.62	0.62	0.62	0.67	0.67	0.67	0.55	0.55	0.55	29,475,866
U-Net	0.71	0.84	0.77	0.87	0.73	0.79	0.49	**0.88**	0.63	31,031,685
Proposed Method	**0.88**	**0.85**	**0.87**	**0.93**	**0.94**	**0.93**	**0.70**	0.78	**0.74**	**7,672,549**



(A) Accuracy and loss during training for the detection stage for the 2StagesCrack dataset.



(B) Accuracy and loss during training for the segmentation stage for the 2StagesCrack dataset.

FIGURE 3.5: Accuracy and loss during training for the detection and segmentation stages for the 2StagesCrack dataset.

loss in both the detection and segmentation phases. In addition, the convergence of the selected model was rapid. This is consistent with the observations of Zhang et al. [28], where less than 20 epochs where shown to be sufficient for convergence. A total of 500 epochs are shown here to demonstrate the stability of the model.

To calculate the final F_1 score as equation 3.4, the $F_{1\text{-Det}}$ and $F_{1\text{-Seg}}$ of the detection phase and the segmentation phase of the different methods are shown in Table 3.2 and Table 3.3. If an original image is of size 750×1900 pixels, there are 155 non-overlapping region samples of size 96×96 pixels, so $N = 155$. Assuming that the number of regions with cracks detected from the detection stage is 10, then $n = 10$. Table 3.4 shows the final F1-score of the various models when combining different detection methods with different segmentation methods when applied to the 2StagesCrack dataset. It is shown that:

- If the segmentation stage is only applied to the regions flagged as containing cracks

TABLE 3.4: F1-score for different combinations of detection and segmentation for the 2StagesCrack dataset.

Segmentation methodsDetection methods	Fan's method	Zhang's method	Proposed Detection stage
SegNet method	0.69	0.74	0.88
U-Net method	0.69	0.75	0.89
Proposed Segmentation stage	0.71	0.76	**0.91**

TABLE 3.5: The MCC score for the examined methods at the Segmentation stage for the three datasets.

Method	CrackIT dataset	DeepCrack dataset	2StagesCrack dataset
Gabor Filter	0.57	0.69	0.25
Adaptive Thresholding	0.60	0.63	0.31
K- Nearest Neighbour	0.44	0.47	0.33
SegNet	0.59	0.61	0.52
U-Net	0.77	0.75	0.72
Proposed Method	**0.85**	**0.92**	**0.74**

by the detection step, the final performance is improved compared to the use of a single stage of segmentation applied to the entire image. While the segmentation model only achieves around 70% when applied to the entire image, the two-stage model achieves about 90% in terms of F_1 score.

- The proposed architecture achieves superior performance in both the detection and segmentation steps alone compared to prior detection or segmentation methods. Hence, the proposed two-stage approach achieves superior performance compared to the other detection/segmentation combinations.

It can be seen that the $F_{1\text{-}2stages}$ score increases as n decreases. So, for the 2StagesCrack dataset, which contains a lower density of cracks and a higher level of noise, this method achieves a high F_1 score in total.

The Matthews Correlation Coefficient (MCC) is another metric for measuring the quality of binary classification [107]. The MCC score provides a less biased evaluation compared to F1-score when the dataset is negatively imbalanced [108]. All datasets used in this chapter are positively imbalanced datasets. However, for further comparison, the MCC score is also calculated in the segmentation stage for the three datasets, as is shown in Table 3.5. The proposed method achieves the highest MCC score for all given datasets.

Table 3.6 shows the total parameters of various models used for detection and segmentation, and testing time for the 2StagesCrack dataset. The testing time is the average time used for processing one sample in milliseconds. It can be seen that the proposed architecture has a smaller number of total parameters and a shorter testing time in comparison with that of the combination of Zhang's method and the U-Net model. In all experiments, the hardware configuration for the testing phase are the same. In addition to the number of total model parameters, there are various factors that affect the testing time, including the software library used (in these experiments the *numba* library is used instead of *numpy*), the hardware platform (CPU or GPU, these experiments used the CPU) and the pixel stride chosen during testing. After investigation, it is found that a sizeable percentage of the testing time was taken by the time for software and hardware to extract samples from the images for testing. Hence testing time scales with the number of parameters only to a certain extent.

TABLE 3.6: Total parameters and testing time for the 2StagesCrack dataset.

Model		Total parameters	Average testing time per sample
Detection	Zhang's model for detection	205,466	19.2ms
	Proposed stage for detection	58,404	**13.5ms**
Segmentation	U-Net model for segmentation	31,031,685	153ms
	Proposed stage for segmentation	7,672,549	**103ms**
Two stages model	Two stages of Zhang and U-Net	31,052,151	172.2ms
	Proposed model	**7,730,953**	**116.5ms**

Figures 3.6,3.7,3.8,3.9, 3.10 and 3.11 show some typical examples of detection and segmentation results produced by the different methods. The first five original images are from the 2StagesCrack dataset and have a size of 750×1900 pixels, the remaining images are from the CrackIT dataset and have a size of 600×800 pixels. Each of these figures contains 6 sub-figures: the original image, ground truth image, the detection result obtained by the method of Zhang et al. [28], the result obtained from combining the detection method from Zhang et al. [28] and the segmentation method from Ronneberger et al. [73], the detection result from the first stage of the proposed model, and the final image is the result by the proposed two-stage architecture. For post-processing, the Otsu method Otsu [48] and some other techniques are applied to obtain the binary images from the segmented results such as detects cracks at different scales [79]. The results from Otsu or a median

filter may not be stable across the different inputs encountered. The proposed method applies an automatic threshold to classify pixels as cracks when their probability exceeds the F1-score of the positive crack segmentation and exports the results as binary images. The experiments in this chapter show that a threshold that is equal to the F1-score achieves the best output images.

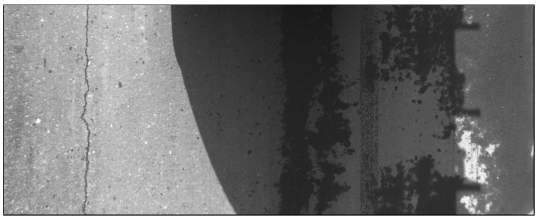
Figure 3.6 shows a typical example. The original image, Figure 3.6a, contains a single crack. There are many artifacts in this image, such as a shadow, and wheel tracks caused by water. While the first stage of the proposed method detects the true crack with the surrounding area, the method from Zhang et al. [28] also identifies one side of the shadow area as a crack. The proposed method is more robust compared to prior methods and less likely to misidentify shadows and artifacts on the road surface as cracks. Figure 3.7 shows a similar example with a single crack and once again, we can observe that the proposed method works very well.

Figure 3.8a is an example of connected cracks. The method from Zhang cannot detect all regions containing cracks, and detects a number of false negative regions. For the detected crack in Figure 3.8c, the U-Net model also has poor segmentation in some parts that results in false negative samples.

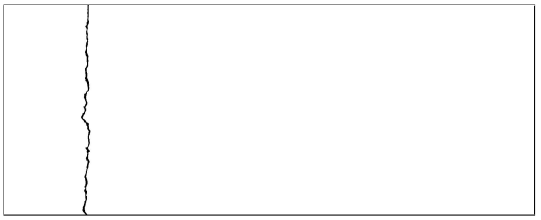
Figure 3.9 shows other experimental results on connected cracks. The image in figure 3.9a is a challenge because it was captured in weak light and the road surface is covered with water in many places. The first stage of the proposed model detects all patches that contain cracks but also detects some background pixels that surround the true crack pixels. Following this, the second stage removes the remaining non-crack pixels. Compared to the proposed method, the two other methods ignore many true positive cracks.

Figure 3.10 is a special instance because the cracks are thin and in the shadows. These cracks were captured under poor illumination conditions because of the shadows, and this image contains white noise around the cracks. The proposed method detects and segments most of cracks, while the other methods recognize only a small proportion of the cracks.

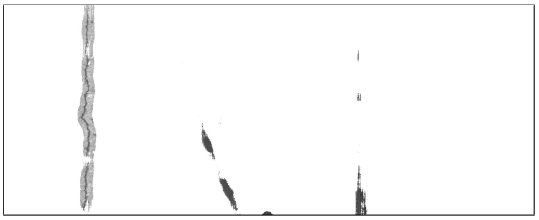
Figure 3.11 shows an example from the CrackIT dataset with thin cracks with a width of only one or two pixels. The results of Zhang's method shows a number of false positives,



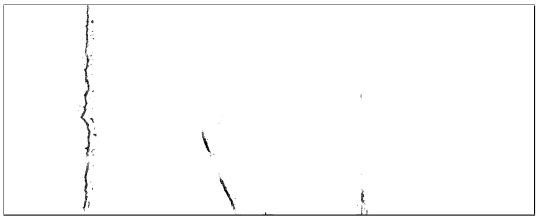
(A) Original image.



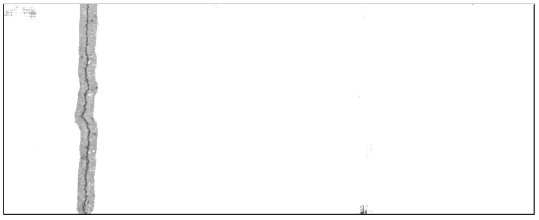
(B) Ground truth.



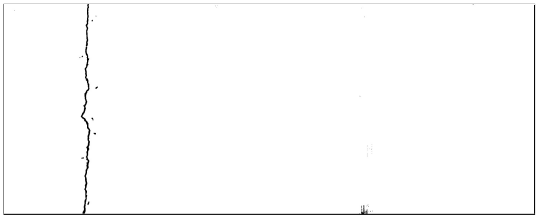
(C) Detection result from the method of Zhang et al. [28]



(D) Segmentation result from the combination of Zhang et al. [28] and U-Net [73].



(E) Detection result from the first stage of the proposed method.



(F) Final segmentation result from the two-stage model.

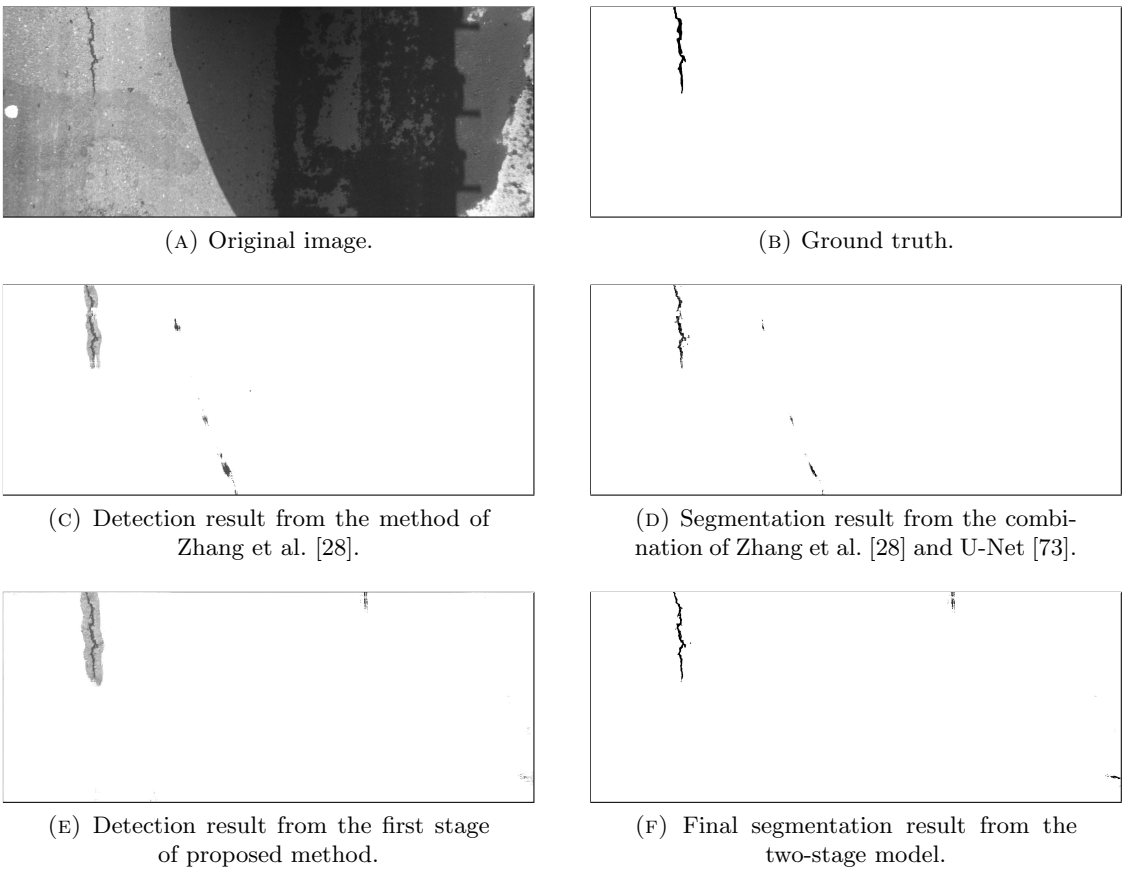
FIGURE 3.6: Experiment on an image with a long, single crack and a large shadow.

and the U-Net model segments many noise pixels as cracks. The proposed method detects and segments the thin cracks with few false positives.

3.4 Chapter Conclusions

This chapter proposed a novel two-stage model based on CNN for segmentation at the pixel level of cracks in road images. This method integrates state of the art detection and segmentation into a single unified framework that outperforms existing approaches while significantly reducing the effective imbalance in the data and achieves this with reduced computational requirements.

Experimental results show that the proposed method achieves a higher accuracy than either detection or segmentation alone. The experiments show that the two-stage model



(A) Original image.

(B) Ground truth.

(C) Detection result from the method of Zhang et al. [28].

(D) Segmentation result from the combination of Zhang et al. [28] and U-Net [73].

(E) Detection result from the first stage of proposed method.

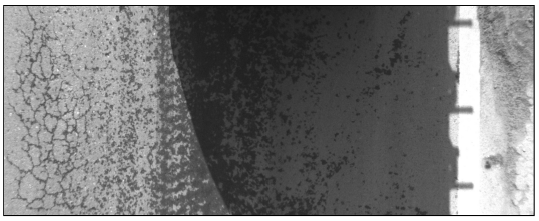
(F) Final segmentation result from the two-stage model.

FIGURE 3.7: Experiment on an image with a short, single crack and a large shadow.

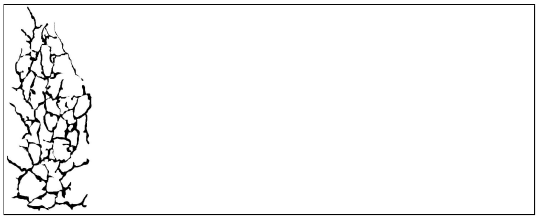
works well for noisy, low resolution road images, and imbalanced datasets with artifacts with an F_1 score of more than 90% compared to a result of under 80% from other state of the art methods. This architecture may also be considered for other object detection problems that use low quality input data.

The following link corresponds to the 2StagesCrack dataset:

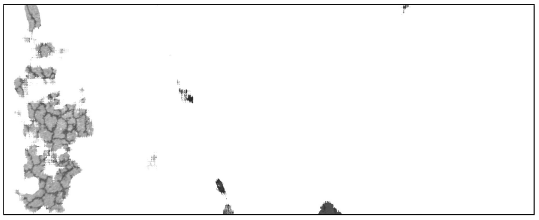
https://drive.google.com/drive/folders/1Pz0ogI9nWQdF3fvTZ7fUppjVXrVKc4HW?usp=sharing



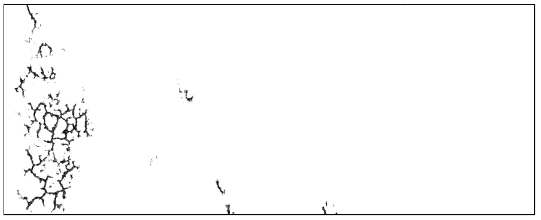
(A) Original image.



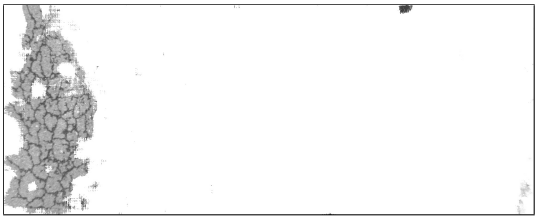
(B) Ground truth.



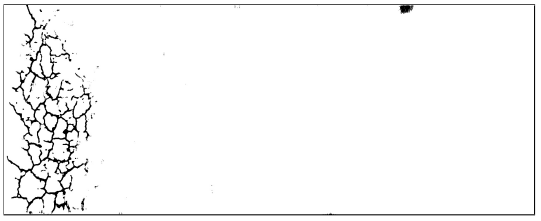
(C) Detection result from the method of Zhang et al. [28].



(D) Segmentation result from the combination of Zhang et al. [28] and U-Net [73].

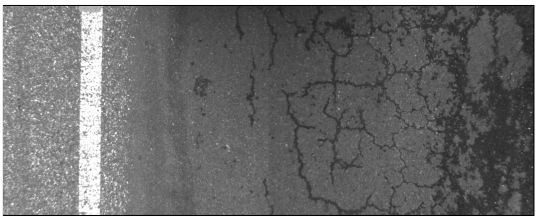


(E) Detection result from the first stage of proposed method.

