Brunel University London

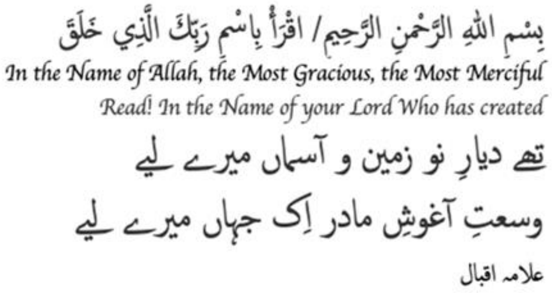
DOCTORAL THESIS

AI based Automated Detection, Measurement and Classification of Pits and Cracks on Rail Axles for Microscopic Visual Inspection in Non-Destructive Testing

Juvaria Fatima Syeda

A thesis submitted to Brunel University London for the degree of
Doctor of Philosophy in the Department of Electronic and Computer Engineering

February 2020



بِسْمِ اللّٰهِ الرَّحْمٰنِ الرَّحِيمِ/ اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ

In the Name of Allah, the Most Gracious, the Most Merciful

Read! In the Name of your Lord Who has created

تھے دیارِ نو زمین و آسماں میرے لیے

وسعتِ آغوشِ مادر اِک جہاں میرے لیے

علامہ اقبال

** Dedicated to my lovely mama and her research work on Hggs Boson

Who instilled love and passion for Science & Engineering in me

Declaration of Authorship

I, Juvaria Syeda, declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: ________________________________ DATE: ______________
(Signature of student)

Juvaria Syeda,

In Cambridge, February 2020

Abstract

In recent years, a lot of interest has been generated in structural health inspection automation. Although a number of computer-vision based crack detection techniques have been designed but further research is needed. Firstly, they are crack focused so there is a requirement to research for Pits. Secondly, these techniques have not been used in practical visual inspection as they are often developed under near-ideal conditions, not in an on-site environment, which contain more complexities such as variable shapes, glare, and loss of focus due to curved surfaces. Thirdly, there is a need to detect pits and pit-to-cracks of microscopic level for early assessments.

This research presents an automated inspection technique that is able to perform detection and measurements of on-surface micro-flaws caused by corrosion fatigue, including classification of pits and cracks by using artificial intelligence (AI) technology. It is an industry-driven application that can be used for both rail axles and pipelines.

It has been used for two purposes. First purpose, is to serve as part of a tool to enable early detection of cracks and assess the remaining service life of a corroded axle. The measurement of the damage depends on the detection of small microscopic cracks (around 0.1-0.3mm long) within the corroded area. This serves as an input value into the remaining-life software in the RAAI project. The results have been validated by the Polimi data showing that the implemented method is able to provide a prompt outcome including highlighting location, measuring and counting specific features, using on-site data. The second purpose, is to create a tool that can provide assistance to a corrosion assessment operator. The outcome includes highlighting, measuring and counting specific features plus it is able to classify between pits and cracks, using on-site data. It also implements industrial standard, considering the closeness of the pits, their size and density. Thus, the tool reduces the skill level requirement of an operator, as the algorithm sets a standard for the desired defects to be counted in quantitative measures. The applicability of the proposed method has been evaluated on images taken from the field. The evaluation results confirm the high adoptability of the proposed method for defect inspection in an on-site environment.

Firstly, new databases have been designed and created for the project that includes data handling, data gathering and data labelling. Database of such nature doesn't exist per existing research for the specific research problem. The data has been gathered by multiple sources that includes three site visits as well as data by performing bending testing at TWI, which expands the depth of the database. Pixel-wise labelled database of 165,888,000 data inputs, consisting of 115 microscopic pit images and 20 crack images has been created for this research task.

Secondly, advance unsupervised image segmentation based detection, localisation, measurement and assessment is built that produces better results than the state-of-the-art algorithm, for this specific industry-driven problem. Watershed and Morphological-based (shape) algorithms are implemented for analysing and assessment. This includes tasks such as finding shapes, detecting edges, removing noise, object detection, counting objects, segmentation, filtering and region analysis. It is able to show quantitative results such as number of flaws detected, along with the flaw's length, area, eccentricity, and their location shown on the image. They were then tested at different sites such as data from Ireland, Italy and UK; and validated with real data such as number of flaws, their average length and the longest flaw. Detailed indicative measures were applied to test and validate the system's performance such as 95.2% accuracy, 55% precision, f1 score 56%, probabilistic rand Index (PRI) 91.7%, CV is 42.8%, VOI as low as 41.08%, and Global consistency error (GCE) as low as 2.6%.

Thirdly, novel supervised machine learning methods are proposed. Especially, deep learning model is implemented as an image-wise classification model, pre-trained with AlexNet. The resultant outcome displays the class of the image to which it belongs. If the image has a pit, then the Pit model is able to pick it with 91.4% accuracy, and if the image has a crack, it is picked by the Crack model with a high accuracy of 98%. Another deep learning model implemented is a pixel-wise segmentation, based on a combination of two state-of the art models, UNet with VGG16. It shows performance with global accuracy of 93%. With a validation accuracy of 95% on validation training dataset and testing mean accuracy of 91%, mean IoU is 63.71% with a weighted IoU being 90.4%.

Lastly and most importantly, a Defect Detection System (DDS) has been designed, implemented, tested and verified in a real-industrial application. It is a practical solution with a ready-to-use and resource-efficient design. The setup requires a laptop, portable microscope and an automated scanner. The laptop is attached to the microscope which capture flaws with up to 0.08mm size length sensitivity and saves data in images/video format. The microscope is mounted on the scanner, that auto-rotates the camera circumferential as well as in axial axes, along the structure component being inspected. It is based on the combination of both supervised and unsupervised learning methods by merging their strengths. Detects, measures and localises by unsupervised image segmentation, so it doesn't need to perform lengthy pixel-wise labelling; and classifies the flaws by using deep learning so it doesn't need to hard-code complex computational values. It is simple yet efficient.

Acknowledgements

This thesis is the result of the love and support of numerous people in my life. I want to take this opportunity to express my gratefulness towards them & acknowledge the significance of their support.

I would like to express my deepest appreciation to my supervisor Dr Hongying Meng for his support, constant encouragement, and guidance. Especially his willingness to schedule untimely meetings to meet deadlines throughout the course of this research.

My most heartfelt gratitude goes to my wonderful industrial supervisor, Dr John Rudlin. His guidance and energy in working through all my ideas and his dedication to discuss all aspects of my research have made this work possible, not to forget our fun times with cricket banter.

This research was made possible by the sponsorship and support of EPSRC ICASE. The work was undertaken at NSIRC and TWI, Cambridge. It has been an honour for me to work within NDT section led by Ivan Pinson, who provided a friendly and comfortable environment for research. With special thanks to Dorothee Panggabean and Catherine Wells for their support.

No words can ever express my immense love towards my dearest parents, Syed Mohammad Saeeduddin and Syeda Rafiq Fatima, for their selfless devotion for my sister Zainub (Love my nieces) and I. I will always cherish the values you both instilled in me. Mama you are the most gentle soul I know, my buddy, my heart and soul, my world, I always look up to you…. You're my inspiration for life!

My gratitude is extended to all my dear friends with whom I enjoyed my research journey especially Jenny, Muntasir, Alenka, Shaheer, Jessica, Saadia, Fatima, Zuni, Nancy, Rabya. Thank you for all the mad times together.

Great thanks to all my in-laws, particularly AchiApa who supported me to take on this project. I will always remember the enthusiasm and prayers of AaliJanab and ApaJan, for the completion of my PhD.

Finally, I want to make a very special mention of immense appreciation for the love of my life, my hubby, Ghazali Syed. He has been by my side whenever I needed him and has helped me achieve what I have achieved today. He has been my rock, always showering me with unconditional kindness, encouragement and belief. Words are actually insufficient to express what I feel and wish for you. I am truly blessed to have you by my side as a friend and husband, all together in one loving soul. Just want you to know that I adore your love and support from the core of my heart and wish to be always with you my entire life ameen. Thank you for making me feel so special and making my life so colourful… to all the great times and rolling ☺

You all have made my dream come true...

Table of Contents

List of Figures

List of Tables

Nomenclature

Following are some of the main keywords and acronyms frequently used throughout the report.

2d/ 2D	Two dimensional
3d / 3D	Three dimensional
ADS	Automatic Detection System
ADDS	Automatic Defect Detection System
ANN	Artificial Neural Networks
API	American Petroleum Institute
CLAHE	Contrast Limited Adaptive Histogram Equalisation
CNN	Convolutional Neural Network
DDS	Defect Detection System
DL	Deep Learning
DT	Destructive Testing
EAC	Environmentally Assisted Cracking
FFS	Fitness-For-Service
GUI	Graphical user interface
IP	Image processing
ML	Machine Learning
MATLAB	Matrix Laboratory (Mathworks, Inc.)
MPI	Magnetic Particle inspection
NDE	Non-Destructive Evaluation
NDT	Non-destructive testing
NN	Neural Networks
PR	Pattern Recognition
SCC	Stress Corrosion Cracking
SCF	Stress Concentration Factor
SE	Structuring Element
SI	Structural integrity
SIF	Stress Intensity Factor
SOM	Self-organising map
USB	Universal serial bus
UT	Ultrasonic Testing

Keywords: image processing (IP), image segmentation, morphological, Watershed, machine learning (ML), ensemble classifiers, deep learning(DL), UNet model, VGG16 model, AlexNet model, artificial neural networks (ANN), pattern recognition (PR), on-surface flaws, microscopic defects, detection, localisation, measurements, classification, pixel-wise, labelled data, ground truth, pits, cracks, corrosion, structural integrity (SI), non-destructive testing (NDT), automated visual inspection.

Chapter 1
Introduction

This chapter presents the project motivation, aims & objectives and background of the project and also introduces some of the main concepts related to the project in a downward hierarchy at the end of this chapter. This project is about analysing the structural integrity of on-surface flaws like pits and cracks as a non-destructive testing method, achieved by developing and applying traditional image-processing techniques and then later on using more advanced methods like machine learning.

1.1 PROJECT MOTIVATION AS AN INDUSTRY NEED

In order to solve a problem, the first step towards it is to understand what the problem is. The research area lies within Structural Integrity (SI), which is a huge subject area in itself. There are many areas within it including inspection, monitoring and assessment; it also includes non-destructive testing (NDT) which include methods like Magnetic Particle Inspection (MPI), Eddy Current; then there are different kinds of flaws; and many diverse applications.

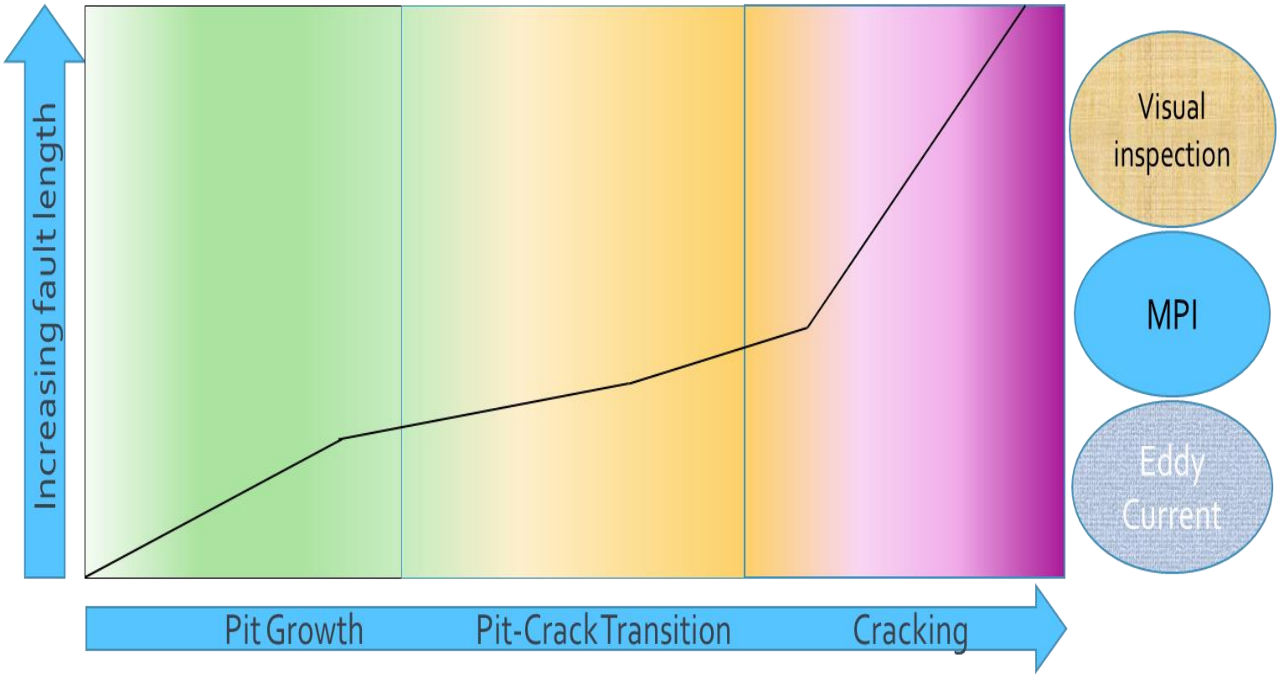


Figure 1-1-Target research area showing corrosion fatigue stages and inspection methods

The target area for this research is specifically to look into flaws such as cracks and pits, more specifically to look into the initial stages of the flaws. The first two stages as shown in figure 1.1, are the focus of the research. Some work has been done for crack stage but very limited work is available for pits and pit-to-crack stages. If the flaw can be detected at an early stage then it will be more useful in predicting the overall life of the structure and hence damages might be prevented.

The reliability of structures is an important factor in the construction phase as well as in the maintenance phase. For structural health inspection, a need has been felt for systems to become automated [1]. By developing an effective system to assist in structural reliability assessment, it is potentially possible to reduce the maintenance costs and still extend the useful life of a structure. Moreover, the condition of the structure health can be judged, in a more objective way by acquiring and processing relevant data. This will be achieved by developing image-processing and machine learning tools to better enable the visual corrosion assessments.

The European rail network is targeting a considerable expansion of passenger and freight traffic by 2020. In order to achieve this, increased reliability and availability of rolling stock is necessary whilst maintaining the same or a better level of safety. The axle life is a crucial part of both the safety and economic performance of the vehicles and the axle deteriorates through its lifetime by means of fatigue and corrosion mechanisms. Periodic inspection is used to ensure that these mechanisms have not compromised the axle safety however inspection that takes a vehicle out of service, impacts on the economic aspects of train operation. Hence a portable device needs to be considered that can be taken for on-site inspection.

For corrosion assessment, generally there are a number of measurements and classification to be done by following an industry standard which is API-579-1 FFS pitting assessment. This standard will be applied if pipelines are used but different analyses are required for high cycle fatigue such as rail axles. These assessments are restricted by the need to carry out manual assessments of the pits.

Table 1-1 Current Inspection methods with their detection limitations

Systems	Detection limit	Need
Industry standard (MPI, Eddy Current etc.)	1-2 mm length	0.5mm or less
Assessment	Manual visual assessment and pit counting	Automation

The existing system in the industry for identifying cracks during the initiation stage is less established than is generally believed. The current inspection methods for both pits and cracks are shown in table 1.1 along with minimum requirements for the proposed system. Currently for cracks, existing methods are MPI and Eddy Current which are able to pick flaws up to 1-2mm of length. While for pits, it is done by manually counting and measuring them as shown in figure 1.2. Hence, a system is needed that can detect flaws of size around 0.5mm or smaller and also needs to shift more towards automation from manual inspection.

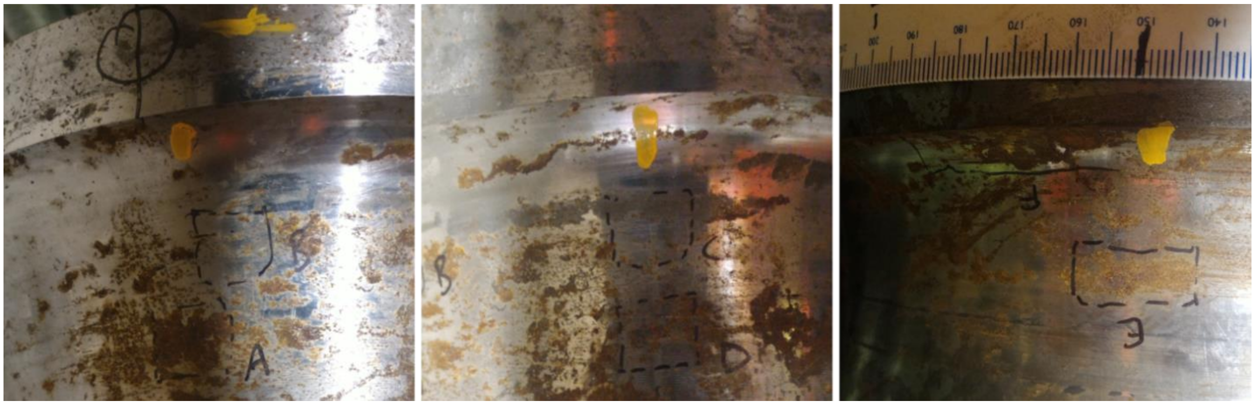


Figure 1-2 Example of manual visual assessment and pit counting

A new method must be devised that can effectively assist visual tests and improve detection rates. TWI has always been at the forefront of the research, development and application of many advanced and conventional NDT technologies. The reasons behind pursuing this project are as follows:

- Time: It will give results in less time as compared to the currently time-consuming method to count the number of pits and cracks individually by the skilled operator.
- Classification: It will be able to classify cracks and pits
- Objectivity: The assessment of microscopic pits is a subjective matter and the suggested method could present an objective approach for inspection and assessment.
- Consistent: Results will be in quantitative terns and therefore will be consistent as human error is possible when manual assessment is performed.
- Cost: It will be cost effective in relation to the expensive equipment used in other methods.
- On-site inspection: Portability of the equipment as it can be taken anywhere as compared to other methods which involve large instrument equipment.
- Skill: A relatively less skilled operator could inspect in comparison to some methods that require an absolute professional and skilled operator for the inspection task
- Assessible equipment: The equipment is easily accessible for operators to conduct the inspection, which is a microscopic 2D camera with LED lights.
- Scale: Detection of early stages of corrosion fatigue will be possible.

Data that will be collected could be used for two purposes. First purpose is to help create a tool that can provide assistance to a corrosion assessment operator. This means that as an output, Defect Detection System (DDS) should be able to display the processed image by highlighting the flaws visually plus save the flaw information, like flaw count, dimensions, area covered, in a file. This could be done by using image processing techniques. It will be ideal if it is able to distinguish between a crack and a pit flaw. This could be done by using machine learning or deep learning to classify between the flaws. Hence the system will be able to show where and what kind of flaw is present, on the investigated area.

The second purpose of the system is to estimate the remaining life of the axle given the presence of corrosion fatigue. It does this by detecting microscopically small cracks as shown in figure 1.3, which appear originating from corrosion pits in the corrosion fatigue process. Then the life is estimated from the average length of the cracks. This means that as an output, the defect detection system (DDS) result will have dimensions of all the cracks and its average. This will serve as an input value into the remaining-life software.

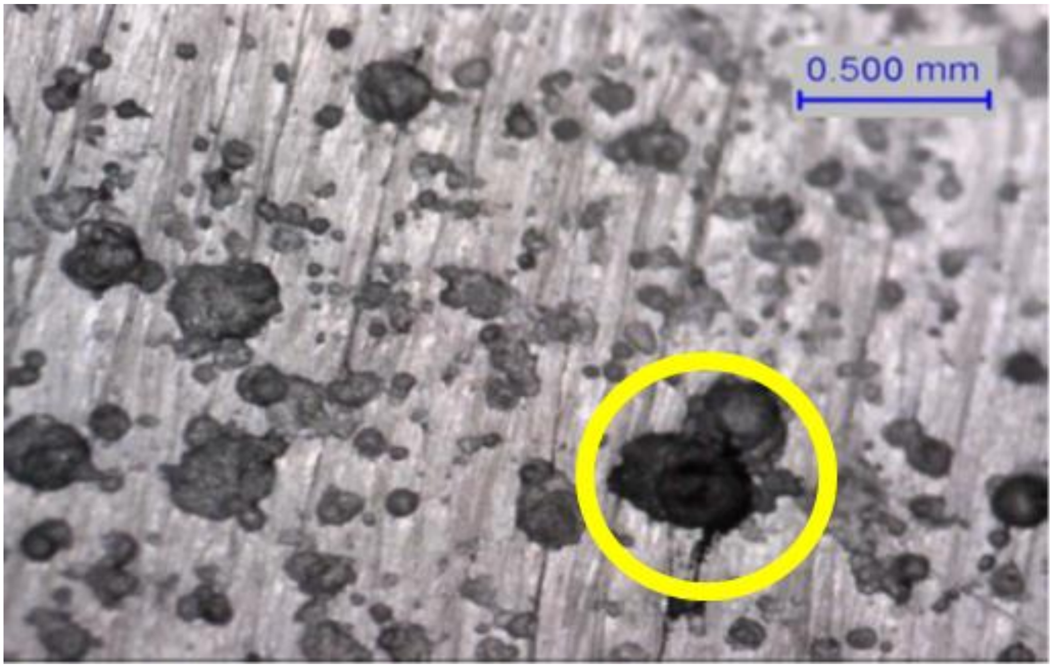


Figure 1-3 Example of a crack length within the size of the research need captured

For the DDS requirements to be filled, the first purpose will be followed as it will serve for both purposes. The second purpose just needs crack length information which is also needed for the first purpose.

1.2 Project Aim And Objectives

The aim of the project is to develop image acquisition and analysis techniques specifically for corrosion assessment by using an NDT method, and to apply them in useful situations for example for rail axle or pipeline inspection. The computer vision and image processing techniques that will be applied in this work are time-efficient and relevant in industrial applications. The rail axles suffer from high cycle fatigue during operation and also corrosion due to the environment, which modifies the fatigue properties. In order to identify when the cracking process is beginning; a microscopic image of the axle surface will be used.

The project will require taking images from real axles, which contain a mixture of corrosion and cracks and to develop an image analysis technique to automate the identification of pits and cracks within the corrosion. It is expected that the outcomes of this research will have a significant impact on automatic detection of such flaws using images.

Objective 1

Analysis of Images of Corrosion to facilitate assessment and classification for an operator.

- Acquire images
- Analyse images with image processing methods
- Create a map/profile of the image

Objective 2

Development of algorithm for detection of micro-cracks in pitted corroded areas (example rail-axle) in order to locate and measure the features. The information generated from the program could be analysed by other experts such as corrosion or fracture experts.

Objective 3

Implementation of Pattern Recognition such that input images with specific features such as pitting or cracking could be detected automatically with the software and classed according to their pattern.

- Use of machine learning
- May also apply deep learning methods

Deep learning methods will be explored as they are known to be effective for solving problems from unstructured data. They are particularly useful as they have an ability to automatically learn features from a large abstract data set.

1.3 BASIC FRAMEWORK

In simple terms, the main concept to solve this industrial problem is to collect data and run image processing on it to extract useful information required as shown in figure 1.4.

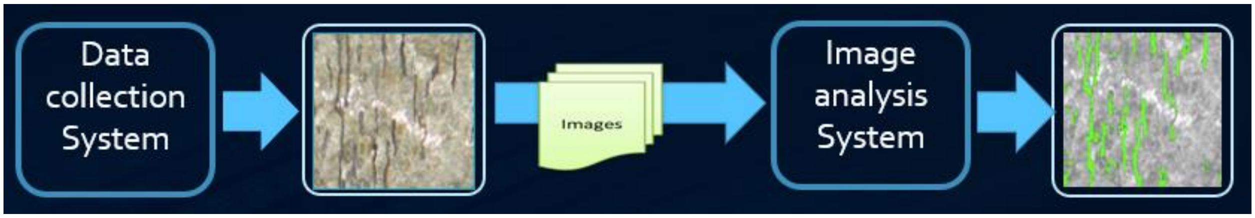
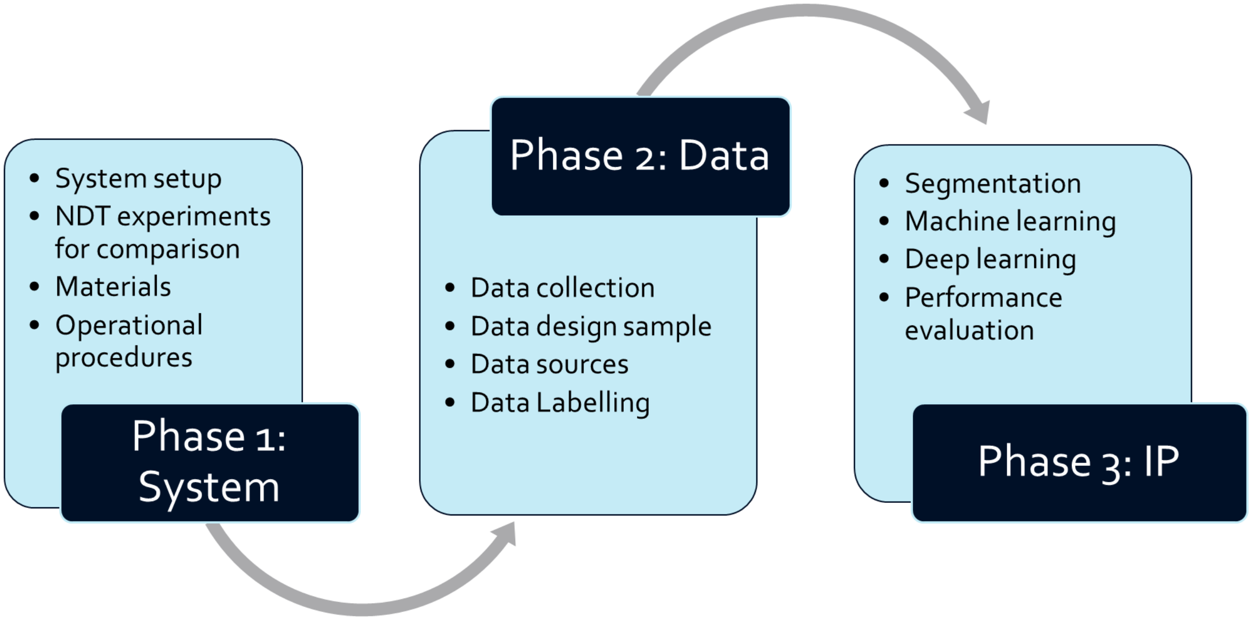


Figure 1-4 Initial basic framework of the project

1.4 RESEARCH PHASES

The research was divided into three major phases which are system setup, data and image processing illustrated in figure 1.5.



Phase 1: System
- System setup
- NDT experiments for comparison
- Materials
- Operational procedures

Phase 2: Data
- Data collection
- Data design sample
- Data sources
- Data Labelling

Phase 3: IP
- Segmentation
- Machine learning
- Deep learning
- Performance evaluation

Figure 1-5 Initial research design based on key project phases

1.5 PROJECT KEY TERMS

Structural health inspection plays an important role during the different phases of structures especially during the operational phase and construction phase for large scale constructions.

Hence a need has been felt for such systems to become automated [1]. It has been seen that a huge part of construction depends on proper management, and management requires specific skills and expertise in their particular area. This increases the cost of construction, maintenance and time consumed for structural health monitoring and inspection. Automation also becomes crucial for remote regions of structures. If a system could collect data and automatically analyse the data, the cost of maintenance could be greatly reduced as this will help in reducing the skilled labour requirement and the time consumption. This is the reason why in the recent times, automation in structural health inspection has gathered a lot of interest. Keywords related to the project are introduced in this section, while their detailed explanation will be given in the next chapter literature review.

1.5.1 Structural integrity defects

There are many different kinds of defects. This project will deal with the effects of corrosion damage like pits and cracks. Corrosion is typically explained in terms of chemical or electrochemical oxidizing process whereby a metal gradually wears away or is altered from its usually desired chemical composition. The chief concern in many applications is the capability of these closely related forms of corrosion to lead to accelerated failure of structural components by damage [2], or by acting as an initiation site for cracking, causing severe loss of functionality or even catastrophic failure [3]. Important studies have been carried out and some investigations made show that all crack nucleation initiate at corrosion pits [4] [5]. At the beginning of the corrosion process, pitting appears and further in the process cracking is generated. This suggests that understanding pits along with cracks is highly important for assessment as corrosion costs billions of pounds to the government each year [6].

Cracks may develop because of the cyclic stress that the structures are subject to and then these cracks will grow and may possibly lead to an in-service failure [7]. The reasons for the initiation of a crack are several and diverse. Initiation often occurs at discontinuities especially with the existence of corrosive agents like water, oxygen and other chemicals. Working at high-cyclic rates will cause components to suffer from cyclic fatigue, which increases the size of cracks and leads to failure. Mobile structures, including all sorts of vehicles will always likely to be highly stressed and will be at risk of fatigue failure. Cracking is an indication that an area is experiencing more stress than it can handle. Finding and assessing those stress areas will determine whether the parts need to be repaired or replaced.

1.5.2 Non-destructive testing

There are two different ways of testing the material structures; one is destructive and the other one is non-destructive. Nondestructive testing (NDT) is a wide group of analysis techniques used in science and industry to evaluate the properties of a material, component or system without causing damage while destructive testing (DT) includes those methods that cause the material some loss or damage and are usually done on a sample of the material to find out its properties such as tensile strength, hardness and fracture toughness. Testing is, mandatory for safety critical components in applications such as aerospace, oil and gas, pressure vessels and nuclear power, for the verification of weld quality or the monitoring of corrosion damage in service. The common general NDT methods being used in the industry include visual inspection, liquid penetrant, ultrasonic, eddy current, radiographic and magnetic-particle inspection.

The current NDT method for the assessment of corrosion on an accessible surface is by performing visual inspection which means manual counting of the pits. It is at best supported by a pit depth gauge or similar device. Although corrosion can be measured in the laboratory by optical methods to a high degree of accuracy but these measurements tend to be slow and require precision scanning of small areas [8]. For corrosion assessment, there are a number of measurements and classification to be done by following an industry standard API-579-1 FFS pitting assessment [9]. This standard is applied to the pipeline assessment but these are not related to high cycle fatigue such as the axles. Therefore, there is a need for an instrument that can be used directly for a quantitative on-site inspection for such a scenario.

1.5.3 API 579-1-2007 Standard

This is an American Petroleum Institute’s Recommended Practice 579, Fitness-For-Service (FFS). Part 6- Assessment of Pitting corrosion will be applied (for pipelines), as it relates to the specifics of pit assessments. The FFS assessment deals with the structural integrity evaluations. It also provides methodologies for conducting these assessments and guidelines to make decisions on whether the component should remain in run, go for repair or needs to be replaced. The assessment procedures in part 6 can be used particularly for evaluating pitting corrosion, which is the area of our research interest. When a structural component faces cyclic stress then such pits usually become cracks, so the pits need to be monitored from early on. There are three levels of assessment and for this project the main focus will be on level 1 and level 2, as these levels are difficult to automate at present.

There is a software tool developed by TWI called the Integrity-wise which uses the calculations and formulas on the basis of the API-579. The API standard for its calculations, needs the depth of the flaw as well. The pits that have been counted and measured manually are entered into the system to assess the pitting corrosion but this is limited to higher graded levels only and the measurements of all the pits must be entered manually.

1.5.4 Image processing (IP)

The rapid development of digital computers has led to a vast expansion of applications for computer vision systems. Processes that previously had to be done manually can now be automated using computers and cameras. Computer vision and image analysis have very high demands for processing power, something that still is a problem for real time applications. While the computers and vision algorithms are getting very advanced, to a level where they supersede human capacity for certain tasks, there are still tasks that have been deemed too difficult [10]. To be able to replicate the functionality of an human eye has been a challenge because the human visual cortex is a vastly advanced and highly trainable computer that has an amazing ability to analyse images, and the tasks it performs are in many cases difficult, if not impossible, to replicate digitally. It however is limited by a shorter endurance. This is true for most kinds of real-world applications; there is seldom a one best way to solve a computer vision problem, and it is therefore important to be able to quickly draft an algorithm and test its effectiveness.

There are two principal applications of image processing because of which there is a huge excitement for the field. One is for human perception and the other for machine perception. For example, improving pictorial information is for the humans and processing scene data for automatic machines [11].

Nowadays, image processing technology shows more and more power in many fields such as medical, industrial and commercial areas. In recent years, image processing technology is widely used in medical science to help understand and gather information from biomedical images of nature of human biological systems. Transformation from 2D to 3D images, automated feature finding and image comparison is the magnificent outcomes of the image processing technology. Moreover, image processing is also applied in textile industry to detect yarn parameters, the roughness of textile surface and the defect of textile, which is proved to be very effective.

It is a vast field dealing with manipulation and interpretation of the contents of digital images, and involves varied algorithms for many different purposes. Many of the different steps could be restoring the effects of corruptions during image acquisition; enhancing an image aid visualization and display; segmentation to identify regions and objects in an image on the basis of homogeneity criteria, such as colour, intensity or texture; and deriving properties and features of the regions that can be used to interpret the image.

1.5.5 Machine learning (ML)

Machine learning is a data-driven technique which is able to perform classification based on pattern recognition. It comprises of a series of methods, that enable computers to adopt certain desired behaviours based on the training of given example datasets instead of programming the computer with hard rules. This approach is beneficial especially for complex problems, because clear rules or equations are rarely available or known in those such cases.

Many of the techniques of image processing can serve as a step in the overall classification and analysis of the objects in an image, though pattern recognition techniques have been developed independently of image processing. A set of properties or features extracted using the appropriate Image Processing algorithms are the input data for the Pattern Recognition algorithms. But the Image Pre-processing stages will often be minimal as most enhancement and segmentation operations will change the values of the original image data and so affect the accuracy of the classification system. While these techniques have been used successfully over the years to a variety of applications, very little work has been described using defects especially pit images, and these data may offer new challenges to Pattern Recognition and Classification.

Neural networks process information in a similar way the human brain does, that is NN tries to replicate the functionality of a human brain [12]. The network is composed of a large number of highly interconnected processing elements that are called neurons [13], working in parallel to solve a specific problem. The neural networks resemble the brain mainly in two respects;

- Knowledge is acquired by the network through a learning process [14]
- Inter-neuron connection strengths known as synaptic weights are used to store the knowledge [15].

They are typically used in problems that may be couched in terms of classification, or forecasting. Some examples include image and speech recognition, textual character recognition, and domains of human expertise such as medical diagnosis, geological survey for oil and financial market indicator prediction. This type of problem also falls in the domain of classical artificial intelligence (AI) so that engineers and computer scientists see neural nets as offering a style of parallel distributed computing, thereby providing an alternative to the conventional algorithmic techniques that have dominated in machine intelligence [15].

Although many of the features extracted from the images can be used as input to Pattern Recognition systems, intelligent techniques such as Artificial Neural Networks are now being used for complex datasets where traditional techniques have been unable to provide satisfactory or reliable classifications [13]. Neural networks are often used for statistical analysis and data modelling, in which their role is perceived as an alternative to standard nonlinear regression or cluster analysis technique [16]

1.5.6 Deep learning (DL)

Deep learning is a branch of machine learning that attempts to model high level abstractions in data with multiple processing layers composed of multiple linear and non-linear transformations. The focus of deep architecture learning is to automatically discover such abstractions, from the lowest level features to the highest level concepts. The algorithms may be supervised or unsupervised and applications may include pattern analysis (unsupervised) and classification (supervised). Deep learning methods aim at learning feature hierarchies with features from higher levels of the hierarchy formed by the composition of lower level features. The ability to automatically learn powerful features will become increasingly important as the amount of data and range of applications to machine learning methods continues to grow [17]. There are different kinds of DL architectures such as deep neural networks, convolutional deep neural networks, deep belief networks and recurrent neural networks discussed in the next chapter in detail.