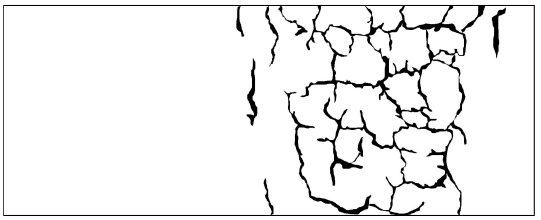


(F) Final segmentation result from the two-stage model.

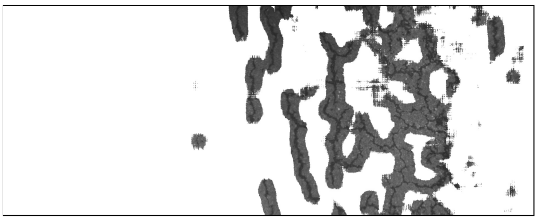
FIGURE 3.8: Experiment on an image with a connected crack on a wet surface with dotty noise.



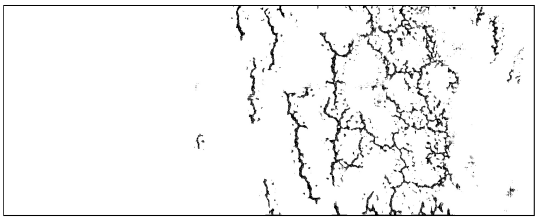
(A) Original image.



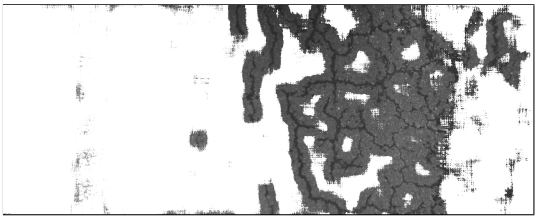
(B) Ground truth.



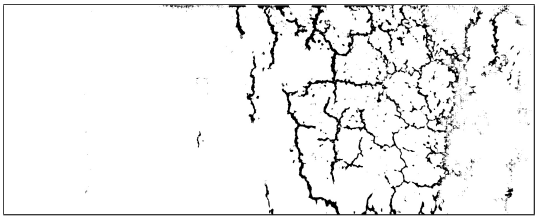
(C) Detection result from the method of Zhang et al. [28].



(D) Segmentation result from the combination of Zhang et al. [28] and U-Net [73].



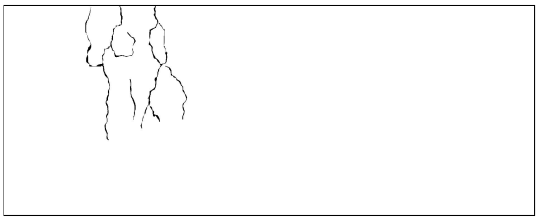
(E) Detection result from the first stage of proposed method.



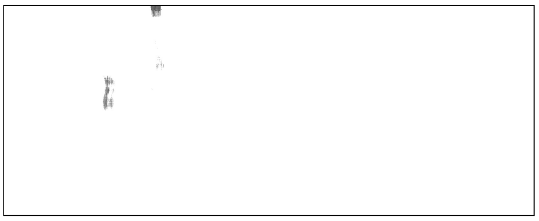
(F) Final segmentation result from the two-stage model.

FIGURE 3.9: Experiment on connected, wet cracks captured under weak light conditions.

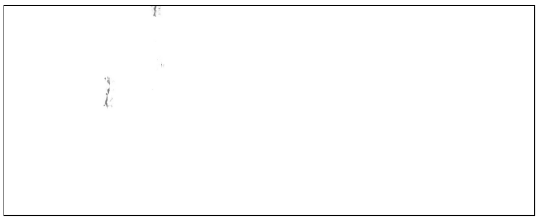
(A) Original image.



(B) Ground truth.



(C) Detection result from the method of Zhang et al. [28].



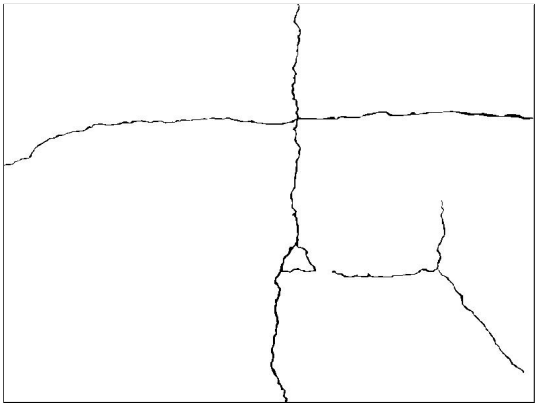
(D) Segmentation result from the combination of Zhang et al. [28] and U-Net [73].

(E) Detection result from the first stage of proposed method.

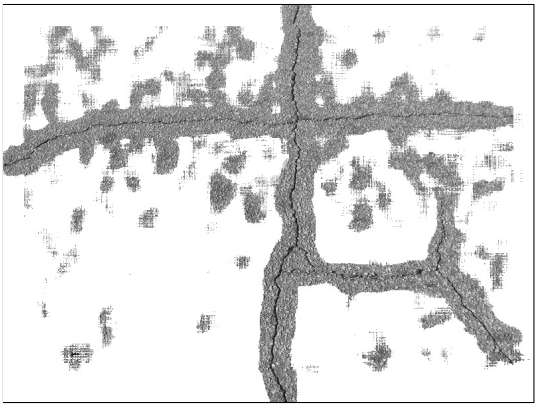
(F) Final segmentation result from the two-stage model.

FIGURE 3.10: Experiment on a crack under a shadow.

(A) Original image.

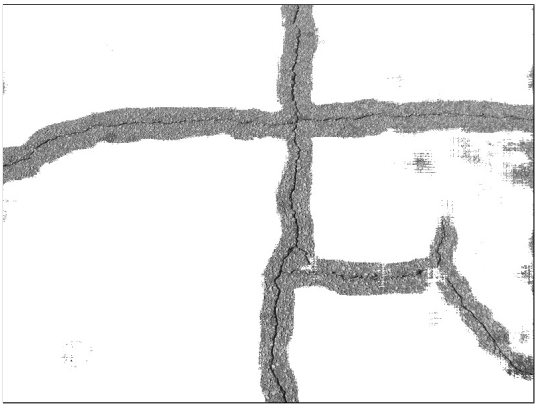


(B) Ground truth.



(C) Detection result from the method of Zhang et al. [28].

(D) Segmentation result from the combination of Zhang et al. [28] and U-Net [73].



(E) Detection result from the first stage of proposed method.

(F) Final segmentation result from the two-stage model.

FIGURE 3.11: Experiment on thin cracks in the CrackIT dataset.

Chapter 4

Merging 2D and 3D Data for Crack Detection

4.1 Introduction

Crack detection and segmentation are essential for maintaining civil infrastructure, such as roads, bridges, and buildings. The application of digital imaging methods to this problem has resulted in significant advances. However, 2D crack images do not contain the rich information from three dimensional (3D) data such as point cloud data which can be highly useful when examining complex structure or thin cracks. Using 3D data has the potential to significantly improve crack detection and segmentation [109]. One of the difficulties when processing point clouds is that the low density of points limits the achievable resolution of features required for good detection. Upsampling of the point cloud data offers the possibility of addressing this problem, but current upsampling methods struggle to achieve the required fidelity.

Point cloud upsampling is a topic that challenges researchers in computer vision, and has attracted increasing interest in recent years [83–85, 110]. In these papers, the authors focus on upsampling low-resolution point clouds by learning features from similar point clouds. The results from the above work are effective for upsampling many kinds of point

cloud objects. However, there are some limitations that still need to be addressed, such as upsampling of very sparse point clouds, and filling voids in point clouds.

In this chapter, the information from low-resolution point clouds is combined with features derived from high-resolution 2D images of the same scene to produce an upsampled version of the low-resolution point clouds. The features from the high-resolution images are combined with point clouds to augment the point cloud information for the upsampling process. Several different approaches to the upsampling were explored, such as combination at the point-pixel level, or using transfer learning for combination at the feature level. Apart from improvements in the fidelity of the upsampled point cloud, the method also seeks to increase the uniformity of the sampling and fill voids in the point cloud data.

Crack point clouds may be considered as 2.5D point clouds as they take the form of almost planar surfaces with a height map associated with each pixel representing principally the cracks. 2.5D point clouds such as this lack complex structure such as closed surfaces. Similar 2.5D point clouds are important in a range of different applications, from the construction industry [111, 112] to health monitoring. However, high-density point clouds or 3D data are expensive and sometimes impractical to collect, so collecting low-density point clouds and then upsampling them offers advantages in practice.

There are two main types of 3D scanning technologies used for crack analysis. The first type is light amplification by stimulated emission of radiation (laser) triangulation 3D scanning technology and the second type is structured light 3D scanning technology. Crack 3D data can be collected using terrestrial laser scanning [113, 114] or a mobile laser scanning [115]. The laser technique is limited in resolution, so 3D data from laser data sources is used to detect surface distresses more than 1 cm wide [113]. Therefore, cracks that are smaller than 1 cm should be detected on upsampled data point clouds. The ground truth 3D data used in this paper are collected by a scanner using the structured light 3D scanning technology that has accuracy at 0.1 mm. The accuracy of 0.1mm refers to the minimum distance between two points on the object surface can be captured by the scanner. Given this accuracy, our point cloud dataset can represent a tiny crack with depth or width of 0.1mm or above. The structured light 3D scanner has an advantage in resolution, but setting up these systems for capture can sometimes be impractical in

real world applications. Therefore, collecting high-resolution point clouds for training and using upsampled point clouds for real world applications has beneficial applications in real world scenarios.

In this chapter, "transfer learning" is used to refer to the extraction of knowledge from 2D crack images (feature extraction) and then the combination with additional point cloud data. Torrey et al. indicated that "The goal of transfer learning is to improve learning in the target task by leveraging knowledge from the source task." [116]. In a survey of transfer learning [117] techniques, transfer learning is defined as follows "Given a source domain D_S with a corresponding source task T_S and a target domain D_T with a corresponding task T_T, transfer learning is the process of improving the target predictive function $f_T()$ by using the related information from D_S and T_S, where $D_S \neq D_T$ or $T_S \neq T_T$". In 2D and 3D crack processing, D_S is the domain of 2D features and T_S is the source task of crack segmentation, D_T is the domain of 3D features and T_T is the task of crack point cloud upsampling. In this thesis, the term "transfer learning" is used to indicate that knowledge from the 2D crack images domain is referred to the 3D crack point domain. The transferred knowledge is then used to improve crack features and the crack features are input to the generator process.

Figure 4.1 shows the proposal to apply transfer learning between point clouds and images in a type of Generative Adversarial Network (GAN) architecture with two parts; the *generative model* and *discriminative model* [118] for crack point cloud upsampling. The point cloud samples and image samples are created and matched one by one. Image features that are extracted from images by a crack detection model are combined with point cloud features before becoming input data for a GAN architecture. The output is a high-resolution uniformly sampled point cloud.

The main contributions of this chapter are:

- A novel approach is proposed based on GAN and transfer learning that is effective for upsampling sparse point clouds.

- The experiments show that 2D images can augment low-resolution point clouds, and the performance of this architecture is superior to architectures that use point clouds only.
- The proposed model achieves a superior performance compared to prior art point cloud upsampling approaches, demonstrating that the specifically tailored combination of 2D images and point clouds takes full advantage of features from both images and point clouds.
- A new dataset of concrete crack point clouds and their corresponding images is presented and that will be made available to the research community.

The proposed method aims to create high-density point clouds from sparse point clouds and their corresponding images. It requires the output point clouds to contain a high number of points, and have a uniform distribution of points in order to support the subsequent crack analysis. To do that, an architecture is proposed based on GAN and uses a combination of point clouds and images as input data.

The proposed method focuses on two main parts. The first part is the proposed method for combining images and point clouds. This chapter presents in detail how images and point clouds are aligned and combined to become input data to a GAN model. The second part is the proposed GAN model which uses the combined data from images and point clouds to upsample the low-resolution point clouds. The three major components of this model are the generative model, discriminative model, and the loss functions. Each of these components are described in this section. The generative model takes input from one of two feature combination methods; point-pixel and feature-feature combination and generates an upsampled version of the point cloud. The discriminative model ensures the fidelity of the upsampled point cloud by comparing the characteristics of the upsampled point cloud to that of the ground truth. Both the generative and discriminative models work together through the formulation of an appropriate loss function as will be described below.

4.2 Combining 2D Images and Point Clouds

In this chapter, two ways to combine image data and point cloud data for crack point cloud upsampling are proposed. The first approach is called "point-pixel combination", and the second is called "feature-feature combination".

Point-pixel combination. Point clouds often have no regular structure, however images have a regular order, and if their matched point clouds belong to a 2.5D surface such that the implicit surface is single-valued, then there is an alignment of the point cloud and the image such that each point in the point cloud can be matched to a pixel in the image unambiguously. An image has no explicit information about the depth of an object, but the image pixel's intensity value in the image often contains implicit depth information in the case of crack images with darker areas in a crack often corresponding to the deepest parts of the imaged crack. For this reason, it is expected that the information from an image can be combined with point cloud information as an additional channel. The additional channel can be built from the image grayscale value or can be created from other features derived from the image.

For *Point-pixel combination*, a four-channel point cloud from each low-resolution point cloud input and its corresponding image is created. From each point $(x_k^{pc}, y_k^{pc}, z_k^{pc})$ in the input point cloud sample, the first step is finding the matched pixel (x_k^{im}, y_k^{im}) (where the grayscale value is $img[x_k^{im}, y_k^{im}]$) in the corresponding image. The new point contains four channels, $(x_k^{pc}, y_k^{pc}, z_k^{pc}, img[x_k^{im}, y_k^{im}])$.

To enhance this method, some additional channels using other image features are also implemented. The first kind of feature is the Sobel gradient feature information. Sobel features are produced by applying a Sobel operator [119] to the image. Gamma features extracted from the image using Gamma filters [21, 120] are also used in these experiments with $\gamma = 0.2$. Gamma features are strong features utilized in image enhancement. The last kind of image feature used are Difference of Gaussians (DoG) features [121, 122]. DoG features are used in various methods of image processing such as edge detection and image matching.

Feature-feature combination. The feature-feature combination is implemented by summarising knowledge from the image domain, and then combining it with knowledge from the point cloud domain by a layer concatenation operation. There are two different models used to extract features from point clouds and from images. Point cloud features are extracted from the same model that is used for *Point-pixel combination.* Image features are extracted from a crack detection model that was pre-trained in a previous model for crack detection [60]. This model is better suited for the method presented in this chapter than other existing models for image feature extraction because it was trained from a crack dataset. To take image features from the crack detection model, several of the first layers are frozen and features from the last max-pooling layer are retrained.

Features from images provide extra useful information for the point cloud features. The image feature extraction component extracts features that can enhance the point cloud feature extraction such as crack edge features. Using transfer knowledge from a crack detection model has another advantage, it saves training time, because it uses information that has already been learnt from an existing model and learns from the last layer features with more complex representations specifically suited to the upsampling problem.

While the "point-pixel combination" method can be considered as the combination of point clouds and the *local* image features, the "feature-feature combination" method is the combination of global point cloud features and *global* image features.

Figure 4.2 shows the architecture for feature extraction. The workflow in figure 4.2a is used for point-pixel combination, and figure 4.2b shows the proposed network for combining point clouds and images at the feature level.

In the *point pixel combination* architecture as shown in figure 4.2a, the input can be considered as a 4-channel point cloud. The first three channels are from the point cloud, and the fourth channel is an additional channel that comes from the matched image such as the grayscale value or other features. Then point cloud features are extracted from the 4-channel point clouds by a convolutional neural network. Twelve convolutional layers with an increment of the number of kernels are used. There are four concatenated layers placed after every third layer of the Convolutional Neural Network (CNN) to improve the global features by combining with local features.