1.6 RESEARCH CHALLENGES

There are several difficult challenges to be addressed in the project, including:

- The corrosion assessment technique will need to follow the API 579 standard [2] including many complex calculations for the structural integrity due to pitting corrosion on the pipelines for different levels and grades

- In order to use image analysis for rail axle inspection, the main issue of background “noise” must be solved. This can be achieved through image processing methods (e.g. edge detection, image enhancement, etc.) and image pre-conditioning/filtering.

- In order to identify cracks within the corrosion on the axle surface, the area and its properties need to be detected and identified. It can be solved using advanced image analysis and pattern recognition methods (e.g. segmentation, deep learning, etc.).

- The information gathered in the project will be used to develop and propose design guidance for automatically rail axle visual inspection.

- Image acquisition especially to get clear images is difficult. Some of the factors which affect the images to be of good quality could be the lighting, glare, brightness, blurring, surface material, surface roughness, scratches and grinding marks.

- An appropriate size of a flaw needs to be decided on the basis of which the system will detect and count.

- To make ground truth images is a challenge in itself as it is a subjective area on the matter of which different experts opinions exist.

1.7 Thesis Contributions

The major contributions, as shown in figure 1.6, that have been made to the research area are as follows:

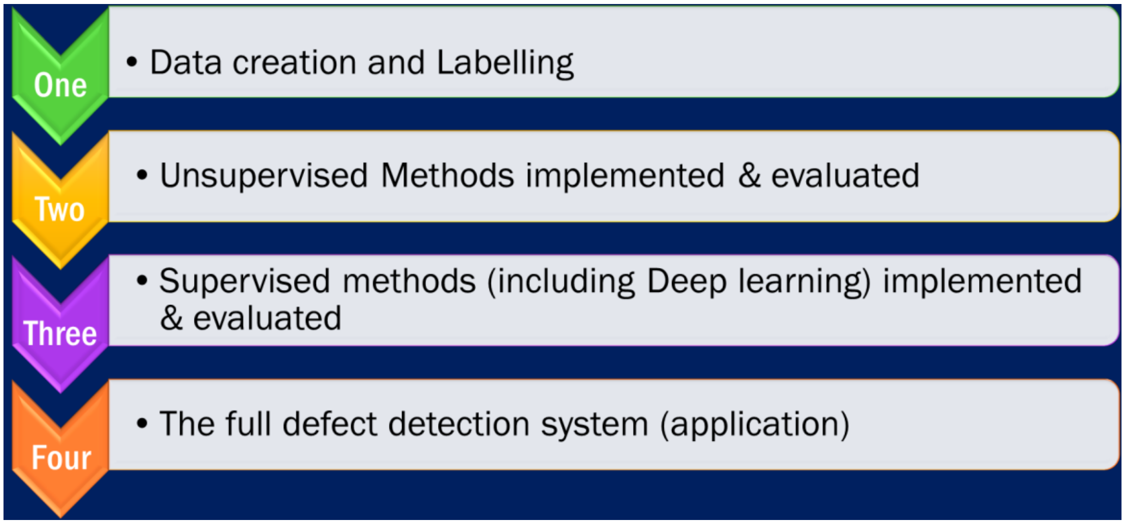

One
• Data creation and Labelling
Two
• Unsupervised Methods implemented & evaluated
Three
• Supervised methods (including Deep learning) implemented & evaluated
Four
• The full defect detection system (application)

Figure 1-6 Major contributions to the research area

1. **Data (Chapter 3 and 4) –** Data creation, collection (plus improvement) and labelling for pixel-wise pit data set of 141 million training data points (attained from 115 images) and

100 image-wise classification has been acquired. Practical requirements were collected and analysed, a protocol for data acquisition was designed and implemented. The databases have been built and ground truth labels have been made. No existing data set of such nature is present as per my knowledge.

2. **Unsupervised Learning (Chapter 5)** – Design and implementation of unsupervised learning method that produces better results than the state-of-the-art algorithm, for this specific industry-driven problem. Based on the database collected, advanced image segmentation methods have been investigated without using ground truth in the methods. Comparison has been done and performance was fully evaluated.

3. **Supervised Learning (Chapter 6 and 7)** – Deep learning methods implemented is a combination of two high-end models based on UNet and VGG16 which produces results that are better than its individual counterparts. Machine learning techniques have been investigated firstly, supervised learning has improved the performance because the labels have been applied in the training process. Further, deep learning methods have been explored in this particular application and the performance has been improved significantly.

4. **DDS Application (Chapter 8)** – The implementation of advanced AI based image processing techniques and novel machine learning techniques, as an industrial on-site application, in the research area, is a contribution on its own. As per my knowledge, such a system has not been built for detection and measurement of corrosion fatigue using NDT methods of microscopic visual inspection on rail axles.

1.8 Publication List

1. Beretta S, Sangalli F, Syeda J, Panggabean D, Rudlin J. RAAI Project: Life-prediction and prognostics for railway axles under corrosion-fatigue damage. Procedia Structural Integrity. 2017;4:64-70. doi:10.1016/j.prostr.2017.07.010.

Conference presentations:

2. Syeda J, Rudlin J. Image Processing and deep learning possibilities for detection of small cracks within corrosion. ESIS (European Structural Integrity Society) TC 24 DGZIP Conference 2017, Workshop on Integrity of railway structures at Wittensberge, Germany

3. Panggabean D, Syeda J, Beretta S, Archer R, Rudlin J. Data collection for corrosion fatigue life estimation. ESIS (European Structural Integrity Society) TC 24 DGZIP Conference 2017, Workshop on Integrity of railway structures at Wittensberge, Germany
4. Syeda J. Image Analysis for complex data in Non-destructive testing, NSIRC (National Structural Integrity Research Centre) conference 2017
5. Syeda J. Image processing and machine learning for detection of pits and cracks for rail axles using NDT. 57th Annual British Conference on Non-Destructive Testing (BINDT) 2018, East Midlands Conference Centre and Orchard Hotel, Nottingham, UK
6. Syeda J. Machine learning for pits and cracks in Non-destructive testing, NSIRC (National Structural Integrity Research Centre) conference 2019

Research engagements:

Presented the research work along with the improved project equipment used for data collection, to the following esteem dignitaries:

1. European Committee for welding of Rail Vehicles (ECWRV) tour
2. Sir Mark, Government Chief Scientific Adviser in the United Kingdom, (https://www.twi-global.com/news-events/news/2018-01-technology-in-action-as-ukri-chief-executive-professor-sir-mark-walport-visits-twi/)
3. Hayaatun Sillem, CEO of the Royal Academy of Engineering (RAE).
4. Regional Growth Fund (RGF) Tour visitors

1.9 Thesis Structure

This will give an overview of the following chapters. The structure of this report is as follows:

Chapter 2 This chapter serves as a literature-based background to understand the complete project problem from the NDT perspective, such as understanding the structural flaws that need to be inspected, their morphology, different stages of corrosion, existing non-destructive techniques being used in the industry such as MPI, Eddy Current and ultrasound along with their limitations to this specific industry-problem. It reviews existing literature by discussing all methods that are relevant for the later chapters such as image segmentation, machine learning and deep learning methods for baseline research.

Chapter 3 This chapter discusses different methods and materials to setup a data acquisition system. It starts by discussing reasons of selecting the chosen NDT technique verifying through experimental investigations such as MPI. Later on, discusses the selected materials and equipment, and the pre-requisites for the data collection such as the sample selection, camera selection, preparing the surface and then system setup. The design framework of the project is explained in depth.

Chapter 4 Image collection is the first and foremost step of the defect detection system. The correct sample collection and selection has been discussed in this chapter in detail. It also includes creating a ground truth database which is essential in order to analyse the performance of the algorithms. This is the first main contribution as there are no existing pixel-wise labelled database for a microscopic pit data from rail axles.

Chapter 5 This chapter discusses three main unsupervised image segmentation algorithms implemented for this research, such as watershed, Morphological and Gaussian based FCM clustering. For each of them, it shows individual steps along with visual outcomes and also presents the effects of variation of flaw-size parameter. Later on in the chapter, flaw detection results are shown along with measurements and then evaluation of these methods is implemented on the basis of multiple metrics to quantify their performances. In the end, a comparison to one of the state-of the art segmentation methods is made vs the findings of this research.

Chapter 6 This chapter provides an overview of the design steps that were necessary to set up supervised classification and begin training. This includes discussion of classifier algorithms, datasets used, model implementation, spatial and textural feature extractions such as local binary patterns, gradients and other parameters that were applied for the detection system in terms of machine learning. It then discusses experimental results based on the metrics for comparing and analysing different classifier's performance.

Chapter 7 This chapter provides an overview of the design steps that were necessary to set up the network and begin training for deep learning methods. It discusses three main supervised deep learning models implemented for this research such as Image-wise classification model, Pixel-wise UNet model and a novel model which is a combination of UNet_VGG16. For each of them it discusses their detailed architecture, implementation, hyper parameters such as learning rate, batch size, then displays visual results and conclusions.

Chapter 8 This chapter presents the Defect Detection system (DDS) along with its industrial impact (RAAI). It shows the workflow and application of the system including verified results from on-site data.

Chapter 9 This chapter summarises and concludes on some of the major points discussed in the previous chapters. This list includes data collection, creation and labelling and then all the experimental results produced by unsupervised image segmentation, supervised learning with extracted features such as local binary pattern, and deep learning performing pixel-wise segmentation as well as image-wise classification. In the end, it discusses possible improvements to enhance performance for future research.

Chapter 2
Project background through literature review

This chapter serves as a literature-based background to understand the complete project problem from the NDT perspective, such as understanding the structural flaws that need to be inspected, their morphology, different stages of corrosion, existing non-destructive techniques being used in the industry such as MPI, Eddy Current and ultrasound along with their limitations to this specific industry-problem. It also reviews existing literature by discussing all methods that are relevant for the later chapters such as image segmentation, machine learning and deep learning methods for baseline research.

2.1 INTRODUCTION TO THE PROJECT BACKGROUND

In this chapter, project background will be laid through literature research. In order to solve a problem, the first step is to understand what the problem is. Then look into existing methods and if they can't give satisfactory required results then new methodologies are proposed. In order to solve the problem in hand, different disciplinary aspects had to be considered. This includes:

- Understanding the flaw (section 2.2 - structural integrity)
- Explore general inspection methods in the industry (section 2.3 - non-destructive testing)
- Existing inspection method used for rail axles and pipelines application and its limitations for this specific problem
- Research state-of-the-art detection systems dealing with somewhat similar problem (section 2.4 – image anomaly detection systems)
- Investigate traditional image processing methodologies (section 2.5)
- Research different machine learning models (section 2.6)
- Understand the capability of neural networks and its working (section 2.7)
- Proposed inspection methodology (NDT + image processing + machine learning) based on the above literature review and investigate how to implement in rail axles and pipelines application for corrosion fatigue (section 2.8)
- After performing in-depth research in this chapter, project's conceptual framework and research design will be discussed in the next chapter (Chapter 3)

Many structures like building, pipelines, axles, containers may have problems due to the environment and the material they are made of. Due to this, failures like pits and cracks are expected which will affect the integrity of the structure. In order to be sure they are working fine, they need to be monitored and also assessed. Maintenance costs goes into this procedure to be able to prevent any failure or accidents. There are many methods by which they can be checked but they might be damaging to the material hence we use the non-destructive testing (NDT) methods in this project. There are many NDT methods, which can be selected on the basis of different things like surface detection, size of defect etc. But these methods either require the component to be placed in a container or they need to be taken apart from the rest of the model or require a highly skilled labour. For this project, we are interested in on-surface flaws and to be able to detect an early formation of pits and cracks with a flaw size less than a millimetre. Currently the situation here in United Kingdom is that the axles are withdrawn because of corrosion, sometimes they may have no cracks or low percentage of cracks. But still, once they detect a crack or pits they are put aside out-of-service. They are not utilised properly and are being wasted. In order to secure them back, a proper assessment has to be done in order to confirm their working condition.

Currently when the visual method is used they have to count each and every pit and crack manually, which is a very long and tedious job so if they find defects they don't do the further assessment and just take it out of service. If this process could be automated it will hugely improve on the speed and consistent accuracy of the inspection. If it is done with a simple microscopic camera then the issue of portability will also be solved also such equipment does not require highly skilled labour.

In this project, the on-surface defects will be monitored and assessed by using a simple 2D microscopic camera. It will count the number of defects and also tell the size of the defect in terms of length and area. There are many challenges in getting a good quality clear image because of the image acquisition restrictions like correct lighting, camera angle. The images may be shiny, blurry and have surface roughness. Image processing techniques will be used to try and solve these issues. Moreover, images may have other objects in the frame, which need to be ignored. Another challenging task is to be able to make the computer understand and differentiate between different defects like cracks and pits. Tasks like these could be achieved by using intelligent systems like neural networks, which are based on feature extraction algorithms. In order to analyse the images of the defects, the first step is to understand the defects properly especially the geometry and shape of the defects in a 2D image and maybe 3D for the later stage. For this project, pipelines for pits with industry standard and rail axles for pits and cracks are taken as the typical sample for experiments. The images used are: standard images from the industry of different grading; real images which include normal corrosive pitting method and also pin-needle method. The image processing techniques and neural networks methodology have several applications especially in the field of structural integrity like bridge, roads and on other materials as well.

Following sections dissect the problem through literature research to understand key factors involved to lay the foundation of a defect detection system (DDS). It starts with an explanation of cracks and pits, then explores different existing NDT methods, explain different image processing techniques and lastly investigates into the advance field of neural networks.

2.2 UNDERSTANDING THE FLAW

In order to analyse the defects by image processing techniques, the first step is to understand the flaw's working and morphology to be able to identify them correctly. The material of the structure is also vital as different materials respond differently due to their structural properties.

The choice of material of construction is a compromise between investment (expensive resistant material) and maintenance costs. Economic consequences are to repair or replace; over-design to avoid it; preventive (time-based) or predictive (condition-based) maintenance like painting; shutdown of equipment; contamination of product; loss of efficiency; loss of valuable products; damage to equipment adjacent to corrosion failure; and worst of all, loss of life. Also the life cycle costs are to be managed like maintenance, NDT, visual, condition monitoring, unplanned or planned shutdowns, repair costs, and warehousing of spare parts.

2.2.1 Material used: Steel

The major material used in structures in the industry is steel. There are two major types of steel, which are carbon and stainless. The stainless steel has a passive layer consisting of some percentage of chromium. So when it gets exposed to the corrosive environment, then pitting starts and that is why generally it has localised form of pitting. Carbon-steel has a lower corrosion resistance [18] as it doesn't have a passive layer so it is directly exposed to the corrosive environment. The sample material that we'll be dealing with in the project is carbon steel, which is used both in pipelines and rail axles.

2.2.2 Kinds of defects

An important part of maintenance has to do with the visual inspection of the structures. These structures can be affected by different kinds of defects typical of steel surfaces and structures, such as cracks and corrosion. These two kinds of defects are clear indicators of the condition of the metallic surface and thus are of great interest. In order to monitor and assess them they need to be detected [19]. If the crack remains undetected and unrepaired, it can grow to a size where it can cause sudden fracture. Therefore, care is needed to visually discover such occurrences in areas prone to high stress concentration [3].

Corrosion is the undesirable deterioration of materials of construction due to chemical or electro-chemical influences initiating from the interface material or environment [20]. There are two different kinds of corrosion associated with cracking corrosion fatigue, and stress corrosion cracking. This project deals with pitting due to corrosion and corrosion fatigue, where cracks initiate from pits

2.2.3 Types of corrosion

Corrosion has a huge part to play in the deterioration of structures and components and so it would be beneficial if we can have a solution where the components like rail axles do not need to be taken out of service and they can be monitored and assessed with NDT method applied.

Many research journals have discussed corrosion being the vital source of damages. The prime source of failure is environment-assisted fracture in the industries like in oil and gas pipelines, chemical plants, bridges, power plants, aircraft and civil engineering structures [21]. Many of the metallic structural failures and degradation are caused by corrosion fatigue [22]. The reason of axle failures in number of accidents has been corrosion fatigue [22] because of which the axles are taken out of service. Different types of corrosion could be better understood through a pictorial view, figure 2.1 to understand the varied classification of corrosion.

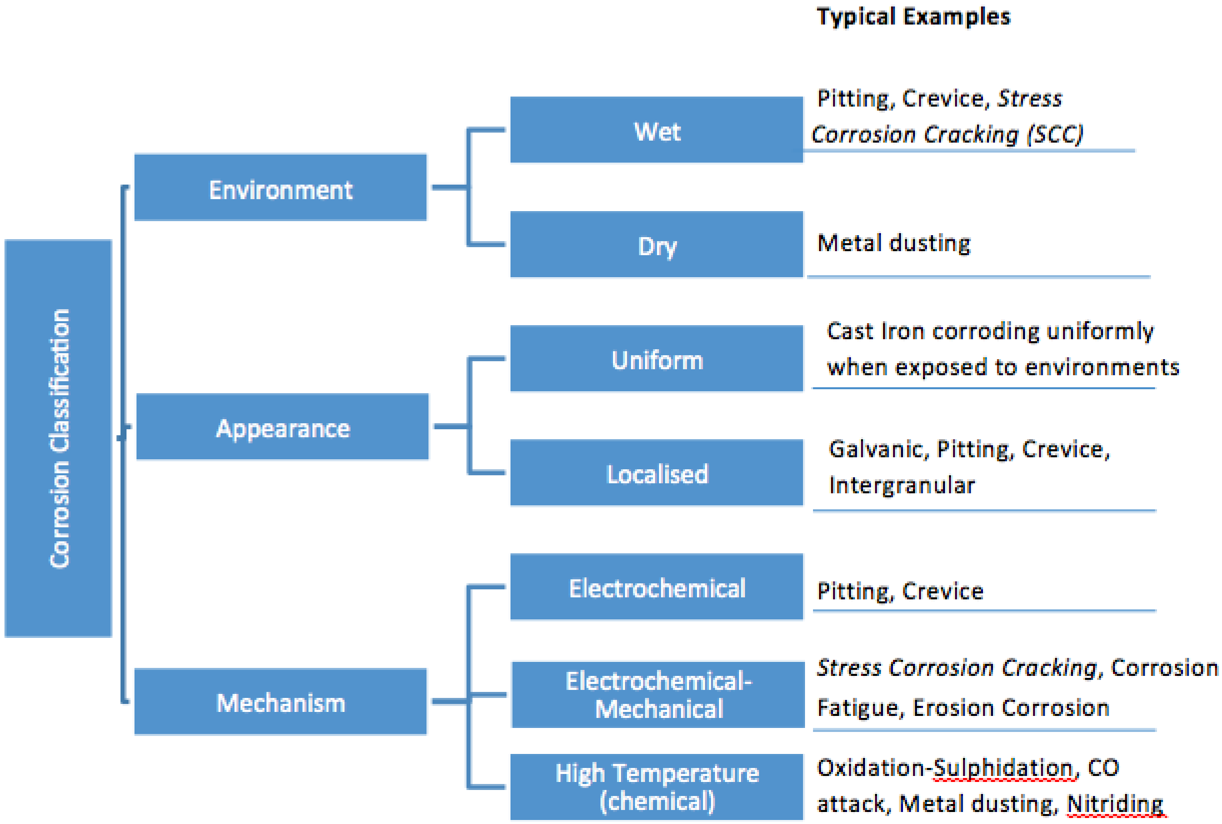

Typical Examples
Corrosion Classification
Environment
Wet
Pitting, Crevice, *Stress Corrosion Cracking (SCC)*
Dry
Metal dusting
Appearance
Uniform
Cast Iron corroding uniformly when exposed to environments
Localised
Galvanic, Pitting, Crevice, Intergranular
Mechanism
Electrochemical
Pitting, Crevice
Electrochemical-Mechanical
Stress Corrosion Cracking, Corrosion Fatigue, Erosion Corrosion
High Temperature (chemical)
Oxidation-Sulphidation, CO attack, Metal dusting, Nitriding

Figure 2-1 Corrosion Classification adopted from [20]

In this project, we'll be dealing with the following kinds of corrosion: pitting corrosion, corrosion fatigue and stress corrosion cracking. Pitting corrosion is formed due to the exposure of the corrosive environment; corrosion fatigue and SCC occur due to the mechanical mechanism added like cyclic stress in the rail axles or stress leading to SCC.

2.2.4 Stages of corrosion

Corrosion fatigue is a term used to describe cracking including both initiation and growth, in materials subject to the combined actions of a fluctuating (cyclic) stress and corrosive environment [23]. Different stages in fatigue cracking in air and in a corrosive environment are shown in figure 2.2 and it can be seen that the number of cycles required to cause failure is much less in a corrosive environment than in air. Although there will be no surface film breakdown as we are dealing with carbon steel. The major factors influencing corrosion fatigue are surface conditions, stress/load/cycle frequency parameters, environmental conditions as well as the material properties. It is assisted by the corrosive environment through the different stages; pit initiation, pit growth, the transition from pit to crack, crack growth and fracture.

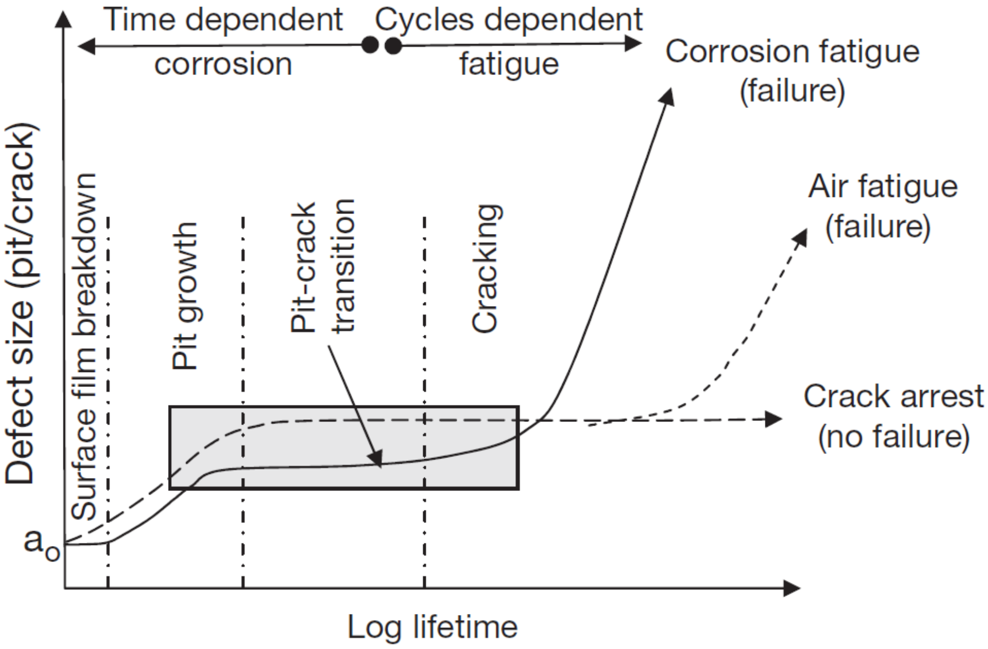


Figure 2-2 Stages in fatigue cracking in air and in a corrosive environment [23]

If a defect can be detected at an early stage [24] then it may be more useful in predicting the overall life of the structure and hence extreme catastrophic deterioration may be prevented. For this purpose, a lot of industry-based research is being done in the area of identifying the different stages of corrosion fatigue [8] as shown in figure 2.3 and predicting early stages of damage development.

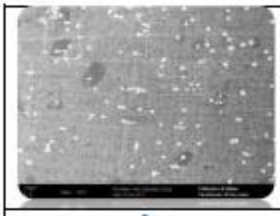
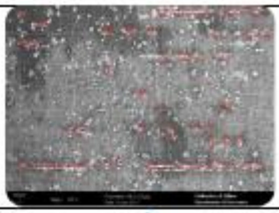
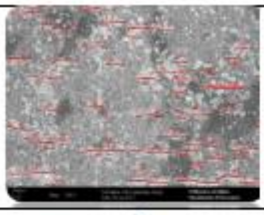
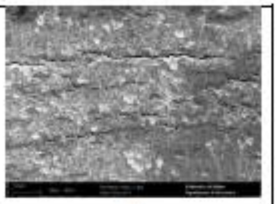
			
Test ML2 (10^5 cycles) (11%)	Test ML3 (2.10^5 cycles) (22%)	Test ML5 (4.10^5 cycles) (44%)	Test ML9 ($8x10^5$ cycles)(88%)
Pitting only	Formation of Microcracks	Coalescence of microcracks when depth exceeds 0.3mm	Growth of macrocracks detectable by conventional NDT

Figure 2-3 Four stages of crack growth in high cycle corrosion fatigue [8]

An important point that has been well established is that in materials undergoing pitting corrosion, all crack nucleation originates from pits [25]. At the beginning of the corrosion process, pitting appears and further in the process cracking is generated. The transition from pit to crack is the most significant stage in terms of predicting the damage process. The pit size is dependent on the electro-chemical and mechanical conditions and it varies with the increase in exposure time [26]. The transition from the pit to crack is initiated when two most important criterias are satisfied which are; the stress Intensity factor (SIF) and when the crack growth rate is greater than the pit growth rate [27]. A corrosion fatigue crack is generally straight lined [28] and not branched. Data from such type of experiments are shown in the form of an S-N curve where S is the cyclic stress range and N as the number of cycles to failure. By using such graphs, the lifetime of a structure under cyclic load may be predicted [20].

2.2.5 Morphology of the defect

By now we know what cracks and pits are and how they form but now we need to know the geometric features of the defects so that we can exploit those by using our processing filters. The morphology of the defects could be applied for the image processing as a region marker and for neural networks processing as a feature classifier. The reason why we need to understand and study the basics of corrosion and its morphology is so that the computer can apply correct processing steps accordingly and take efficient and effective steps to detect them.

The morphology of pitting may vary considerably [29], in terms of geometric shape [30] and depth-width ratio also known as the aspect ratio, as shown in figure 2.4 and this is why it is so challenging to detect them. In many studies, the pits have been represented by a semi-elliptical shape [31]. They propagate with complex directionality, as a function of local stress, microstructure and chemistry. They can act as local stress raisers in a material and hence can participate in the failure of a material through fatigue and/or stress corrosion cracking (SCC) mechanisms.

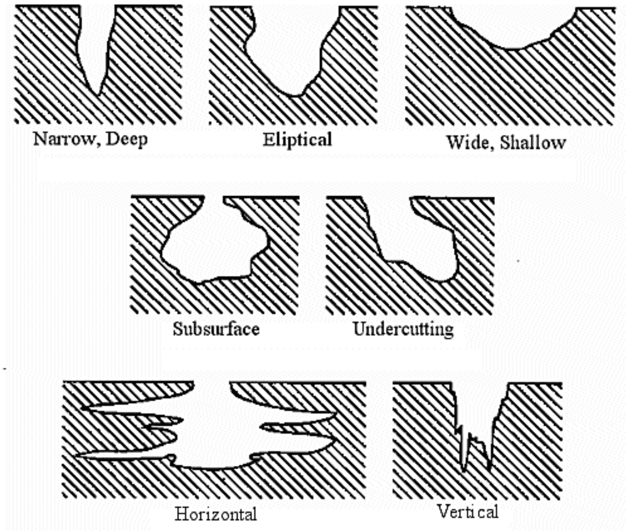


Figure 2-4 Varied Pit Shapes according to ASTM assessment Guide [32]

Experiments done by [33], show that the shape of the pits growth is nearly hemispherical until they reach the pit-to-crack transition stage and the cracks seem to be activated due to a secondary pit or crack at the bottom of the primary pit.

The properties given to vessel-like structures are very closely related to the properties of a crack [34]. While carrying out image processing for crack detection on underground pipelines [35] define three basic properties of cracks: they branch like a tree; more or less have a constant width; and the intensity distribution of crack feature cross section looks like a specific Gaussian curve.

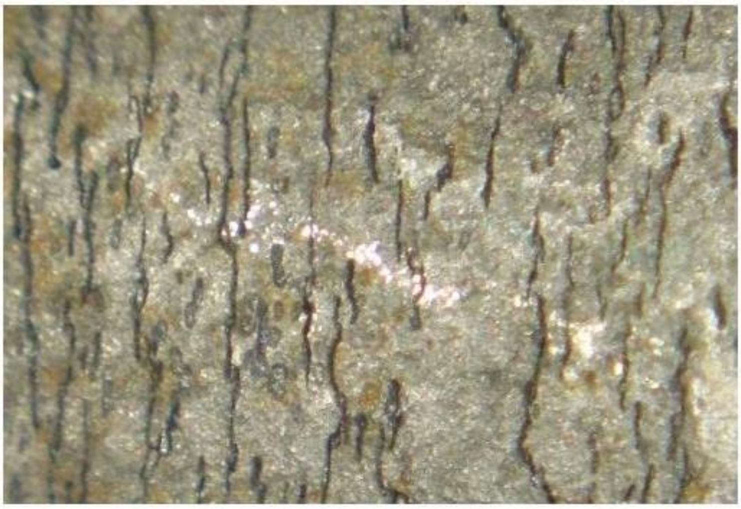


Figure 2-5 Typical example of micro-cracks in an image taken from an axle

For a component under-going cyclic stress like a rail axle, these cracks tend to be thin [36] vertical lines as they are usually along the rotational direction as seen in figure 2.5 above.

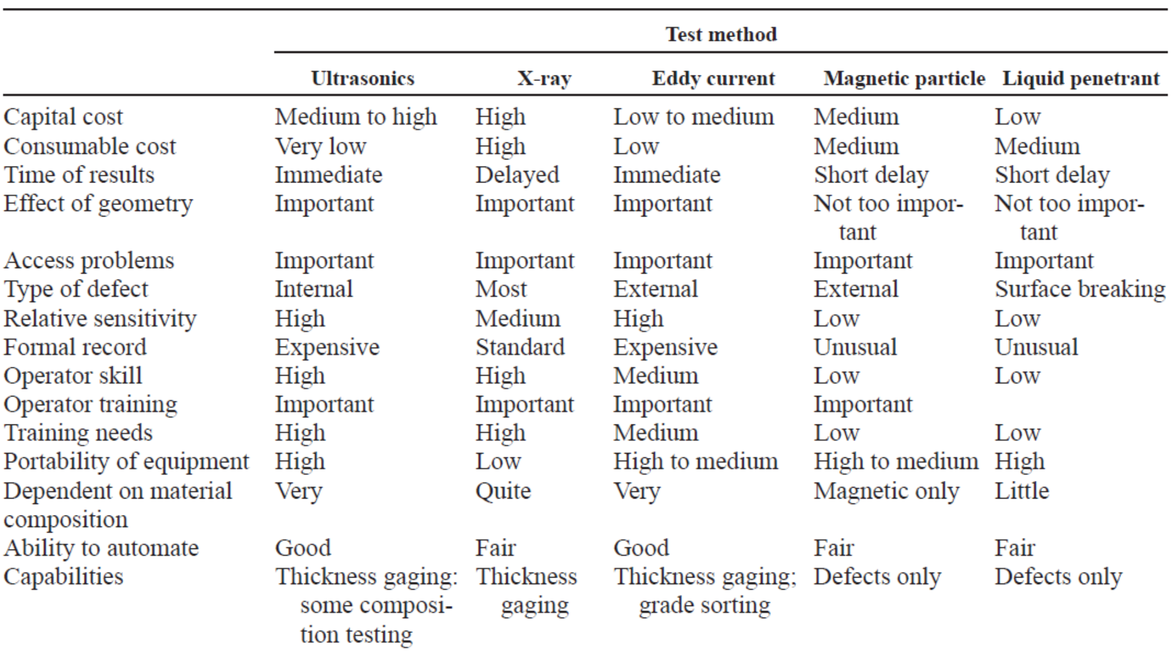
2.3 EXISTING INDUSTRIAL NDT METHODS & THEIR LIMITATIONS IN RESPECT TO PROJECT PROBLEM

The selection of the most appropriate NDT method, for a particular component (in this case, rail axles and pipelines) is based on a number of factors such as:

- Material of the structure and method of assembly (e.g welding, casting etc)
- Thickness of section
- Geometry of the component
- Type of defect to be detected
- Access
- Cost inspection equipment availability
- Operator skill level of inspection

Each sub-section discusses the method for general purpose and in the end of the paragraph also explains as to why it is not suitable for this project i.e discussing the limitation of the method in this case.

Nondestructive testing (NDT) does not permanently alter the article being inspected hence it is a highly valuable technique that can save both money and time in product evaluation, troubleshooting, and research. A brief summary of some of the advantages and disadvantages of the very basic methods for general purposes is shown in figure 2.6 below.



	Test method				
	Ultrasonics	**X-ray**	**Eddy current**	**Magnetic particle**	**Liquid penetrant**
Capital cost	Medium to high	High	Low to medium	Medium	Low
Consumable cost	Very low	High	Low	Medium	Medium
Time of results	Immediate	Delayed	Immediate	Short delay	Short delay
Effect of geometry	Important	Important	Important	Not too important	Not too important
Access problems	Important	Important	Important	Important	Important
Type of defect	Internal	Most	External	External	Surface breaking
Relative sensitivity	High	Medium	High	Low	Low
Formal record	Expensive	Standard	Expensive	Unusual	Unusual
Operator skill	High	High	Medium	Low	Low
Operator training	Important	Important	Important	Important	
Training needs	High	High	Medium	Low	Low
Portability of equipment	High	Low	High to medium	High to medium	High
Dependent on material composition	Very	Quite	Very	Magnetic only	Little
Ability to automate	Good	Fair	Good	Fair	Fair
Capabilities	Thickness gaging: some composition testing	Thickness gaging	Thickness gaging; grade sorting	Defects only	Defects only

Figure 2-6 Summary of basic Non-Destructive Testing Methods [37]

Descriptions of the features of common NDT methods including visual inspection, liquid penetrant, magnetic-particle, radiographic, ultrasonic and eddy current which are currently used in industry are discussed below. They are sometimes used in combinations to improve and verify the assessment. Research effort is being applied all the time to come up with more effective ways of achieving reliable and cost-effective inspections. In this project, most likely a combination of methods will be applied such as microscopy and MPI.

2.3.1 Visual Inspection

It is one of the most widely practiced and probably the first method of NDT. It checks for the surface condition, roughness, defects or other changes like dimensions. It is basically reliant on the inspector's eyesight, attention and judgement so if the defects are too small then they might be too difficult to detect. Also, it is extremely time-consuming to count all of defects and of course continuous viewing is restricted with frequent rests. Much of the success of this method is dependent on the surface condition and the lighting arrangements. It requires surface preparation like cleaning. Traditional methods usually quantify corrosion by visual comparison of the area under study with dot patterned charts or grading charts as shown in figure 2.7. The methods like MPI and/or Eddy currents are used when the defects are big enough like greater than 1-2mm to locate the flaw. Then visual inspection is performed for further assessment.