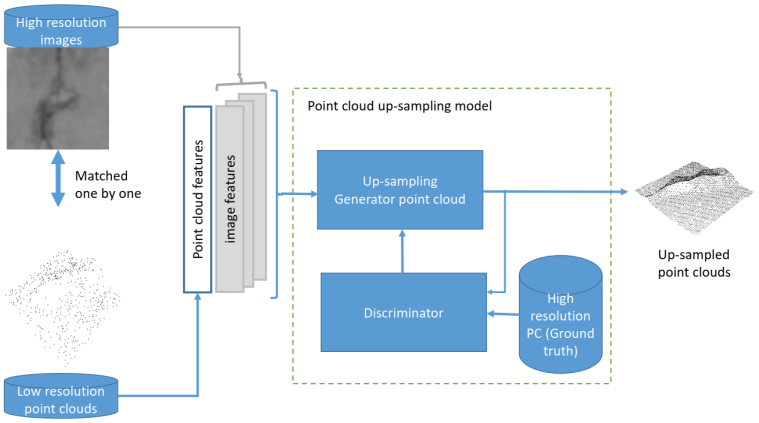
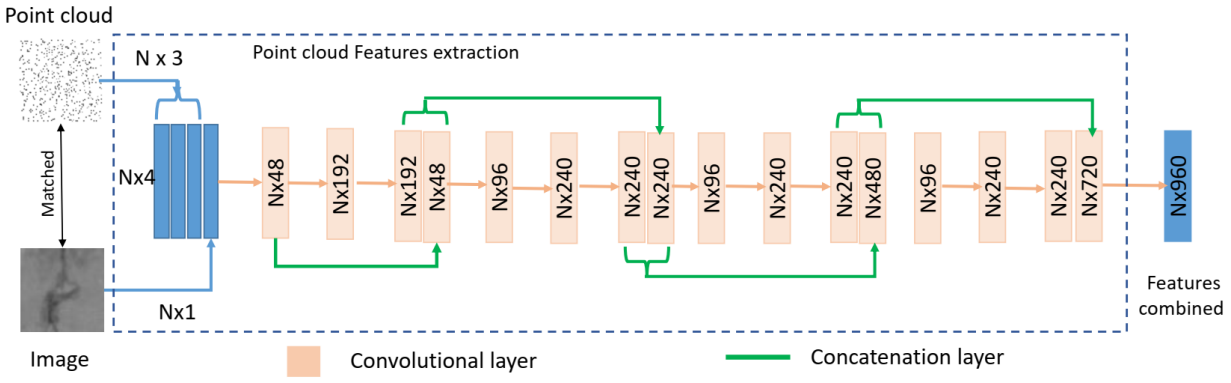
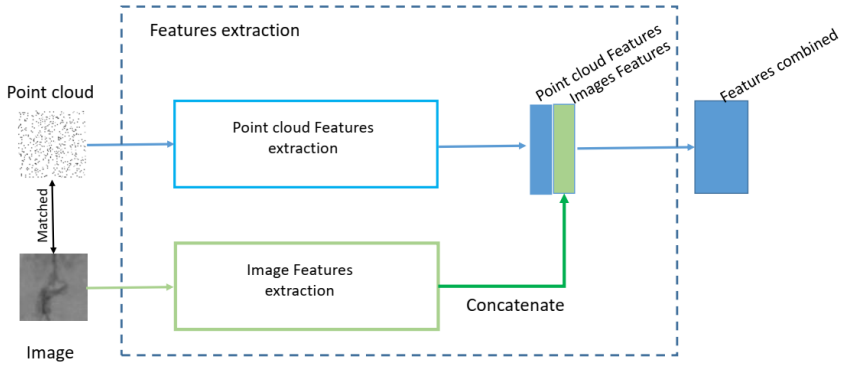


FIGURE 4.1: Proposed architecture for upsampling a point cloud by combining a low-resolution point cloud with a 2D image. The input is the combined data from high-resolution images and their matched point clouds. The point cloud upsampling model is based on a GAN framework that contains both generative and discriminative sub-models. The expected output is a high-resolution point cloud.



(A) Point-pixel combination.



(B) Point cloud feature - Image feature combination.

FIGURE 4.2: Two proposed methods for point cloud and image combination. (a) Point-pixel combination combines an image and point cloud to create 4-channel input data and then extracts features from the 4-channel data. (b) Point cloud feature - Image feature combination combines the point cloud and image by concatenating features that have been extracted separately from the point cloud and image.

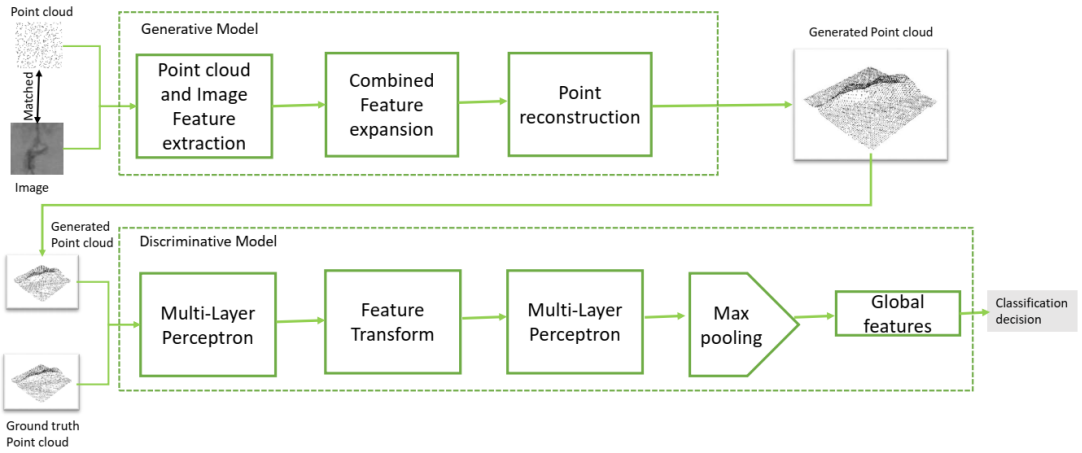


FIGURE 4.3: Point cloud upsampling framework using the combination of point clouds and images based on a GAN architecture. The generative model takes point clouds and images as inputs and comprises components for feature extraction, feature expansion and point reconstruction. The output of the generative model and the ground truth point clouds are inputs to the discriminative model that comprises a feature transform layer, MLP module, max pooling and global features extraction module.

Figure 4.2b shows the combination of point cloud features and image features. The point cloud samples and their matched 2D images are processed separately by the point cloud extractor and the 2D image extractor. Then, global point cloud features are combined with global image features following the two feature extraction models. The output of the *feature-feature combination* is the combined features with a diversity of the features from both point clouds and 2D images.

4.3 Proposed Generative Adversarial Network for Crack Point Cloud Upsampling

4.3.1 Generative model

In a general GAN architecture, the generative model aims to generate new data from input data. While some other upsampling data proposals based on GAN used only the low-resolution data as input data [84, 123], a combination of point clouds and images is used as the input. The upper part of figure 4.3 shows the proposed model for the generator.

There are three transformations in the proposed generative model: data feature extraction, feature expansion, and data reconstruction. A three-transformation generative model was also used in other works for upsampling point clouds such as in PU-GAN [84], and for upsampling images [124, 125]. Three operations use convolutional neural networks. The main role of each part is:

- *Feature extraction* extracts features from low-resolution point clouds. Each feature is represented by a high-dimensional vector. All high-dimensional vectors combine together and produce feature maps $P_{Features}$. The number of features in $P_{Features}$ is equal to the number of points in the low-resolution input point cloud.
- *Feature expansion* based on a deconvolution operation, is a transformation that expands the feature maps $P_{Features}$ to a set of new feature maps $P_{Expanded_Features}$ with a higher number of features. The number of features in $P_{Expanded_Features}$ is similar to the number of points in the target point cloud.
- *Point Reconstruction* is the last phase in the generative model that regresses and aggregates features from $P_{Expanded_Features}$ to build a high-resolution point cloud with a given ratio. In this proposed method, the Farthest Point Sampling method [126, 127] is used to optimize the uniformity of points in the output point cloud. The expected output point cloud is a high-density uniform point cloud.

The network detail of each part in the generative part is described below:

Point cloud feature extraction

The point cloud feature extraction aims to extract strong features from the point cloud data. The experiments in this chapter use a convolutional network for point cloud feature extraction. This part uses four consecutive of convolutional layers. A concatenation layer is used to concatenate the outputs from the two first sequences of convolutional layers. The subsequent three convolutional layers use a higher number of kernels to extract more abstract features.

Image feature extraction

The pre-trained weights from a two stages model for crack detection and segmentation [60] are used in this component for image feature extraction. The convolutional network for image feature extraction contains five convolutional layers. Each convolutional layer is followed by a max-pooling layer to reduce the feature input size and reduce the number of weights in the network.

Feature expansion

The feature expansion component produces more diverse point distributions. Hence, the feature variations are enhanced before being used by the point reconstruction component. The multilayer perceptron (MLP) network is used in the feature expansion to regress 3D coordinates, the MLP is also used in the PUGAN [84] and PU-Net [83] networks for feature expansion. Following this stage, the set of 3D coordinates is extracted by two upsampling layers. A downsampling layer between the two upsampling layers is used to learn more hidden features. The output features are the summary of the outputs from the first upsampling layer and the second upsampling layer.

Point reconstruction

The point cloud reconstruction component is used to build the high-resolution point clouds from the set of point cloud features that output from the feature expansion part. There are two convolutional 2D layers followed by a "gather point" layer used for points reconstruction. The "gather point" layer uses a function "farthest point sample" to update the distance among points in a point cloud by *iteratively sampling the farthest point.*

4.3.2 Discriminative model

The discriminative model aims to distinguish the ground truth point clouds and the generated high-resolution point clouds, so it works as a classification model. The discriminator uses the ground truth point clouds as positive samples, and the generated point clouds as negative samples. The MLP [128] is used for point cloud feature extraction. The MLP is also used in other proposals for point clouds upsampling [84] and point cloud classification [110]. A feature transform block is also used to improve the accuracy of point cloud

classification [110]. Before the last activation layer, a max-pooling layer is used for collecting global features. The output of the discriminative model is a decision that indicates whether an input point cloud is a real or artificially generated point cloud. The lower part of figure 4.3 shows the basic framework for the discriminative model.

In the training phase, the generator and discriminator are trained alternately. In each epoch of the training phase, the generator is kept constant while the discriminator is trained, and then the discriminator is kept constant while the generator is trained. The distribution of the ground-truth point cloud is x, and the input data is a low-resolution point cloud and an image, (z, img). Then, the generator space is $G((z, img), \theta_g)$, where G is a generative function represented by a network with parameters θ_g, and the discriminator space is $D(y, \theta_d)$, where $D(x)$ is the probability that x came from the real data rather than (z, img) and $D(x)$ is represented by a network with parameters θ_d.

While the generator tries to minimize the difference between the created point cloud and the ground truth point cloud, the discriminator tries to maximize the probability of assigning the correct label to ground truth examples and samples from G. Finally, the value function $V(D,G)$ is optimized as equation 4.1:

$$\min_G \max_D V(D,G) = E_{x \sim p_x(x)} [logD(x)] + E_{(z,img) \sim p_{input}(z,img)} [log(1 - D(G(z, img)))] \tag{4.1}$$

4.3.3 Loss functions

In a GAN architecture, each part of the generator or discriminator uses two separate loss functions. The generator loss function minimizes the difference between the generated high-resolution point cloud from the input low-resolution point cloud and the high-resolution ground truth point cloud. The discriminator loss function maximizes the classification performance between the ground truth point cloud and the generated point cloud. If the classification score from discriminator is high, that means the generated point cloud is very different compared with the ground truth. So, in the next iteration of the training process, the generator adjusts the weights in an inverse direction to produce a better point

cloud. The quality of the generated point cloud is estimated by its uniformity, so the uniformity loss function is added to the generator loss function.

The proposed method aims to upsample point clouds using a GAN architecture, wherein the output point clouds should be uniform, and each point should be on the underlying object surface. So, the final loss function has contributions from three terms corresponding to each of the above three goals.

Least Squares loss function for GAN. Least-squared loss is a loss function for GAN architectures that was proposed by [129]. Least-squared loss avoids the problem of vanishing gradients. Equation 4.2 and 4.3 show the loss functions for the generator part and discriminator parts of the proposed system.

$$L_{gan}(G) = \frac{1}{2}[D(Q) - 1]^2 \tag{4.2}$$

$$L_{gan}(D) = \frac{1}{2}[D(Q)^2 + (D(\tilde{Q}) - 1)^2] \tag{4.3}$$

where $D(Q)$ is is the confidence value predicted by D from the generator output Q, and $\tilde{Q}$ is the fake sample.

Uniform loss aims to increase the uniformity of the distribution of points in the generated point cloud. Uniformity is a requirement of the output point cloud both globally and locally. To optimise the global uniformity, A number of random points are selected, and then from each random point an area (denoted as A_i) is generated on the underlying surface. The number of points in each area is then optimised to make the number as uniform as possible across all areas. To optimise the local uniformity in each small area A_i, the nearest distance from point to point is optimised.

Reconstruction loss aims to encourage the generated points to lie on the target surface. Consider Q as the ground truth point cloud. Then Q has the same number of points as the generated point cloud $\tilde{Q}$. The reconstruction loss uses the Earth Mover's [130] distance as per equation 4.4.

$$d(Q, \mathcal{Q}) = \min_{\varphi: Q \to \mathcal{Q}} \sum_{p_i \subset Q} \|p_i - \varphi(p_i)\| \tag{4.4}$$

where $\varphi : Q \to \mathcal{Q}$.

Finally, the model is trained by minimizing the $L_{gan}(G)$ which combines the adversarial, uniform distribution and reconstruction loss terms with weights $\omega_{gan}, \omega_{uni}, \omega_{rec}$, respectively as shown in equation 4.5.

$$L_{gan} = \omega_{gan} L_{gan}(G) + \omega_{uni} L_{uni} + \omega_{rec} L_{rec} \tag{4.5}$$

4.4 Experiments and Results

4.4.1 Data preparation

There is no available crack point cloud dataset or dataset with point clouds and corresponding images. The proposed method is implemented on a crack dataset that was collected specifically for this research. This dataset has two parts. The first part consists of point clouds, and the second part consists of images corresponding to each point cloud. Point clouds and images were collected in a laboratory at the University of Technology Sydney, Australia. The data was collected from concrete blocks having dimensions of $10cm \times 10cm \times 30cm$ with the point clouds captured using an EinScan-SD scanner, and the 2D images captured using a Canon EOS 5D Mark IV camera.

Ground truth point clouds. The original point clouds from the scanner are very large. The original 20 scans of individual concrete blocks are divided into nearly 2000 point clouds for training and 200 point clouds for testing. Each divided point cloud contains 4096 points and one or more cracks.

Input point clouds. The ground truth point clouds are down-sampled randomly to sparse point clouds that each contain 512 points. The proposed method aims to upsample these down-sampled point clouds by a factor of 8 to match the sampling of the original point clouds.

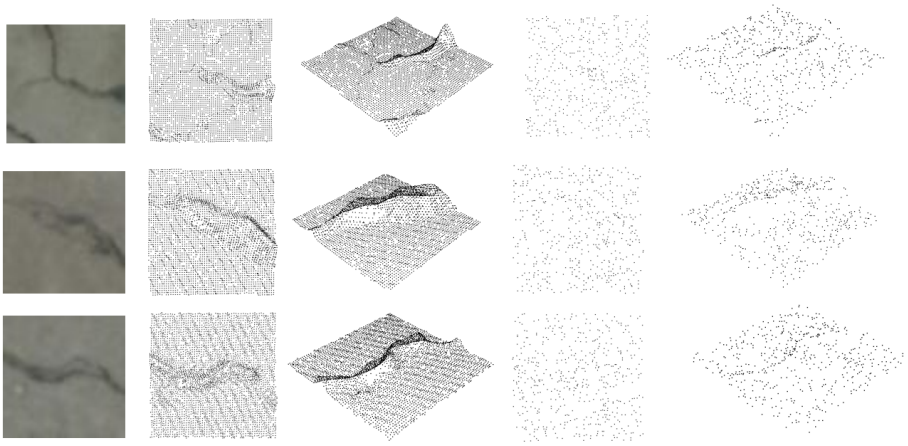


FIGURE 4.4: Examples of images and their corresponding point clouds. The first column shows the original images. The second and the third columns show ground-truth point clouds viewed from the top and an oblique angle respectively. The last two columns show input low-resolution point clouds viewed from the top and an oblique angle respectively.

Images. From the original images and original point clouds, the Meshlab software [131] is used to align the images with the corresponding point clouds. The output of the alignment process is a point cloud and its matched image. For each point cloud tile obtained by dividing the original large point cloud, a corresponding tile of the pre-aligned image of the concrete block is extracted by cropping.

Each ground truth point cloud contains 4096 points, and the corresponding image has a size of 96x96 pixels. For a sparse point cloud containing 512 points, the corresponding image used has a size of 32x32 pixels.

Some examples of point clouds and their corresponding images are shown in figure 4.4.

4.4.2 Evaluation metrics

Two evaluation metrics are used to assess and compare the proposed method with other point cloud upsampling methods.

Chamfer distance (CD) is an evaluation metric for assessing the similarity of two point clouds. It was defined and used in the previous work of Fan et al. [130], and PU-GAN [84]. In this chapter, this distance measure is used to compare point clouds and assess the model.

Given two point sets $P = \{p_1, p_2, ..., p_n\}$ and $Q = \{q_1, q_2, ..., q_m\}$, the CD is calculated as equation 4.6.

$$CD(P,Q) = \frac{1}{P}\sum_{p\in P}\min_{q\in Q}||p-q|\,|_2^2 + \frac{1}{Q}\sum_{q\in Q}\min_{p\in P}||p-q|\,|_2^2 \tag{4.6}$$

Hausdorff distance (HD) was introduced by Hausdorff [132]. Berger describes the application of HD to surface reconstruction in an upsampling proposal [133], and HD was also used to evaluate a method for upsampling point clouds [84]. HD measures how far two non-empty subsets are from each other. HD is the greatest of all the distances from a point in one set to the closest point in the other set. In point cloud comparison, given two point sets $P = \{p_1, p_2, ..., p_n\}$ and $Q = \{q_1, q_2, ..., q_m\}$. Hausdorff distance from A to B is defined as in equation 4.7.

$$H(P,Q) = \max_{p\in P}\min_{q\in Q}||p-q|\,| \tag{4.7}$$

Uniformity measure To calculate the uniformity of generated point clouds from all methods, the standard deviation of the number of points belonging to equal areas on the point cloud surface is calculated. To do that, for each generated point cloud, 1000 sample points are randomly chosen. Then from each point O_j, the number of points inside a circle defined by $(O_j, radius)$ are counted. A point cloud has a higher uniformity if the standard deviation of the counted number of points is small. The uniformity of point clouds is evaluated with two values of *radius, r = 0.4 and r = 0.5.* The proposed method achieves a better uniformity by this measure and this is qualitatively evident in the visualization shown in figure 4.5.

Based on these metrics, a comparison of the performance of the proposed method to that of a number of other established point cloud upsampling methods [83–85] are conducted. The results of this comparison are summarised in table 4.1. The lower values of CD, HD, and uniformity of the proposed method clearly demonstrate the superior performance of the proposed approach compared to these methods. Figure 4.6 shows some examples of the image input, low-resolution point cloud input, ground truth point cloud, and the generated point clouds from the PU-Net, PU-GAN, PCSR, and the proposed method.

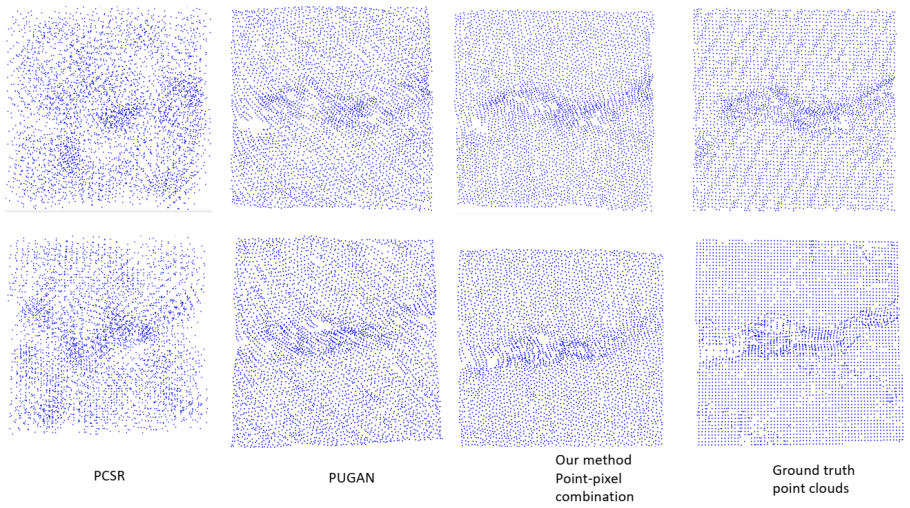


FIGURE 4.5: Examples of generated point clouds with different uniformities created using two existing methods and the proposed method compared with the ground truth. The first column shows results from the Point Cloud Super Resolution (PCSR) method, the second column shows results from the PU-GAN method. The third column shows the proposed method's results using point-pixel combination, and the last column shows the ground truth point clouds.

Methods		CD (10^{-3})	HD (10^{-3})	Uniformity	
				r = 0.4	r = 0.5
PU Net		328.86	1650.18	-	-
PCSR		0.59	14.47	8.32	12.01
PU-GAN		0.33	6.07	4.08	5.9
The proposed methods	Point-Pixel with grayscale	**0.30**	**5.74**	**3.61**	**5.6**
	Point-Pixel with Sobel	**0.29**	**5.25**	**3.39**	**5.1**
	Point-Pixel with Gamma	**0.28**	**4.96**	**3.42**	**5.2**
	Point-Pixel with DoG	**0.28**	**4.95**	**3.17**	**4.9**
	Feature - Feature	**0.31**	**5.75**	**3.8**	**5.6**

TABLE 4.1: Comparisons of Chamfer distance and Hausdorff distance measurements.

The CD and HD value of the PU-Net model is very large as compared to other methods as shown in table 4.1. The PU-Net model used a feature aggregation method that does not consider spatial information when performing the feature expansion operation [83]. It is speculated that this is the reason that the PU-Net model produces eight disjoint point clusters when used to upsample the crack dataset, resulting in very poor performance (the fourth column in figure 4.6). As a result, the PU-Net model obtained a very high value of CD and HD. So, it is not necessary to further compare the PU-Net model with other methods. Figure 4.6 shows the proposed method's results when using *point-pixel combination* with DoG features from images.

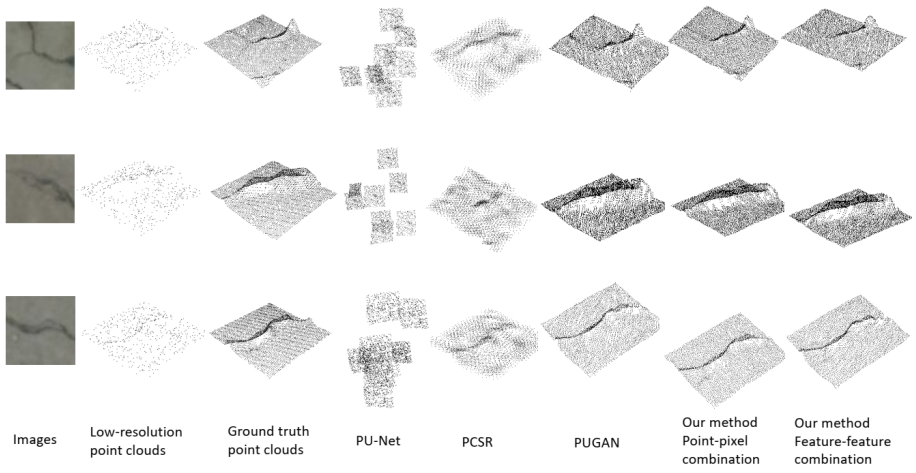
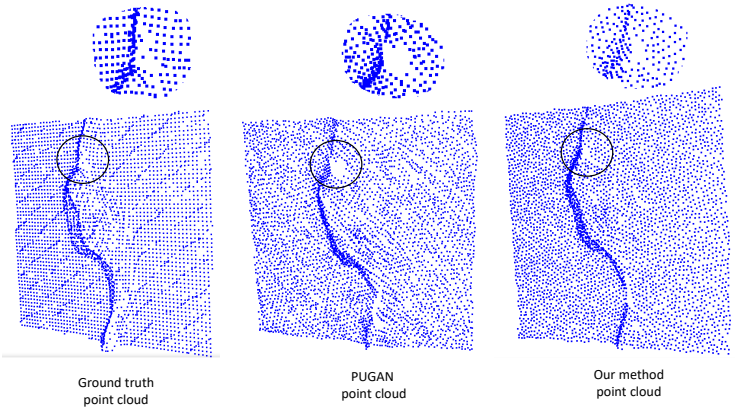


FIGURE 4.6: Examples of generated point clouds from five different methods and three different sets of input data. The two first columns contain input images and input low-resolution point clouds. The third column contains ground truth point clouds. The subsequent three columns are results from three previous methods, PU-Net, PCSR and PU-GAN. The last two columns are results from the proposed method with the two different approaches for combining image and point cloud information in the GAN architecture.

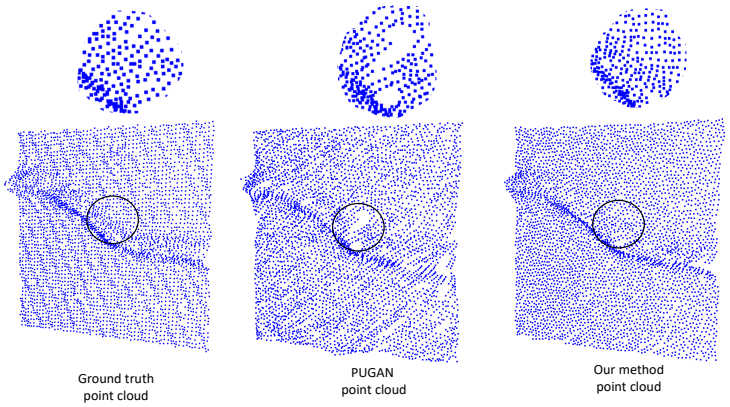
The crack point cloud data are 2.5D point sets that describe the surface of concrete blocks. However, the generated point clouds from the PCSR [85] are watertight in some areas, resulting in the cracks of the generated point clouds from PCSR being filled over with points, so they do not show the correct surface of the concrete in the vicinity of the cracks.

The PU-GAN model [84] achieved good results in these experiments. However, the fusion of high resolution image data achieved by the proposed approach significantly improves upon the results from the PU-GAN method. By combining point cloud and image data at two levels (point-pixel level and feature-feature level), the proposed method achieves better results against the CD and HD distance metrics. The proposed method also generates point clouds with more uniformly distributed samples.

The proposed method is better at filling the gaps in the point cloud surface as shown in figure 4.7. Some examples from PU-GAN, the PCSR method, and the proposed method using "point-pixel combination" with DoG features are taken to compare. The two sides of a crack are often sparser than other areas of the point cloud. The areas surrounded by black circles in the figure can be compared in figure 4.7. On the ground truth surface, this area is sparse, and on the point cloud generated by PU-GAN, there is a hole. The proposed method can generate this area with a higher point density.



(A) a



(B) b

FIGURE 4.7: Comparing results from the proposed method and other methods in terms of filling a gap in the point clouds. The first column shows ground truth point clouds, the second shows results obtained by PU-GAN, and the last column shows results obtained from the proposed method. The ground truth in figure (a) shows a visible gap on the edge of the crack, while the ground truth in figure (b) shows a sparse crack edge.

In the "point-pixel combination", the grayscale and other features from images such as Sobel, Gamma, DoG features are used. In comparison to the "feature-feature combination", the "point-pixel combination" is the better combination method. Among the "point-pixel combination" options, the combination of point cloud with a channel of DoG features extracted from the images is the best combination for upsampling.

4.5 Crack Point Clouds Classification

To assess the impact of the point cloud's resolution on point cloud classification, the Pointnet architecture [110] is applied for classifying the point clouds into crack and non-crack point clouds in low-resolution and high-resolution. Positive samples are point clouds that contain crack points, negative samples do not contain crack points. The Pointnet network takes a point cloud as input, then followed by a feature transform between two MLP modules and then aggregates point features by a max-pooling layer. The classification output is the score for the two classes of crack point cloud or non-crack point cloud. The low-resolution point clouds contain 512 points as the input of the upsampling model. The high-resolution point clouds containing 4096 points are the ground truth point clouds, and the generated point clouds from the different upsampling methods. Table 4.2 shows the proposed method's results for the classification of point clouds. Precision (Pr), recall (Re), F-score, and accuracy are used to evaluate the classification model. The results indicate that the high-resolution point clouds can be classified better than the low-resolution point clouds. The generated point clouds from the proposed method are better for classification than other upsampling methods.

The experiments in this paper show that the number of points in the point cloud affects the accuracy of crack detection. The experiment with a small number of points (512 points for each cloud) got a lower accuracy (63%). The experiment with a higher number of points (4096 points for each cloud) got a higher accuracy (up to 81%). In recent multispectral LiDAR point cloud classification research [134], the experiments also show that the higher number of points, the better the accuracy of the point cloud classification. Point cloud data that has a high uniformity where the number of points is large enough to represent small details will give the best results. The number of points should be optimized depending on the dataset and the number of points should accurately capture the shape of the object [110]. Our ground truth data and our upsampled data display a high uniformity and sufficient density to achieve a reasonable accuracy in crack detection.

The results shown in table 4.2 also indicate that the upsampled point cloud improves the classification task compared to crack image classification. In this chapter, the detection model from the 2StagesCrack method [60] is used for the crack image classification task.

Data		Pr	Re	F1-score	Accuracy (%)
High resolution point clouds from	Ground truth	**0.99**	**0.73**	**0.85**	**81**
	PCSR	1	0.5	0.66	50
	PU-GAN	0.98	0.61	0.75	67
	Proposed method	**0.94**	**0.69**	**0.80**	**76**
Low-resolution point clouds		0.90	0.59	0.71	63
Low-resolution images		0.61	0.86	0.72	74

TABLE 4.2: Results in low-resolution and high-resolution crack point cloud classification.

The experiment outputs a high number of false positives (FP), so the value of Pr is very low (0.61 as shown in table 4.2). In the low-resolution image, some artefacts such as holes in the concrete are very similar in appearance to cracks, which leads to the high number of FP and the low Pr. Although the accuracy of the generated high-resolution point cloud classification is not particularly higher compared with that of the image classification (2% higher), the F1 score is significantly improved (from 0.72 to 0.80). In an unbalanced dataset such as crack road image dataset, the PRC metric is a valuable metric for evaluation.

The PUGAN [1] technique was modified from PU-Net [2] and PCSR [3] was modified from PUGAN and PU-Net. Their architectures were designed for three channel point cloud datasets, and 2D image information cannot be added to these models by a simple modification. The GAN architecture used in my thesis has improved on previous models with many changes and modifications for adapting to the combination of 2D and 3D input data. The experiments show that the combination of 2D and 3D data is necessary to improve the point cloud upsampling and the point cloud classification results.

4.6 Chapter Conclusions

There are some existing works that combine 2D images and 3D data for other applications, and there are some contributions using the combination of 2D and 3D data for upsampling point clouds at the low-level feature level such as gray-scale features. The method proposed in this chapter combines 2D and 3D data at the high-level features level to upsample

[1] https://github.com/liruihui/PU-GAN
[2] https://github.com/yulequan/PU-Net
[3] https://github.com/wuhuikai/PointCloudSuperResolution

point clouds and demonstrates significantly better results compared to existing point cloud upsampling methods.

A GAN architecture is employed for the combination in two ways: the first way integrates points from 3D space and corresponding pixels from 2D space, the second way integrates point cloud features and image features that are transferred from different models. The proposed method also illustrates that images and point clouds can be combined at both low-level and high-level feature stages. The generated high-resolution point clouds show that the combination of low-level features first, followed by the extraction of high-level features, is more effective than combining the high-level features that have been extracted separately.

The experiments in this chapter generate high-resolution point clouds with high uniformity. The Chamfer and Hausdorff standard distance measures are used for evaluating point clouds and the proposed method obtains superior results compared with current state of the art methods. In terms of the detection of cracks in the point clouds, the up-scaled point clouds from the proposed method perform better in terms of classification compared to other methods.

The high-resolution point clouds are generated from low-resolution point clouds, so they save cost in the collection and storage of the point clouds and allow for the use of less expensive and more practical scanning equipment. The generated point clouds are also shown to improve the detection of cracks in the point clouds when compared to the use of the low resolution point clouds. The proposed method can be used in different 2.5D point clouds of other constructions and surfaces such as concrete or steel bridges and tunnels in both upsampling and classification. This method has the potential to contribute significantly to civil engineering applications.

The proposed method is tailored for 2D images and associated 2.5D point clouds. In this thesis, combining images and full-scale complex structured 3D point clouds for upsampling has not been investigated. The extension to 3D point clouds with complex structure is complicated by the difficulty of capturing the necessary structural information in a 2D image. However, with further work, this method could be adapted to more general 3D point clouds if the 3D object can be treated as a set of 2.5D surfaces with associated point

clouds and images. Future work could thereby extend the proposed approach to point clouds of more complex 3D structures.

This chapter proposed a new method for upsampling crack point clouds using a combination of point clouds and two dimensional images as input data. This method uses a GAN architecture and combines point clouds and images in two ways: the first way involves concatenating points from 3D space and the corresponding pixels from 2D space, the second way involves concatenating point cloud features and image features that are transferred from different models. Comparisons of the proposed approach to other established methods based on both distance metrics and a measures of uniformity of the point samples in the point cloud demonstrates that the combination of point clouds and images can improve the performance of upsampling of very low density point clouds. The focus in this chapter is on crack point clouds such as those from concrete bridges, and road pavements, but the approach could be generalised to other 2.5D point cloud datasets. The dataset used in this chapter will be made available to the research community.

Chapter 5

Analysis and Classification of Crack Characteristics

5.1 Introduction

In chapter 2, table 2.2 shows that cracks can be classified based on the type or severity level of the cracks. The crack features such as crack length, and crack width help to classify cracks into one of three types *fatigue crack, block crack, edge crack*, or a severity level of *low, moderate, or high level*. In addition, the direction of cracks can be used to divide cracks into longitudinal cracks and transverse cracks. This chapter will focus on analyzing and calculating two features of cracks; crack width and crack length.

Figure 5.1 shows a simple workflow for calculating and classifying cracks. To calculate the length of cracks, firstly the segmented cracks from the model presented in chapter 3 are used. Then, the thinned skeletons of the segmented crack areas are taken, and the length of these thinned skeletons are considered the length of the cracks. The width of a crack is the distance between the two edges of the crack.

An example image from the CrackIT dataset [15] and its processing is shown in figure 5.2. The processing includes segmenting the crack, finding the thinned skeleton of the crack, and calculating the crack width and length. The width of crack changes along the crack

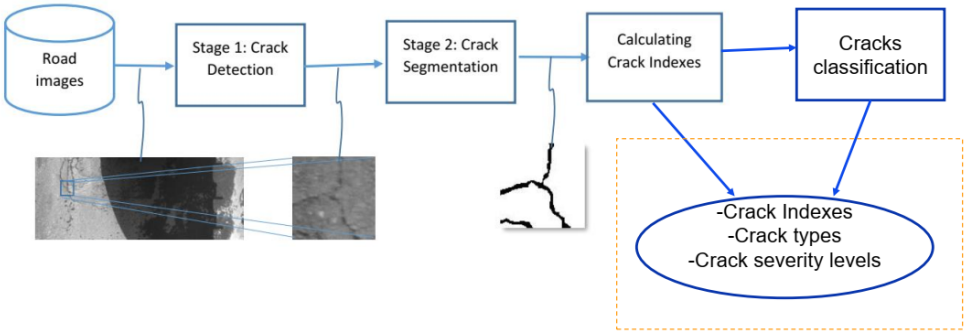


FIGURE 5.1: Workflow for the analysis and classification of crack characteristics. The segmented results from the segmentation model are used to analyze and calculate crack indexes. Following this, cracks are classified based on crack features such as width and length.