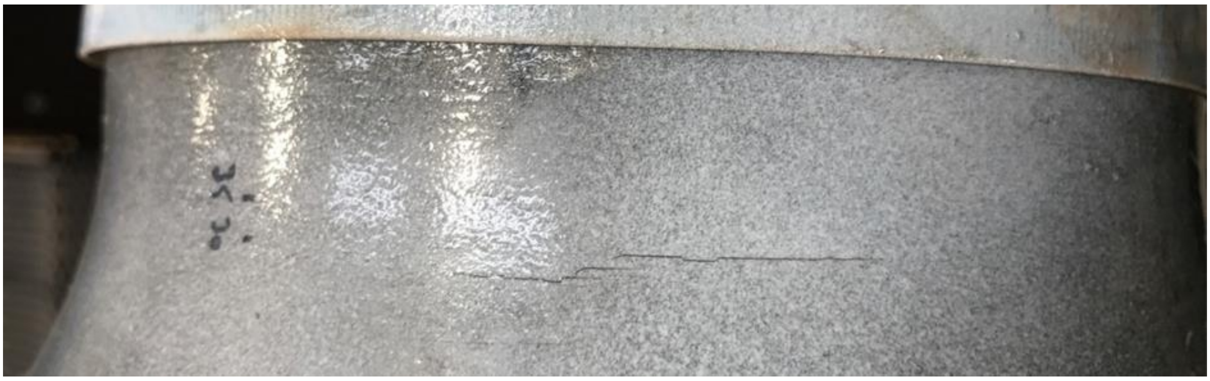


Figure 2-7 Visual inspection using a mobile camera

2.3.2 Liquid Penetrating testing (PT)

Another method for finding surface cracks and flaws is to use a penetrating dye as shown in figure 2.8. This is an old technology but it still works well. It can be used on a relatively smooth and non-porous material. Its use is confined to the detection of surface breaking discontinuities. This method requires proper surface cleaning before applying the penetrant and then cleaning preparation as the contaminants can mask the defects; it uses a considerable amount of consumables; post cleaning is essential to remove the chemicals; and requires very stringent safety procedures like the chemical handling precautions. It has a limited resolution for very fine cracks so it is not suitable in regards of the scale we are looking for in this project.

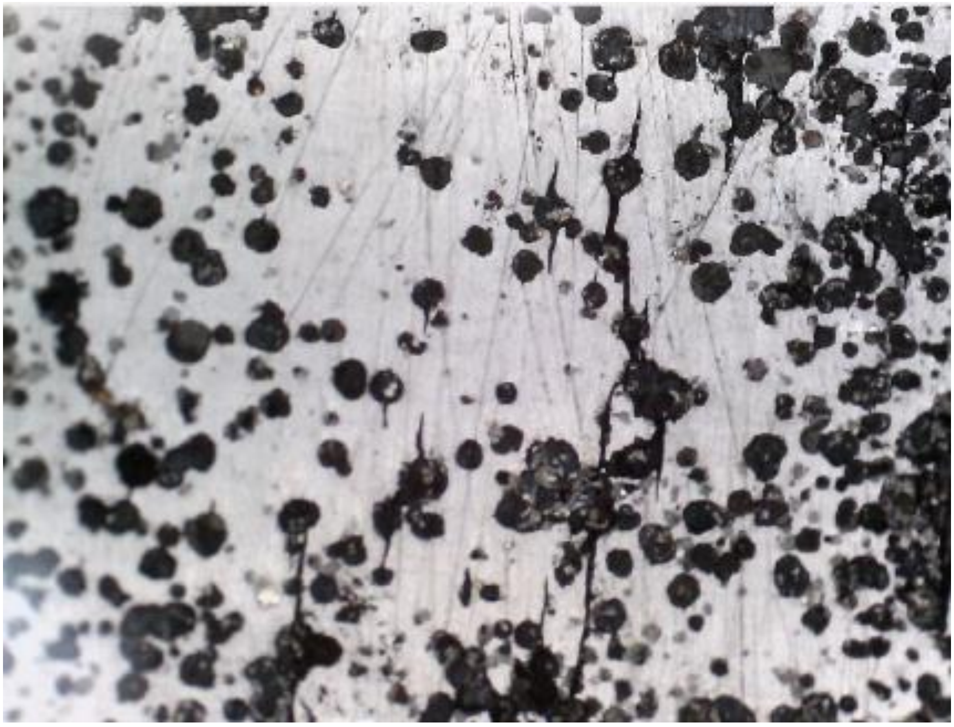


Figure 2-8 Liquid penetrating testing (PT) example

2.3.3 Magnetic Particle Inspection (MPI)

This method is the most widely used in industry. It is somewhat similar to the Liquid Penetrant method but is only applicable to magnetic materials. This method is used for detecting surface discontinuities as shown in figure 2.9. This testing method consists of establishing a magnetic field in the part to be tested, applying magnetic particles to the surface of the part, and examining the surface for accumulations of particles that are attracted to the discontinuities. Few points considered for this method are the surface must be cleaned thoroughly before the test; the method can only work on magnetic materials; it cannot detect deep internal flaws; proper alignment of the magnetic field and defect is critical [6]; large currents are required for very large parts; if very high currents are applied then it may cause damage to the component; paint and other non-magnetic coverings adversely affect the sensitivity; and the components will have to be demagnetised and cleaned later on. This method is insufficient for the size we are aiming for in this project as this can be done only when it is more than at least 1mm.

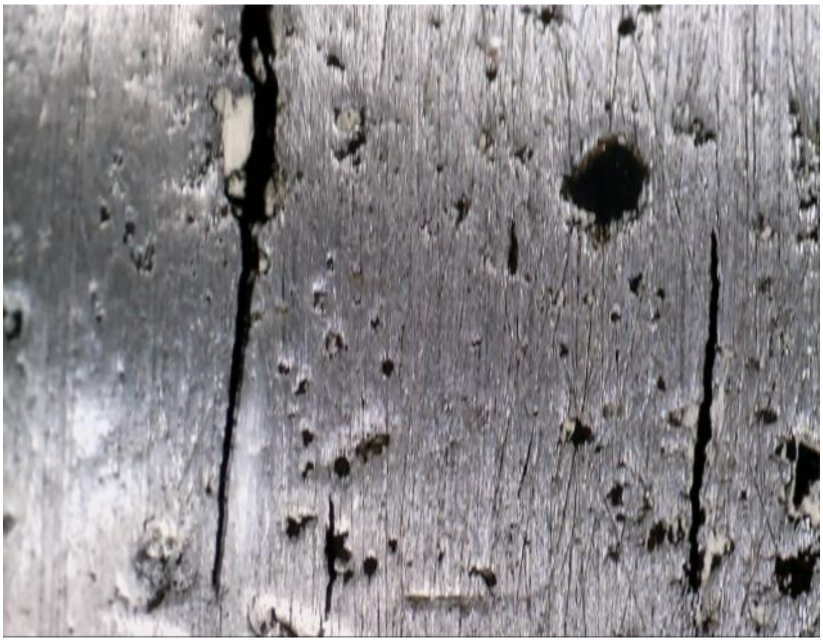
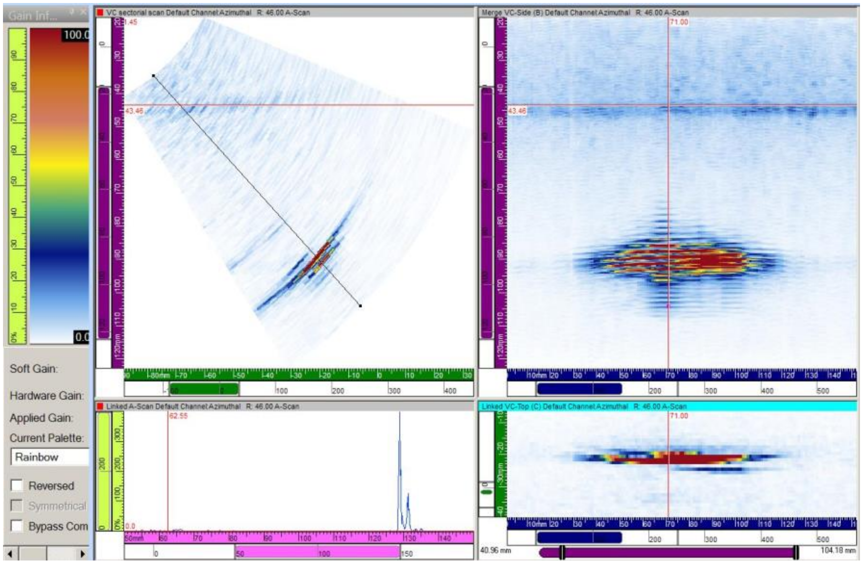


Figure 2-9 Magnetic particle inspection (MPI) example

2.3.4 Ultrasound Testing (UT)

High frequency sound waves have the ability to transmit through solids and liquids and the associated technologies are known as ‘ultrasonic’ when the frequencies are above the hearing range of humans. The UT method is very commonly used for inspection for internal flaws and for precise flaw detection in relatively small areas of components as shown in figure 2.10. It requires considerable skill to interpret the display for example the relationships of flaw size, flaw distance and flaw reflectivity so the skill required by the operator is relatively high. There are also some limitations in its ability to detect certain kinds of cracks like those in very thin sections [38], and it is very sensitive to the orientation of flaws.



Soft Gain:
Hardware Gain:
Applied Gain:
Current Palette: Rainbow
□ Reversed
□ Symmetrical
□ Bypass Com

Gain Inf...
100.0
0.0
VC sectorial scan Default Channel Azimuthal R: 46.00 A-Scan
Merge VC-Side (B) Default Channel Azimuthal R: 46.00 A-Scan
Linked A-Scan Default Channel Azimuthal R: 46.00 A-Scan
Linked VC-Top (C) Default Channel Azimuthal R: 46.00 A-Scan
43.46
71.00
62.55
40.96 mm
104.18 mm

Figure 2-10 Phased array Ultra sound (UT) example

2.3.5 Eddy Current testing (ECT)

This method uses the principle of induction. Generating a high frequency current in a coil and holding that coil close to the surface of a metallic object generates 'eddy-currents' in the surface of the object [39]. It is generally used when a flaw is anticipated so it is used over small areas rather than for large-scale testing. Figure 2.11 shows a sample outcome of the eddy current testing (ECT) method and figure 2.12 shows eddy current array (ECA) example. A high degree of operator skill is required as the results can only be confidently received from a very experienced operator; only conductive materials can be inspected; requires calibration on a set of specimens; and many other parameters can affect the responses [40]; The tests generally are restricted to surface breaking conditions and slightly subsurface flaws.

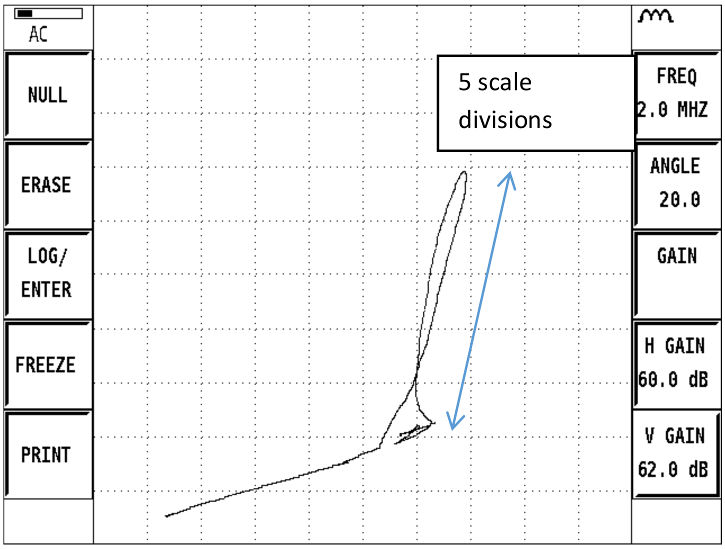


Figure 2-11 Eddy current testing (ECT) example

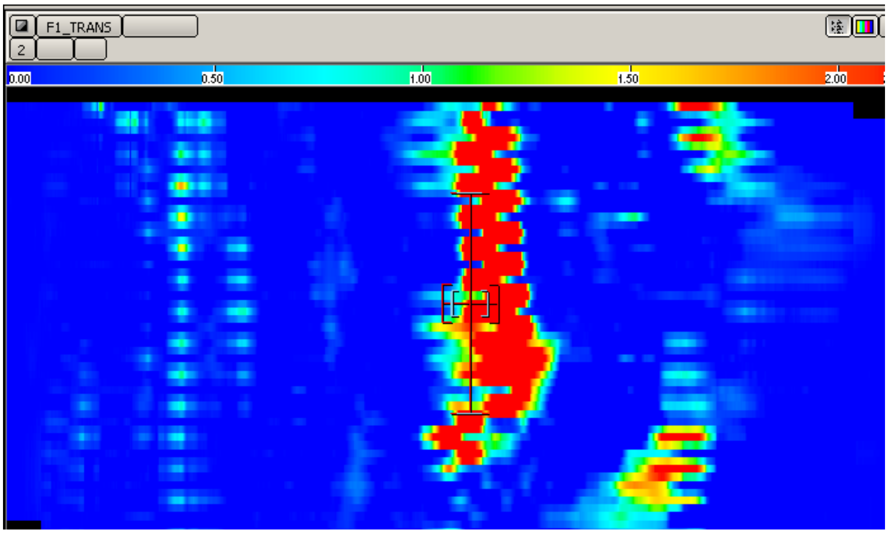


Figure 2-12 Eddy current array (ECA) example

2.3.6 Radiography (RT)

It uses X-rays or gamma rays to produce an image of an object on film. Figure 2.13 shows a sample outcome of radiographic image. The main key points are that there are safety issues like possible radiation hazard for personnel; the equipment is cumbersome; it requires access to both sides of an object which is not possible in on-site case; usually two operators are needed; qualified staff must be employed; the rays are dangerous so they must be used inside a protective enclosure; a relatively expensive equipment investment is required; a film or image processing facility to develop, wash, fix and dry is usually required before results can be seen; and it is not good at defining the presence or through thickness dimensions [41] of small cracks. For purposes of this project, it is not portable and cannot be taken on-site and is not accessible from both sides. For above mentioned reasons, this method is not considered.

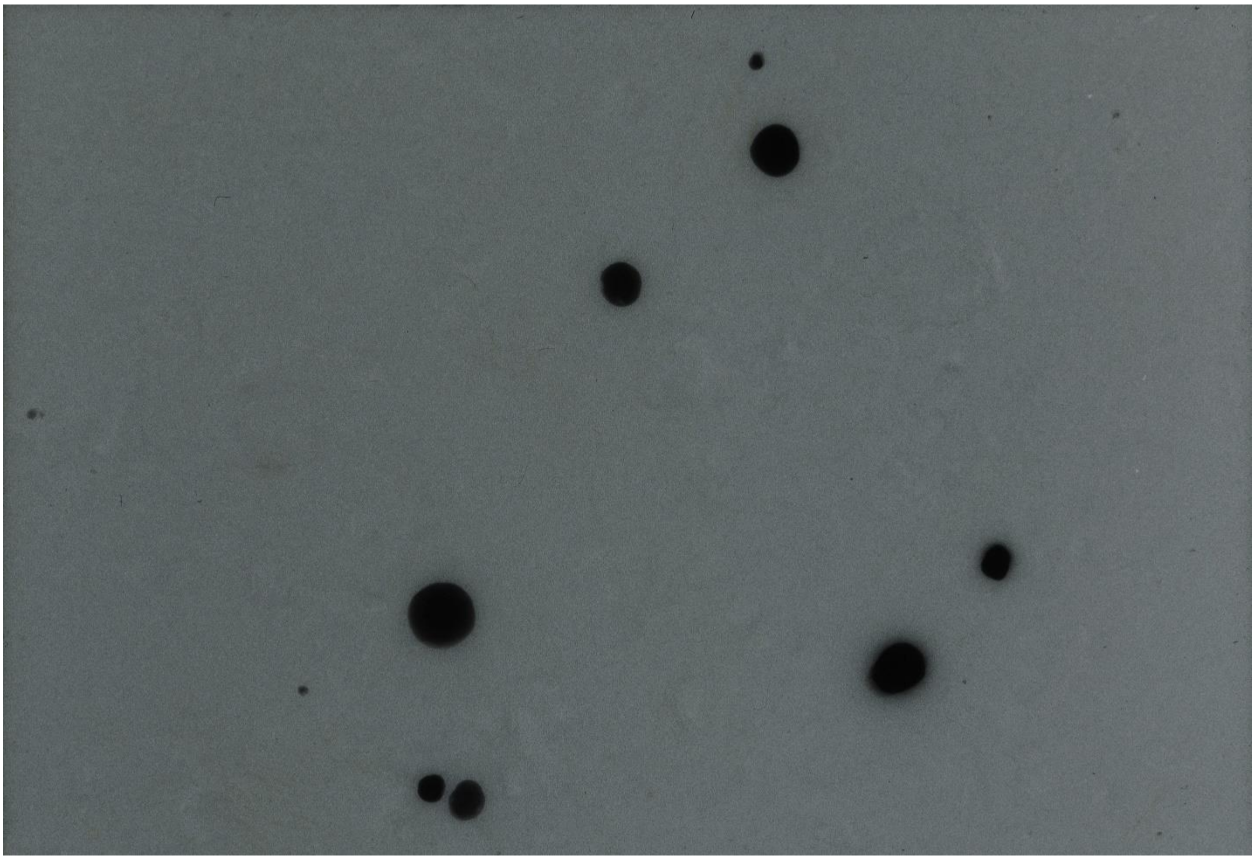


Figure 2-13 Radiographic (RT) image example

2.4 State of the art techniques used for defect detection

Different perceptions have been taken in order to take in account the problem of defect detection in materials. These could be broadly categorised as image processing methods and learning-based methods [3]. The former entails methods based on image filtering like Gabor filters [42], wavelets-based approaches [43] or methods based on the geometry of the defect

[35] while the latter makes use of typical pattern recognition tools, such as neural networks [44] self-organizing maps [45] or support vector machines [46]

Another perspective into this problem is in terms of the data collection device. There are some research groups that are working particularly in the area of defects detection using microscopic imaging such as Aalen University as mentioned in 2.4.6 in detail. For this aspect, the device portability requirement of the project as well as the size of the defect being investigated, needs to be considered carefully. There are many good quality microscopes but most of them are not suitable for on-site application as they are not portable and also most of them are not cost-effective for every day industry usage.

Another way of the classification could on the basis of the kind of defects detected. Mostly the studies are based on a solution for a specific type of defect [47] [48] but there are some researches which are for general applications. These are mainly focused on the crack detection mostly for concrete [1] [49] [24] , asphalt [50] or pavement [51] but as far as corrosion is concerned, there are very few studies regarding this problem such as described in [29] refers specifically to detecting corrosion in metals.

This shows that a lot of work still needs to be researched in this area. Currently the system being used in the industry is by manually counting the pits using visual inspection and for the cracks using the MPI method. Emphasis has been laid for the crack defects but not much has been applied in terms of the analysis of pits [52].

2.4.1 Detection of cracks using fuzzy logic and an ANN model

The study [1] presents fuzzy logic and artificial neural network based models for accurate crack detection on concrete. Features are extracted from digital images of concrete surfaces using image processing which incorporates the edge detection technique. The properties of extracted features are fed into the models for detecting cracks. Two kinds of approaches have been implemented in this study: the image approach which classifies an image as a whole, and the object approach which classifies each component or object in an image into cracks and noise. The models have been tested on 205 images and evaluated on the basis of five measures of performance.

The limitation of the work is that it sets criteria that the image of concrete should not contain any object other than concrete. Otherwise, due to the use of edge detection technique, the models detect the edges of these objects as cracks. Also, if the human eye itself cannot see the crack in the image, neither can any of the models in this research. The crack has to be at least visible no matter how much noisy the surface of the concrete is, only then is it possible to detect cracks using these models. But in case of the project problem, most of the times, the flaw is not visible at all or sometimes not distinct with extra sample features present.

2.4.2 Detection of cracks using image processing and ANN

Research was carried out by [53] to evaluate the cracks which are sometimes formed when molten steel is solidified to make slabs. Removal of these cracks save time, effort and production expense; and reduces costs if detected at an early stage. They have implemented an image analysis system, and its block diagram is shown in figure 2.14. It consists of four major steps which are the edge enhancement and noise reduction; binary segmentation; colour segmentation and Hough transform. These follow as: wavelet transform (2D discrete Haar), non-linear diffusion (Anisotropic diffusion), adaptive Gaussian filtering; then edge detection (SUSAN- Smallest Univalue Segment Assimilating Nucleus), morphological filtering to find exact location; then k-means clustering (HSV – Hue, saturation, value), feature extraction to find the area perimeter length, Artificial Neural Networks (ANN); and then in the end used Hough transform to calculate the size and position of the defect. The experiment results were 97% sensitive, 96% specific and 96.5% accurate.

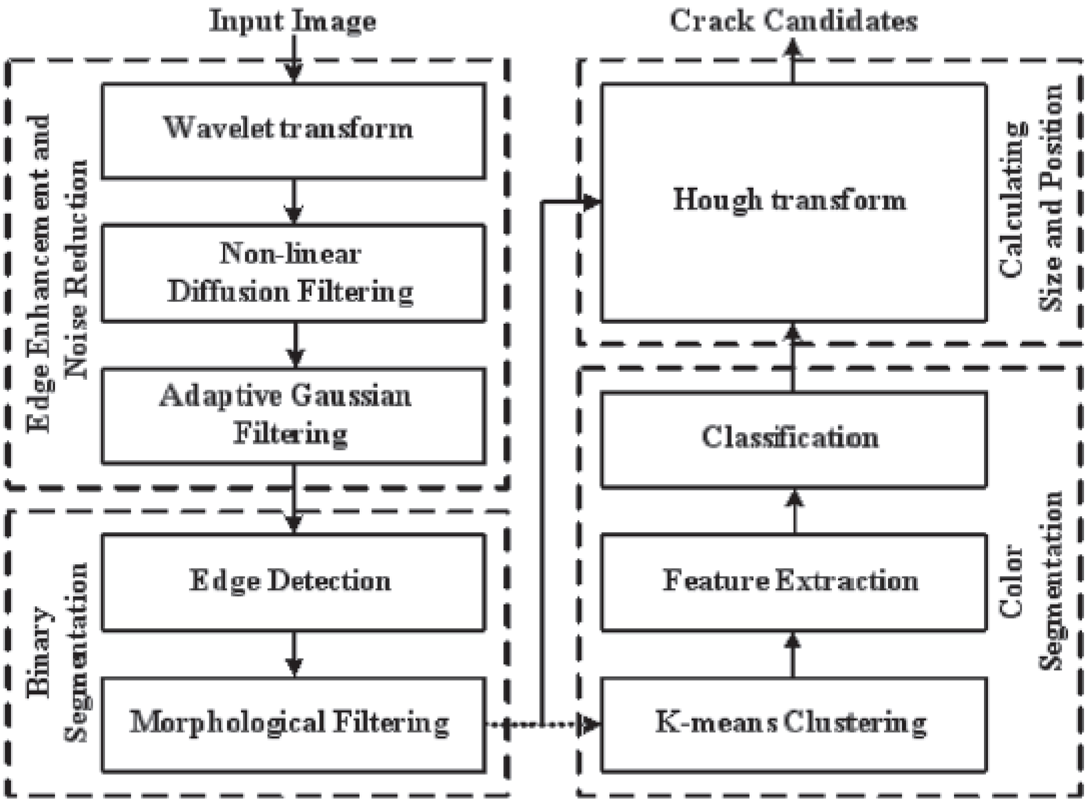


Figure 2-14 Block diagram of a proposed algorithm by [53]

This detection method is based on the features extracted from image processing and then fed into the ANN so it is more of an object detection approach.

2.4.3 Micro-cracks detection using an image segmentation technique

An image segmentation technique was used by [46] to detect micro-crack defects in the multi-crystalline solar cells. They have used electro-luminescence (EL) and high-resolution cameras to capture the defects with dimensions smaller than 30micron. The micro-crack pixels and the background are not significantly different in their grey scale values but the micro-cracks can still be identified as they appear in the form of strong lines with a low intensity and high gradient. There are four major things to consider: bus-bars which are the dark vertical lines; fingers which are the light horizontal lines; micro-cracks that appear in the form of a line or an intersection of lines like a star and the background. Block diagram of the algorithm can be seen in figure 2.15. Following are the steps involved in this system.

1. The first step is the image pre-processing where filtering was applied in the frequency domain to remove the periodic interruption of fingers by removing high-frequency components and retaining the low-frequency domains components like micro-cracks to get inverse Fourier transform image and further normalised to 128 to minimise the error due to the inconsistency of grey-level between cells.
2. The second step was to implement the anisotropic diffusion filtering which gives equal response to any pixels so it was programmed to consider both the intensities, gradient as well as grey level of each pixel, in order to produce a smoothed image with the important edges intact and then the resulting image obtained is by subtraction to produce a new, enhanced image that makes the micro-crack line more visible.
3. The third step is the post- processing which consists of the following steps: apply segmentation using a high threshold value that produces image consisting of mainly incomplete but noise-free edges and then using a low threshold value that gives an image that contains complete edges and noise; reconstruction of the binary image followed by dilation and opening; next the intensity tracing and thresholding are applied to further reduce the noise and unwanted shapes such as scratches and clusters since the grey values of the micro-cracks are relatively less in comparison to the artefacts. The pixels which have the same grey intensity value at the same location bounded by the same contour are traced and extracted then the mean value for each pixel group is computed which is used for thresholding the shapes like if the mean value of any shape is less than the specific threshold, retain the shape otherwise eliminate it. The last step helps to reduce the number of shapes significantly and hence helps to improve in the feature extraction.
4. The fourth step is the shape analysis which is performed to distinguish between micro-cracks and other objects from the dataset examples of varied micro-cracks shapes and will produce features from angular radial transform (ART) shape descriptors that could

be used later in machine learning and classification. The micro-crack has a more distinct fluctuation which increases the average distance between the two spectrums hence resulting in a better discrimination of the shapes. The features are used to train the artificial classifier.

The performance of the algorithm is quantitatively evaluated in terms of three measurable metrics which are sensitivity, specificity and accuracy as the previous journal discussed.

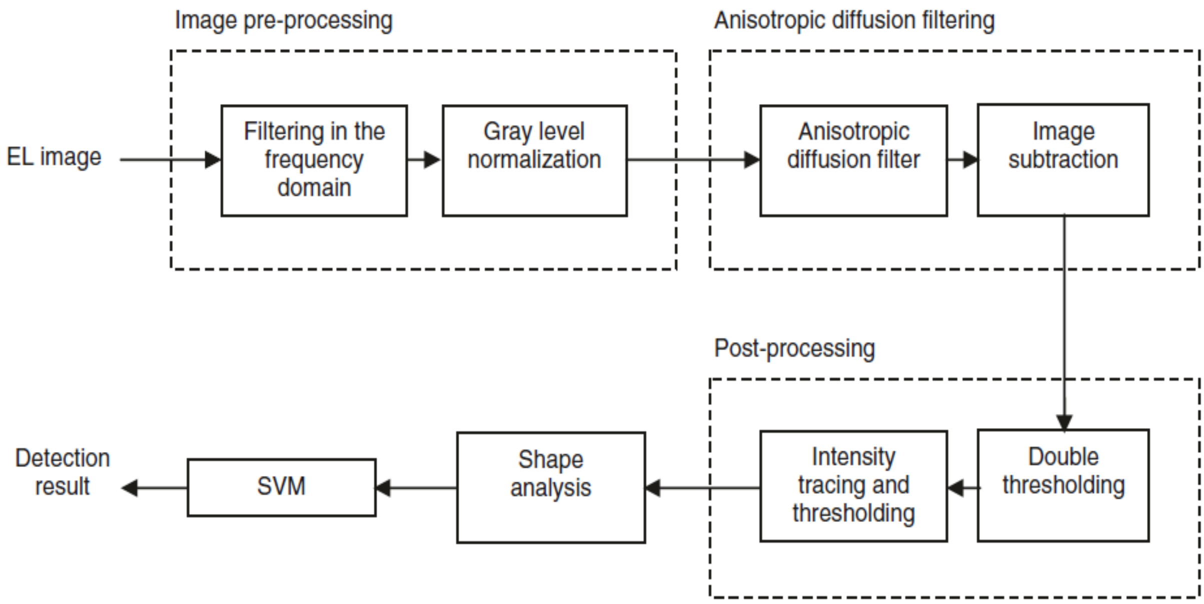


Figure 2-15 Block diagram of the proposed algorithm by [54]

This method is based on data collected by high-resolution cameras and also have used electro-luminescence (EL) which is able to capture high quality images but such cameras are often not portable to be able to taken on-site.

2.4.4 Classification of corrosion based on colour analysis technique

One of the studies on corrosion is based on the colour analysis technique[55]. A scanner-based image analysis was generated for the concrete corrosion that is a simple and cost-effective technique [56]. The second approach was based on artificial intelligence, in particular using Deep Learning methods. The framework is specifically suited for image processing, offering good speed and great flexibility. It also offers the opportunity to easily use clusters of GPUs support for model training which could be useful in the case of large networks. The first step was to collect a good dataset to be used to train the network. They were able to collect around 1300 images for the “rust” class and 2200 images for the “non-rust” class. Around 80% of the images were used for the training set, while the rest was used for the validation set. The dataset was very small so they used an existing model.

This method is restrictive based on the colour of the rust. Though it is much easier to collect and label more data in such a case but usually when microscopic inspection is done, the rust material is cleaned off the specimen to get a good idea of the on-surface cracks and pitting corrosion under the rust layer, as will be discussed in chapter 3 in detail. So, for this project, labelling the data will be much more challenging and complex.

2.4.5 Image classification based on a CNN model

Convolutional networks are powerful visual models that yield hierarchies of features. One of the researches [57] show that convolutional networks by themselves, trained end-to-end, pixels- to-pixels, exceed the state-of-the-art in semantic segmentation. The key insight was to build “fully convolutional” networks that take input of arbitrary size and produce correspondingly-sized output with efficient inference and learning. Fully convolutional networks were defined in depth. They adapted contemporary classification networks like AlexNet, the VGG net, and GoogLeNet, into fully convolutional networks and transferred their learned representations by fine-tuning to the segmentation task. A skip architecture is defined that combines semantic information from a deep, coarse layer with appearance information from a shallow, fine layer to produce accurate and detailed segmentations. For project purposes, pixel-wise classification method might be considered and implemented in the later stages of the research.

2.4.6 Inspection applying deep learning models on microscopy images

The research [58] on prospective identification of hematopoietic lineage choice by using deep learning, uses image patches from brightfield microscopy and cellular movements. It shows the potential of deep learning with microscopy using a high-end device.

Machine learning research group at Aalen University Germany have been involved in research [59] that deal with microscopic images and deep learning. The study [60] based on research from top journals and conferences, presents a review of medical image analysis using deep learning. It discusses the advantages of using deep learning on medical images along with a short summary of a few applications. Another research [61] at Aalen University discusses an efficient machine learning based detection of heart disease using Random Forest algorithm. The dataset consists of patient’s information such as age, gender, body mass index, smoking condition, previous familiar cases, diet, cholesterol, glucose, high blood pressure etc.

The research [62], more relevant to our problem, is an additive manufacturing application that uses deep learning-based model for defect detection in laser-powder bed fusion. The data set has been collected using thermographic camera and infrared. The model architecture is shown in figure 2.16, that performs on an entire image with an image-wise processing approach. The model uses these thermographic imaging to detect printing defects such as delamination and splatter with an accuracy of 96.8%. However other defects such as cracks and pores have not been evaluated.

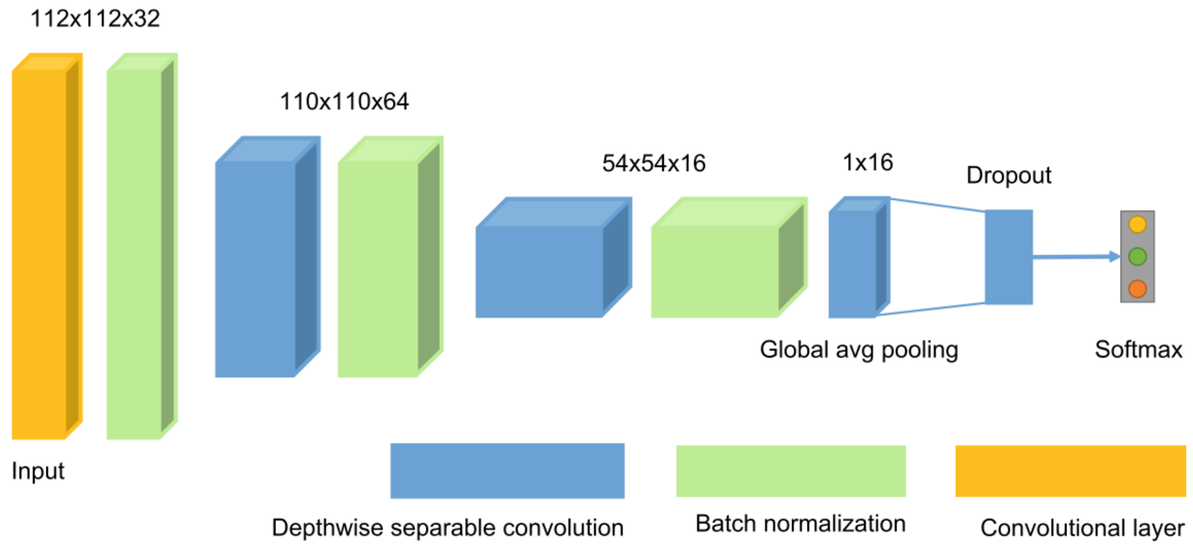


Figure 2-16 Model architecture of the research at Aalen University [62]

2.5 Image Processing Methods

This project is about analysing the structural integrity defects like pits and cracks as a non-destructive testing. After acquiring the images, they will be analysed by using traditional image processing methods. Currently the method being used is the visual inspection method [63] by manually counting the number of pits.

Previously, different image processing techniques have been discussed and used depending on the material and the purpose of the project. A lot of work has been done in the medical field [64] towards analysing images. Some work has been done for detecting cracks in different industry applications with either different scale of the defect area or different high-end optical technique, which are not suitable for commercial use keeping the economical aspect in mind. However, in the area of corrosion very less work has been done, and that too for micron level resolution pitting is extremely limited.

Automatic detection of defects on various surfaces is an area of active research, which is based on digital images. Image processing and threshold based decision making is the basis of most of the research in this field. Understanding some of the key methods and techniques existing in the IP area would be beneficial to do a research of the complex methods. Following are some of the techniques concerned with image pre-processing that typically involves restoring the effects of disruptions that have occurred during the acquisition of the images, reformatting the image data, and then enhancing the image for visual inspection and interpretation or segmentation of regions and objects in an image on the basis of similarity criteria that permit features to be extracted for further tasks, such as classification which is the next section of this chapter.

2.5.1 Image enhancement

These methods are mainly used to improve the perception of a piece of information in images depending on the task at hand [65]. The pixel values are manipulated to achieve desired enhancement, and is used mainly for adjusting the image contrast, histogram equalization, de-blurring, filtering, noise reduction and edge sharpening. As a consequence of these techniques the pixel grey-level values that is the intensities of the output image will be modified according to the transformation function applied on the input values. It should be noted that in some image classification applications, enhancement techniques are avoided [65] since modifying the grey-level values will affect the accuracy of the classification system.