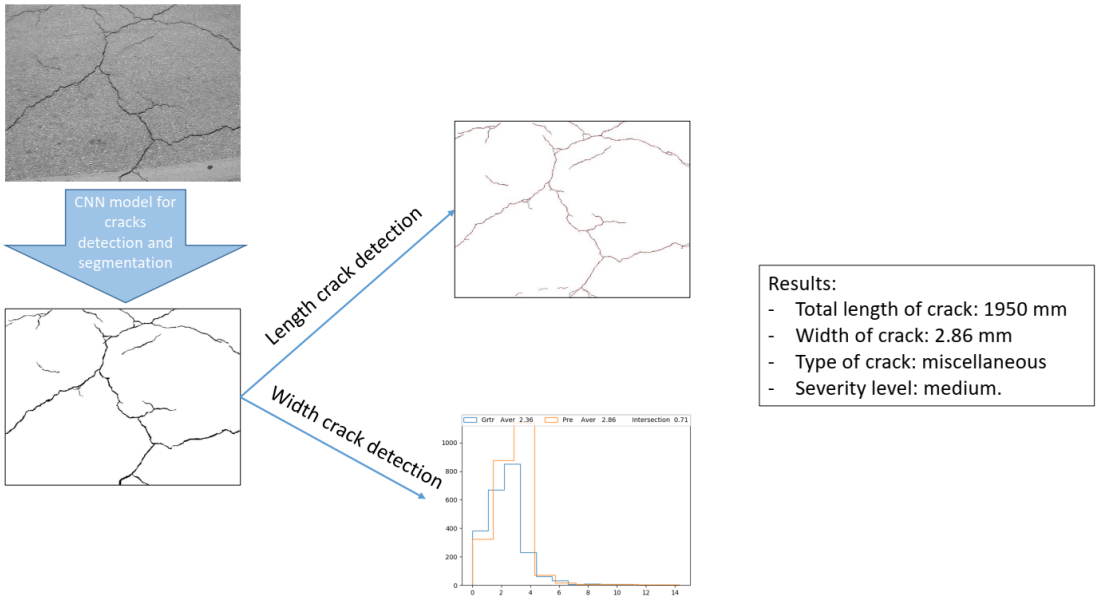


FIGURE 5.2: Example of crack characteristics calculation. The input image is processed with the detection and segmentation model. The thinned skeleton of the crack is extracted from the segmented results and used for crack length calculation. Crack width is calculated using the distance from the two edges of the crack to the thinned skeleton.

length, so the distribution of the crack width is also calculated. From these features, the results are calculated as the length and width of the crack in millimeters. The type of crack in this example is miscellaneous, because it consists of both longitudinal and transversal cracks. The average width of cracks at 2.86 mm indicates that the cracks in this example have a medium severity level.

In the following sections, the proposed method to calculate the crack length and crack

width is presented. The results are also evaluated using an advantageous measure that is suitable for one-pixel width objects such as the thinned skeleton crack data.

5.2 Crack Length Calculation

To calculate the length of the crack from an original crack image, the first step is to segment the cracks at the pixel level by using the two-stage model presented in chapter 3. The segmented results are binary images. Then, the thinned skeletons of the segmented crack areas are extracted.

There are some methods that have been used for thinning an object in an image [82, 135, 136]. However, these methods often not only detect the main skeleton but also detect many branches of the main skeleton. In the proposed method, the thinning method of Zhang and Suen (ZS) [82] is improved, and this method prevents the detection of insignificant branches of cracks that can bias or disrupt the calculation of length.

The thinned skeleton describes the main part of each crack, and this skeleton is in the middle of the crack. So, the length of the detected thinned skeleton can be considered as the length of the crack. Using the thinned skeleton for crack length detection is also better than using the edge of the crack, especially with large and non-symmetrical cracks.

Figure 5.3 illustrates three ways to estimate the length of cracks. In this figure, sub-figure (a) shows a crack with a skeleton containing many branches. If this skeleton was used to calculate the crack length, the estimate may be longer than the real crack. The sub-figure (c) shows two asymmetrical edges (red lines) of the crack. If these edges are considered as the length of the crack, it can lead to inaccurate or inconsistent results due to the differing lengths of the two edges. The sub-figure (b) shows the main skeleton, or the middle line of the crack. The middle of the crack is used to define the length of the crack [17]. This dissertation uses a thinning method taking the main skeleton only with a width of one pixel, so the estimated results for the length of the crack are more accurate than other methods.

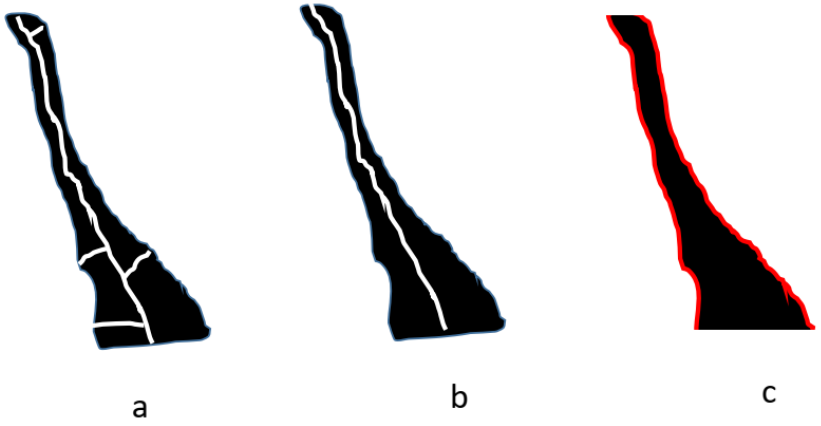


FIGURE 5.3: Three ways to estimate the crack length: (a) the skeleton of a crack with spurious branches, (b) the main skeleton or the middle line of the crack, (c) the two edges of the crack

The final crack length is measured by counting the number of pixels of the middle line. From the number of pixels and the spatial resolution, the length of the crack in millimeters can be defined.

5.2.1 The disadvantage of applying traditional evaluation metrics to a detected crack line

After the thinning process, the middle line of a crack is detected with a width of one pixel. When compared to the ground truth crack line, some of detected pixels are coincident with ground truth pixels, others are near the ground truth and others are far from the ground truth.

There are some metrics for evaluating object detection, such as precision-recall (PRC) and accuracy. Accuracy is the ratio between the number of correct detected samples and the total number of samples. However, this definition of accuracy is not suitable for imbalanced data like crack datasets. In some crack datasets, the percentage of the background pixels is nearly 100% (as shown in table 3.1), so if a model classifies all samples as negative samples, the accuracy is still high at nearly 100%. So, the above definition of accuracy should not be used for evaluating, training or testing systems on typically imbalanced crack datasets.

PRC including precision *Pr*, recall *Rc*, and F_1 are calculated using equation 3.1, 3.2, and 3.3, respectively. The number of True Positive (TP), False Positive (FP), False Negative (FN), and True Negative (TN) are defined as in table 5.1.

TABLE 5.1: TP, FP, FN and TN in traditional PRC

	Positive Prediction	Negative Prediction
Positive Class	True Positive (TP)	False Negative (FN)
Negative Class	False Positive (FP)	True Negative (TN)

If there is a strict requirement that the TP pixels must be the ground truth pixels, then there is a danger of missing many correct pixels that have slightly shifted positions. In previous work this was avoided by allowing for a relaxed definition of TP such that "a detected crack pixel is still taken as a true positive if it is no more than 2 pixels away from human annotated crack curves" [137]. So, the traditional definition of TP, FP, TN, FN as in table 5.1 is not a good definition for estimating the detected crack line results.

5.2.2 A new measure based on PRC for crack line detection

In this section, a new method for evaluating the detected line is presented as an improvement on the traditional evaluation method - PRC. The proposed measure uses the same equations 3.1, 3.2, and 3.3 for Pr, Re and F_1, respectively. However, the proposed method uses adjusted definitions to count TP, FP, FN, and TN pixels more accurately.

The proposed definition of a TP, FP, FN, and TN pixel can be defined based on the distance from the detected pixel to the nearest ground truth pixel. Assume that **s** is a distance in pixel units, then TP, FP, FN, and TN pixels can be defined as below:

- True Positive: a pixel is a TP sample if it is a detected crack pixel and the distance from it to the nearest ground truth pixel is smaller than **s**
- False Positive: a pixel is a FP sample if it is a detected crack pixel and the distance from it to the nearest ground truth pixel is larger than **s**
- False Negative: a pixel is a FN sample if it is a ground truth pixel and distance from it to the nearest detected crack pixel is larger than **s**

TABLE 5.2: New definitions of TP, FP, FN and TN

	Detected crack pixels	Non-detected crack pixels
Distance to the ground truth smaller than **s**	True Positive (TP)	False Negative (FN)
Distance to the ground truth larger than **s**	False Positive (FP)	True Negative (TN)

- True Negative: a pixel is a TN sample if it is a non-detected crack pixel, non ground truth pixel, and the distance from it to the nearest ground truth pixel is larger than **s**. TN is not used in calculating Pr and Re.

The new definitions of TP, FP, FN and TN are suitable for the one-pixel-width line detection problem due to the following reasons:

- A detected crack line and the corresponding ground truth line are one-pixel-width lines. In some applications such as crack line detection, only pixels near the ground truth are counted. So, if only the number of detected pixels that are also ground truth pixels are counted, valuable pixels that are close to the ground truth may be missed.
- The ground truth is created by humans. So, the accuracy of the ground truth depends on skills and perspective of technicians. Using the new definition can avoid the over-dependence on human judgement.

Table 5.2 shows a summary of the new definitions of the TP, FP, FN and TN. The new parameters of TP, FP, FN and TN are then used to calculate Pr, Re and F_1 as per equation 3.1, 3.2, and 3.3, respectively. Experiments were conducted that calculated the thinned skeleton from the ground truth crack areas, and defined the thinned skeleton as the crack skeleton ground truth. Then, a comparison of the ground truth crack lines with the crack lines from the segmented crack areas is performed. The PRC matrix is calculated with different values of **s**. Some results are shown in the experiments section 5.4 of this chapter.

5.3 Crack Width Calculation

Crack width is defined as "the average gap in millimeters (inches) between the two edges of a crack measured at points along the crack" [17]. In this dissertation, the segmented

results are considered as crack areas. Then, for each crack, the two edges are found and the distance between them calculated. There are three steps for calculating the crack width from the segmented crack areas *sca*:

1. Determine the boundary of the *sca*
2. Determine two separated edges of the boundary
3. Calculate the shortest distance from each pixel on one edge to the other edge. These distances are width of the crack at different points. Then the width of crack can be observed as the distribution of distance between the two edges or at different points the average of these distances.

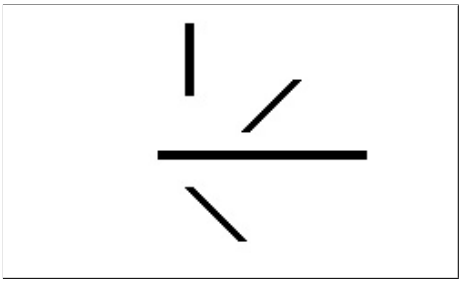
Figure 5.4 shows an example of the width calculating procedure. Figure 5.4a is a binary image that shows the original segmented areas in black. The boundaries of cracks are shown in figure 5.4b. Figure 5.4c shows the thinned skeletons of all cracks, and figure 5.4d shows two edges of each crack in black and red. The final step is calculating the distance from each pixel that belongs to a black edge to the closest pixel on the red edge or vice versa. The result is an array of shortest distances, which can be described as a distribution or average value as shown in figure 5.4e.

Determining the boundaries of the segmented crack areas

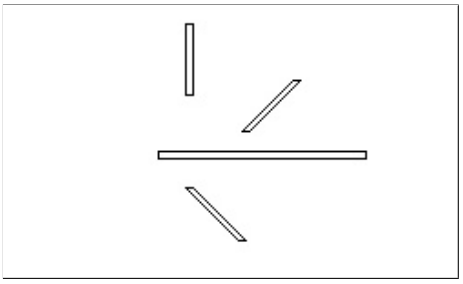
The segmented crack images are binary images. So, applying the Canny edge detection method [32] is an easy way to find the boundary of all cracks. Each continuous crack contains two edges or two sides, which means, each edge of a crack has a corresponding edge on the other side of the crack. Canny's method just detects the edges of the crack, but this technique cannot distinguish between the two sides of a crack. So, this dissertation proposes a technique to separate the two sides of each crack.

Determining the two corresponding edges of each crack

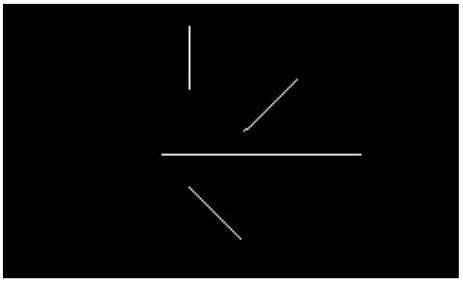
The thinned skeleton of a segmented crack is considered as the middle of the crack. The two edges of the crack are two sides of the boundary and the thinned skeleton is in the middle of the two edges. With each pixel T in the thinned skeleton, two pixels, B and C,



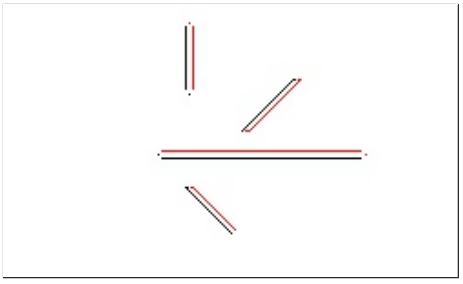
(A) Original segmented areas.



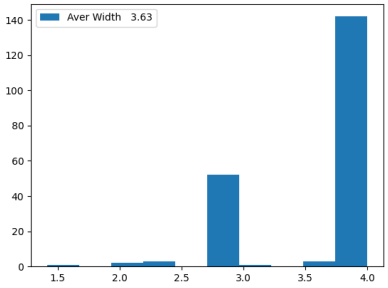
(B) Boundaries of the cracks.



(C) Thinned skeletons of the segmented areas.



(D) Edges of the two sides of the cracks.



(E) Distribution of crack widths.

FIGURE 5.4: An example of the crack width calculation. The original image contains cracks in many different directions. The boundaries of the cracks are determined by edge detection using the Canny method. Then the middle crack and two separated edges are extracted. Finally, crack width and length are calculated and shown in a distribution.

belonging to the two adjacent edges are spaced on either side of T (either up and down or right and left) and are detected as the two pixels belonging to the pair of edges.

There are many cracks in an image. To calculate the width of each crack, a pair of edges belonging the crack must be defined. To avoid the problem that a pair of detected edges contains one edge of a crack and one edge of another different close crack, another condition is used. This condition uses the Bresenham line [138]. In this case, two edge pixels satisfy the Bresenham line condition if the line between them contains only crack pixels. Finally,

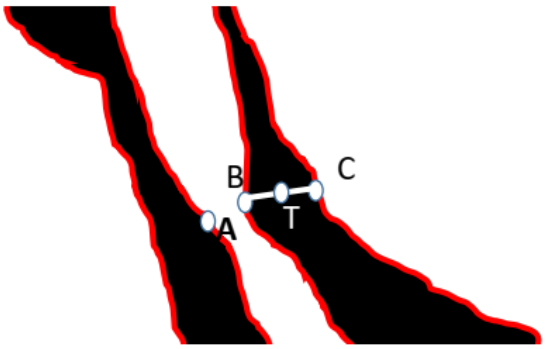


FIGURE 5.5: The Bresenham line is used to define a pair of edges of a crack. A and B are two edge pixels close together. The Bresenham line determines correctly that B and C are corresponding edge pixels and A is not a corresponding edge pixel for either B or C.

a edge pixel corresponds with another edge pixel if the line between them is a Bresenham line and contains a thinned skeleton pixel.

Figure 5.5 is an example of using the Bresenham line condition to define two corresponding edge pixels. There are two close cracks marked in black with the edges of cracks marked in red. A, B, and C are edge pixels and T is a thinned skeleton pixel. B and C are the two closest edge pixels belonging to two corresponding crack edges. Point A is closer to B than C, but C is the corresponding edge point of B, because the straight line BC is a Bresenham line that contains only crack pixels and contains a thinned skeleton pixel. A and B are very close but they are not corresponding crack edges because background pixels lie between A and B.

5.4 Experiments and Results from the Calculation of Crack Characteristics

5.4.1 Data set preparation

The **CrackIT data set** was published and used in a Ph.D. dissertation [15]. The original data set has ground truth cracks that are lines one-pixel in width. For implementing the methods in this chapter, segmented ground truth images at the pixel level were made, containing the entire width of the cracks.

The **UTSConcrete data set** was captured at the TechLab, UTS. These images are concrete crack images. Cracks on the concrete samples caused by the alkali–silica reaction (ASR). This data set had a high spatial resolution, where 1 square mm on the concrete surface is represented by a square area of 16×16 pixels in the captured image. The UTSConcrete data set is a diverse data set, that contains multiple levels of crack width, noises types and brightness levels.