Linear

This method is used [66] to enhance the contrast of an image whose pixels are of a similar grey level like in a really dark image. In order to ameliorate the contrast in a particular grey level band, linear transformations map a certain grey level band to a wider one. All pixels below a certain grey level will appear black while all pixels with a grey level greater than a certain level will be white. Pixels between these values will be mapped to values between 0 and 255.

Power and Log

These transformations map narrow grey levels to wider ones. For example, a log transformation [67] will map low narrow grey levels to a much larger band. If most pixels have a grey level between 0 and L/4 where L is the maximum grey level, mapping them to a 3 times wider scale will enhance contrast. However, pixels with an original grey level between L/4 and L will be reduced to a 3 times smaller band. Those transformations are interesting in cases where the image is clear or dark. The log transformation can enhance details in a dark image. By mapping the darkest pixels to brighter ones, more details appear. Gamma correction is a way to accentuate contrast, just as log transformation. Gamma transformation’s impact is related to the value of gamma, as changing it makes different bands of grey levels wider or smaller. A low value of gamma will make dark pixels brighter, while a high value will make bright pixels darker. Gamma correction maps narrow bands of pixels to larger ones.

Filters

To modify an image, we can use filters. A filter will alter a pixel in the image according to its neighbors. Using filters can help sharpen or smooth an image. There are many kinds of filters that can be used such as mean, median filter, Gaussian filter, Wiener filter, order-statistic filter, edge-emphasizing filter, Laplacian operator filter and linear motion filter. For example median filtering [68] is a nonlinear method used for the removal of impulsive noise. It is executed by choosing a mask of odd length which moves over the image and at each center pixel the median value of the data within the window is taken as the output. Wiener filter [68] often produces better results as it is applied to an image adaptively tailoring itself to the local image variance, performing little smoothing where the variance is large and more smoothing where the variance is small. It works best when it is used to remove Gaussian noise.

Histogram equalization

Enhancement can be accomplished by using this which can stretch the image contrast between the background and foreground pixels and help highlight specific details in the image. The main objective of histogram equalization is stretching out the grey levels of an image matching a specified histogram, uniform distribution by default [69]. For example, if an image is mainly dark, making the brightest pixels of the image actually bright can produce a brighter image. But in some scenarios it might over saturate the area causing an object to wash out.

Contrast Limited adaptive histogram equalization

It is known as CLAHE for short. Unlike histogram equalisation, it operates on smaller data regions like tiles rather than on the entire image. Each tile’s contrast is enhanced so that the histogram of each output region approximately matches the specified histogram [70], uniform histogram by default. The contrast enhancement can be limited in order to avoid amplifying the noise which might be present in the image so the problem of over saturation is solved by this enhancement technique. It does not cause some of the intensities to be too bright but still manages to make the subtle smaller features more pronounced. But if the tiles are kept too small it might enhance the noise as well.

Thresholding transformation

These transformations take an input image and return a black and white image. Each pixel above a certain grey level is returned white while the others are returned black. Thresholding transformations isolate objects from the background. The object is made white while the background stays black. It obviously works better when the object to show and the background have different colors [71]. The main problem with this technique is to find the threshold so that all the pixels of the object are white and all the pixels of the background are black. Different thresholds can be chosen given the details we want to enhance.

2.5.2 Edge detection

In the active field of computer vision, edge detection is considered a well-developed area on its own. Edges help in segmentation and object recognition by defining the boundaries between regions in an image so the main objective of these techniques is to find the edges of the objects of an image [72]. Often there is a sharp change in intensity at the region boundaries and that is why edges and region boundaries are closely related. The basic theme to find the edges is to look for places in the image where the intensity changes rapidly or suddenly. It works by detecting discontinuities in brightness. Edge detection is used for image segmentation and data extraction in areas such as image processing, computer vision, and machine vision.

Many studies have applied and researched these methods [73] [1]. The most common edge-detectors are named after their inventors such as Sobel, Prewitt, Roberts and Canny. The basic edge-detectors usually decide whether or not a pixel lies on an edge independently of the neighbouring pixels, and as a result, noise can fragment the boundary of a region, or miss a region. Canny method is considered one of the most powerful edge detectors [74]. It differs from the other common ones, as it uses two different thresholds to detect strong and weak edges. It includes the weak edges in the input only if it is connected to a strong edge. Therefore it is more likely to detect true weak edges rather than getting noise.

2.5.3 Morphological filtering

It is a broad set of operations that process images based on shapes. The above discussion on edge-detection and clustering indicates that further processing is required to correct any insufficiencies or errors in the segmentation due to noise in the image, and a full segmentation of an image usually involves several different segmentation algorithms. The binary image after thresholding may contain noises and small residual spots that mask the useful information to be interpreted. To eliminate these artefacts, morphological processing can be applied. However they might also be applied for pre-processing purposes to clear the image before taking any steps.

There are basically two morphological operations [75]. They perform the shrinking and/or expanding operations with regard to a certain structuring element (SE). Mathematical morphology could be used to detect the boundaries of objects and their skeletons for example, by taking the dilation of an image and then subtracting away the original image, thus highlighting just those new pixels at the edges of objects that were added by the dilation. These filters may also be used for thinning or pruning of the edges as either a pre or post processing technique.

In the morphological operations [76], a specific rule appropriate to the requirement is applied to the pixels in the neighbourhood and a value is assigned to the corresponding pixel in the output image. The choice of the structuring element is of high importance as they probe the image, they are typically much smaller than the image. The rule used to process the pixels defines the operation. The basic filters used are as follows:

For an erosion operation the value of the output pixel is the ‘minimum’ value of all the pixels in the input pixel’s neighbourhood. So in a binary image if any value is set to ‘0’ the output pixel is set to ‘0’. For dilation operation, the value of the output pixel is the ‘maximum’ value of all the pixels in the input pixel’s neighbourhood. So in a binary image if any value is set to ‘1’ the output pixel is set to ‘1’. Opening operation is a combination of two basic operations, erosion and dilation with the same structuring element. Opening by a disk-structuring element can smooth the boundary, break narrow parts and remove small objects. The SE should be large enough to remove the small objects in the image but not large enough to remove the larger ones. Closing operation is also a combination of two basic operations, dilation and erosion, but in reverse order. Closing by a disk-structuring element can fill narrow bays, eliminate small holes and also smooth the boundary.

Reconstruction operation [77] uses two images as inputs, a marker and its mask, instead of one image and its SE. Hence, it is based on the concept of connectivity rather than an SE. The reconstruction applies a dilation or erosion on one image, called the marker, based on the characteristics of another image which is called the mask. This is repeated until the application acquires stability that is does not change the result anymore. It could be understood as repeated dilations of the marker image until the contour of the marker image fits under the mask image.

There are many other techniques that can be used like Skeletonization [43] is used to reduce all objects in an image to lines, without changing the essential structure of the image. Perimeter determination is used to find the perimeter pixels of the objects in a binary image. A pixel is considered a parameter pixel when the pixel is on and one or more of the pixels in its neighbourhood is turned off. Top-hat filter [35] could be used to eliminate particular features from an image. The basic method considers; applying opening to an object, and then doing a subtraction with the original image. It can also be used to enhance contrast in an image. Bottom-hat can be defined as the difference between the ‘closing’ of the original image and the original image.

2.5.4 Image Segmentation

Image segmentation techniques are found out to be strongly application dependent as each is applied and adapted according to the application features involved [66]. The basic steps involved in the detection could be: Smoothing to suppress as much noise possible without the true edges being destroyed; Enhancement to enhance the quality of the edges in the image; Detection to make a decision on which of the edge pixels to keep and which to eliminate as noise; and Localisation gives the exact location of the edge. This technique is used to isolate regions and objects of interest depending on its similar properties such as grey-scale, colour or texture. The process is to divide an image in to multiple parts to identify relevant information from the digital image. In order to characterise the defects in an image like its size, locations and shapes, they need to be segmented from the background.

Thresholding

The simplest technique is thresholding. It can be regarded as partitioning pixels into foreground object and background based on the comparison between the grey level of a pixel and a threshold. A large number of algorithms have been proposed like Otsu and histogram because they are efficient in performance and simple in theory. The algorithm automatically computes a threshold based on a given distribution or histogram of grey levels [78]. There are different parameters to consider to threshold an image. The basic idea is based on a clip level or value to turn the grey scale image into a binary image. The most important problem is to choose optimal thresholds [71]. Otsu is an effective method as it uses the calculated optimum threshold value to separate the two classes, the foreground and background, by maximizing the interclass variance [79].

Histogram thresholding could be used if the histogram of an image includes some peaks by separating it into a number of modes. Each mode corresponds to a region, and a threshold exists at the valley between any two adjacent modes. An appropriate threshold value is found in an iterative fashion in this midpoint method [80] with the basic steps being: apply a reasonable initial threshold value; then compute the mean of the pixel values below and above this threshold, respectively; also compute the mean of the two means and use this value as the new threshold value; continue this until the difference between two consecutive threshold values are smaller than a pre-set minimum value.

Clustering

This segmentation method is used for good recognition for image processing. It is a specific method as it can make characteristic classification of the pixels by computing similarity between them. It basically organizes the objects into groups based on certain attributes. There are many methods which are effective for this task. Some of the popular ones are k-means and FCM.

In K-means clustering [81] algorithm, data vectors are grouped into predefined number of clusters. At the initial stage, the centroids of the predefined clusters are initialized randomly. The dimensions of the centroids and the data vectors are the same. The assignment of each pixel to the cluster is done on the basis of how close they are in terms of the Euclidian distance measure. Once all the pixels have been assigned to their respective cluster, recalculation of the average of each cluster is performed. This process is continued for a fixed number of iterations or until no significant changes result for each cluster mean [53]. An image segmentation method was proposed by [82] based on the collaboration of self-organisation map (SOM) neural networks. It clusters the pixels in image according to colour and spatial features with the SOM neural networks. Results produced show that it is better than K-means or single SOM neural network, but it also has a drawback which is to manually set the number of regions to be segmented.

Fuzzy C-means clustering is also known as FCM in common. It is a method of clustering which allows one piece of data to belong to two or more clusters unlike the k-means clustering which assign them to a single cluster. This allows the pixels to belong to multiple classes with varying degrees of membership.

Watershed

This technique is a fine example to see how effective it can be to combine many morphological operations to achieve a segmentation task [83]. Separating objects in an image where objects are touching each other is segmentation. Watershed is often used in images where it is required to segment touching objects.

2.5.5 Feature Extraction

In machine learning terms, it can be defined as an individual measurable property or characteristic of a phenomenon that is being observed. They are the input which is fed into machine learning models like K-nearest neighbour or Adaboost ensemble, to receive a prediction or classification output, so the model is only as good as the features provided to it. This means that selecting good features is an important task that should not be underestimated hence traditionally a good research is done for manual features selection and engineering. It relies on domain knowledge or partnership with domain experts, to create features that will make algorithms work better. The basic idea is to transform the raw data into feature vectors to show the learning algorithm how to learn the desired characteristics

[84]. Some of the handcrafted descriptors that have been previously proposed in studies are features like Scale-Invariant Feature Transform (SIFT) [85] and Histogram of Oriented Gradients (HOG) [86].

The input to the model are images that are comprised of pixels. These raw pixel values from an image can be transformed into substantially more meaningful and useful information. They could be a pattern or a distinct structure within the image, referred as an image patch that contrasts from its immediate surroundings by either texture, colour, or intensity. Such texture regions can be defined as the statistical spatial distribution of their pixel intensities [87]. Texture analysis can help in segmentation or by extracting information about the defects in the images with the help of characteristics such as brightness, colour, shape and size [88]. Detectors that rely on gradient-based and intensity variation approaches are able to detect good local features. There are two approaches, structured and statistical, to analyse an image texture in computers.

Many texture segmentation methods include the extraction of texture descriptors that are based on local responses to hand-designed filter banks typically with a set of scales and orientations such as GLCM descriptors, Gabor filters [42], local spectral histograms [89], Edge detection [64] [73], spatial frequency and an average grey level [87]. Other popular texture features used include statistical descriptors such as Local Binary Patterns (LBPs) [48], co-occurrence matrices, and wavelet transforms. Segmentation of texture descriptors include approaches like k-means, region splitting and watershed [76]. Deep learning methods have been able to perform better than most of these methods in classification and segmentation tasks especially the CNNs. These approaches have enabled to learn multiple levels of abstraction, by replacing these hand-crafted descriptors to trainable filters combined in a cascade of layers. Hence the research was shifted towards this technology, which is discussed in detail in the next chapter.

Similarly, there are many colour based models such as RGB and HSV that are used for machine learning purposes. They can be classified into groups based on Wu and Sun (2013) research. The hardware-orientated spaces are based on the hardware equipment used to reproduce the colours, such as RGB, YIQ, and CMYK. Another is the human-orientated spaces that are not sensitive to small colour changes but rather based on hue and saturation such as HSI, HSL, HSV and HSB which resemble to the concepts of tint, shade, and tone. Then there is the Instrumental spaces group, that are based around the property that the colour coordinates of an instrumental space are the same on all output media, with models such as XYZ, L*a*b*, and L*u*v* as examples. Each colour space has a particular reason for its development and so each of them have an advantage over the other when applied for classification and recognition tasks [90]. In short, there are various features based around texture, intensity or colour descriptors. Images have different colour spaces such as RGB, LAB, HSV and different texture variations can be extracted like LBP, HOG and gradients

2.5.6 Histogram of Oriented Gradients (HOG) feature

It is one of the effective methods to extract features from pixel for classification purposes [91]. Once image gradient vector has been researched it is easier to understand the functionality of the HOG feature, as it is based around it. The method starts by performing pre-processing that includes re-sizing and colour normalisation, then compute gradient vector of every pixel along with its magnitude and direction.

Later on divide the image into many 8x8 pixel cells. In each cell, the magnitude values of these 64 cells are binned and cumulatively added into 9 buckets of unsigned direction. This means no sign hence 0-180 degree rather than 0-360 degree, which is a more practical choice based on empirical experiments. For improved robustness, if the direction of the gradient vector of a pixel lays between two buckets, its magnitude proportionally splits between the two bins instead of going into the closer one. To better understand this, let’s say if a pixel’s gradient vector has magnitude 8 and degree 15, it is between two buckets for degree 0 and 20 and value ‘2’ would be assigned to bucket 0 and value ‘6’ to bucket 20. This makes the histogram much more stable when small distortion is applied to the image.

Then for the final step, slide 2x2 cells block that means 16x16 pixels, across the image. In each block region, four histograms of four cells are concatenated into one-dimensional vector consisting of 36 values and then normalised to have a unit weight. The final HOG feature vector is the concatenation of all the block vectors. Then this can be fed into a classifier for learning pattern recognition tasks.

2.6 MACHINE LEARNING

Machine learning is one of several methodologies of the grand scale Artificial intelligence field. It teaches machines tasks that are easy for humans to perform but hard for them to formalize how it was performed [92]. Formally, machine learning can be defined as, a computer program that is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E” [93]. In simpler terms, it is an application of Artificial Intelligence (AI) that provides computer systems the ability to learn and improve from experience without explicit programming. Once a computer algorithm is trained, the algorithm can apply the relationship learnt during training to solve similar problems. Basically, there are two types of machine learning techniques, one is unsupervised learning and the other is supervised learning. Supervised learning further falls down in to two categories; regression and classification.

Classification refers to a procedure that assigns data objects to a set of classes. Classifiers are pattern recognition methods that perform partitioning based on certain features derived from an image using data with known labels [94]. The most common example of a feature is the image intensities like a histogram can provide feature with two apparent classes. Several methods have been researched on classification of images. Following sections explain different popular classifiers such as Tree, SVM, KNN, Ensemble, and Random Forest.

2.6.1 Linear discriminant analysis (LDA)

It is one of the simpler methods of supervised classification that classifies data by finding linear combinations of features. It classifies the samples into groups that are mutually exclusive with the objective to attain maximised between–group variance and minimised within-group variance. It assumes that different classes produce data based on the Gaussian distributions so the model is involved in searching the parameters for such distribution. The parameters are used to calculate boundaries that can be linear or quadratic functions and these boundaries are then used to conclude the new data class. If the available variables are two then the separators between the groups will become lines. If there are three variables available then the separator is a plane. When the number of variables is higher than three then the separators become a hyperplane.

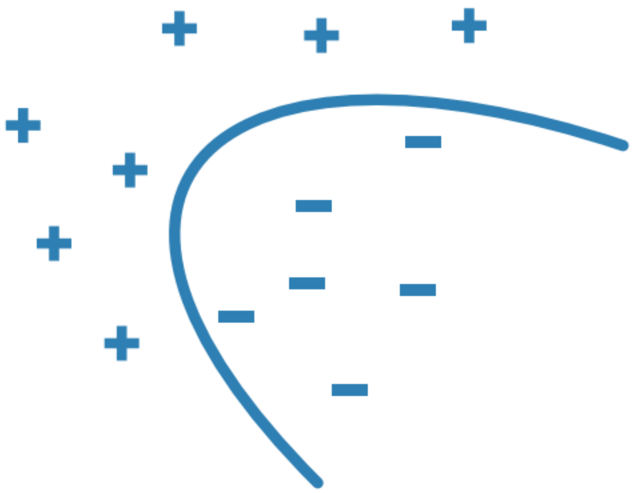


Figure 2-17 Illustration of machine learning linear discriminant analysis

It is best to use this model when a simple model is required that is easy to interpret, or when memory usage during training is a concern or when model is needed that is fast to predict.

2.6.2 Logistic Regression

This model is commonly used as an initial start for binary classification problems due to its simplicity. It can predict the probability of a binary response belonging to one class or the other.

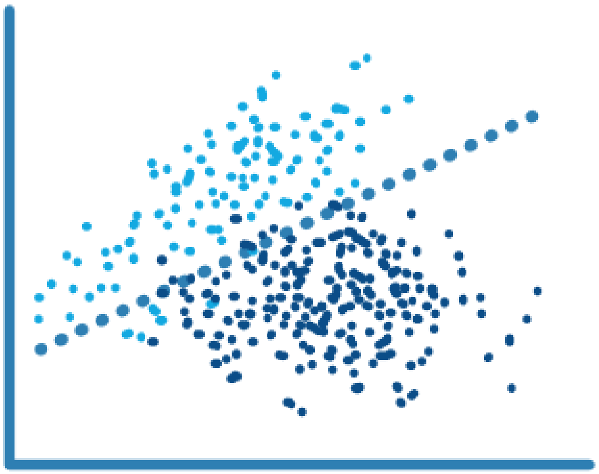


Figure 2-18 Illustration of machine learning Logistic Regression

This model is best to be used when data can be clearly separated by a single, linear boundary or if it's being used as a baseline for evaluating more complex classification methods.

2.6.3 Naive Bayes

This is also one of the simpler methods which is better suited for less complex problem cases. It's a high-bias with a low-variance classifier. When there is a limited amount of data to be trained then using this model might have a slight advantage over logistic regression. Naïve Bayes makes calculations based on Bayes' theorem. Naïve Bayes assumes the features are independent of each other [95]. If two classes are being dealt with then high probability to the observations belongs to one class, and low probability to the other class. A threshold is defined on the basis of which the separation between the two classes are made.

It is best to use this model for a small data set with many parameters, or when a classifier that's easy to interpret is needed or when CPU and memory resources are a limiting factor. If the data grows in size as well as variance then it is better to work with a more complex model. Also, its simple analysis is not a good basis for complex hypotheses.

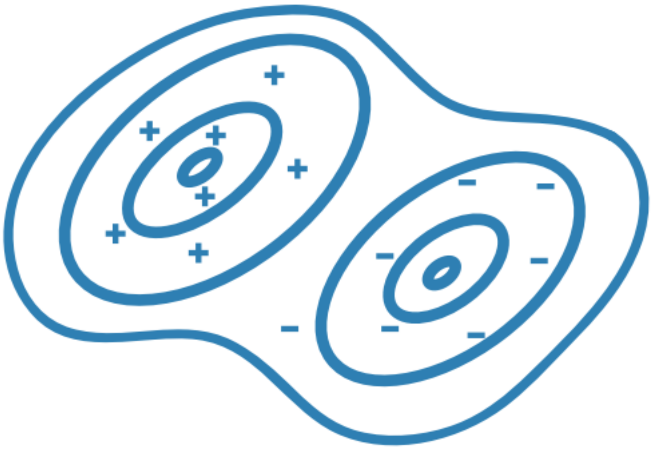


Figure 2-19 Illustration of machine learning Naïve Bayes model

2.6.4 K-Nearest Neighbour

KNN is a type of instance-based lazy learning, meaning there is no actual training phase involved. It starts when there is a new data entry where it looks to classify objects based on the classes of their nearest neighbours in the dataset. The model looks for the specified k number of nearest neighbours so if the value of k is 5, then the class of 5 nearest neighbours are found. If a label or class is being applied then the model takes a vote to see where it should be classed.

KNN predictions assume that objects near each other are similar hence it is known to be a 'guilty by association' algorithm. It categorises data points based on their distance metrics such as Euclidean, to other points in a training dataset. Basically it looks for K number of training observations closest to a test observation, counts how many observations in each class out of K, and return an estimate of the likelihood this test observation belonging to a particular class.

One of the drawbacks to this method is that it can be deceived by irrelevant attributes that might obscure important attributes. But there are ways to correct this issue, such as applying weights to the data. Weights are assigned to the contributions of the neighbours so that the closer neighbours contribute more to the average than the more distant ones.

When the class distribution is skewed, it causes a downside affect to the 'majority voting' classifier. A more frequently prevailing class will dominate the prediction of the new example because they are more common in the k-nearest neighbours just due to their large numbers.

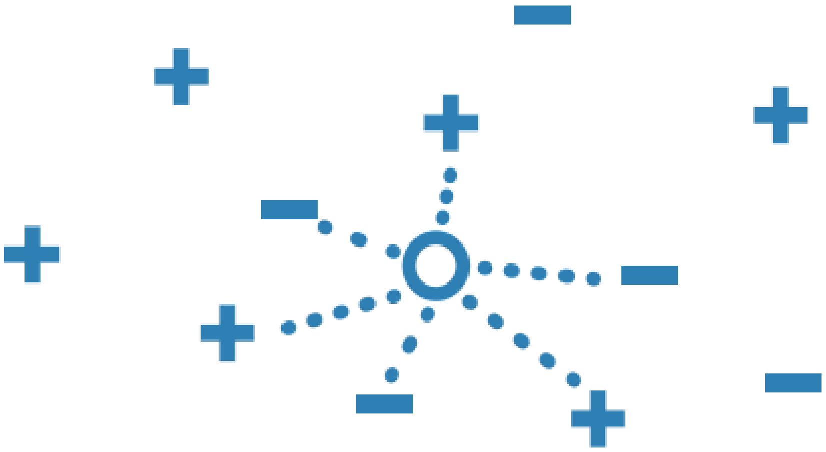


Figure 2-20 Illustration of machine learning K-Nearest Neighbour classifier

Since all the training data is kept so as the number of data points grow, the requirement for both, the time taken for testing and memory usage increases. Although the training time in comparison might be less. It doesn’t make an assumption about the data so it is useful and robust with regards to the search space, for instance, classes don't have to be linearly separable. It has a relatively high accuracy but not competitive in comparison to better supervised learning models. It is versatile as it is suitable for classification or regression.

2.6.5 SVM

It classifies data by finding a linear decision boundary or a hyperplane to separate the classes on the basis of maximum margin. Margin can be defined as the amount of space, or separation, between the two classes as defined by a hyperplane. The best hyperplane being the one with the largest margin between the two classes. When a linear boundary separation is not possible then the algorithm employs a loss function to penalise points on the wrong side of the hyperplane. It uses a kernel transform to transform nonlinearly separable data into higher dimensions, where a linear decision boundary can be found. In simpler terms, in case of dealing with more than two classes, the model creates a set of binary classification sub-problems, with one SVM learner for each of the sub-problem.

It provides a better generalisation to the model. This means that the classifier accomplishes high performance on the training data as well as it produces high predictive accuracy for the future data from the same distribution as the training data [96]. But on the other hand, they need to be trained and tuned up front, so a lot of time needs to be invested in the model before it can be used. Also, its speed is heavily impacted if the model is used for more than two classes.

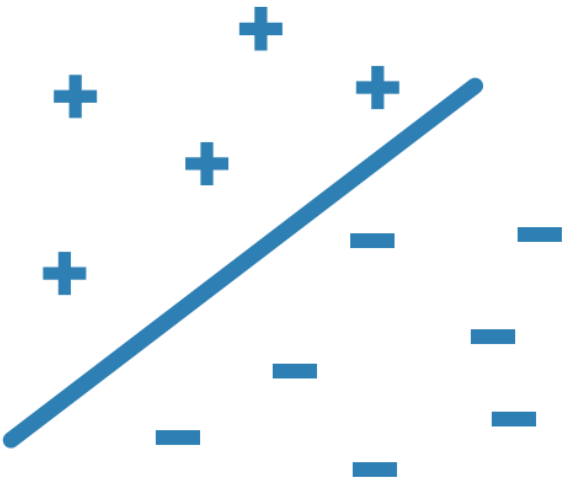


Figure 2-21 Illustration of machine learning SVM Model

2.6.6 Decision Trees

The models are created in the form of a tree structure that can deal with both classification and regression, also known as Classification and Regression Tress (CART). This technique keeps splitting the dataset into smaller subsets until all the observations in one subset belong to one class [97].

They are used to build a model by a recursive binary partition of a labelled dataset into increasingly homogeneous nodes. Different measures can be used to measure this homogeneity. One of them is known as Gini's diversity index. There are other split criterion options like Twoing rule and Maximum deviance reduction (also known as cross entropy). The classification tree tries to optimize to pure nodes containing only one class so when all observations belong to the same class, this index is minimised. At each step the node with the highest G value is split. The splitting process continues until no further subdivision can reduce the Gini index [98]. The final result should be a fully-grown classification tree whose lower nodes include cases belonging to just one class.

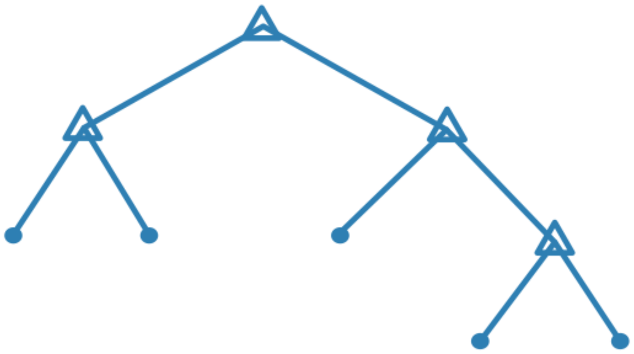


Figure 2-22 Illustration of machine learning Decision Tree Model

The number of branches and the values of weights are determined in the training process. Additional modification, or pruning, may be used to simplify the model. Interpretation of classification trees increases in complexity as the number of terminal nodes increases. It allows to predict responses to data by following the decisions in the tree, from the beginning of the root, right down to a leaf node. Classification trees give responses that are in nominal form such as true or false, while regression trees give the response in numeric form. The complexity of the tree is defined by the number of branch splits and depending on its complexity, they have quick training and prediction speeds, moderate predictive accuracy and low computational memory requirements. Full representation of the path taken from root to leaf can be easily tracked which is useful if results need to be shared.

2.6.7 Ensemble

When unique trees are used for classification, they are highly sensitive to the input data and with small changes in the dataset, they can produce completely different models. Ensemble learning techniques have recently received much interest as a tool to overcome this limitation of decision trees, to obtain better predictive performance. These models are a combination of weighted base classifiers that achieve higher predictive performance in comparison to other individual underlying classifiers.

The choice of algorithm selected is important as it affects the quality of the ensemble. They are tend to be slow to fit because they often need many learners. The information needed to create a classification ensemble is Predictor Data (X), Responses (Y), Ensemble aggregation method, number of ensemble learning cycles and weak learners. We can divide the ensemble methods into two groups:

- Boosted trees - Base learners are biased and are generated sequentially. Basic motivation is to exploit independence between the base learners. The overall performance can be boosted by weighing previously mislabelled examples with higher weight. For example, AdaBoost with Decision Tree learners and Gradient boosting Machines [99]
- Bagged trees– Base learners try to decrease variance and are generated in parallel. Basic motivation is to exploit independence between the base learners since the error can be reduced dramatically by averaging. For example, Random Forest

One of the examples is AdaBoost that works on the same principle of using many smaller, weaker models and combine them together into a final summed prediction. It is called adaptive because it uses multiple iterations to generate a single composite strong learner. The basic idea of boosting is to add new models to the ensemble in a sequence for a number of sequences. In each iteration, a new weak model is added to the ensemble and a weighting vector is adjusted to focus on examples that were misclassified in previous rounds. The result is a classifier that has higher accuracy than the weak learners' classifiers. AdaBoost is more effective at reducing bias than bagging, but bagging is more effective than AdaBoost at reducing variance [100].

2.6.8 Random Forest

One of the most used ensemble learning methods is Bagging method. It consists of trees that are trained independently by re-sampling on the same by bootstrapping. That is generating new datasets of the same size as the initial one by random sampling with replacement. Some of the original samples occur again in the newly generated dataset. The observations that are not included in any of the new datasets are called out-of-bag observations. The trees obtained are not pruned and are used to classify the out-of-bag observations. As each initial observations is included inside the out-of-bag of several trees, its class is estimated several times. The final estimation assigns each observation to the most voted class [101].

Random forest (RF) classifier is a bagging based method. It is an ensemble of large number of decision trees around 500 to 2000 [102] by using bootstrapping that run randomly for a specified number of times. Each run votes for the most accurate results given the training set passed through the classifier. The result is based on the most voted for tree.

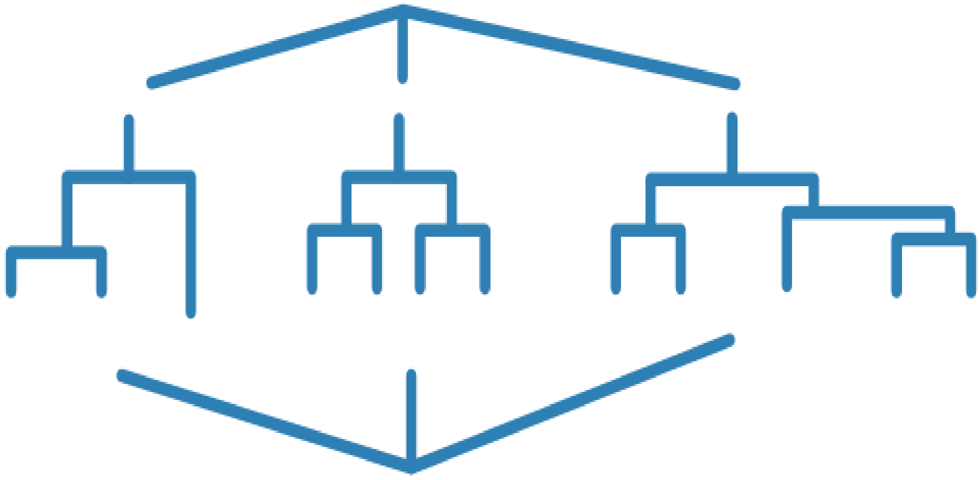


Figure 2-23 Illustration of machine learning Random Forest model

The major reason for the popularity of a random forest instead of a decision tree, is that it combines the predictions of many decision trees into a single model. The reasoning is that a single ensemble model built, with even many mediocre models, will still be better than one good model. They are less prone to overfitting since the classification error of one permutation can be overcome by the ensemble of permutations. This way the large number of trees reduces generalisation error. Over-fitting can occur with a flexible model like decision trees where the model memorises the training data and learn any noise in the data as well, which makes it difficult to predict the test data. It is less computationally expensive and does not require a GPU to finish training. A random forest can give you a different interpretation of a decision tree but with better performance.

2.7 Neural Networks

Deep learning (DL) methods are slowly gaining popularity over traditional machine learning methods. One of the big challenges with traditional ML models is a process called feature extraction. It can be challenging to determine in advance which features should be used for a particular problem, as such decisions require domain knowledge and are naturally heuristic. All successive steps can become futile with a poor choice of features [93]. The capability of learning to focus on the right features by themselves, requiring little guidance, makes deep learning an extremely powerful tool for modern machine learning. At present, deep learning is the state-of-the-art approach to machine learning, and is prominent in numerous fields, such as computer vision, speech recognition, and natural language processing. Hence the potential of deep learning is growing especially for heavily complicated use cases like image recognition.

It is also known by the name of deep structured learning or hierarchical learning. One high-level definition is that it is a class of machine learning techniques that exploit many layers of non-linear information processing for supervised or unsupervised feature extraction and transformation, and for pattern analysis and classification. Since its emergence in 2006 as a new area of machine learning, it is being applied in a vast range of applications. The key note is that these layers of features are not designed by human engineers, they are learned from data using a general- purpose learning procedure. The significant aspect of deep learning algorithms lies in neural networks which are inspired from neurons in human brain and, try to replicate human learning functionality.

2.7.1 ANN

Artificial neural network (ANN) can be defined in many different ways such as, it is a collection of nodes or units that are inter-connected via adaptive weights to form a directed weighted graph, which can learn distributed representations and ultimately the task at hand. They are loosely modelled after the structure of biological neural networks, such as the neural network in a human brain. They can also be defined as, a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs. In general, such systems are able to improve their performance by considering previous examples, and do not require any programming specific to the task they are learning about. They were found to be particularly useful in domains where knowledge and decision processes are poorly understood, or the data is subject to uncertainty [103].

They have been successfully used in a wide range of applications involving pattern recognition and classification, object recognition in image processing, control systems in engineering, forecasting and in applications in commerce, retail industries, engineering and biomedicine [104]. Some of the major reasons for their popularity are as follows:

Because they have a proven ability by which they can derive meaning from complicated or imprecise data, they can be used to extract patterns and detect trends [105] that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyse. This expert can then be used to provide projections given new situations of interest and answer those situations.

Another big advantage is its ability to learn how to do tasks based on the data given for training or initial experience known as adaptive learning [106] as shown in figure 2.24. Artificial neural networks can modify their behaviour in the response to their environment. This factor, more than any other, is responsible for the interest they have received. When they are shown a set of inputs, which perhaps have specific desired output, they self-adjust to produce consistent responses.

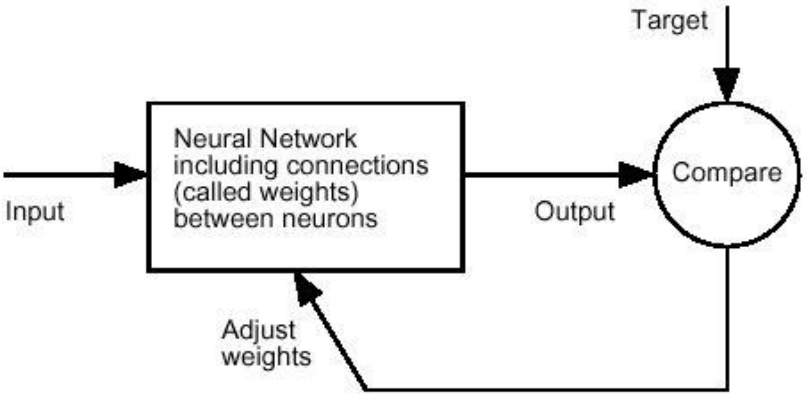

Target
Neural Network including connections (called weights) between neurons
Input
Output
Compare
Adjust weights

Figure 2-24 An ANN performing adaptive learning; [106]

Thirdly an ANN can create its own organisation or representation of the information it receives during learning time which means that it can self-organise itself [107].

2.7.2 Basics of neural networks

The inventor of one of the first neuro-computers, Dr Robert Hecht-Nielsen has provided the most basic and simple definition of an artificial neural network. He has defined it as a computing system made of a number of simple and highly interconnected processing elements that process information by their dynamic state response to external inputs.

ANNs are processing devices, algorithms or actual hardware that are roughly modelled after the neuronal structure of the mammalian cerebral cortex but on much smaller scales. A large ANN might have hundreds or thousands of processor units, whereas a brain has **billions** of neurons with a corresponding increase in magnitude of their overall interaction and emergent behaviour.

Simple neuron

An artificial neuron is a device with many inputs and one output. The neuron has two modes of operation; the training mode and the using mode. In the training mode, the neuron can be trained to fire or not fire, for particular input patterns. In the using mode, when a taught input pattern is detected at the input, its associated output becomes the current output. If the input pattern does not belong in the taught list of input patterns, the firing rule is used to determine whether to fire or not. Figure 2.25 shows a single neuron with its working.

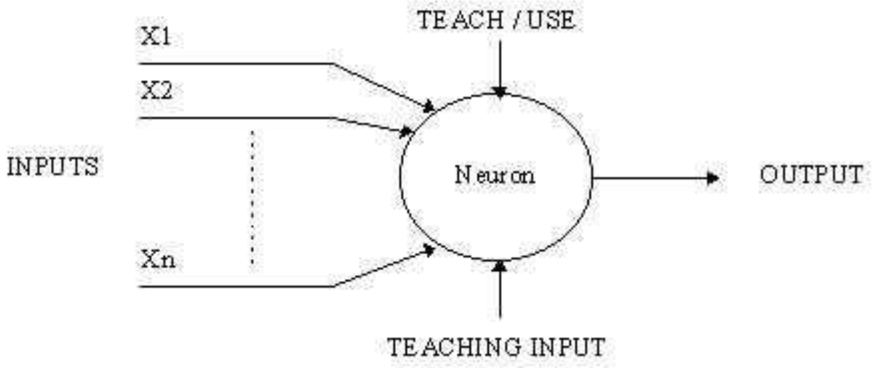


Figure 2-25 A single simple neuron [108]

Complicated neuron

The previous neuron doesn't do anything that conventional computers don't do already. A more sophisticated neuron shown in figure 2.26 is the McCulloch and Pitts model (MCP). The difference from the previous model is that the inputs are 'weighted', the effect that each input has at decision making is dependent on the weight of the particular input. The weight of an input is a number which when multiplied with the input gives the weighted input. These weighted inputs are then added together and if they exceed a pre-set threshold value, the neuron fires. In any other case the neuron does not fire.

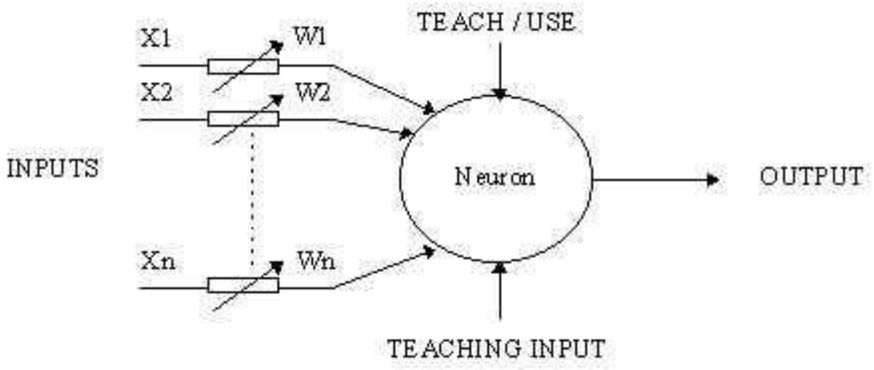


Figure 2-26 Workflow of an MCP neuron [108]

In mathematical terms, the neuron fires if and only if

$$X_1W_1 + X_2W_2 + X_3W_3 + \cdots > T$$

The addition of input weights and of the threshold makes this neuron a very flexible and powerful one. The MCP neuron has the ability to adapt to a particular situation by changing its weights and/or threshold. Various algorithms exist that cause the neuron to 'adapt'; the most used ones are the Delta rule and the back error propagation. The former is used in feed-forward networks and the latter in feedback networks.

Network layers

The most common type of artificial neural network consists of three layers of units: a layer of "input" units is connected to a layer of "hidden" units, which is connected to a layer of "output" units [109]. The activity of the input units represents the raw information that is fed into the network. The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units. The behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and output units. This simple type of network as shown in figure 2.11 is interesting because the hidden units are free to construct their own representations of the input. The weights between the input and hidden units determine when each hidden unit is active, and so by modifying these weights, a hidden unit can choose what it represents.

Architecture

Feed-forward networks: These networks allow signals to travel one way only [70] from input to output. There is no feedback loop i.e. the output of any layer does not affect that same layer. Feed-forward ANNs tend to be straight forward networks that associate inputs with outputs. They are extensively used in pattern recognition. This type of organisation is also referred to as bottom-up or top-down.

Feedback networks: They can have signals travelling in both directions [110] by introducing loops in the network. Feedback networks are very powerful and can get extremely complicated. Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point. Feedback architectures are also referred to as interactive or recurrent. This technique is not a new technique. The mathematical model of back-propagation was first developed in ‘70s and was originally reused by Yann LeCun. This was one of the first real applications of Deep Learning. However a major step forward was made in 2012 when Geoff Hinton won the ImageNet competition by using Deep Learning network, outperforming other more classic algorithms.

Transfer Function

The behaviour of an ANN depends on both the weights and the input-output function that is specified for the units. These input-output functions are also called transfer function [45]. This function typically falls into one of three categories: linear (or ramp) where the output activity is proportional to the total weighted output [103]; threshold where the output is set at one of two levels, depending on whether the total input is greater than or less than some threshold value [51]; or sigmoid where the output varies continuously but not linearly as the input changes [46]. Sigmoid units bear a greater resemblance to real neurones than do linear or threshold units.

To make a neural network perform some specific task, we must choose how the units are connected to one another and we must set the weights on the connections appropriately. The connections determine whether it is possible for one unit to influence another. The weights specify the strength of the influence [107].

Training models

There are two kinds of training models: supervised and unsupervised methods. The supervised methods are those which incorporate an external influence, so that each output unit is told what its desired response to input signals ought to be. The idea behind this model is to compare the output with the desired output. Paradigms of supervised learning include error-correction learning, reinforcement learning and stochastic learning. An important issue concerning supervised learning is the problem of error convergence that is the minimisation of error between the desired and computed unit values. The aim is to determine a set of weights which minimises the error.

The unsupervised methods are those which don't use any external human influence and are based upon only local information. It is also referred to as self-organisation, in the sense that it self-organises data presented to the network and detects their emergent collective properties.

A neural network learns off-line if the learning phase and the operation phase are distinct and it learns on-line if it learns and operates at the same time. Usually, supervised learning is performed off-line, whereas unsupervised learning is performed on-line. Supervised learning procedures have achieved a reputation for producing good results in practical applications, which are gaining in popularity [82].

2.7.3 CNN

A Convolutional Neural Network (CNN) is a specific class of deep, feed-forward artificial neural networks that use convolution instead of general matrix multiplication in at least one of [its] layers [92]. They learn to recognize patterns depending upon training classes. The theoretical principles of deep learning, including the ideas behind Convolutional Neural Networks are not new, however they were limited by the computing resources available at that time. The availability of powerful processing units (GPUs) and large quantities of data available for training deep neural networks has helped them gain new prominence in recent years. Hence many significant achievements in this area have occurred in mainly just the last three to four years and there is a large number of research still ongoing on this topic. They have been successfully applied for pattern recognition problems, for example sentence classification and modelling, face recognition, image classification and object detection in images.

Convolutional Neural Networks (CNNs) are specialized for the use with inputs that have a grid-like topology. For example when working with images, each input feature map would be an array that contains one colour channel of the input image, while each output feature map would be the extraction of one particular feature at all locations of the input. Regular ANN, i.e. networks that are made up of only linear and activation layers, are not able to scale to large data, such as large images, very well. For instance, an image with a resolution of 1920 × 1080 and 3 colour channels, would require 1920 × 1080 × 3 = 6220800 parameters for just a single neuron. By foregoing general matrix multiplication in favour of convolution, thereby only connecting parts of an input that have a strong spatial relationship, the number of connections is greatly reduced. This also results in their trained features being translation invariant, i.e. features are being recognized by the network even if they vary in some way [111]. Fully-connected layers are “regular” neural network layers that are connected to all activations in the previous layer. As such, their activation can be calculated with a matrix multiplication.

2.7.4 Pre-trained Network Architectures

State of the art in CNN's Krizhevsky et al. created AlexNet [112], GoogLeNet by Szegedy et al. [112] and VGG16 by Simonyan & Zisserman [113]. Such CNN Net could be applied using transfer learning for the new database, Gao and Mosalam [114] applied the transfer learning using VGG16 on the image based structural recognition.

The number of nodes in the input and output layers can generally be determined by the dimensionality of the problem. There are a few pre-trained networks that are used for segmentation tasks, which are in the following sections.

2.7.4.1 AlexNet

It is one of the pioneer Deep Neural Net that was developed [112]. It won the ImageNet challenge in 2012 by a large margin as one of the first CNNs to be used for image classification problems. It was initially trained to recognize 1000 different objects. It achieved a top5 error rate of 15.3 %, over 10 % better than the runner-up that did not apply CNNs [115]. From then onwards, more or less following nets are based on its architecture.

It contains five convolutional layers from C1 to C5 plus three fully connected layers from FC6 to FC8 with the final one being a softmax output layer. Additionally, it has three layers applying max pooling. It uses ReLU activation function instead of Sigmoid or Tanh functions which makes it speed up by more than 5 times with same accuracy. To reduce overfitting, dropout is applied in the first two fully-connected layers with a probability of 0.5 for each neuron. At test time, the output of each neuron is multiplied by 0.5. It is designed as a fully convolutional network in [57], however it is not able to accomplish the same image segmentation performance as its more complex and innovative rivals, VGG and GoogLeNet architectures.

2.7.4.2 GoogLeNet

This architecture was developed based around the idea of Network in Network’ structures [116] and achieved the new state of the art for image classification in the ILSVRC14. Previously, such as in AlexNet, and VGGNet, conv size is fixed for each layer. But this architecture is based on a concept of inception module that is a technique that have different sizes or types of convolutions for the same input and stacking all the outputs. Hence, 1×1 conv, 3×3 conv, 5×5 conv, and 3×3 max pooling are done altogether for the previous input, and stack together again at output.

When images enter, different sizes of convolutions as well as max pooling are applied. Then numerous kinds of features are extracted. After that, all feature maps at different paths are concatenated together as the input of the next module. It also differs from others as it applies global average pooling at the end of the network instead of using fully connected layers. There are 22 layers in total with numerous inception modules connected together to go deeper. It is already a very deep model compared with previous AlexNet but not too deep compared with ResNet invented afterwards.

2.7.4.3 VGGNet

The VGG network architecture was introduced by Simonyan and Zisserman in their 2014 paper [113], where the authors explored the effect that, depth of a convolutional network, has on its image recognition accuracy. This network is characterized by its simplicity by using only 3×3 convolutional layers stacked on top of each other in increasing depth up to 19 weight layers. This showed a significant improvement on the previous state-of-the-art. The team won first and second places in localization and classification tracks respectively at the ImageNet Challenge 2014 submission. There is a number of different network configurations presented in [113], ranging from 11 to 19 weight layers. But due to its depth and number of fully-connected nodes, they weigh quite large and also it is slow to train them.

VGG is used in many deep learning image classification problems; however, sometimes smaller network architectures are needed such as SqueezeNet, GoogLeNet etc. This VGGNet was adapted by [57] specifically the VGG-16 with 16 weight layers, as fully convolutional. Of the three evaluated architectures, the FCN-VGG16 performs the best.

2.7.4.4 ResNet

The deep residual network, or Resnet in short, is a very deep neural network architecture, also known as network-in-network architecture. It was developed [117] with the deepest evaluated network having a depth of 152 layers, about 8 times deeper than the VGGNet architecture. It demonstrates that extremely deep networks can be trained using standard SGD and a reasonable initialization function, by using residual modules. ResNet is a form of exotic architecture that banks on micro-architecture modules, as shown in figure 2.27, unlike the traditional sequential network architectures such as AlexNet and VGG. The term micro-architecture refers to the set of “building blocks” used to construct the network. A collection of micro-architecture building blocks, including the standard CONV, POOL, etc. layers, leads to the macro-architecture itself.

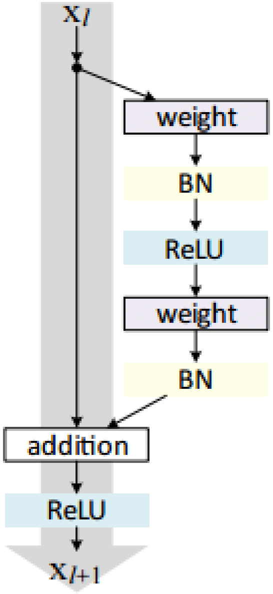


Figure 2-27 The residual module in ResNet as originally proposed by [117]

When using very deep networks, the vanishing gradient problem may prevent weights to update their value, thereby effectively preventing any learning. This effect can cause deeper networks to actually have a higher training error than a shallow network. This problem is solved by the residual networks by adding shortcut connections to the network. Even though ResNet is much deeper than VGG16 and VGG19, the model size is actually substantially smaller due to the usage of global average pooling rather than fully-connected layers.

2.7.4.5 DenseNet

It is a logical extension of ResNet, where the architecture has a fundamental building block where a previous layer is merged into a future layer. Reasoning here is by adding additive merges, the network is forced to learn residuals (errors i.e. diff between some previous layer and current one). In contrast, DenseNet paper [118], proposes concatenating outputs from the previous layers instead of using the summation. It was developed in 2017, so it's a new technique in the field, which got Best Paper Award with over 2000 citations.

In the DenseNet architecture in order to improve accuracy and speed of training, each layer is directly connected to each other layer in a feed-forward fashion. Hence, each layer obtains additional inputs from all preceding layers and passes on its own feature-maps to all subsequent layers. So with the help of concatenation, each layer is receiving a "collective knowledge" from all preceding layers.

2.7.5 Optimisation algorithms

2.7.5.1 Adam

It is derived from the name ‘adaptive moment estimation’, presented in [118], the Adam optimisation algorithm is an alternative to SGD. It has been demonstrated empirically that Adam compares favorably to other stochastic optimization methods in practice.

2.7.5.2 Gradient Descent

In order to find the minimum of the loss function, the gradient descent algorithm can be applied which iteratively moves towards a set of values that minimize the function by taking steps in the negative direction of the function gradient. The learning rate has been set too high in the left image, causing overshooting. In the right image of figure 2.28, the learning rate has been set too small, causing trapping in a local minima [119]

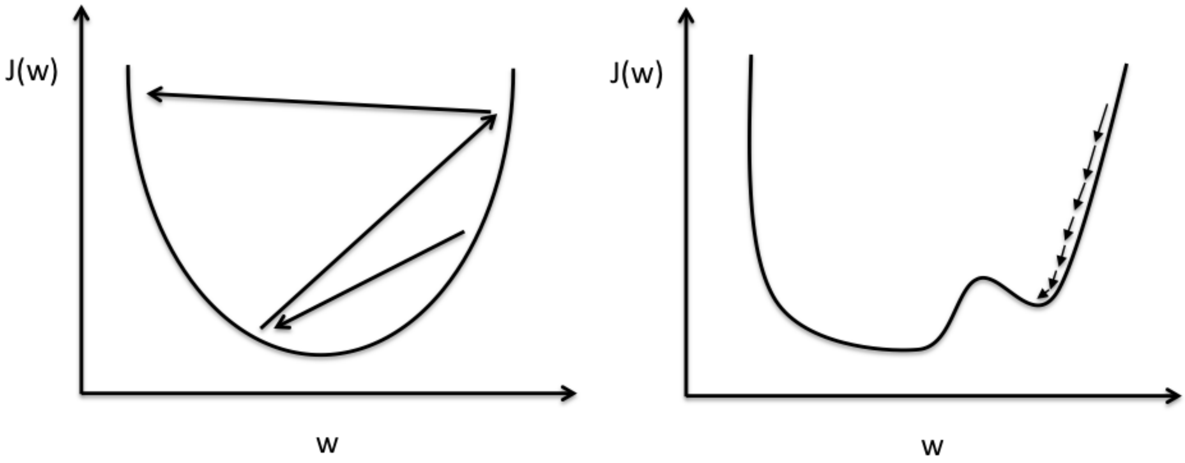


Figure 2-28 Illustration of the gradient descent algorithm. [119]

2.7.5.3 Stochastic Gradient Descent

Stochastic gradient descent (SGD) is a simplification of the regular gradient descent algorithm that does not compute the gradient exactly. Instead, the gradient is estimated for each iteration on the basis of one randomly picked example. By randomly picking each of the examples for each iteration, the hope is that the algorithm will in the long term behave like the regular gradient descent algorithm above.

However, in practice it is usually necessary to decrease the learning rate gradually over time. Unlike the regular gradient descent, the SGD introduces a source of noise by randomly sampling the training examples, which does not vanish even when a minimum is reached. As training with SGD can be slow, by extending it with momentum, i.e. an added velocity to the algorithm, it is possible to accelerate the learning process. The momentum is set to an average of the negative gradient that decays exponentially. [92]

2.7.5.4 Mini Batch Gradient Descent

A variation of gradient descent, the mini-batch gradient descent splits the dataset into small (mini) batches. It serves as a compromise between the regular and stochastic gradient descent, as it computes the gradient for a small number of examples at each step. This has been empirically shown to result in smoother convergence.

2.7.6 Hyper-parameters

For machine learning, there are some parameters that are set before the start of the learning process and they are not influenced by the learning process itself, these are called the hyper-parameters. For deep learning, the training algorithms are almost always iterative, and as such they require a pre-set starting points, that can be used as the basis for the iterations. If this initial state is set badly, the learning time can be greatly increased. It is even possible that the algorithm may never converge at all [92]. There are several strategies for the initial setting of the parameters. The search for the best hyper-parameters is a large part of building any suitable model [120].

2.7.6.1 Initial weights

Weight initialisation has been shown to be one of the most important factors for speeding up neural network training, and as such is an important concept to review in closer detail. With proper data normalization the assumption is that after training approximately half the weights will be set to a positive value, while the other half will be set to a negative value. Accordingly, one strategy might be to set all weights to zero. However, in practice this does not lead to good effects. If every weight is set to zero, then every neuron in the network will compute the same output, which also means all weights will be updated by the same value during backpropagation. This means there is no source of asymmetry between the neurons. Accordingly, while the general question of how to specifically initialize weights is a difficult task and still debated by researchers, it can be said with certainty [92]that the initial parameters for two hidden units, which have the same activation function and are connected to the same inputs, must have different initial parameters. One common approach is the initialization of all weights to a small random value that is not identical for all neurons.

Another approach is to use available pre-trained weights for a model that has been trained on similar data. For example, [57] uses the pre-existing VGG net pre-trained on the ILSVRC and performs fine-tuning.

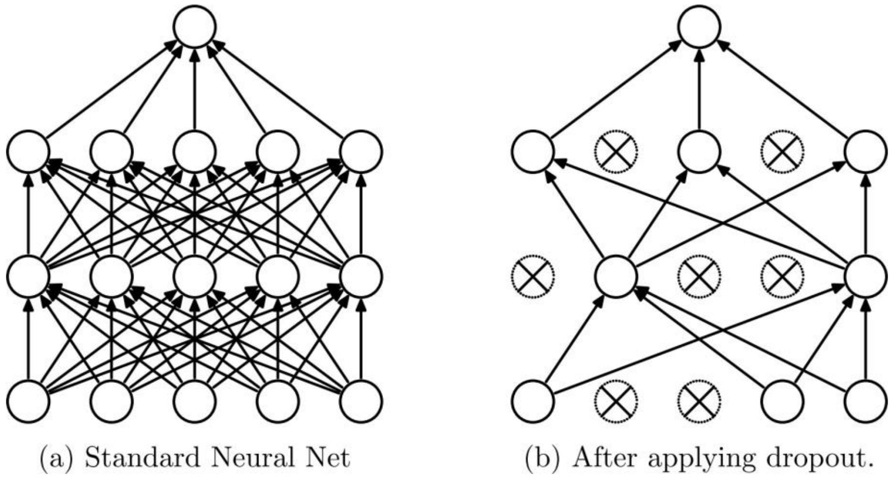
2.7.6.2 Regularisation

Regularization describes “any modification [made] to a learning algorithm that is intended to reduce its generalisation error but not its training error” [92]and as such is an important concern for machine learning in general, that aims to reduce overfitting.

L2 Parameter Regularization: One often-used method of regularization, also referred to as weight decay, adds a penalty to the cost function. The L1 regularization effect is markedly different from L2 regularization. Instead of scaling linearly, the parameter norm penalty is a constant factor. The solutions with L1 regularization are generally sparser, with regards to some parameters having an optimal value of zero. This makes L1 well-suited as a feature selection mechanism, in order to simplify the problem by only taking into account some of the available features.

Dropout: It is a way to prevent overfitting in deep neural networks that have a large number of parameters. By randomly dropping units, as well as their connections, from the network during training, complex co-adaptions on the training data are prevented. This allows training a large number of separate networks in a reasonable amount of time, as all networks can share the same weights for the hidden units that are present. [112]

When testing the network, by using a single un-thinned network that has smaller weights, it is possible to approximate the effect of averaging the predictions of several thinned networks together. The left side shows a standard neural net, while the right side shows how the net can be thinned by applying dropout to it. As shown in figure 2.29, the crossed out units are dropped [121]


(a) Standard Neural Net

(b) After applying dropout.

Figure 2-29 Dropout applied to a neural net. [121]

2.8 Proposed inspection methodology of the project

Based on the above literature review, an inspection methodology is proposed keeping in view aspects of NDT, image processing and machine learning. This has been suggested after careful investigation on how to implement in rail axles and pipelines application for corrosion fatigue. From NDT point of view, it is used as a visual aid to see the component unlike visual inspection which is done by sighting directly by eyes. From NDT point-of-view, it may be classified as a type of visual/optical inspection or taken as a separate type of NDT method known as microscopy. At times, it might be aided with MPI as discussed in chapter 4.

The proposed methodology is a combination of microscopy and advance image processing methods. It will have the advantages of microscopy (and visual inspection) methods and will be able to overcome some of its disadvantages by applying automation. The reason for choosing this particular method is its sensitivity to small surface cracks. Some of the key points of the method are as follows:

1. Portability: The equipment can be taken anywhere as compared to some of the other methods which involve large instrument equipment.
2. Cost of equipment: In comparison to other methods it is cheap hence it is cost effective in relation to the expensive equipment used in other methods like X-ray machines.
3. Used for both defects: Different methods are used for different kind of defects. This method gives good assessment for both the pits and cracks, others may be good for one but not for the other defect.
4. Operator skill: A relatively less skilled operator could inspect in comparison to other methods that require an absolute professional and expert skilled operator for the inspection task.
5. Easy accessibility: The equipment is accessible for operators to conduct the tests, which is a microscopic 2D camera with LED lights in comparison to the large CT scanners.
6. Size of defect: This method deals in micron size defect detection. For such accuracy, either an optical microscope could be used but it is not portable or other methods like MPI that cannot detect that small defects, the defects need to be at least 2mm or bigger to start with to be able to detect anything.
7. Operator cost: The method will minimise the time-consuming manual counting of defects and hence reduces the labour cost required.
8. Time effective: It will give results in less time as compared to the currently time-consuming method to count the number of pits and cracks individually by the skilled operator.

9. Consistent results: Methods which are human-dependent lack the factor of consistency as the interpretation of the same site can differ between two operators.

It is at the high end of the on-surface flaw detection right now as it can be as effective as to be able to detect a micro level defect yet still be extremely portable and cost-effective. This method is suitable for the specific application we require and hence selected due to this reason in comparison to other NDT methods. The other methods like MPI and/or eddy currents are used primarily for crack detection when the flaws are big enough like greater than 1-2mm. This will be further investigated in chapter 3 (section 3.2), by conducting experiments using different NDT methods on same dataset.

2.9 Chapter summary

The knowledge that was gained from the preliminary research helped to establish the fundamentals of this project which included the understanding of the defects to be detected, their morphology, different stages of corrosion; other NDT methods being used in the industry such as MPI, Eddy Current and ultrasound along with their limitations to this specific industry-problem; different image processing methods and their functionalities; and understanding of the key concepts of classification by neural networks and pattern recognition.

Most importantly, it can be seen by the literature review that there is a gap in the industry for an automated inspection system that can analyse and classify cracks and pits as well as make assessments for the pits for 2D images. There is limited research done in this area where they are usually based for crack detection and generally they are not dealing with real on-site complicated images. One of the other research show colour-based segmentation by picking features based on the rust of the component but this doesn’t reflect the actual pits which are hidden underneath the rusting.

The literature survey acknowledges that very few studies have been done for the automation of specifically detecting pitting corrosion. By developing an effective system to assist in structural reliability assessment, it is potentially possible to reduce the maintenance costs and still extend the useful life of a structure. Moreover, the condition of the structure health can be judged in a more objective way. So the system that will be built is based on a real industry problem.

For corrosion assessment generally there are a number of measurements and classification to be done by following an industry standard which is API-579-1 FFS pitting assessment. This standard will be applied if pipelines are used but different analyses are required for high cycle fatigue such as rail axles. For now, these assessments are restricted by the need to carry out manual assessments of the pits.

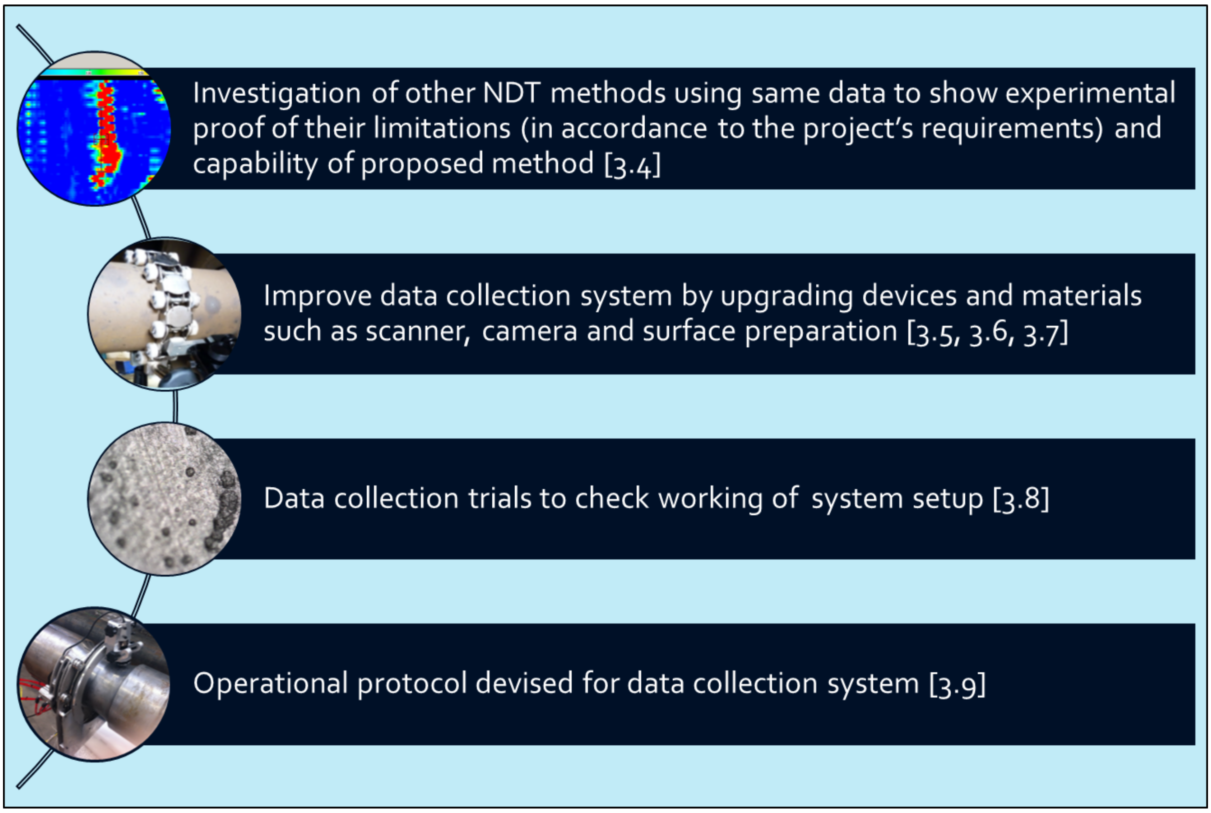
The other purpose of the system is to estimate the remaining life of the axle given the presence of corrosion fatigue. It does this by detecting microscopically small cracks, which appear originating from corrosion pits in the corrosion fatigue process. Then the life is estimated from the average length of the cracks. This means that as an output, the DDS result will have dimensions of all the cracks and its average. This will serve as an input value into the remaining-life software.

Chapter 3
System design and Setup

This chapter discusses different methods and materials to setup a data acquisition system. It starts by discussing reasons of selecting the chosen NDT technique verifying through experimental investigations such as MPI. Later on, discusses the selected materials and equipment, and the pre-requisites for data collection such as the sample selection, camera selection, preparing the surface and then full complete system setup. The initial layout of the detect detection is explained in depth. It further considers the image analysis frameworks to be used for the defect detection system

3.1 INTRODUCTION

In this chapter, project's detailed conceptual framework and refined research design is presented based on the proposed methodology discussed in chapter 2. This chapter is based on phase 1 of the research design called system design and setup, where it discusses the journey of the complete data collection setup including trials, tests, selection and upgrade of devices and procedures. As shown in figure 3.1, it starts with planning experiments with other NDT methods on same data to explore the capacity of other methods in light of the project goal and requirements; secondly it discusses the upgrade of devices and improvements of procedures for data collection; thirdly data capture trials are performed; lastly an operational protocol is devised for data collection system.



Investigation of other NDT methods using same data to show experimental proof of their limitations (in accordance to the project's requirements) and capability of proposed method [3.4]

Improve data collection system by upgrading devices and materials such as scanner, camera and surface preparation [3.5, 3.6, 3.7]

Data collection trials to check working of system setup [3.8]

Operational protocol devised for data collection system [3.9]

Figure 3-1 Design & setup components (Research Design's phase 1)

The process of implementing a strategy for damage detection and characterization for engineering structures is crucial to the safety and continued use of the structures. The goal is to provide a diagnosis of the state of the constituent materials, both of the different parts of the structure as well as the entire structure, at any time during the life of a structure. Specifically, the project is concerned with the identification of cracks and pits of passenger and freight train axles. As the axle deteriorates through its lifetime through fatigue and corrosion mechanisms, it is important to ensure its structural integrity through inspection or monitoring, often using non-destructive testing (NDT) techniques. The methods used are also applicable to other situations involving corrosion or fatigue (e.g. pipework).

3.2 Conceptual Framework

The system expectation, as discussed in project motivation, is that a data collection system is needed that can acquire good quality images keeping in mind that it needs to be portable for on-site visits.

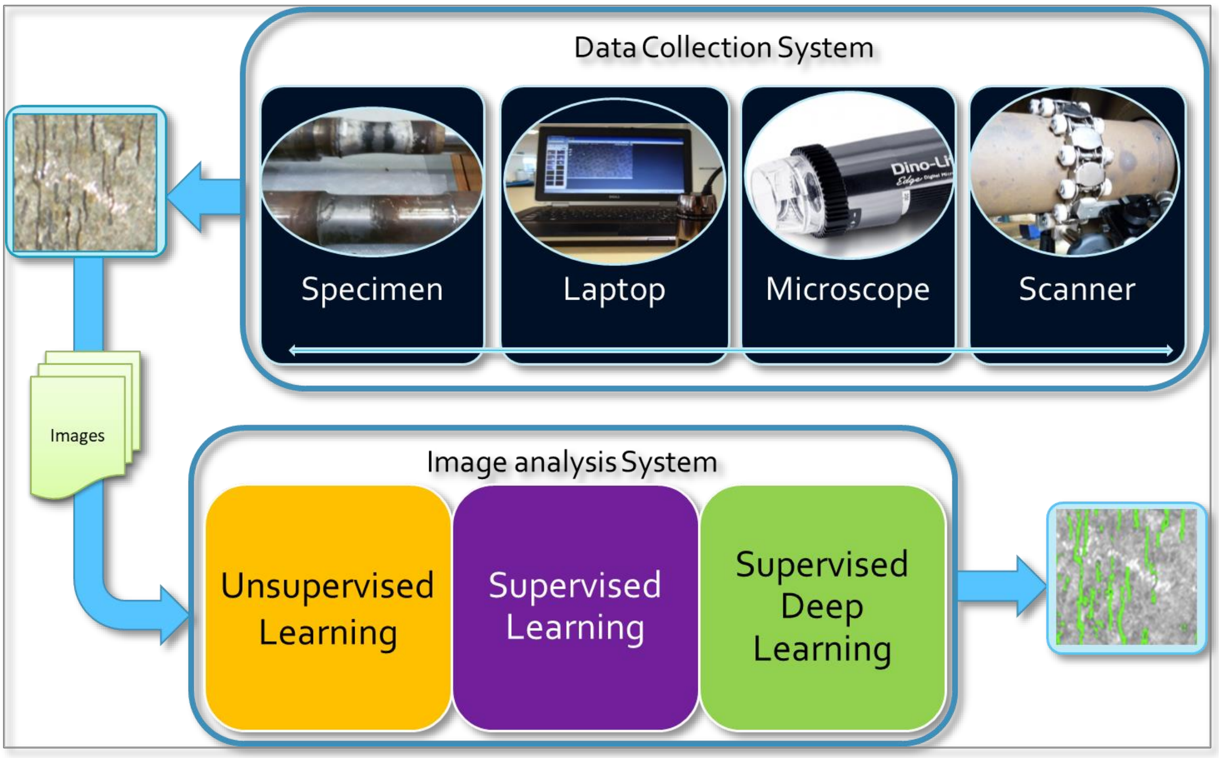


Figure 3-2 Detailed Conceptual Framework of the research project

Once the image has been collected by the system, it can be put in the analysis system that will produce a visual image with localised flaws so that the results don't need to be interpreted. With this note, the conceptual framework for the project, as shown in figure 3.2 was designed. This is an improved and more detailed version of the conceptual framework.