To create ground truth images the *Image Segmenter* tool is used with the "Draw Freehand" function from the *Image Processing and Computer Vision* toolbox in the Matlab software package [105]. The ground truth images are binary images such that the crack pixels appear in black and the background pixels are in white.

All experiments in this chapter are implemented on a PC with a single CPU Intel core i7 processor with a speed of 2.81 GHz, 8GB of RAM, running the Ubuntu operating system.

5.4.2 Results

Figure 5.6, 5.7, and 5.8 show examples of experiments on the CrackIT data set, and figure 5.9, 5.10, 5.11 show examples of experiments on the UTSConcrete data set. Each figure contains six sub-figures: the original image, the ground truth image, the thinned skeleton of the segmented cracks in black and the thinned skeleton of the ground truth cracks in red, the segmented image, a table that shows the new PRC metrics when calculating the crack length with the threshold **s** varied from 0 to 5 pixels, and finally, a graph that shows the distribution of crack width along all points of the cracks and the average width in pixels. In calculating the crack length, the highlight value in the table of results corresponds to a threshold s, and s is equal to 3 pixels. It can be seen that, when **s** increases, F_1 also increases. In all examples, the F_1 score of the crack thinned skeleton detection method is around 90%.

Figure 5.6 and 5.7 show two typical examples. The original image 5.6a and 5.7a contains thinned skeleton cracks on a noisy background with a lot of stains, and black and white dots. Some cracks are thin, at one or two pixels, and the average width of the ground truth is smaller than two pixels. The segmentation method segmented almost all of the

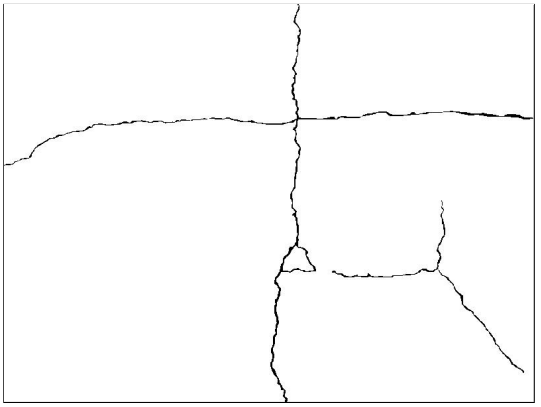
crack pixels. At a threshold of $s = 3$ pixels, the detected thinned skeleton crack method achieved an F_1 of 89% on image 5.6a and 90% on image 5.7a. The intersection of the two distributions from the ground truth width and segmented width are more than 60%. Another example from the CrackIT data set is shown in figure 5.8. Image 5.8a contains many cracks in different directions, and there is paint at the bottom of this image. The F_1 for crack length detection is more than 90%, and the detected width is close to the real width, at 2.36 pixels and 2.69 pixels, respectively.

Figure 5.9 is a testing example from the UTSConcrete data set. The image 5.9b is a small image with a size of 344×299 pixels, that contains a short crack. Although this original image is blurred and noisy, the F_1 for the crack length detection is 99% and the detected crack widths are similar to the ground truth widths for most of points. Image 5.10a has a size of 2759×1353 pixels, and contains a lot of bubbles in the concrete caused by air while the concrete was setting. This image has low brightness and contains dark areas created by high humidity during the setting process. The segmentation method has removed all of the above artifacts. Finally, the F_1 score is 91% for crack length detection in this example. The last example shown in figure 5.11 is a difficult case. The image shown in figure 5.11a has a size of 1327×713 pixels and contains a wide range of crack sizes from as tiny as 1 pixel to as large as about 40 pixels. This image has good contrast, but there is a green ink stain in the image. In addition, some part of the cracks have broken edges. This kind of damage is not considered as a crack but is easily misclassified as crack pixels. The image shown in figure 5.11d demonstrates a good segmentation result. The broken edge and ink trace was segmented as background, and almost all big and thin cracks are segmented.

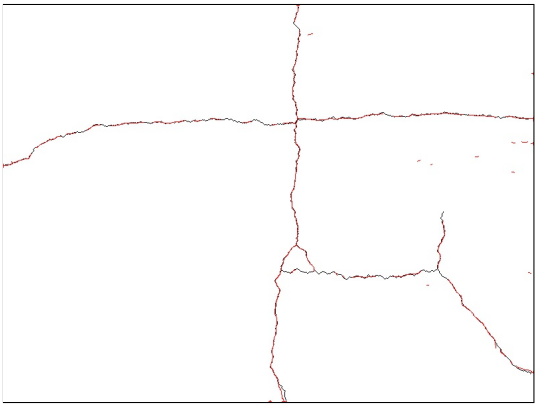
5.5 Chapter Conclusions

This chapter has implemented an algorithm for determining crack characteristics. Crack features such as crack length and crack width can be used for severity level crack classification and the classification of cracks as longitudinal or transverse cracks. A proposed novel metric based on the traditional PRC metric was used to evaluate the length crack detection results. The new metric uses the same equation as the old PRC metric, but uses

(A) Original image.



(B) Ground truth image.

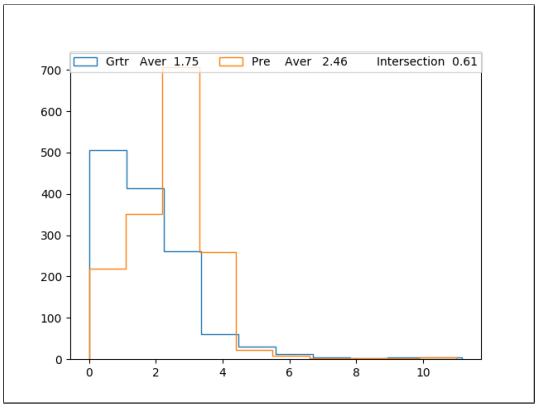


(C) Thinned skeletons of segmented cracks and thinned skeletons of ground truth cracks.

(D) Segmented image.

s	Pr	Re	F1
0	0.44	0.40	0.42
1	0.85	0.75	0.8
2	0.91	0.81	0.86
3	0.93	0.85	0.89
4	0.94	0.87	0.91
5	0.95	0.9	0.92

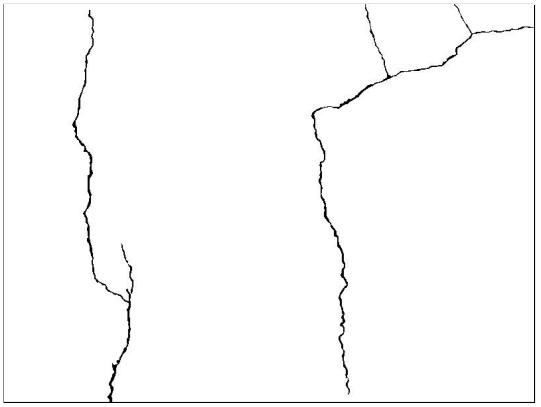
(E) PRC of crack length results.



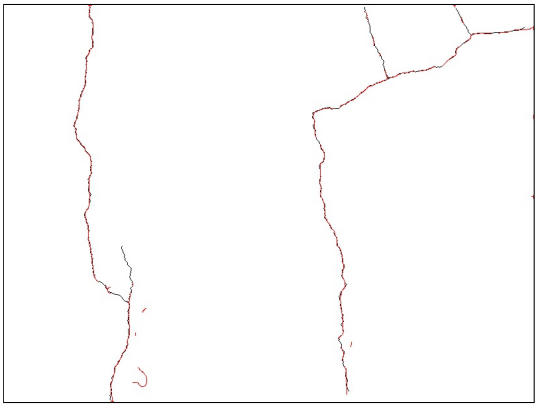
(F) Crack width results.

FIGURE 5.6: Experiment on cracks in a noisy image from the CrackIT data set.

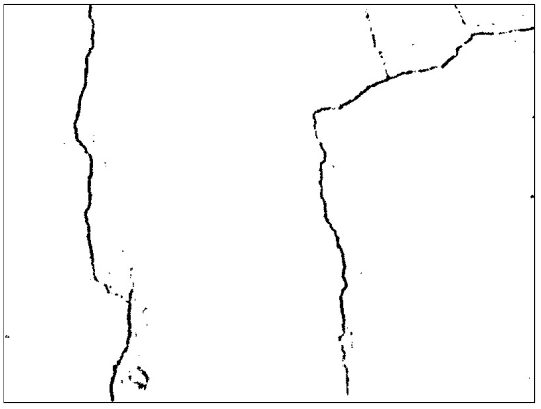
(A) Original image.



(B) Ground truth image.



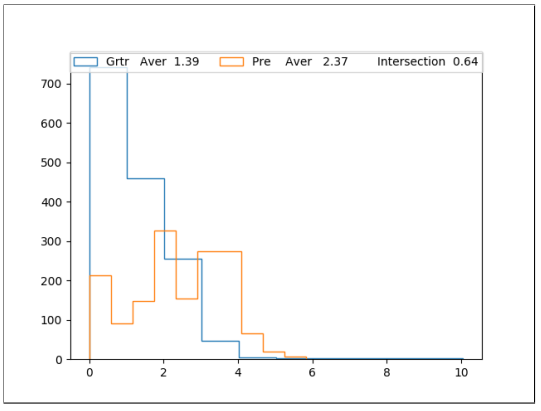
(C) Thinned skeletons of segmented cracks and thinned skeletons of ground truth cracks.



(D) Segmented image.

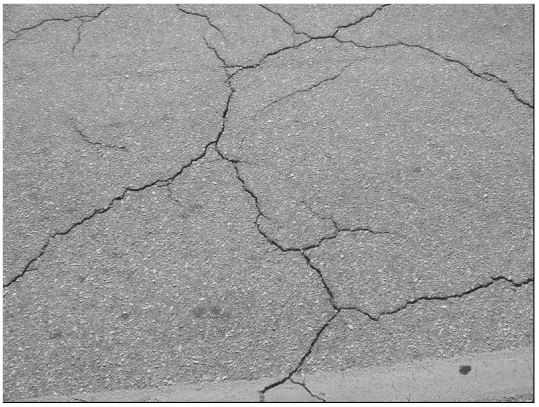
s	Pr	Re	F1
0	0.43	0.42	0.43
1	0.85	0.8	0.83
2	0.9	0.86	0.88
3	0.91	0.89	0.9
4	0.92	0.9	0.91
5	0.93	0.92	0.92

(E) PRC of crack length results.

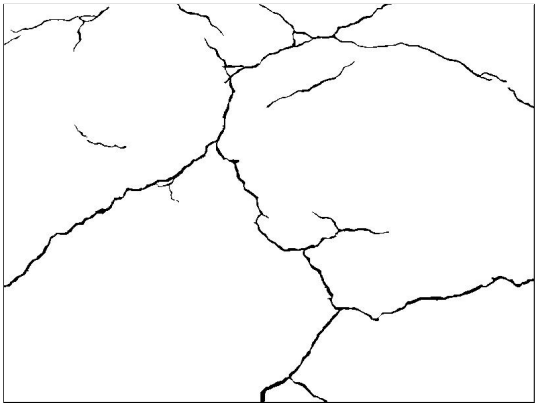


(F) Crack width results.

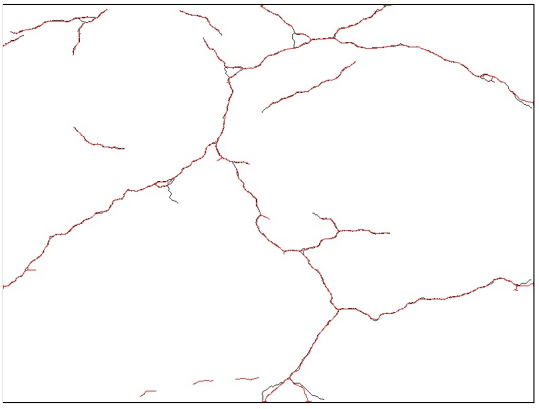
FIGURE 5.7: Experiment on thinned skeleton cracks from the CrackIT data set.



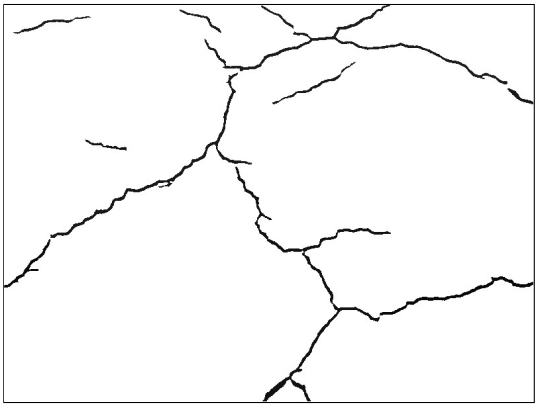
(A) Original image.



(B) Ground truth image.



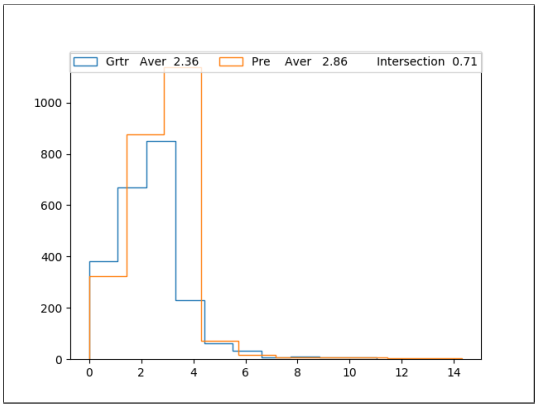
(C) Thinned skeletons of segmented cracks and thinned skeletons of ground truth cracks.



(D) Segmented image.

s	Pr	Re	F1
0	0.41	0.47	0.44
1	0.82	0.85	0.84
2	0.89	0.91	0.9
3	0.91	0.93	0.92
4	0.92	0.94	0.93
5	0.93	0.96	0.95

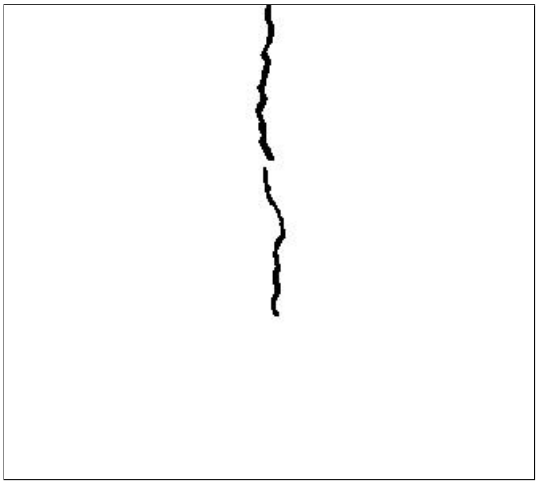
(E) PRC of crack length results.



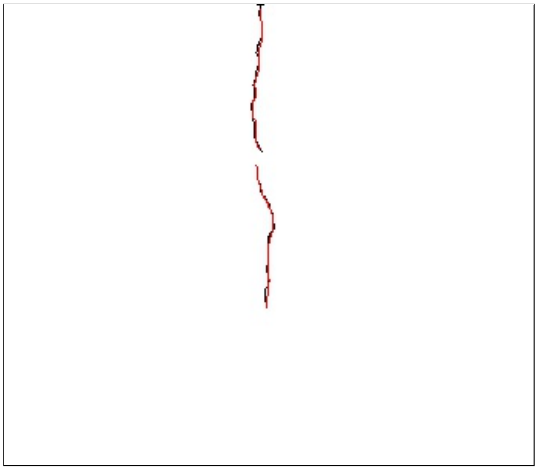
(F) Crack width results.

FIGURE 5.8: Experiment on a image with many cracks and artifacts like paint.

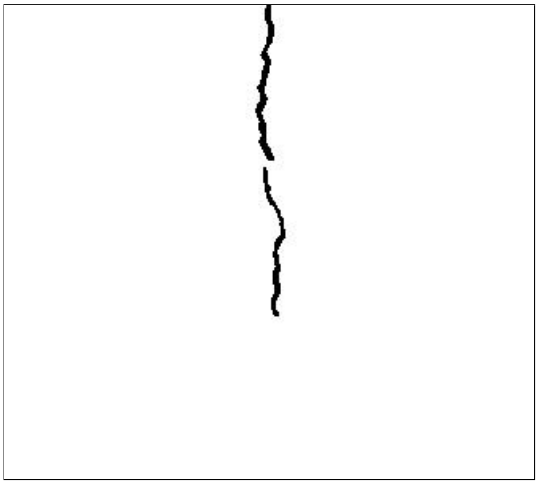
(A) Original image.



(B) Ground truth image.



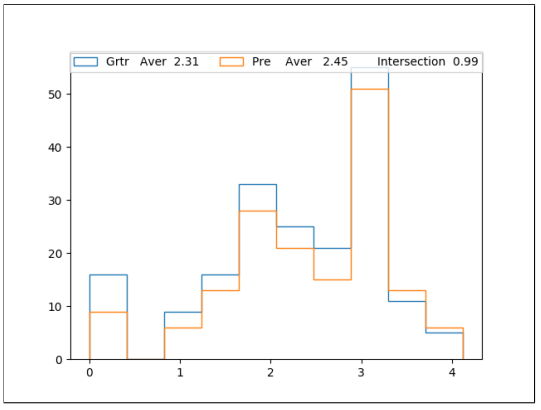
(C) Thinned skeletons of segmented cracks and thinned skeletons of ground truth cracks.



(D) Segmented image.

s	Pr	Re	F1
0	0.50	0.48	0.49
1	0.94	0.90	0.92
2	0.99	0.97	0.98
3	1.00	0.98	0.99
4	1.00	1.00	1.00
5	1.00	1.00	1.00

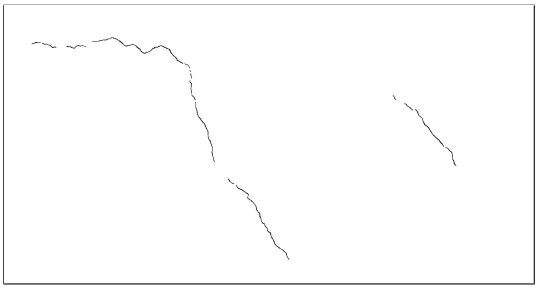
(E) PRC of crack length results.



(F) Crack width results.

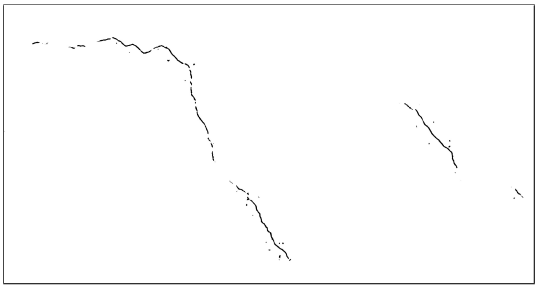
FIGURE 5.9: Experiment on a thinned skeleton and short crack.

(A) Original image.



(B) Ground truth image.

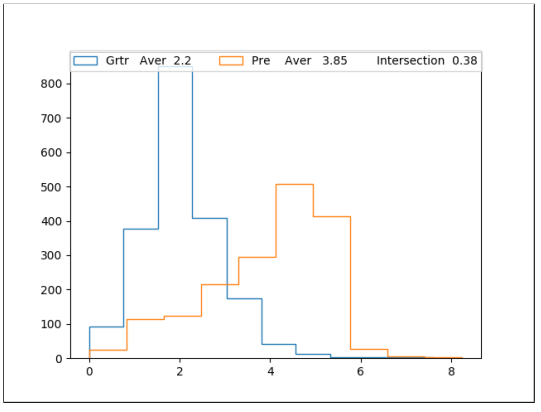
(C) Thinned skeletons of segmented cracks and thinned skeletons of ground truth cracks.



(D) Segmented image.

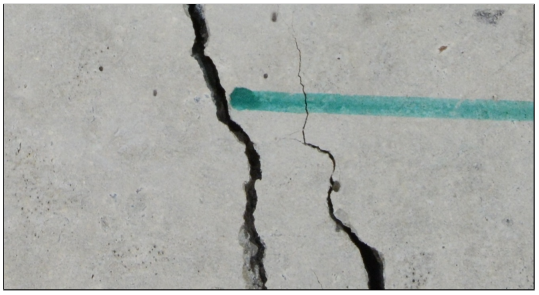
s	Pr	Re	F1
0	0.48	0.41	0.45
1	0.89	0.78	0.83
2	0.95	0.84	0.89
3	0.96	0.86	0.91
4	0.96	0.87	0.91
5	0.96	0.89	0.92

(E) PRC of crack length results.

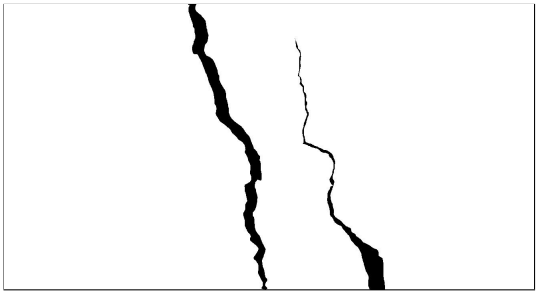


(F) Crack width results.

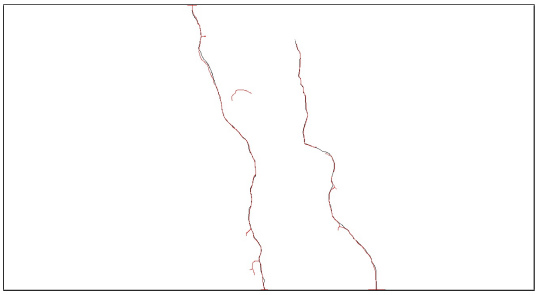
FIGURE 5.10: Experiment on small cracks in a low contrast image containing many bubbles.



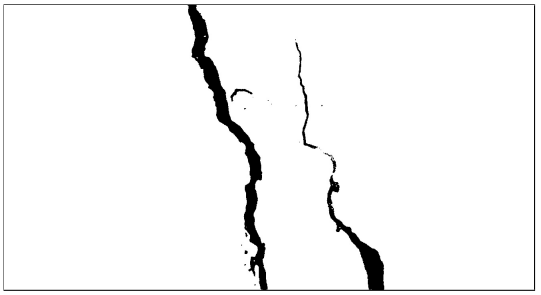
(A) Original image.



(B) Ground truth image.



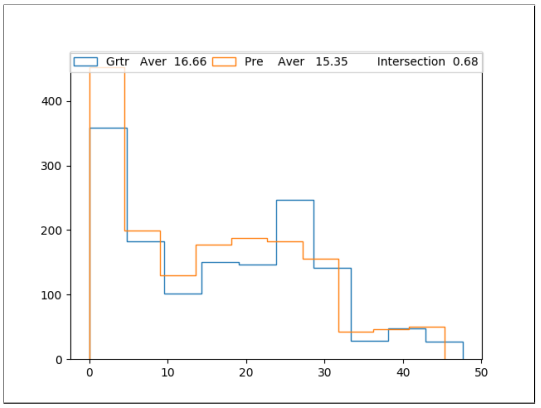
(C) Thinned skeletons of segmented cracks and thinned skeletons of ground truth cracks.



(D) Segmented image.

s	Pr	Re	F1
0	0.32	0.35	0.34
1	0.7	0.77	0.73
2	0.8	0.88	0.84
3	0.84	0.92	0.88
4	0.87	0.96	0.91
5	0.89	0.98	0.94

(E) PRC of crack length results.



(F) Crack width results.

FIGURE 5.11: Experiment on a range of cracks from thin to large in an image contains an ink stain.

new definitions of TP, FP, TN, and FN. These new definitions perform better for very thin objects such as the one-pixel wide thinned skeleton of a crack.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This dissertation has proposed, implemented, and estimated some methods that are valuable contributions to challenging problems in the field of crack data processing, including crack detection, segmentation, and crack point cloud upsampling. The results for crack detection, segmentation and crack point cloud upsampling improved the performance of the crack classification task.

The two-stage model using convolutional neural networks for crack detection and segmentation addressed two main challenges of crack processing. The first challenge is that road images are noisy and contain a variety of artifacts. Crack images are often captured from moving vehicles with the camera moving quickly, and under different conditions of weather and illumination. These conditions cause a diversity of crack images, and a low quality of road images. The second challenge is that crack datasets traditionally have imbalanced data, where the number of background pixels is many times larger than the number of crack pixels. Imbalanced datasets can lead to over-fitting of the classification model. The method proposed in this dissertation solves the above problems by using two separated CNNs in an architecture that allows for both crack detection at the sample level and crack segmentation at the pixel level. The two-stage system takes advantage of both detection and segmentation methods for the combined crack detection and segmentation problem

and is the basis of my contribution to this research field. I have shown that the first stage detection supports the second stage segmentation and vice versa. The first stage detects the crack at a course scale of 96x96 pixels, while the second stage segments the crack at the pixel level. So, the proposed method solves the problem of imbalance in crack datasets. The detection stage removes almost all of the noise and artifacts, and just keeps the area surrounding the cracks, so the proposed method also tackles the problem of noisy images.

One of the most significant contributions of this dissertation is a new approach for crack point cloud up-sampling. The crack point clouds play a critical role in crack damage calculation and crack development monitoring. However, high-resolution point clouds are costly to collect, so the idea of up-sampling the low resolution crack point clouds before investigating the crack is potentially useful. A method to combine 2D crack images and 2.5 crack point clouds in a single up-sampling architecture was proposed. An architecture based on GAN to up-sample the low-resolution point clouds from the combined data was proposed, with the ground truth point clouds being high resolution point clouds. There are two methods to combine images and point clouds that in this work and we denote as "pixel-pixel combination" and "feature-feature combination". In the first method, images and point clouds are merged together at the pixel and point level. In the second method, the features learnt from images are concatenated with the features learnt from point clouds at an intermediate stage of the architecture. The Chamfer distance and Hausdroff distances were used to evaluate the up-sampled results, and it was shown that the combination of images and point clouds improves the quality of the up-sampled point clouds. Experiments indicated that the combination of DoG features from the image with the point cloud data was the best combination for up-sampling crack point clouds. A method for crack point cloud classification was also implemented. The results illustrated that the higher resolution point clouds generated by our method can be classified better than lower resolution point clouds, and obtain a higher score than other approaches for crack classification from point cloud data.

This dissertation also implemented crack width and crack length calculation. From the segmented crack area, the thinned skeleton of the crack was found and used to compute the length of the crack. The distance from two edges of the segmented crack area is used to compute the width of the cracks. I have proposed a new evaluating metric that

is based on the traditional PRC metric by defining the true positive, false positive, true negative, and false negative samples (TP, FP, TN, FN samples) in an advantageous way. A reference distance **s** was defined, and then the distance from the detected pixel and the nearest ground truth pixel within **s** was compared to define the values of TP, FP, TN, and FN. This method avoids the problem of ground truth postional errors made by humans affecting the final evaluation results.

6.2 Future Work

Due to a lack data and time many interesting ideas related to this work have needed to be deferred to future research. Below are some of the more promising areas for future work:

Motion blur problem

As shown in Chapter 1 and Chapter 2, the road inspection system uses a special car, that captures images while moving quickly. So motion blur noise is often present in the road images. Previous work tried to solve the problem of motion blur [139, 140]. However, these works do not calculate the effect of motion blur on the final crack detection and segmentation results. The ideal method is a method that can both detect the cracks and estimate the error caused by motion blur noise.

Crack detection on other surfaces

For future work, the two-stage model for crack detection can be adapted to other surfaces such as concrete tunnels and bridges, or steel surfaces. More studies should be conducted to search for better CNN architectures for the proposed model. Further improvements to our model with the addition of 3D data to create a multi-stage model that is based on CNNs for crack detection and segmentation on three dimensional (3D) images is also possible. In addition, future work in this area will involve automating the calculation of the cracking index to avoid the human judgement component of current methods.

End to end network for detection and segmentation

I will continue to research and design an end-to-end (E2E) network for a hybrid task of crack detection and segmentation. The E2E network may contain two parts: the first part

uses a CNN backbone for crack feature extraction and this part is shared for all machine tasks, and the second part uses a CNN head to produce bounding boxes for crack detection and binary masks for segmentation. The E2E network is expected to save time for training and testing, and it can be used to improve tasks of online crack detection, segmentation, and classification.

Real time crack detection and segmentation

In Chapter 3, a method for crack detection and segmentation is proposed. This method improves the time in training and testing. However, for real time processing, the time consumption should be greatly decreased to allow this method greater utility for future applications.

Monitoring crack development using point cloud segmentation

Crack point clouds can be segmented into two types of point; crack points and non-crack points. Crack development may be monitored by periodically observing the development of crack areas or the increase in the number of crack points classified by systems such as those proposed in this dissertation. This process can help in assessing the severity level of cracks, and predict the time remaining until there is a need to repair the cracked surfaces.

Bibliography

[1] Mohamed Y Shahin and Starr D Kohn. Development of a pavement condition rating procedure for roads, streets, and parking lots. volume i. conditions rating procedure. Technical report, CONSTRUCTION ENGINEERING RESEARCH LAB (ARMY) CHAMPAIGN IL, 1979.

[2] Nhung Thi Hong Nguyen, Thanh Ha Le, Stuart Perry, and Thi Thuy Nguyen. Pavement crack detection using convolutional neural network. In *Proceedings of the Ninth International Symposium on Information and Communication Technology*, pages 251–256. ACM, 2018.

[3] Tien Sy Nguyen, Manuel Avila, and Stéphane Begot. Automatic detection and classification of defect on road pavement using anisotropy measure. In *2009 17th European Signal Processing Conference*, pages 617–621. IEEE, 2009.

[4] Sos Agaian, Ali Almuntashri, and AT Papagiannakis. An improved canny edge detection application for asphalt concrete. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pages 3683–3687. IEEE, 2009.

[5] Kelvin CP Wang, Qiang Li, and Weiguo Gong. Wavelet-based pavement distress image edge detection with a trous algorithm. *Transportation Research Record*, 2024 (1):73–81, 2007.

[6] M Salman, S Mathavan, K Kamal, and M Rahman. Pavemet crack detection using the gabor filter. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*, pages 2039–2044. IEEE, 2013.

[7] Qingquan Li and Xianglong Liu. Novel approach to pavement image segmentation based on neighboring difference histogram method. In *Image and Signal Processing, 2008. CISP'08. Congress on*, volume 2, pages 792–796. IEEE, 2008.

[8] Yong Hu and Chun-xia Zhao. A novel lbp based methods for pavement crack detection. *Journal of pattern Recognition research*, 5(1):140–147, 2010.

[9] Hoang-Nam Nguyen, Tai-Yan Kam, and Pi-Ying Cheng. An automatic approach for accurate edge detection of concrete crack utilizing 2d geometric features of crack. *Journal of Signal Processing Systems*, 77(3):221–240, 2014.

[10] John S Miller, William Y Bellinger, et al. Distress identification manual for the long-term pavement performance program. Technical report, United States. Federal Highway Administration. Office of Infrastructure Research and Development, 2014.

[11] Jyh-Dong Lin, Jyh-Tyng Yau, and Liang-Hao Hsiao. Correlation analysis between international roughness index (iri) and pavement distress by neural network. In *82nd Annual Meeting of the Transportation Research Board*, pages 12–16, 2003.

[12] Bonifacio R Prieto. Road surface structure monitoring and analysis using high precision gps mobile measurement systems (mms). pages 1732–1738. Citeseer, 2015.

[13] H Zakeri, Fereidoon Moghadas Nejad, and Ahmad Fahimifar. Image based techniques for crack detection, classification and quantification in asphalt pavement: a review. *Archives of Computational Methods in Engineering*, 24(4):935–977, 2017.

[14] Arun Mohan and Sumathi Poobal. Crack detection using image processing: A critical review and analysis. *Alexandria Engineering Journal*, 57(2):787–798, 2018.

[15] Henrique José Monteiro Oliveira. *Crack Detection and Characterization in Flexible Road Pavements using Digital Image Processing.* PhD thesis, PhD thesis, Universidade de Lisboa-Instituto Superior Técnico, 2013.

[16] American Association of State Highway and Transportation Officials. Standard practice for quantifying crack in asphalt pavement surface - aashto designation pp44-01, first publication. In *Standard Specifications for Transportation Materials and Methods of Sampling and Testing and Provisional Standards*, pages 194–197, 2001.

[17] American Association of State Highway and Transportation Officials. R 85-18 - quantifying cracks in asphalt pavement surfaces from collected pavement images utilizing automated methods. In *Standard Specifications for Transportation Materials and Methods of Sampling and Testing and Provisional Standards*, pages 1–2, 2018.

[18] American Association of State Highway and Transportation Officials. R 86-18 - collecting images of pavement surfaces for distress detection. In *Standard Specifications for Transportation Materials and Methods of Sampling and Testing and Provisional Standards*, pages 1–2, 2018.

[19] G Zhang, PA Vela, and I Brilakis. Automatic generation of as-built geometric civil infrastructure models from point cloud data. In *Computing in Civil and Building Engineering (2014)*, pages 406–413. 2014.

[20] Mohammad Ebrahim Mohammadi, Richard L Wood, and Christine E Wittich. Non-temporal point cloud analysis for surface damage in civil structures. *ISPRS International Journal of Geo-Information*, 8(12):527, 2019.

[21] Rafael C Gonzalez, Richard E Woods, et al. Digital image processing, 2002.

[22] Tom Michael Mitchell. *The discipline of machine learning*, volume 9. Carnegie Mellon University, School of Computer Science, Machine Learning . . . , 2006.

[23] Markus Svensén and Christopher M Bishop. Pattern recognition and machine learning, 2007.

[24] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[25] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[26] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[27] Jun Li. *A wavelet approach to edge detection*. PhD thesis, Citeseer, 2003.

[28] Lei Zhang, Fan Yang, Yimin Daniel Zhang, and Ying Julie Zhu. Road crack detection using deep convolutional neural network. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 3708–3712. IEEE, 2016.

[29] Zhun Fan, Yuming Wu, Jiewei Lu, and Wenji Li. Automatic pavement crack detection based on structured prediction with the convolutional neural network. *arXiv preprint arXiv:1802.02208*, 2018.

[30] Qin Zou, Yu Cao, Qingquan Li, Qingzhou Mao, and Song Wang. Cracktree: Automatic crack detection from pavement images. *Pattern Recognition Letters*, 33(3): 227–238, 2012.

[31] Djemel Ziou, Salvatore Tabbone, et al. Edge detection techniques-an overview. *Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii*, 8:537–559, 1998.