3.3 Research Design

In order to achieve and follow the conceptual framework, the research was split, as shown in figure 3.3, into three major phases consisting of the System (design and setup), Data (creation and labelling), and DDS (defect detection system). Some of the main tasks based on the phases of research design and thesis chapters are as follows:

Phase 1 includes: System design (Chapter 3); System setup (Chapter 3)

Phase 2 includes: Data acquisition (Chapter 4); Data labelling (Chapter 4)

Phase 3 includes:

- Unsupervised classification (Chapter 5)
- Supervised classification (Chapter 6)
- Intelligent supervised classification (Chapter 6)
- Detection system (Chapter 7)
- Conclusions based on main classifier performances (Chapter 5-8)

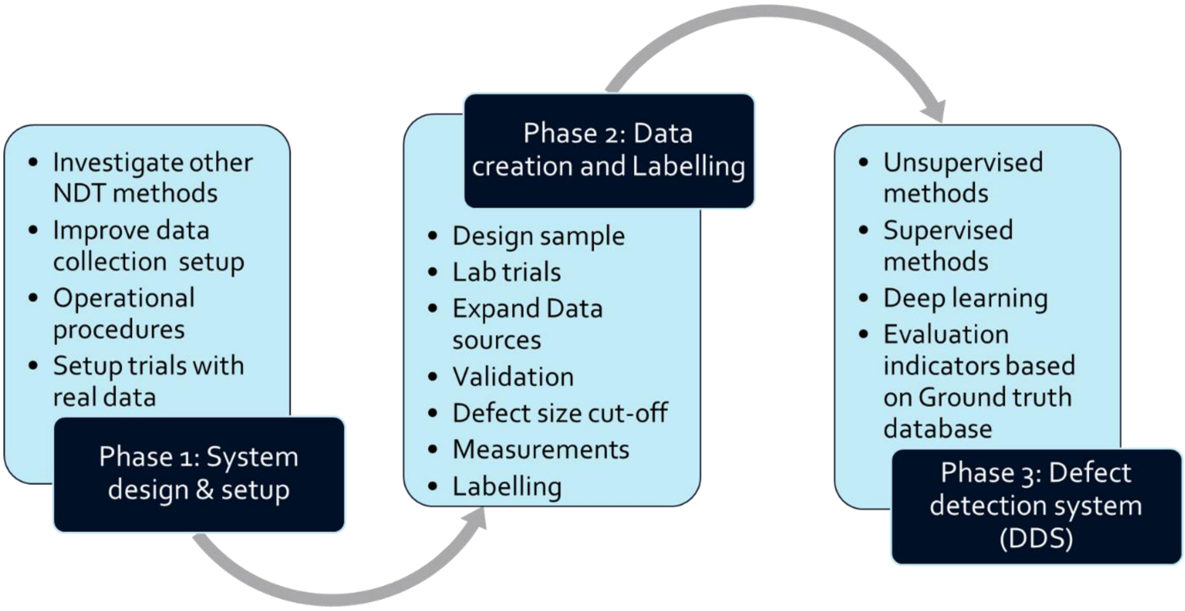

• Investigate other NDT methods
• Improve data collection setup
• Operational procedures
• Setup trials with real data
Phase 1: System design & setup
Phase 2: Data creation and Labelling
• Design sample
• Lab trials
• Expand Data sources
• Validation
• Defect size cut-off
• Measurements
• Labelling
• Unsupervised methods
• Supervised methods
• Deep learning
• Evaluation indicators based on Ground truth database
Phase 3: Defect detection system (DDS)

Figure 3-3 Research design of the project

3.4 Investigation of NDT methods on a same sample data

For the early research stage, different viable and existing NDT techniques were explored in order to carry out corrosion fatigue assessment. This means that a comparison was to be carried out discussing the capability of other NDT methods to detect initial stage corrosion fatigue for the axle crack samples. The prime focus was around the early development of the corrosion fatigue which includes pit initialisation as well as pit-to-crack transition, typically at a stage before conventional NDT detects the cracks. As mentioned earlier, there has not been much work done in this area. The objectives of the work carried out was to establish the best possible method and procedure for detection of pits and micro-cracks within corrosion. The existing NDT method is a visual examination (possibly with a photographic record) and then manual counting and pit assessment. The test sample used for the series of experiments, were two cracks developed by corrosion fatigue in a single axle at TWI (Figure 3.1). The tests were carried out on the 6 o'clock and 12 o'clock rotational positions for the axle.

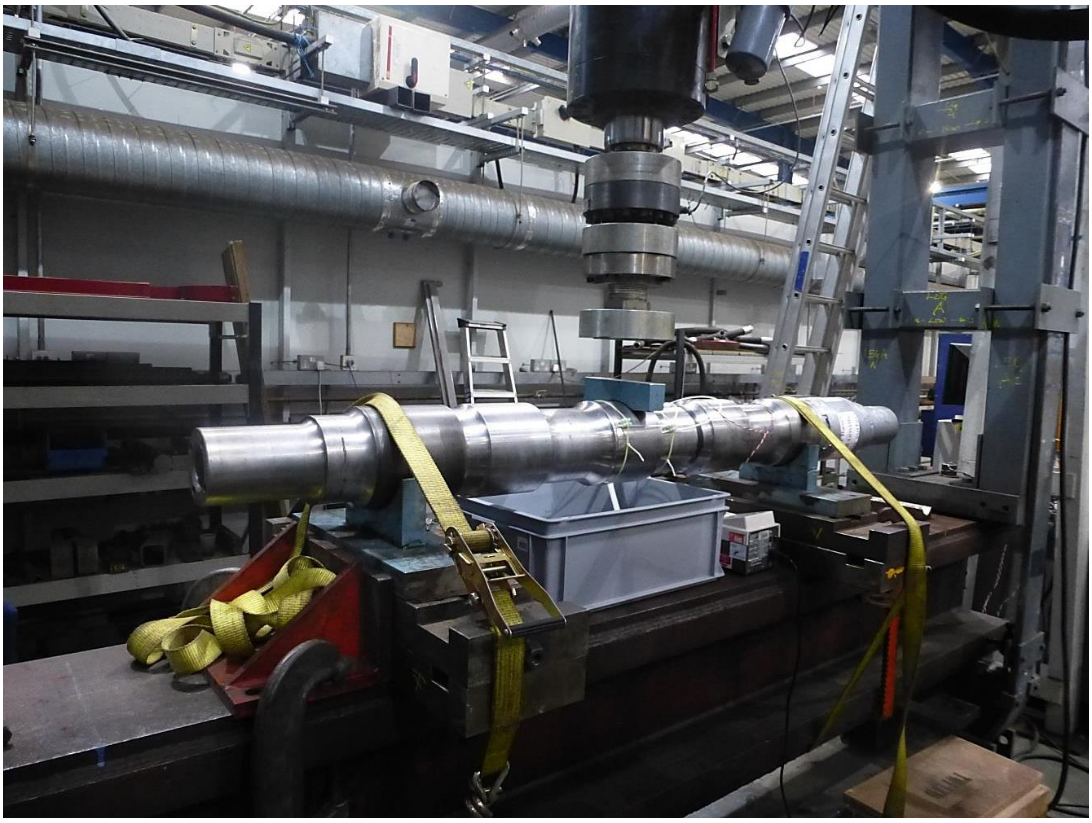


Figure 3-4 Axle set-up on fatigue rig and water circulation

3.4.1 Visual Inspection with USB microscope alone

The USB microscope (see details in section 3.3) was held in a scanner and scans were taken at 5mm square intervals. With details of the microscope magnification used during the tests, it was possible to size the features observed on the microscope images. However the method has its limitations; it is a 2D method with no information of the depth of the pits or the crack. This method depends a lot on the surface condition of the sample. When the sample is shiny due to surface finish, glare can be seen on the image and potentially hides the desired features. (Some of these results are recorded during the monitoring of crack growth and are not directly comparable).

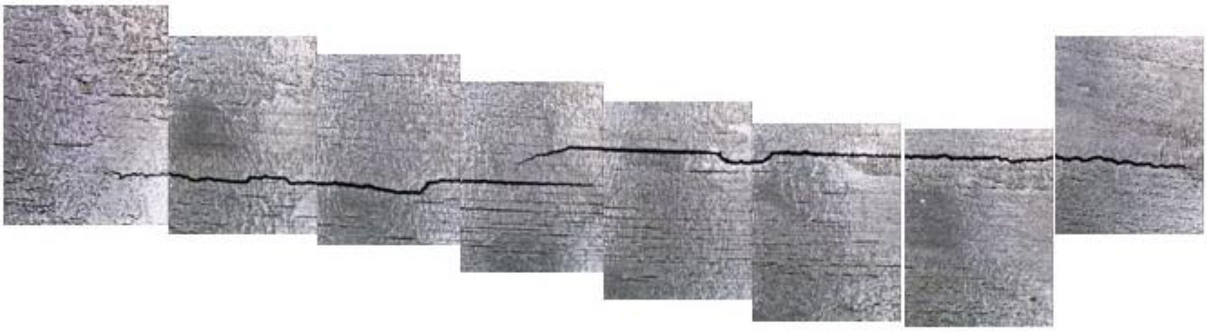


Figure 3-5 Reconstruction of series of images to visually see full crack length

Due to the image size and the interval of shots taken, some of the cracks were stitched as shown in figure 3.5, however it was not applied to all of the images.

3.4.2 MPI with USB microscope without contrast paint

In this case a magnetic yoke was used, in the same way as for standard MPI, except that the residual field was used to observe the cracks. The difference between MPI with contrast paint and without using contrast paint can be seen in figure 3.6. The impact of using contrast paint can sometimes be significant (especially for the detection of cracks).

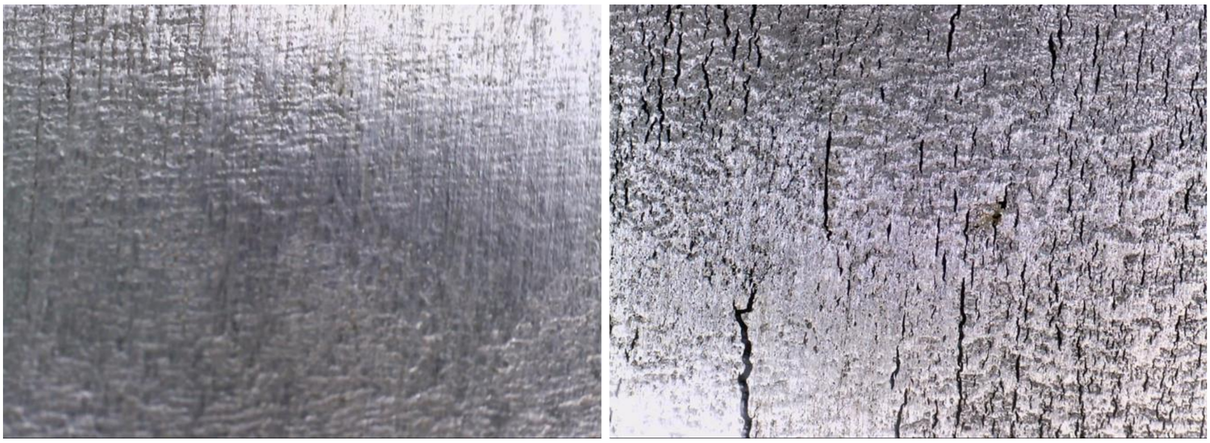


Figure 3-6 - Improvement in crack visibility after applying MPI (without using contrast paint)

3.4.3 MPI by contrast paint and magnetic yoke

This used a standard procedure to BS ISO 9934 where the method can carry out an inspection through non ferromagnetic coatings up to approximately 50 µm. The results are recorded for comparison using a digital camera. The results from the conventional MPI method are shown in figure 3.7 and figure 3.8.

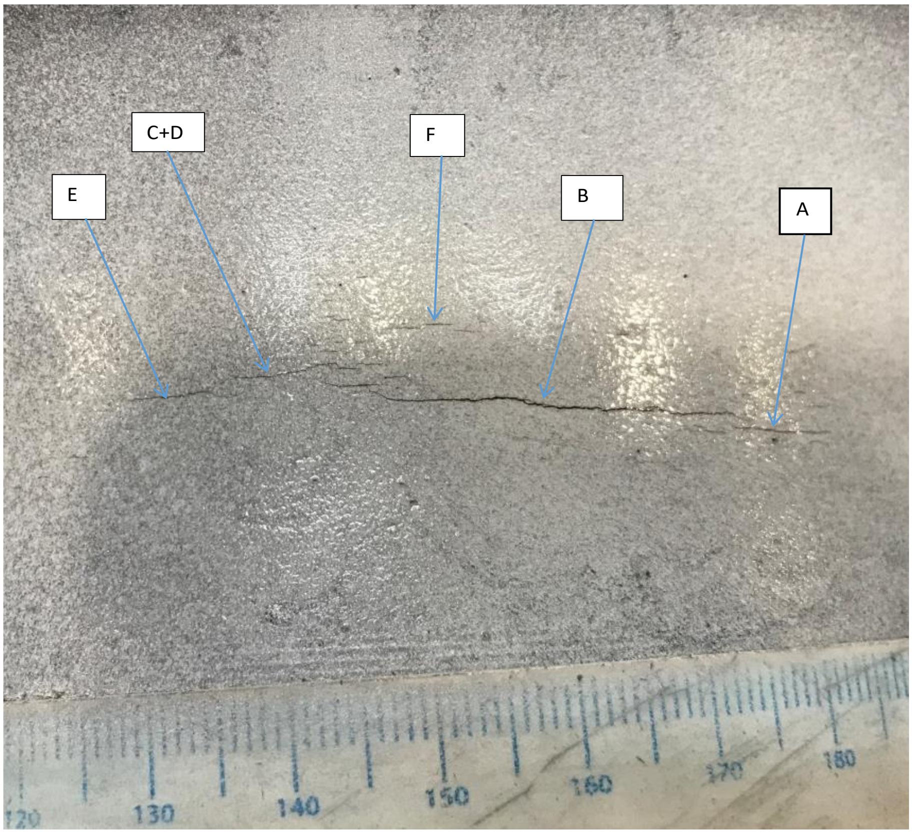


Figure 3-7 Nomenclature for ACPD of Cracks (6 o'clock)

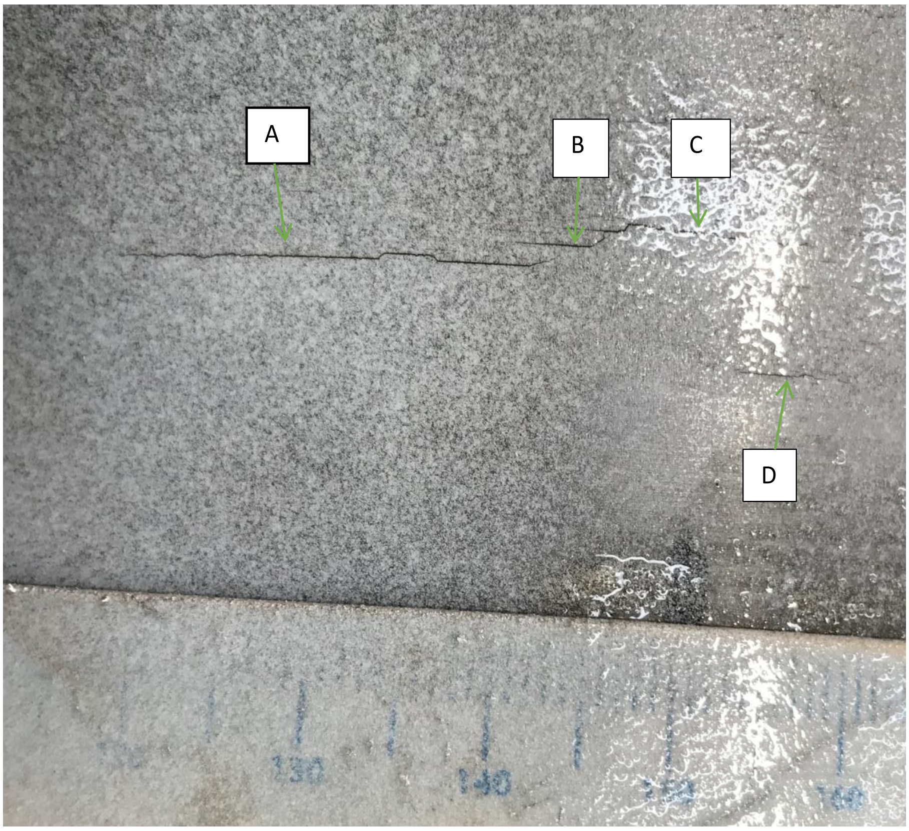


Figure 3-8 Nomenclature for cracks at 12 o'clock

3.4.4 MPI with microscope with contrast paint

The same procedure as above was used except that contrast paint was applied before magnetization.

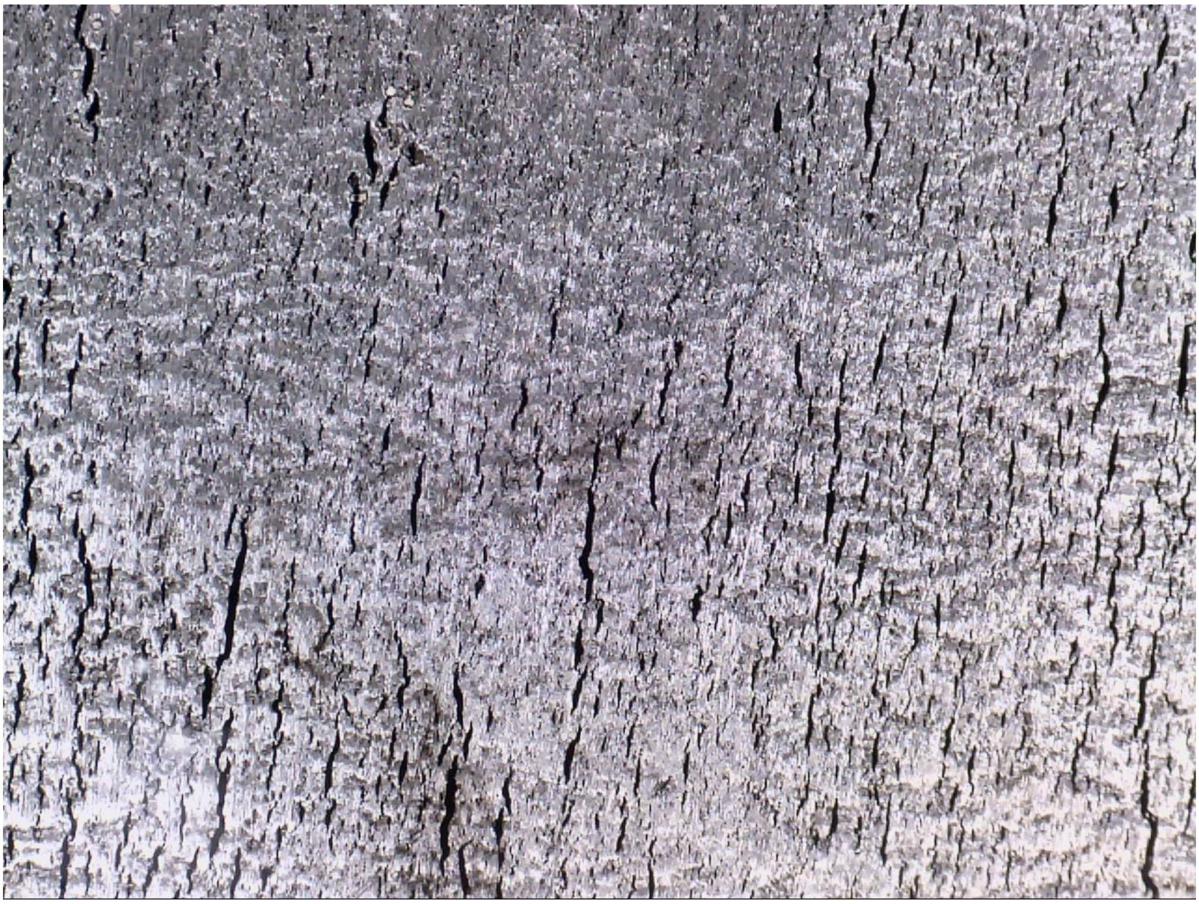


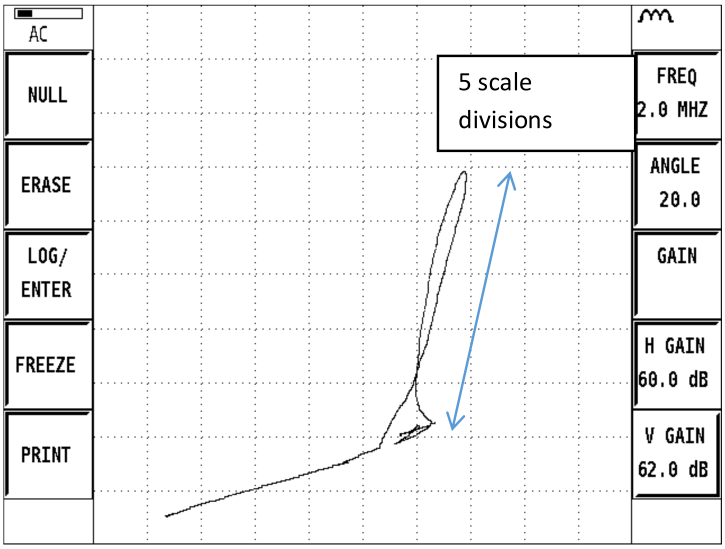
Figure 3-9 Image by microscope after applying MPI with contrast paint

Figure 3.9 displays the results of the microscope and MPI with contrast paint. The contrast paint does highlight the crack. Through the microscope software, measurements of the cracks can be carried out. It can be seen that the use of the magnetic particles (with or without contrast paint) emphasises the cracks.

3.4.5 High Frequency Eddy Current

This method uses an eddy current probe, an instrument typical of the type used to detect cracks in the aerospace industry. The amplitude of an indication from a 0.2mm deep slot is displayed on a screen and is used as a reference, and indications above this value are recorded. This enables a crack length to be obtained by manual marking on the sample of probe position on the sample. A rough estimate of crack depth can be obtained by comparing the amplitude of the crack indication with that from different depth slots.

An example of the display for a crack at the 6 o' clock position is given in figure 3.10. This was obtained using a reference of a 1mm deep slot giving 6 scale divisions. Using the ACPD depth measurement as a reference (see Section 3.2.6), the number of scale divisions for the different crack depths can be shown in figure 3.11. These suggest that the eddy current method produces the most amplitude change for smaller crack depths, with the larger depths (say greater than 1mm) uncertain. The eddy current probe covers a surface area of around 2mm diameter, therefore cracks closer than this will combine to produce the indication. However it is clear that this cannot be used for crack depth sizing.



FREQ 2.0 MHZ

ANGLE 20.0

H GAIN 60.0 dB

V GAIN 62.0 dB

5 scale divisions

Figure 3-10 Eddy current indication for crack A (3mm deep by ACPD) at 6 o'clock

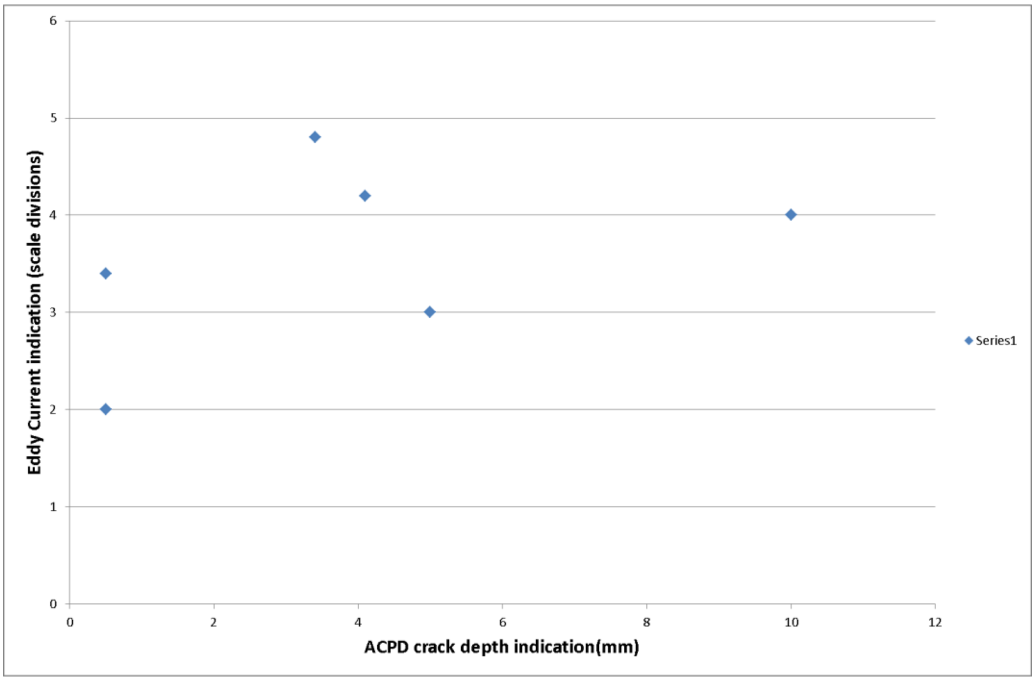


Figure 3-11 Comparison of Eddy Current indication size and ACPD

3.4.6 ACPD (depth measurement)

The ACPD method gives an indication of crack depth. It is used as a measurement rather than a search tool, and has a range of depth measurement up to many mm. The instrument is manufactured by Karl Deutsch. The results of the ACPD measurement using the crack nomenclature used in figure 3.7 and figure 3.8 are shown in figure 3.12 and figure 3.13

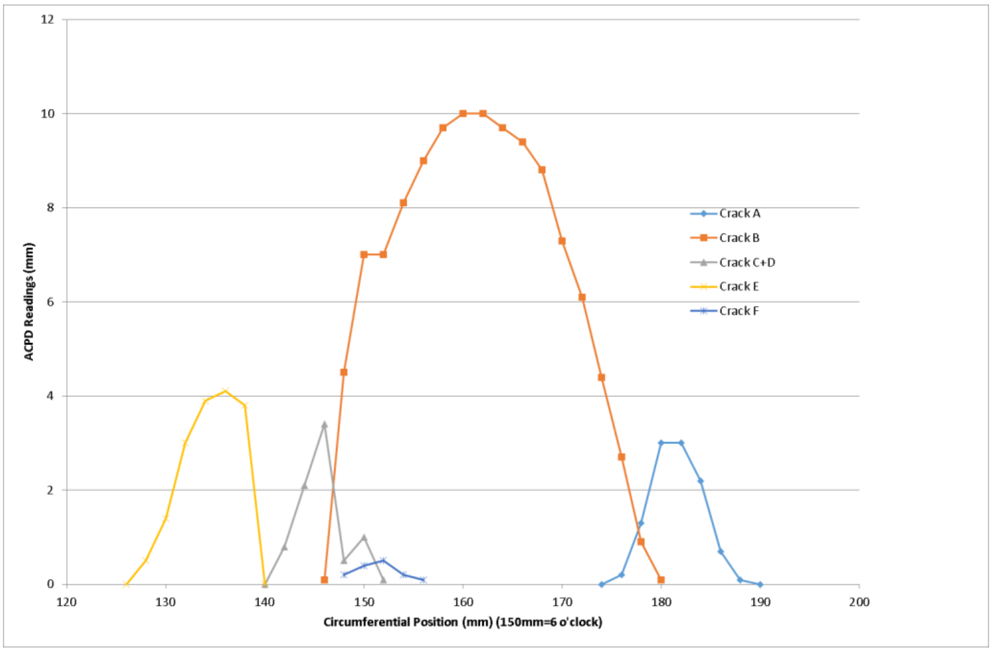


Figure 3-12 ACPD Measurements for cracks at 6 o'clock

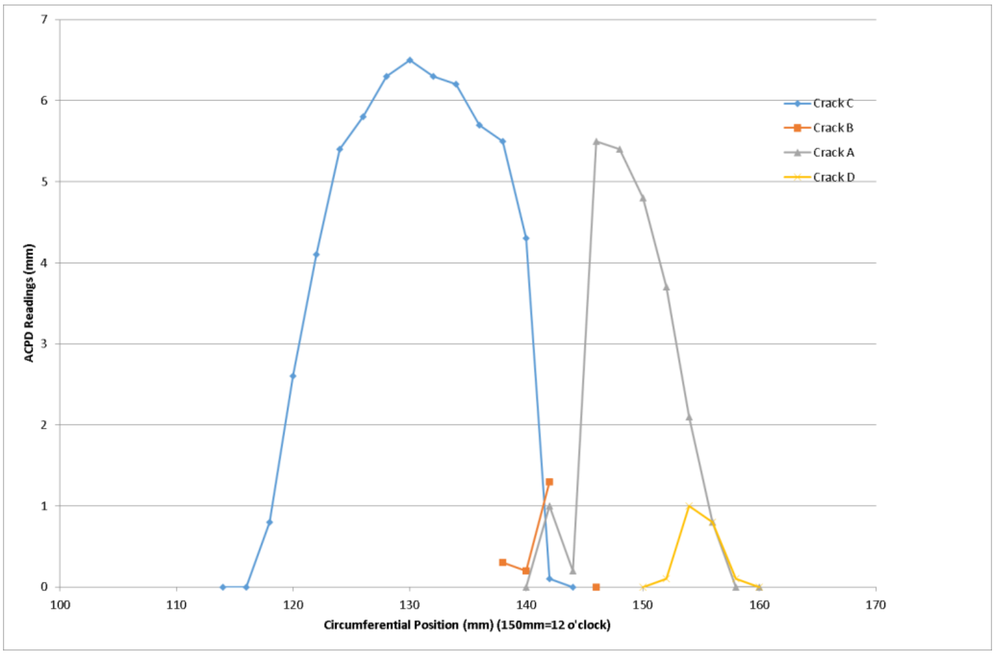


Figure 3-13 ACPD Measurements for cracks at 12 o'clock

The complexity of the cracking may cause some errors in these measurements, as the electrodes are 2mm apart and can sometimes cover more than one crack.

3.4.7 Eddy Current Array (ECA)

Eddy Current Arrays (ECAs) were investigated for the crack detection. The ECAs are multiplexed by the EDDYFI ECTANE instrument and 2 flexible probes were used. The flexible probes were used as the profile of the axle changes. Depending on the area of concern, several scans are required in order to ensure full coverage. The arrays performed scans along the profile of the transition area or along the diameter of the axle. Results from the eddy current array were taken with two flexible array probes, 'short' and 'long'. The probes were set up on a reference block with slots up to 2mm deep. The probes could be scanned around the circumference (with the array axial) or axially (with the array circumferential). The former is better suited for probe handling but the scans on the reference blocks are not valid for comparison as they have to be scanned in the other direction.

Some results are shown in figure 3.14 and 3.15 showing crack at 6o'clock position of axle. Figure 3.14 shows ECA long probe scanned circumferentially axial response (crack direction horizontal above) whereas figure 3.15 shows ECA short probe scanned axially transverse response (crack direction vertical). The crack indication is clear in general but blurred because the effective size of each sensor in the array is several mm.

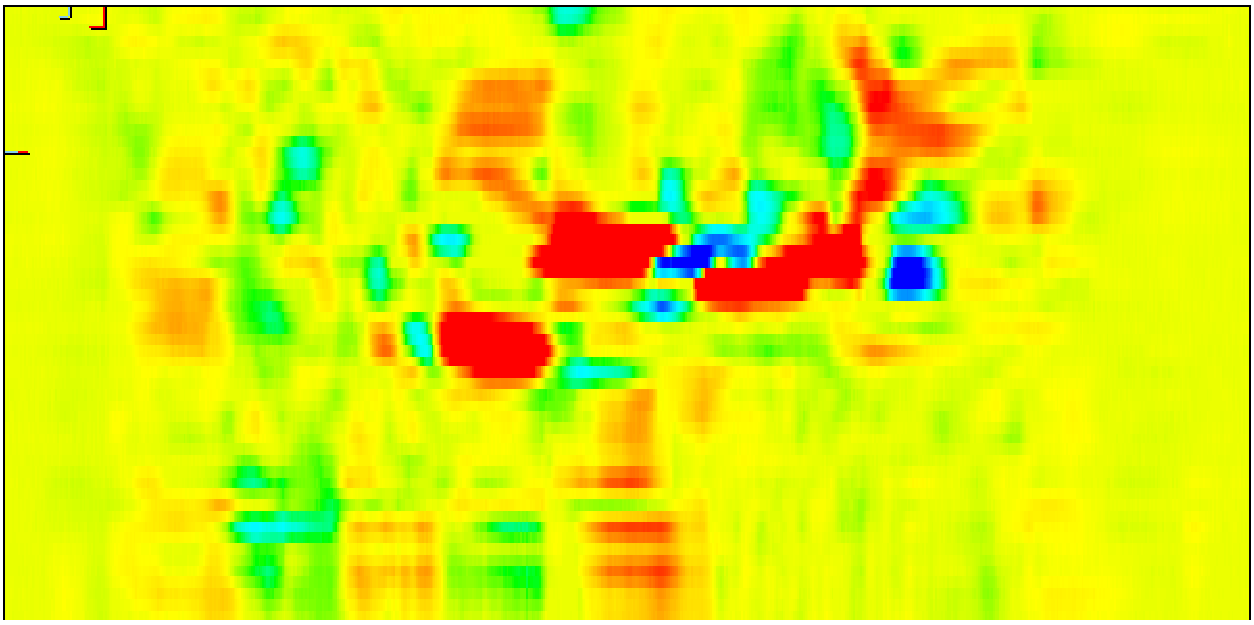


Figure 3-14 ECA long probe scanned circumferentially axial response at 6-o'clock

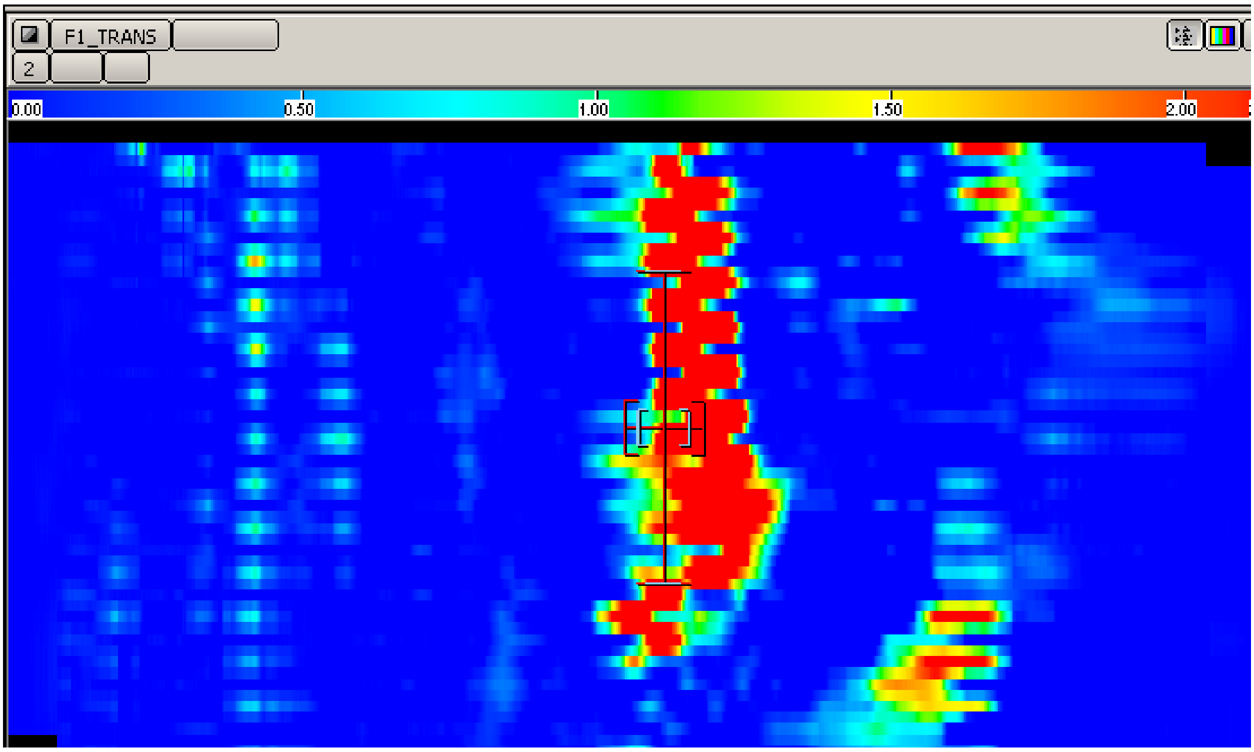
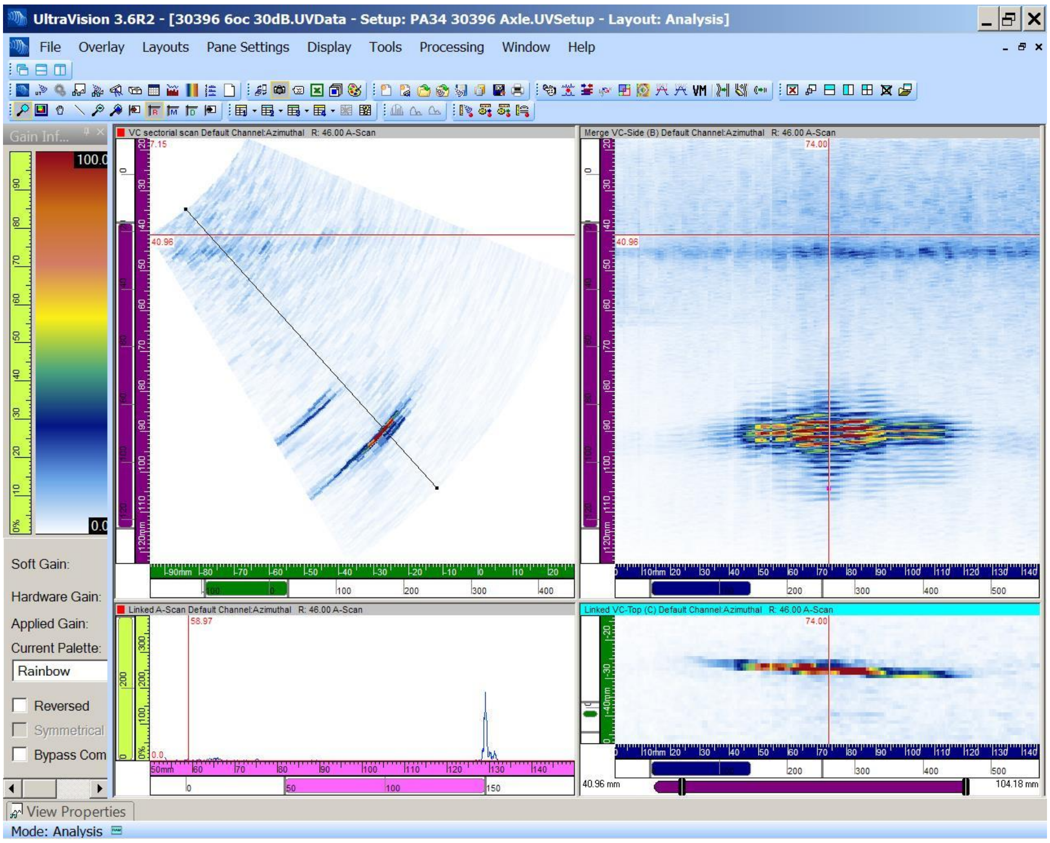


Figure 3-15 ECA short probe scanned axially transverse response at 6-o'clock

3.4.8 Phased Array Ultrasonic Testing (PAUT)

For these tests, ultrasonics with a skip method (where the UT beam was bounced off the interior bore of the axle) was used. This method ensures volumetric coverage where cracks were observed by the methods above. PAUT was the only volumetric method used for the report. These scans, figure. 3.16 & 3.17, show that the corner echo is very clear from these cracks, but the detail of the crack cannot be seen.

PAUT being the only volumetric method used for the tests could display and measure the height of the cracks. The measurements given from the PAUT could be correlated with the ACPD depth measurement. However due to the number of cracks produced and their location, PAUT could not display individually the cracks. In Figure 3.17, where the cracks were located, the CSCAN data grouped the cracks altogether.



UltraVision 3.6R2 - [30396 6oc 30dB.UVData - Setup: PA34 30396 Axle.UVSetup - Layout: Analysis]

Current Palette: Rainbow

□ Reversed

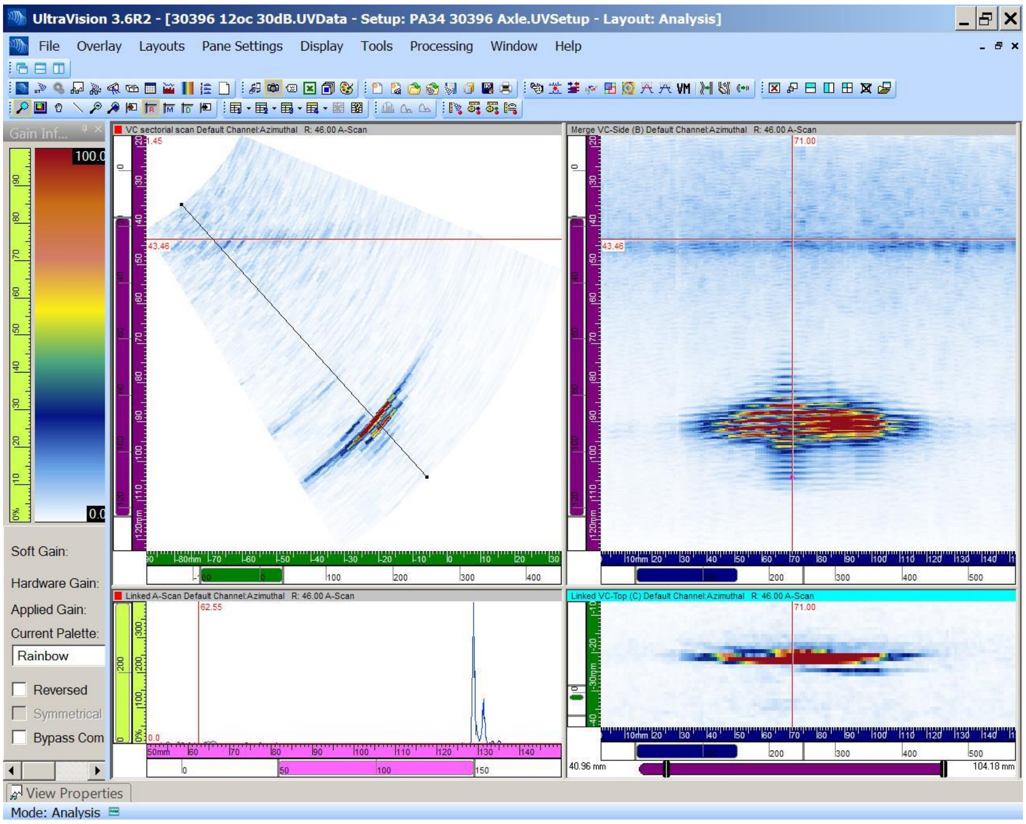
□ Symmetrical

□ Bypass Com

Mode: Analysis

File
Overlay
Layouts
Pane Settings
Display
Tools
Processing
Window
Help
Gain Inf...
100.0
0.0
Soft Gain:
Hardware Gain:
Applied Gain:
View Properties
VC sectorial scan Default Channel:Azimuthal R: 46.00 A-Scan
Merge VC-Side (B) Default Channel:Azimuthal R: 46.00 A-Scan
Linked A-Scan Default Channel:Azimuthal R: 46.00 A-Scan
Linked VC-Top (C) Default Channel:Azimuthal R: 46.00 A-Scan
40.96
74.00
58.97
40.96 mm
104.18 mm

Figure 3-16 CSCAN data from Phased Array Indication at 6-o‘clock



UltraVision 3.6R2 - [30396 12oc 30dB.UVData - Setup: PA34 30396 Axle.UVSetup - Layout: Analysis]

Current Palette: Rainbow

□ Reversed

□ Symmetrical

□ Bypass Com

Mode: Analysis

File
Overlay
Layouts
Pane Settings
Display
Tools
Processing
Window
Help
Gain Inf...
100.0
0.0
Soft Gain:
Hardware Gain:
Applied Gain:
View Properties
VC sectorial scan Default Channel:Azimuthal R: 46.00 A-Scan
Merge VC-Side (B) Default Channel:Azimuthal R: 46.00 A-Scan
Linked A-Scan Default Channel:Azimuthal R: 46.00 A-Scan
Linked VC-Top (C) Default Channel:Azimuthal R: 46.00 A-Scan
43.46
71.00
62.55
40.96 mm
104.18 mm

Figure 3-17 CSCAN data from Phased Array Indication at 12 o'clock

3.4.9 Conclusions

On the basis of the following points, it was concluded that the data will be gathered by using microscopic visual inspection NDT technique with/without MPI.

1. The microscope method is best when the cracking is at a very early stage, typically with crack sizes of the order of 0.2mm.
2. When the cracking is developing, using a magnetic ink and magnetic field enhances the crack indications.
3. The conventional MPI method would be useful for cracks above about 2mm in length (Note this does not necessarily mean that all cracks of 2mm length can be detected).
4. Contrast paint assists by reducing the surface “noise” as well as enhancing the visibility of the crack
5. The conventional high frequency eddy current probe is good for detection of small cracks, but cracks above 1-2mm in depth cannot be distinguished for depth.
6. ACPD- it is suitable for crack depth measurements in individual locations but not for scanning
7. EC – Detects large cracks but is unable to resolve crack details.
8. PAUT is useful to detect the total depth of cracks when the depth has exceeded a certain value and this method does display the content of the volume inspected. However, the resolution of the method is not sufficiently fine to distinguish between cracks. The equipment used for PAUT is more difficult to set-up than the other tests. A skilled technician is required to carry out the tests.

After selecting the appropriate NDT method to gather data (I.e in this case the USB microscope, with and without MPI). The next step is to setup a system for surface inspection, which is covered in the rest of the chapter.

3.5 CAMERA SELECTION

Selection of an appropriate camera is one of the vital steps in detecting and identifying defects in the components by using optical techniques. The USB microscope was initially selected for reasons of cost and portability. It was compared for optical performance with a higher quality optical microscope to ensure that it has sufficient optical quality. They were both used to test the same area of a rail axle containing cracks of the size which needed to be detected for early analysis of crack growth. A replica method was used to obtain samples for the optical microscope. The results showed that the USB microscope chosen had sufficient optical quality for the requirement.

The digital (e.g USB) microscopy images enable features of operation and analysis that may not be available for conventional optical microscopy. They are assisted by dedicated software tools, and permit access to any detail through the computer monitor [122]. They offer modest magnifications (up to about 200×) without the need to use eyepieces, and at very low cost. Images can be recorded and stored using methods similar to a webcam on the computer. The camera is usually fitted with a light source, although extra sources (such as a fibre-optic light) can be used to highlight features of interest in the object. USB microscopes offer the great advantage of being much less bulky than a conventional stereo microscope so they can be used in the field, attached to a laptop computer.

The important factors on the basis of which the camera was selected was minimum sensitivity required and the quality of the images within reasonable economics. From the sensitivity perspective, a microscope was needed that was able to extract initial phases of corrosion fatigue. Information required from the images is to be able detect flaws within corrosion which are around 0.3mm or greater in length as an input for the corrosion fatigue crack growth model. This capability can be seen in figure 3.18 showing Image-A885-L2 as an example.

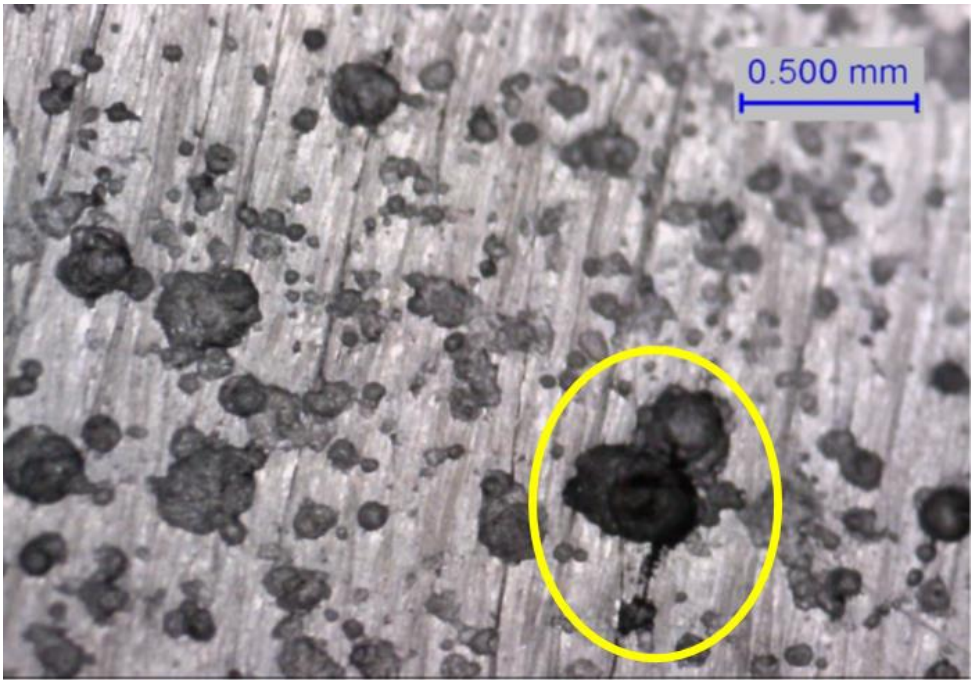


Figure 3-18 Crack (initiating from a pit) with length less than 1mm

Although the camera is satisfactory from the resolution point of view it should be noted that the depth of focus is quite limited, which is important when imaging a curved surface, and the area of an image is only a few mm square, so a large surface requires many images. The impact of these limitations is discussed in more detail below.

3.5.1 Selected model

The camera being used for the visual inspection of the defect is a microscope which includes an adjustable, polarized light source seen in figure 3.19. Reflection and glare are controlled by this advanced polarization feature. It makes this model the ideal choice when examining materials that are reflective or which generate glare such as metals. It is a USB type microscopic camera Dino-Lite AM4113ZT with the following specifications:

- Magnification 10x- 60x, 200x
- Polarizer
- LED lighting
- 1.3 Megapixel resolutions – 1280 x 1024 pixels
- Field of View (FOV) - 19.5 x 15.6mm at 20x to 1.8 x 1.4mm at 220x
- Weight 105g

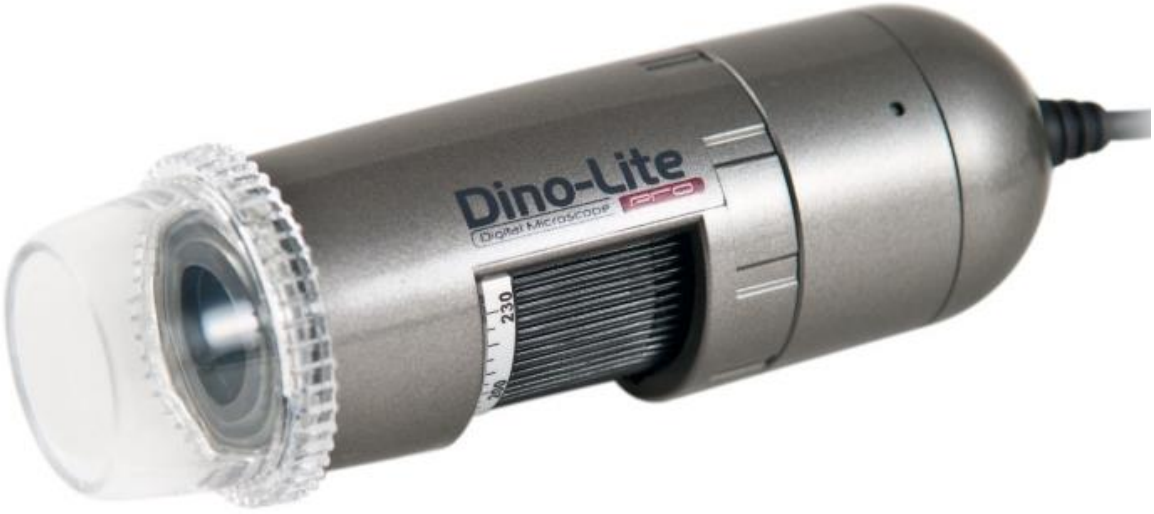


Figure 3-19 Dino-Lite AM4113ZT USB microscope

The microscope detected cracks of similar size to those detected by replica. The number of pixels in the optical microscope per unit distance was roughly three times higher than that in the USB microscope. Therefore at some level measurements of around 2μm the optical microscope will give more accuracy and detail than the USB microscope. However details of this level are not required as the cracks are usually much greater in length than this. Crack width measurements might be affected, although these are not usually used. Although it should be noted that in both cases, there is no direct comparison of the same crack (there are many cracks and it was impossible to identify them), it is clear that the maximum crack size ranges observed by both methods was very similar at each cyclic level. Therefore, it is reasonable to conclude that any measurements of crack lengths will not be affected by using the device selected.

3.5.2 Current upgraded model

A more advanced and sophisticated version of the previously selected microscope was used in the mid stage of the project. The Edge series provide superior image quality and great flexibility in use for professional applications shown in figure 3.20. The high-quality optics provides a very sharp, bright and natural colour image with very low aberration and vignette. The adaptable and exchangeable caps provide for even more flexibility in use for all kinds of applications. By removing the end cap, the full range of magnification can be obtained or more working space can be achieved. A closed cap is included for protection of the lens in dusty environments or when working with liquids and a diffuser cap is supplied to spread the light on the object evenly. The features for the Dino-Lite Edge AM7115MZT USB microscope are as follows:

- Magnification 20x- 220x
- Polarizer
- LED lighting
- 5 Megapixel resolutions - 2592 × 1944 pixels
- Field of View (FOV) - 19.5 x 14.6mm at 20x to 1.8 x 1.3mm at 220x
- Weight 140g

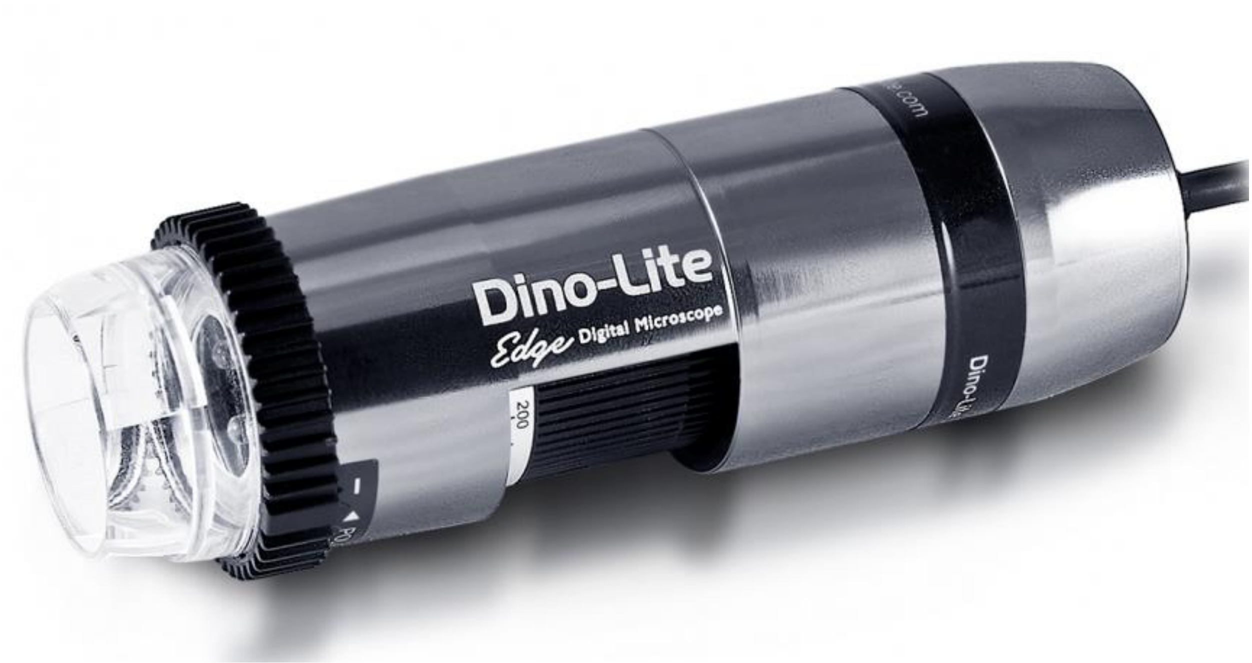


Figure 3-20 Dino-Lite Edge AM7115MZT USB microscope

This camera delivers good image quality with an optical magnification power of 10x - 230x and with illumination flexibility options. With an integrated cutting-edge optics, advanced 5MP sensor, and low-loss MJPEG compression, this camera provides incredibly crisp and fluent imaging with low power consumption. The innovative Flexible LED Control enables partial illumination to enhance edge contrast and provides intensity adjustment for better adaptability. It also has a built-on polarizer with it.

3.5.3 Software installation

The microscope is supplied with its own software called the Dino-Capture. The one which comes with latest edge model is the 'Dino-Capture 2.0'. The Dino-Lite software allows the user to capture image of the inspected specimen then record the image or a video. It also provides features for measuring objects within the image according to the magnification given to the software.

3.5.4 Future upgrade model

A 3D camera for the future will be able to take in account the on-surface depth measurements of the defects as well. This will be useful for the API-579 assessment as one of the requirements of the standard is the depth of pits. The API discusses pipelines as a standard sample that has large scale and dimensions involved. The assessment is not based specifically for rail axles.

3.5.5 Capabilities and challenges of the microscope

The microscope is connected to a laptop via USB cable and can be controlled via the software on the laptop. The camera allows a live preview to be seen on the screen, to obtain a sharp focus when taking images, since the effects of changing the microscope focus control is immediately observable on screen. It could be used directly on the surface structure without any support. But it could also be used with some support like the scanner or the microscope holder.

The magnification values have been studied in order to acquire the appropriate images. It will be interesting to use a higher magnification in order to observe whether other types of information will be available at that value of magnification. Cracks are detectable (visible) at two appropriate magnifications as seen in figure 3.21 showing the same flaw with two pictures, ImageA884-17 and ImageA885-17, to highlight the difference. This gives an option to choose between sensitivity and covered area.

Another camera with 3D capacity could be envisaged with different or higher specs nevertheless the USB microscope is sufficient for the work related to 2D images. The specification of the microscope enabled inspections with different lighting like UV and polarized. It was noted that the UV light was not the first choice however this light could have been optimized if special phosphorescent liquids were used. The quickest way to inspect the components was by using the polarized microscope which required only good cleaning.

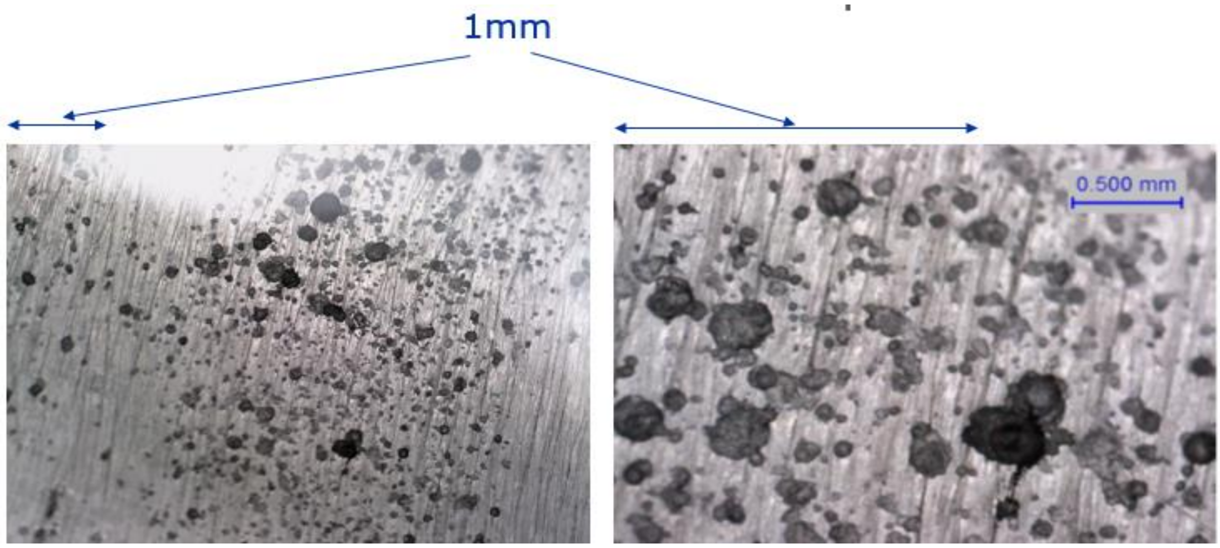


Figure 3-21 Example of same flaw captured at two different magnifications

Some of the challenges of the data acquisition is that the data set is from real rail axle specimen. With this setup, especially when it involves data from on-site axles, the images acquired are not that straight forward to process which include the following problems to acquire a quality image.

1. Uneven lighting causes a blotch of white shade as shown in figure 3.22, by using two pictures, ImageB409-16 and ImageB678-16, which then causes to effect on the results of image analysis based on thresholding method.

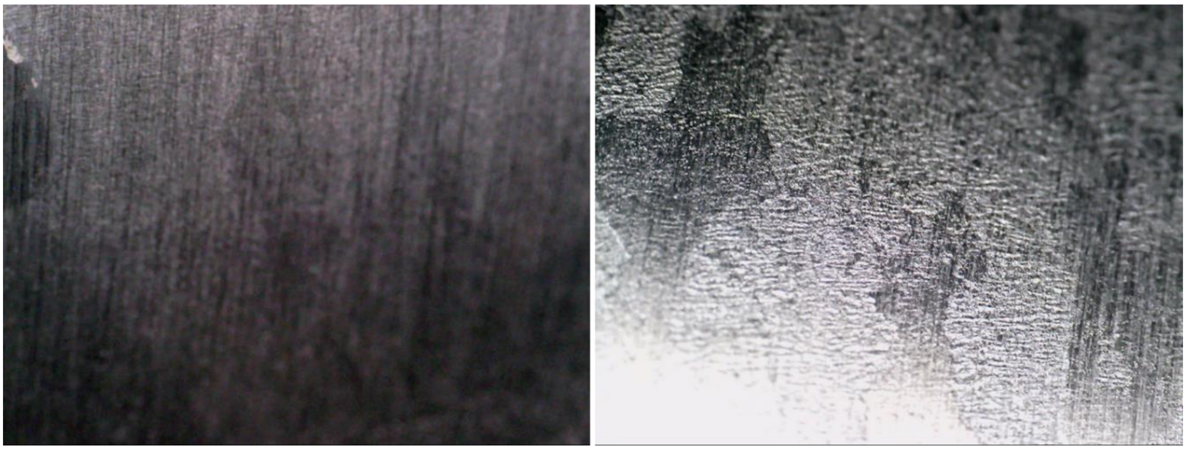


Figure 3-22 Example of uneven lighting issue using two different images

2. Depth of focus limitation gives blurred images as shown in figure 3.23, by using two pictures, ImageC152-16 and ImageA832-17

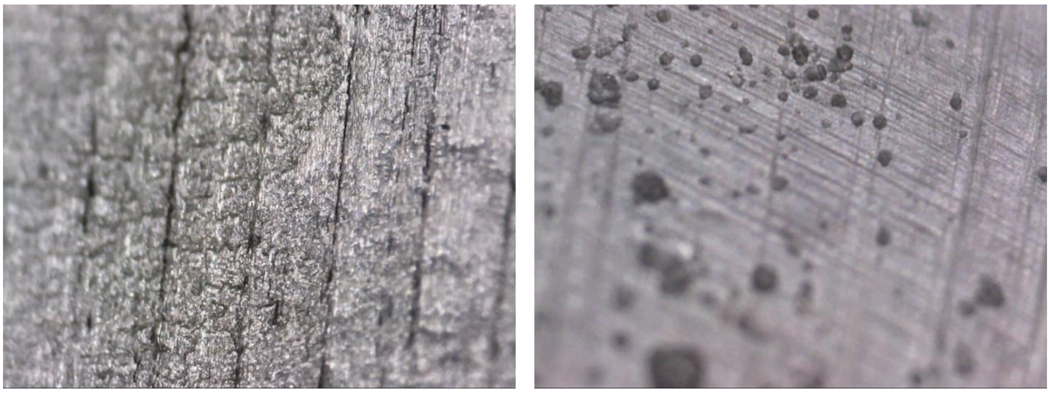


Figure 3-23 Example of restrictive depth of focus issue using two different images

3. Due to the lack of variation in contrast, sometimes the flaws might be missed. MPI method can be used to enhance the cracks as shown in figure 3.24. It shows image ImageB079-17 with lack of contrast and then ImageB098-17 taken after applying MPI

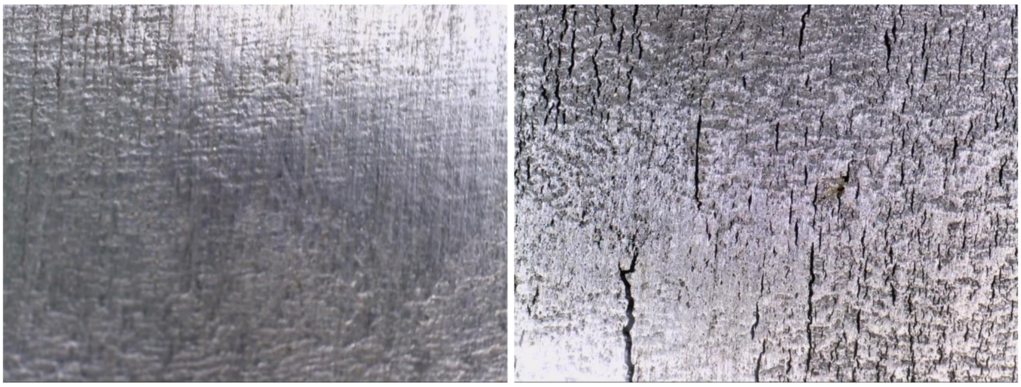


Figure 3-24 Example of lack of contrast issue and MPI method applied on it

4. The surface roughness of the material might come across as a flaw, as can be seen in figure 3.25 using ImageC667-L1. But they are actually just some sketching marks on the surface.

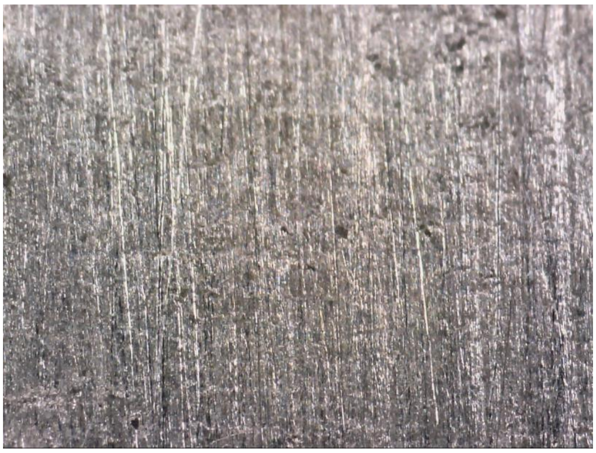


Figure 3-25 Example of rough surface of a material on a sample image

3.6 Scanner improvement

To make the data collection process more inclined towards automatic, a scanner or holder was required to hold the microscope. These parts are optional and are only to be used to support the microscope for improved data collection.

3.6.1 Initial scenario

Initially data was collected with holding the camera as a hand-held device as shown in figure 3.26. This required a lot of laborious and steady work. As the sensitivity of the microscope is quite high, a slight bit of shake of the hands made a big difference, resulting in a distorted image.

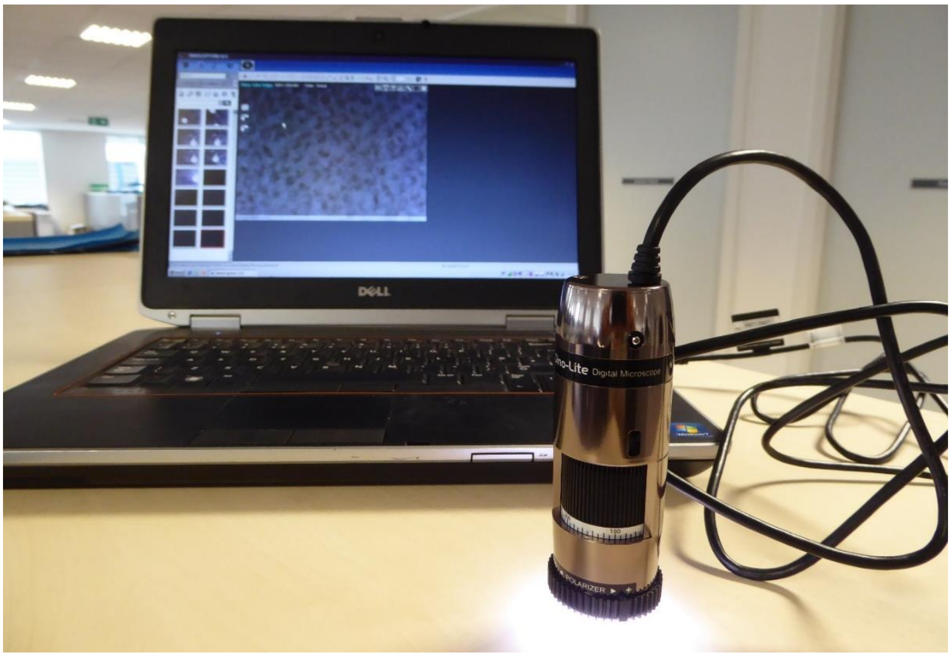


Figure 3-26 Setup of the data collection system without support

Then a microscope stand holder was used in order to increase stability as there were many wasted blur images caused due to the shaky hands. The stand holder proved to be better than the hand-held method. But it still could not be used on the rail axle sample as it could not be rotated on the component. Apart from the fact that the system was nowhere near towards automatic data collection.

There were a few other points to be considered in general. In order to see the growth of a specific pit through its stages, the pit position will need to be properly tracked on the component. This could be something where for each cycle on the bending testing, all images taken will be matched to a certain specific position on the component like by making markings or using a transparent grid. One other consideration was to take images with correct orientation and alignment from both the vertical as well as horizontal sides.

3.6.2 Manual chain scanner

Mechanical scanner was used to improve data collection by making it more robust. The scanner consists of a number of small wheeled trolleys together with a screw attachment to hold on to the component with ease, for example an axle or a pipeline as shown in figure 3.27.