[32] John Canny. A computational approach to edge detection. In *Readings in Computer Vision*, pages 184–203. Elsevier, 1987.

[33] Huili Zhao, Guofeng Qin, and Xingjian Wang. Improvement of canny algorithm based on pavement edge detection. In *Image and Signal Processing (CISP), 2010 3rd International Congress on*, volume 2, pages 964–967. IEEE, 2010.

[34] Yi-Chang Tsai, Vivek Kaul, and Russell M Mersereau. Critical assessment of pavement distress segmentation methods. *Journal of transportation engineering*, 136(1): 11, 2010.

[35] Turgut Aydin, Yucel Yemez, Emin Anarim, and Bulent Sankur. Multidirectional and multiscale edge detection via m-band wavelet transform. *IEEE Transactions on image processing*, 5(9):1370–1377, 1996.

[36] Zhen Zhang, Siliang Ma, Hui Liu, and Yuexin Gong. An edge detection approach based on directional wavelet transform. *Computers & Mathematics with Applications*, 57(8):1265–1271, 2009.

[37] Piotr Porwik and Agnieszka Lisowska. The haar-wavelet transform in digital image processing: its status and achievements. *Machine graphics and vision*, 13(1/2):79–98, 2004.

[38] A Cuhadar, K Shalaby, and S Tasdoken. Automatic segmentation of pavement condition data using wavelet transform. In *Electrical and Computer Engineering, 2002. IEEE CCECE 2002. Canadian Conference on*, volume 2, pages 1009–1014. IEEE, 2002.

[39] Stephane G Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE transactions on pattern analysis and machine intelligence*, 11 (7):674–693, 1989.

[40] Javier R Movellan. Tutorial on gabor filters. *Open Source Document*, 2002.

[41] Lia Ma, Yunhong Wang, Tieniu Tan, et al. Iris recognition based on multichannel gabor filtering. In *Proc. Fifth Asian Conf. Computer Vision*, volume 1, pages 279–283, 2002.

[42] Wai Kin Kong, David Zhang, and Wenxin Li. Palmprint feature extraction using 2-d gabor filters. *Pattern recognition*, 36(10):2339–2347, 2003.

[43] Khagswar Bhoi and Dipesh Kumar Solanki. *Texture Segmentation Using Optimal Gabor Filter*. PhD thesis, 2011.

[44] Eduardo Zalama, Jaime Gómez-García-Bermejo, Roberto Medina, and José Llamas. Road crack detection using visual features extracted by gabor filters. *Computer-Aided Civil and Infrastructure Engineering*, 29(5):342–358, 2014.

[45] Derek Bradley and Gerhard Roth. Adaptive thresholding using the integral image. *Journal of graphics tools*, 12(2):13–21, 2007.

[46] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[47] Rui Fan, Mohammud Junaid Bocus, Yilong Zhu, Jianhao Jiao, Li Wang, Fulong Ma, Shanshan Cheng, and Ming Liu. Road crack detection using deep convolutional neural network and adaptive thresholding. *arXiv preprint arXiv:1904.08582*, 2019.

[48] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[49] Jagat Narain Kapur, Prasanna K Sahoo, and Andrew KC Wong. A new method for gray-level picture thresholding using the entropy of the histogram. *Computer vision, graphics, and image processing*, 29(3):273–285, 1985.

[50] HN Nguyen, TY Kam, and PY Cheng. Automatic crack detection from 2d images using a crack measure-based b-spline level set model. *Multidimensional Systems and Signal Processing*, 29(1):213–244, 2018.

[51] Roberto Medina, José Llamas, Eduardo Zalama, and Jaime Gómez-García-Bermejo. Enhanced automatic detection of road surface cracks by combining 2d/3d image processing techniques. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 778–782. IEEE, 2014.

[52] Vishal Mandal, Lan Uong, and Yaw Adu-Gyamfi. Automated road crack detection using deep convolutional neural networks. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 5212–5215. IEEE, 2018.

[53] BV Sobol, AN Soloviev, PV Vasiliev, and LA Podkolzina. Deep convolution neural network model in problem of crack segmentation on asphalt images. *Vestnik of Don State Technical University*, 19(1), 2019.

[54] Hiroya Maeda, Yoshihide Sekimoto, Toshikazu Seto, Takehiro Kashiyama, and Hiroshi Omata. Road damage detection using deep neural networks with images captured through a smartphone. *arXiv preprint arXiv:1801.09454*, 2018.

[55] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[56] Yoav Freund, Robert Schapire, and Naoki Abe. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

[57] Yong Shi, Limeng Cui, Zhiquan Qi, Fan Meng, and Zhensong Chen. Automatic road crack detection using random structured forests. *IEEE Transactions on Intelligent Transportation Systems*, 17(12):3434–3445, 2016.

[58] Sylvie Chambon and Jean-Marc Moliard. Automatic road pavement assessment with image processing: Review and comparison. *International Journal of Geophysics*, 2011, 2011.

[59] Allen Zhang, Kelvin CP Wang, Baoxian Li, Enhui Yang, Xianxing Dai, Yi Peng, Yue Fei, Yang Liu, Joshua Q Li, and Cheng Chen. Automated pixel-level pavement crack detection on 3d asphalt surfaces using a deep-learning network. *Computer-Aided Civil and Infrastructure Engineering*, 32(10):805–819, 2017.

[60] Nhung Hong Thi Nguyen, Stuart Perry, Don Bone, Ha Thanh Le, and Thuy Thi Nguyen. Two-stage convolutional neural network for road crack detection and segmentation. *Expert Systems with Applications*, 186:115718, 2021.

[61] H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[62] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[63] Laha Ale, Ning Zhang, and Longzhuang Li. Road damage detection using retinanet. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 5197–5200. IEEE, 2018.

[64] Somin Park, Seongdeok Bang, Hongjo Kim, and Hyoungkwan Kim. Patch-based crack detection in black box images using convolutional neural networks. *Journal of Computing in Civil Engineering*, 33(3):04019017, 2019.

[65] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[66] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[67] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[68] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.

[69] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[70] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[71] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[72] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[73] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[74] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[75] Joseph Bullock, Carolina Cuesta-Lázaro, and Arnau Quera-Bofarull. Xnet: a convolutional neural network (cnn) implementation for medical x-ray image segmentation suitable for small datasets. In *Medical Imaging 2019: Biomedical Applications in*

Molecular, Structural, and Functional Imaging, volume 10953, page 109531Z. International Society for Optics and Photonics, 2019.

[76] Yahui Liu, Jian Yao, Xiaohu Lu, Renping Xie, and Li Li. Deepcrack: A deep hierarchical feature learning architecture for crack segmentation. *Neurocomputing*, 338:139–153, 2019.

[77] Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. *IEEE transactions on pattern analysis and machine intelligence*, 35(6):1397–1409, 2012.

[78] Cuong Nguyen Kim, Kei Kawamura, Hideaki Nakamura, and Amir Tarighat. Automatic crack detection for concrete infrastructures using image processing and deep learning. *MS&E*, 829(1):012027, 2020.

[79] FuTao Ni, Jian Zhang, and ZhiQiang Chen. Zernike-moment measurement of thin-crack width in images enabled by dual-scale deep learning. *Computer-Aided Civil and Infrastructure Engineering*, 34(5):367–384, 2019.

[80] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[81] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[82] TY Zhang and Ching Y. Suen. A fast parallel algorithm for thinning digital patterns. *Communications of the ACM*, 27(3):236–239, 1984.

[83] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2018.

[84] Ruihui Li, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-gan: a point cloud upsampling adversarial network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7203–7212, 2019.

[85] Huikai Wu, Junge Zhang, and Kaiqi Huang. Point cloud super resolution with adversarial residual graph networks. *arXiv preprint arXiv:1908.02111*, 2019.

[86] Bisheng Yang, Ronggang Huang, Zhen Dong, Yufu Zang, and Jianping Li. Two-step adaptive extraction method for ground points and breaklines from lidar point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 119:373–389, 2016.

[87] Ruihui Li, Xianzhi Li, Pheng-Ann Heng, and Chi-Wing Fu. Point cloud upsampling via disentangled refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 344–353, 2021.

[88] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Ec-net: an edge-aware point set consolidation network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 386–402, 2018.

[89] Wang Yifan, Shihao Wu, Hui Huang, Daniel Cohen-Or, and Olga Sorkine-Hornung. Patch-based progressive 3d point set upsampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5958–5967, 2019.

[90] Rajat Sharma, Tobias Schwandt, Christian Kunert, Steffen Urban, and Wolfgang Broll. Point cloud upsampling and normal estimation using deep learning for robust surface reconstruction. *arXiv preprint arXiv:2102.13391*, 2021.

[91] Hyukseong Kwon, Kyungnam Kim, and Jean Dolne. Improving 3d registration by upsampling of sparse point cloud through fusion with high-resolution 2d image. In *Unconventional and Indirect Imaging, Image Reconstruction, and Wavefront Sensing 2017*, volume 10410, page 104100I. International Society for Optics and Photonics, 2017.

[92] Hyukseong Kwon and Kyungnam Kim. Three dimensional model generation using heterogeneous 2d and 3d sensor fusion, April 7 2020. US Patent 10,614,579.

[93] Georg Krispel, Michael Opitz, Georg Waltner, Horst Possegger, and Horst Bischof. Fuseseg: Lidar point cloud segmentation fusing multi-modal data. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1874–1883, 2020.

[94] Shuo Yang, Min Xu, Haozhe Xie, Stuart Perry, and Jiahao Xia. Single-view 3d object reconstruction from shape priors in memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3152–3161, 2021.

[95] S Bozinovski and A Fulgosi. The influence of pattern similarity and transfer learning upon training of a base perceptron b2. In *Proceedings of Symposium Informatica*, pages 3–121, 1976.

[96] Yi-Chang James Tsai and Feng Li. Critical assessment of detecting asphalt pavement cracks under different lighting and low intensity contrast conditions using emerging 3d laser technology. *Journal of Transportation Engineering*, 138(5):649–656, 2012.

[97] Qingquan Li, Dejin Zhang, Qin Zou, and Hong Lin. 3d laser imaging and sparse points grouping for pavement crack detection. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 2036–2040. IEEE, 2017.

[98] Allen Zhang, Kelvin CP Wang, Yue Fei, Yang Liu, Siyu Tao, Cheng Chen, Joshua Q Li, and Baoxian Li. Deep learning–based fully automated pavement crack detection on 3d asphalt surfaces with an improved cracknet. *Journal of Computing in Civil Engineering*, 32(5):04018041, 2018.

[99] Yue Fei, Kelvin CP Wang, Allen Zhang, Cheng Chen, Joshua Q Li, Yang Liu, Guangwei Yang, and Baoxian Li. Pixel-level cracking detection on 3d asphalt pavement images through deep-learning-based cracknet-v. *IEEE Transactions on Intelligent Transportation Systems*, 21(1):273–284, 2019.

[100] Allen Zhang, Kelvin CP Wang, Yue Fei, Yang Liu, Cheng Chen, Guangwei Yang, Joshua Q Li, Enhui Yang, and Shi Qiu. Automated pixel-level pavement crack detection on 3d asphalt surfaces with a recurrent neural network. *Computer-Aided Civil and Infrastructure Engineering*, 34(3):213–229, 2019.

[101] Alessandro Giusti, Dan C Ciresan, Jonathan Masci, Luca M Gambardella, and Jurgen Schmidhuber. Fast image scanning with deep max-pooling convolutional neural networks. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 4034–4038. IEEE, 2013.

[102] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[103] Stanford University. *Convolutional Neural Networks for Visual Recognition*, 2017 (accessed August 3, 2018). URL `http://cs231n.github.io/convolutional-networks/#add`.

[104] Jason Brownlee. A gentle introduction to the rectified linear unit (relu). *Machine learning mastery*, 6, 2019.

[105] Inc. The MathWorks. *Image Segmenter, Image Processing and Computer Visions*. Natick, Massachusetts, United State, 2019. URL `https://www.mathworks.com/help/symbolic/`.

[106] Leif E Peterson. K-nearest neighbor. *Scholarpedia*, 4(2):1883, 2009.

[107] Brian W Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405 (2):442–451, 1975.

[108] Davide Chicco and Giuseppe Jurman. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, 21(1):1–13, 2020.

[109] Chenglong Jiang and Yichang James Tsai. Enhanced crack segmentation algorithm using 3d pavement data. *Journal of Computing in Civil Engineering*, 30(3):04015050, 2016.

[110] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[111] Mohammad R Jahanshahi and Sami F Masri. Adaptive vision-based crack detection using 3d scene reconstruction for condition assessment of structures. *Automation in Construction*, 22:567–576, 2012.

[112] Sizeng Zhao, Fei Kang, Junjie Li, and Chuanbo Ma. Structural health monitoring and inspection of dams based on uav photogrammetry with image 3d reconstruction. *Automation in Construction*, 130:103832, 2021.

[113] Ziyi Feng, Aimad El Issaoui, Matti Lehtomäki, Matias Ingman, Harri Kaartinen, Antero Kukko, Joona Savela, Hannu Hyyppä, and Juha Hyyppä. Pavement distress detection using terrestrial laser scanning point clouds–accuracy evaluation and algorithm comparison. *ISPRS Open Journal of Photogrammetry and Remote Sensing*, 3:100010, 2022.

[114] Xiangyang Xu and Hao Yang. Intelligent crack extraction and analysis for tunnel structures with terrestrial laser scanning measurement. *Advances in Mechanical Engineering*, 11(9):1687814019872650, 2019.

[115] Mianqing Zhong, Lichun Sui, Zhihua Wang, and Dongming Hu. Pavement crack detection from mobile laser scanning point clouds using a time grid. *Sensors*, 20 (15):4198, 2020.

[116] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.

[117] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global, 2010.

[118] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[119] FG Irwin et al. An isotropic 3x3 image gradient operator. *Presentation at Stanford AI Project*, 2014(02), 1968.

[120] Charles A Poynton. Rehabilitation of gamma. In *Human Vision and Electronic Imaging III*, volume 3299, pages 232–249. International Society for Optics and Photonics, 1998.

[121] David Marr and Ellen Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 207(1167):187–217, 1980.

[122] Tony Lindeberg. Image matching using generalized scale-space interest points. *Journal of mathematical Imaging and Vision*, 52(1):3–36, 2015.

[123] Ryan Dahl, Mohammad Norouzi, and Jonathon Shlens. Pixel recursive super resolution. In *Proceedings of the IEEE international conference on computer vision*, pages 5439–5448, 2017.

[124] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015.

[125] Xiaobin Hu, Wenqi Ren, John LaMaster, Xiaochun Cao, Xiaoming Li, Zechao Li, Bjoern Menze, and Wei Liu. Face super-resolution guided by 3d facial priors. In *European Conference on Computer Vision*, pages 763–780. Springer, 2020.

[126] Teofilo F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical computer science*, 38:293–306, 1985.

[127] Carsten Moenning and Neil A Dodgson. Fast marching farthest point sampling. Technical report, University of Cambridge, Computer Laboratory, 2003.

[128] Luis B Almeida. C1. 2 multilayer perceptrons. *Handbook of Neural Computation C*, 1, 1997.

[129] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2017.

[130] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017.

[131] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. MeshLab: an Open-Source Mesh Processing Tool.

In Vittorio Scarano, Rosario De Chiara, and Ugo Erra, editors, *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008. ISBN 978-3-905673-68-5. doi: 10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136.

[132] Henry Blumberg. Hausdorff's grundzüge der mengenlehre. *Bulletin of the American Mathematical Society*, 27(3):116–129, 1920.

[133] Matthew Berger, Joshua A Levine, Luis Gustavo Nonato, Gabriel Taubin, and Claudio T Silva. A benchmark for surface reconstruction. *ACM Transactions on Graphics (TOG)*, 32(2):1–17, 2013.

[134] Zhuangwei Jing, Haiyan Guan, Peiran Zhao, Dilong Li, Yongtao Yu, Yufu Zang, Hanyun Wang, and Jonathan Li. Multispectral lidar point cloud classification using se-pointnet++. *Remote Sensing*, 13(13):2516, 2021.

[135] Zicheng Guo and Richard W Hall. Fast fully parallel thinning algorithms. *CVGIP: Image Understanding*, 55(3):317–328, 1992.

[136] Charles R Dyer and Azriel Rosenfeld. Thinning algorithms for gray-scale pictures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (1):88–89, 1979.

[137] Qin Zou, Zheng Zhang, Qingquan Li, Xianbiao Qi, Qian Wang, and Song Wang. Deepcrack: Learning hierarchical convolutional features for crack detection. *IEEE Transactions on Image Processing*, 28(3):1498–1512, 2018.

[138] Jack E Bresenham. Incremental line compaction. *The Computer Journal*, 25(1): 116–120, 1982.

[139] Ronghua Fu, Hao Xu, Zijian Wang, Lei Shen, Maosen Cao, Tongwei Liu, and Drahomír Novák. Enhanced intelligent identification of concrete cracks using multi-layered image preprocessing-aided convolutional neural networks. *Sensors*, 20(7): 2021, 2020.

[140] Riccardo Chianese, Andy Nguyen, Vahidreza Gharehbaghi, Thiru Aravinthan, and Mohammad Noori. Influence of image noise on crack detection performance of deep convolutional neural networks. *arXiv preprint arXiv:2111.02079*, 2021.