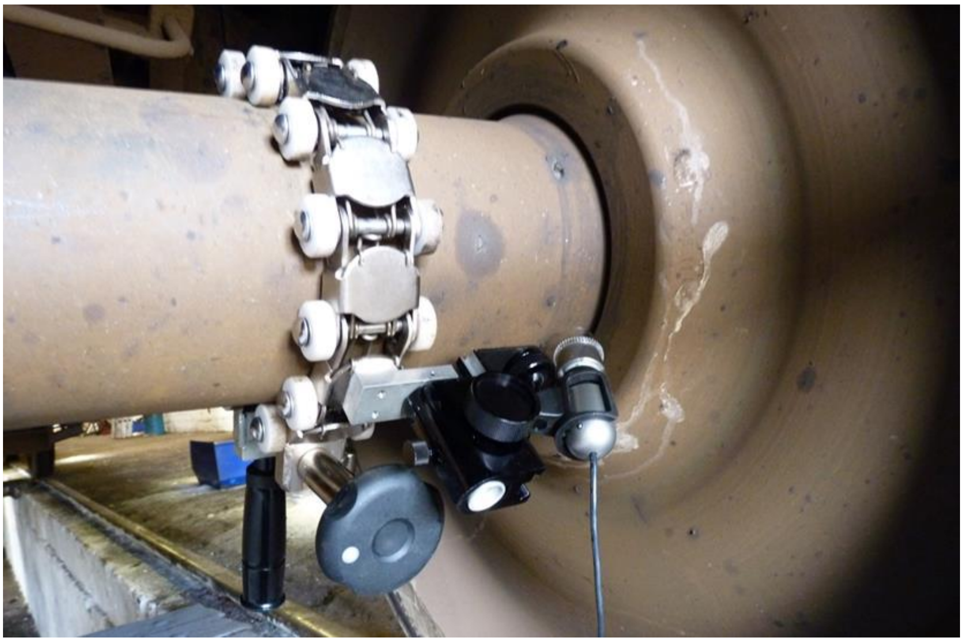


Figure 3-27 Manual chain scanner along with the microscope and its holder

This enables it to scan around and move along the component. The microscope could be attached with the scanner in two possible positions. One for axle body and the other for radii as shown in figure 3.28.

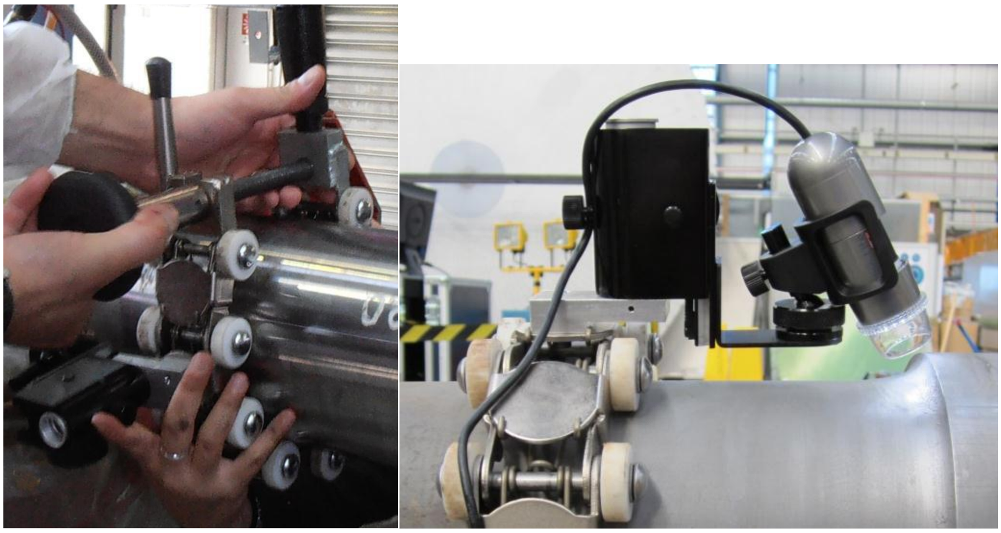


Figure 3-28 Possible positions of the microscope on the component

This setup can be fastened on different sizes of the component since the wheels can be removed and the belt can be tightened around the component. This shows that it is able to be adjustable for wheel seat scan. The system setup can be seen in figure 3.29 with the support of a scanner along with the microscope holder. The microscope holder enables a more stable and accurate positioning and orientation of the microscope onto the specimen at different curvatures. The scanner has been tested during image acquisitions and has proven to be flexible and adaptable to different types of components and is modular. The microscope holder was also tested and was proven to be stable while acquiring the images.

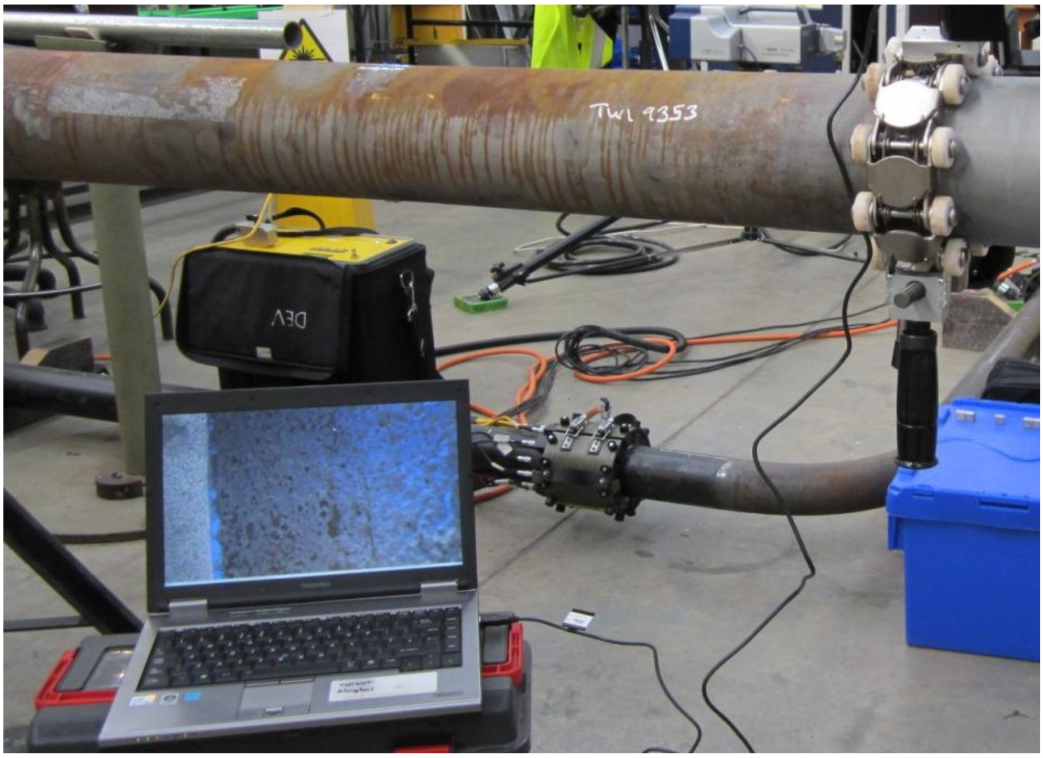


Figure 3-29 Setup of the data collection system along with a manual chain scanner

3.6.3 Semi-automated scanner

The first and far most benefit of using this scanner was that the acquisition method is semi-automatic. It moves along length and around circumference. It has adjustable length to fit between seats from about 100mm to 1.5m. It also has an adjustable angle to enable microscope to point to wheel seats. Sketch design of the semi-automated scanner is shown in figure 3.30.

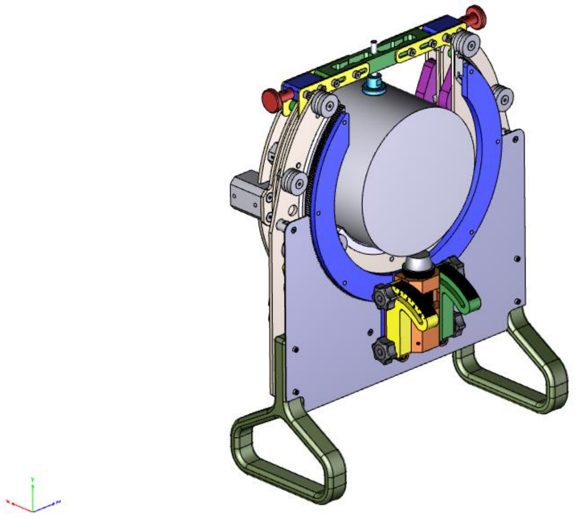


Figure 3-30 Sketch design of the semi-automated scanner

The scanner was designed as to clamp directly on the axle as shown in figure 3.31. The motors enabled lateral motion in order to move along the axis of the scanner. Finally, the ring where the scanner sat rotates around the axis of the axle. The speed of motors was optimized in order to acquire a sequence of images along the circumference of the axis. The fingers could angle the probe in order to scan the radii on the axles. The position of where the camera will settle when using the scanner is depicted in figure 3.32.

The sequence of the motor was programmed prior the validation tests; when the motor finished the sequence and stopped, the software would start the acquisition of data. A larger area could be covered with the scanner in comparison to manual testing. The collected data could then be post-processed. The amount of manual manipulation was reduced with the scanner. In addition, constant speed could be reached which allows constant acquisition of data. The addition of a microscope holder, the microscope was held steady during the tests.

As to program scanning sequence, a control box was designed; different switches control the clamping motions, lateral movements and circumferential scanning. The control box was designed for the microscope to cover an area of 100mm^2 per sec.

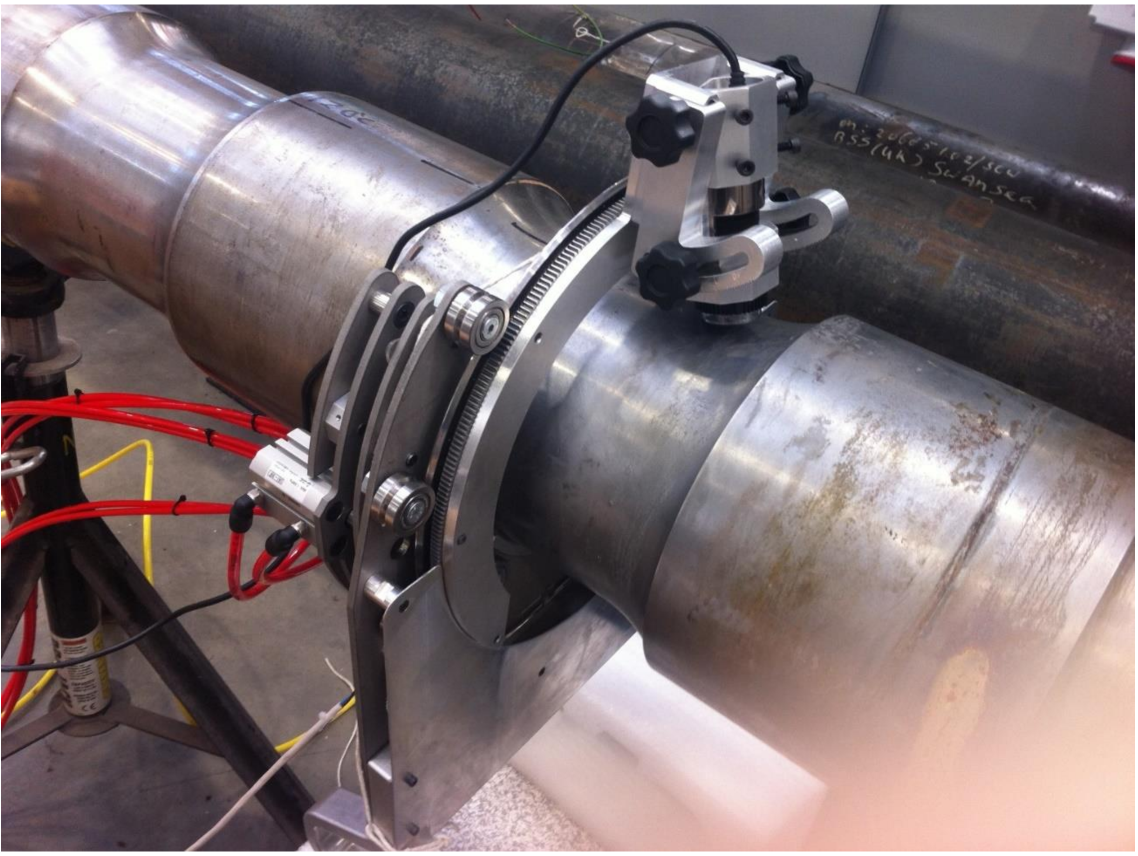


Figure 3-31 Setup of the data collection system along with the automated scanner

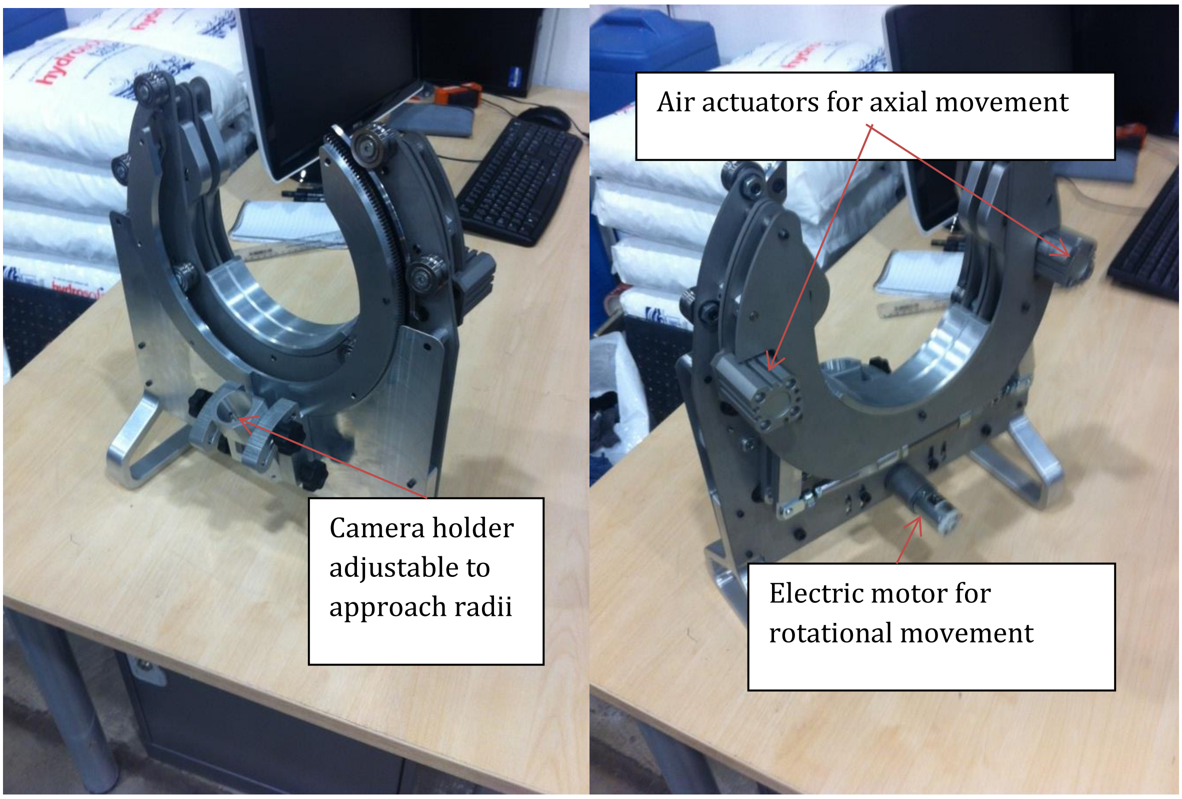

Air actuators for axial movement
Camera holder adjustable to approach radii
Electric motor for rotational movement

Figure 3-32 Position of where the camera settles when using the scanner

3.7 SURFACE PREPARATION MATERIAL

The quality of surface preparation is critical since optical techniques are used to detect and identify the defects in the components. The success of visual inspection, with or without aids at detecting indications, relies highly on the surface condition and lighting arrangements. For this purpose, various chemical surface treatments were investigated by TWI in an earlier stage and the most efficient solution, in terms of being more reliable, effective and rapid, was chosen. Commercially available products were selected to facilitate on-site cleaning operation for the operator to account for the limitations of the on-site inspection. Hence, laboratory techniques such as ultrasonic cleaning or temperature assisted bath solutions that may be more effective and quicker were not considered.

Rust is of high concern as they can compromise the results of the visual inspection by forming a thin layer of oxide on the surface and so the experiment was performed on a very heavily corroded steel plate. The chemical products compared were DE.SOLV.IT®, rust remover wipes supplied by Mykal; KURUST, water based rust converting primer supplied from Hammerite; and DEOX-C concentrated rust remover powder to be dissolved in water, rust remover solution but it is also available in gel and supplied by Bilt Hamber. The advantage of using liquid solutions is that they penetrate into any crevices and flaws which aid in removing corrosion completely. This is a big support when an inspection is done by optical techniques.

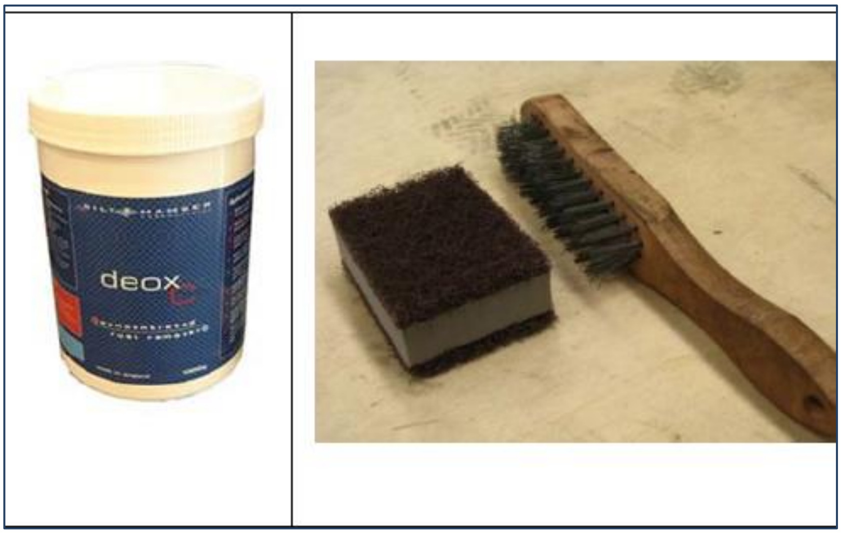


Figure 3-33 Chemical rust remover and abrasives used

The results show that Deox-C (20% concentrated) is the commercial product offering the highest performance between the three products initially tested. Note that in this case Deox-C was in solution, and therefore might be difficult to apply for the inspector who is on-site. However, the same product can be found as a gel and as a consequence might be more appropriate to use on site. The advantage with the current liquid solution is that the concentration can be modified according to the strength of the solution that is needed to remove the rust. This will not be possible with the gel.

The selected chemical rust remover and the abrasives that can be used are shown in figure 3.33 above. Following are the basic steps suggested to prepare the surface before running the software which are as follows:

1. Use wire brush or other abrasives to remove excess rust on the surface.
2. Apply Deox-C solution using a paint brush (20% concentrated) for 15 to 90 minutes depending on the level of rust present on the surface. To increase treatment speed, the solution sitting on the component surface should be renewed occasionally and rust which appears to be still attached to the surfaces can be brushed away.
3. Remove the solution using lint-free cloth and rinse treated component surface with clean water and finally wait for the surface to dry after rinsing process.

3.8 Data sample Trials & Testing

Data collection is one of the most important steps in the project. The first step required is to find an appropriate sample for testing the image analysis tool, keeping the data specifications of the project in mind plus taking into account the stage of the software development.

In the early stages it was necessary to devise set procedures and to know how the various features work with each other. Examples of these are: how to setup the camera, lighting, how to hold the camera at certain angles (for angled surfaces), how to gather flaw location information on the sample, and to be used as good input images by the analysis tool. For the initial first algorithm, a very simple and clear image, with either pits or cracks only, was required so that the working of the algorithm could be checked. There was a rail axle sample, but it was too heavily corroded and hence the pictures were not apt for the initial testing of the simple algorithm. Later in the project, data was collected from different real samples. The relevant features of these will be explained in the next chapter under data sets. Following are some of the samples that were used, to test the initial tool as well as to gain practice with the microscope.

3.8.1 Sample 1: Real images

These were real images of pitting and cracking on rail axles. These images were way too complicated for the initial stage of the algorithm though a couple of the images shown in figure 3.34 and figure 3.35, did produce some sensible results. The detailed results will be shown in Chapter 5 but they still need to be verified by manual assessment by an experienced operator.

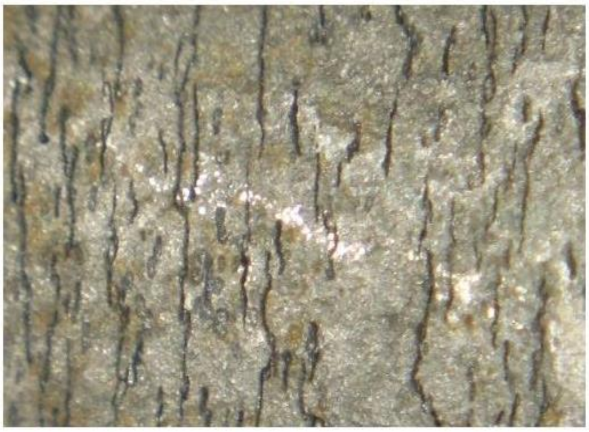


Figure 3-34 Sample of micro-cracks on an axle

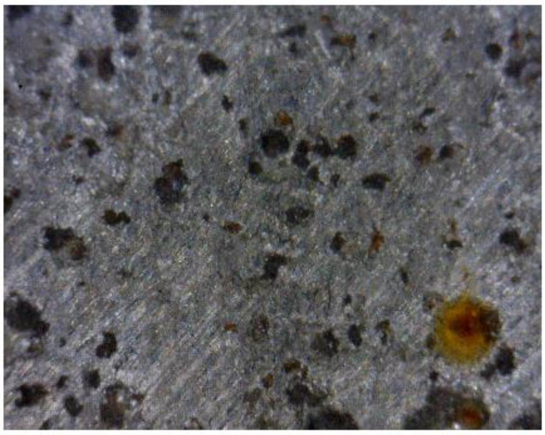


Figure 3-35 Sample 1: Pitting on an axle

3.8.2 Sample 2: Standard images

The next set of images that were used was from the industry API-579 standard. The algorithm worked well with these images. There are different levels and grades for pitting corrosion assessment, with figure 5.36 showing grade 1. The initial defect detection algorithm was tested on the API standard images from grade1 to grade 3. They were all verified using manual counting of the pits.

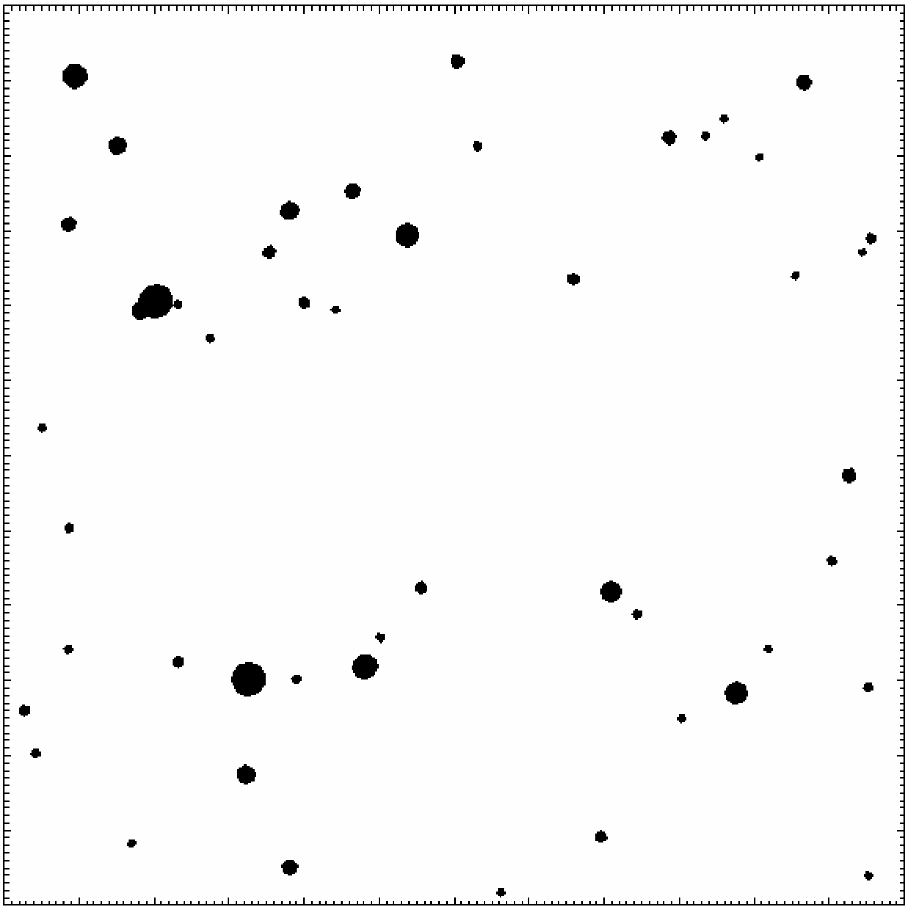


Figure 3-36 Sample 2: Grade 1 pitting

3.8.3 Sample 3: Real image - Pipeline

As the algorithm worked on sample 2, as shown in figure 3.36, the algorithm needed to be tested with some real images. For the initial experimentation, less pitted and simple images were required to be able to check that the algorithm was working. But the image acquisition has been a bit of a problem as the images have heavy pitting.

For this a pipeline was used which seemed to be less pitted. But after cleaning the surface, it was seen that it was in fact very heavily pitted as seen in figure 3.37 so didn't test with the software. Though some other images were taken from a simple steel block calliper but they were too shiny and scratched, as shown in figure 3.38

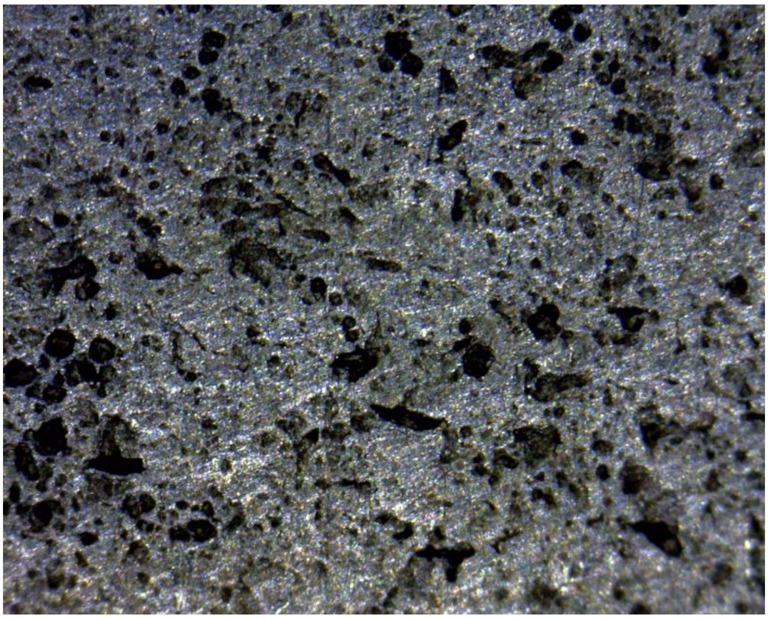


Figure 3-37 Sample 3: Heavily corroded pitting on pipeline

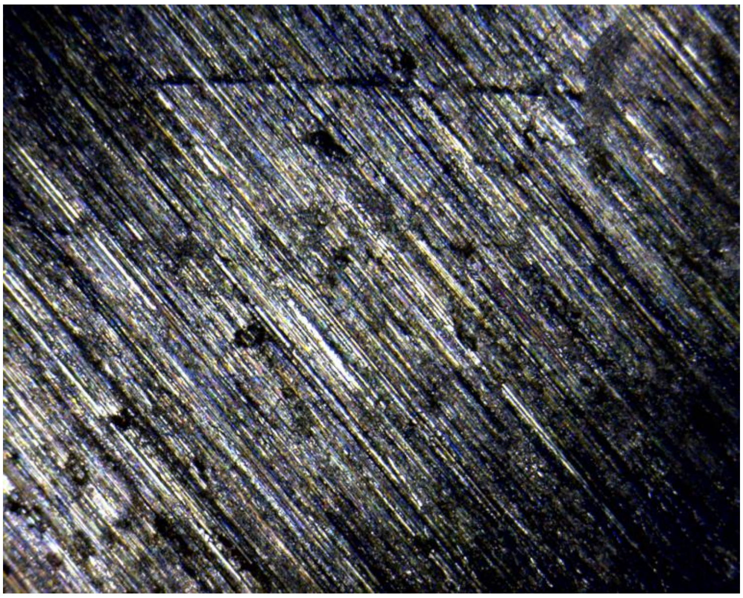


Figure 3-38 Sample 4: shiny and scratchy simple steel block calliper

3.8.4 Sample 4: Real image - Carbon steel block

This time it was decided that the pits should be created by using a self-design so that the exact number of pits and the dimensions of each pits will be known already. For this to be made possible, some discussions were held with an operator familiar with pit assessment, and a researcher who is working on different methods of pit formation. Firstly, two different designs were conceived; one was made with number of pits equal to six and the other with ten pits. Both had pits in different locations and a few different pit dimensions, which can be seen in figure 3.36. These patterns were used to test the accuracy of the algorithm.

Some small 50mm by 50mm samples of carbon steel material were supplied, and a tape “mask” was made. The tape was selected and then holes were drilled in the chosen tape. The design part of the tape was kept on one side of the material and then the rest of the sample was covered with tape. A solution of 3.5%Nacl solution was prepared, to serve as the corrosive environment for the exposed surface areas.

First Method

For the first method as shown in figure 3.39, a strong tape was used which was left dipped in the solution as seen in figure 3.40, but by the second day the water had started to leak through it which meant that it was not water-resistant. The methods which were decided after considering suggestions were unfortunately not that successful. The water started to leak inside through the tape as seen in figure 3.41.

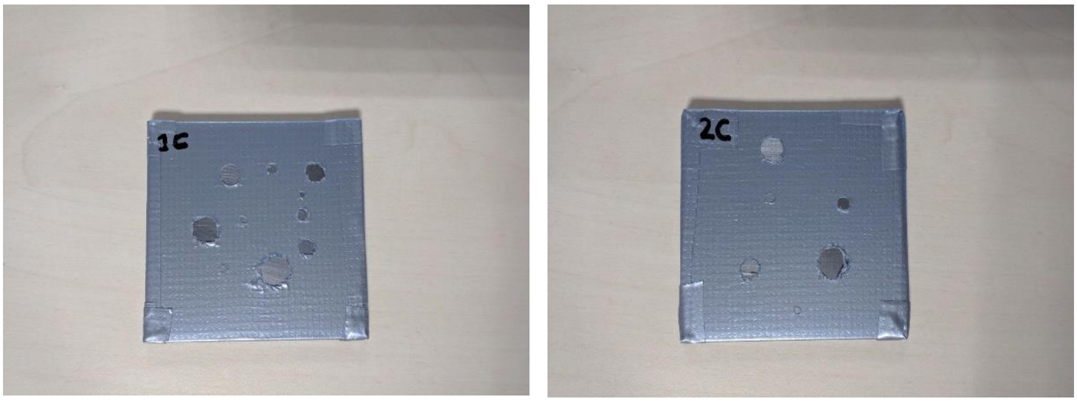


Figure 3-39 Pit formation Method 1 step 1

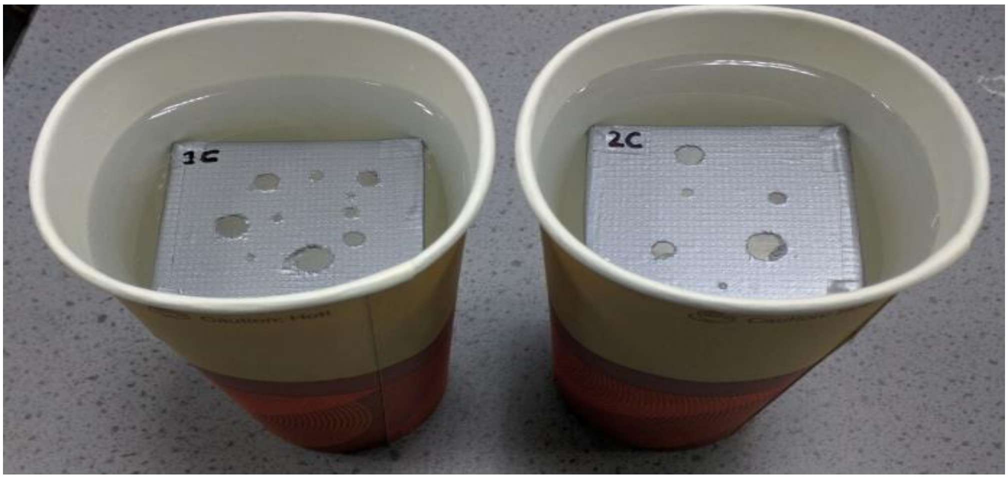


Figure 3-40 Pit formation Method 1 step2

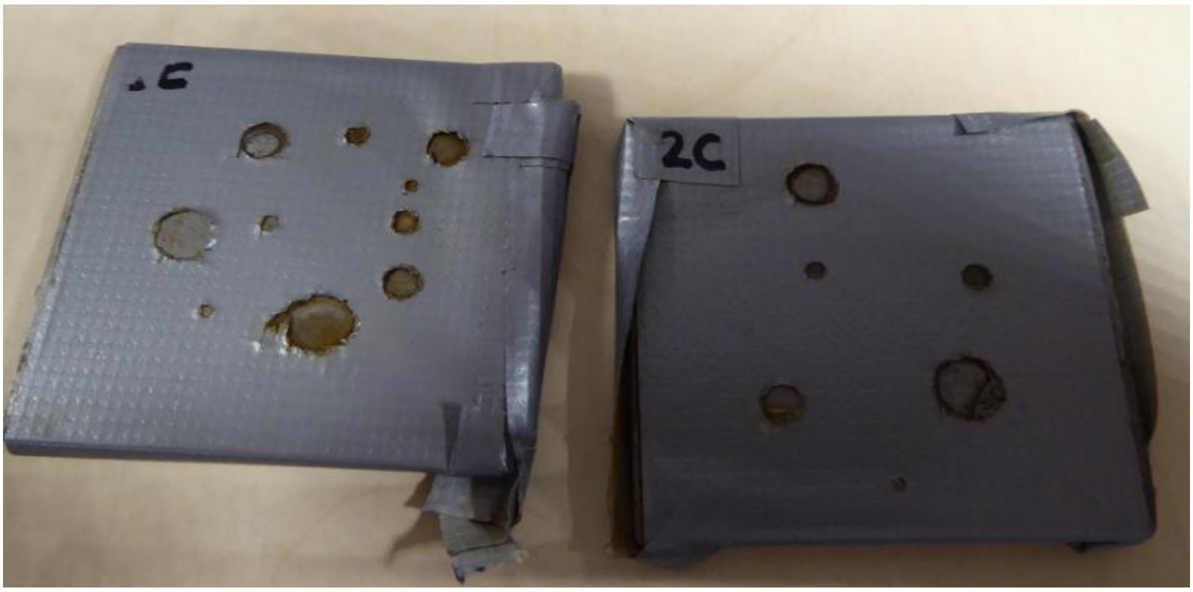


Figure 3-41 Pit formation Method 1 Outcome

Second Method

For the second method, water resistant tape was used. This was checked initially by making three simple holes using a needle. Then the material was tested by keeping the two holes on one side and then marking them. The markers seen in figure 3.42 are just for the indication of the location of the pin needle hole, and do not represent the size of it. This time an extra layer of protection for the edges with multiple layers of tape was made. After that it was left for exposure. However the tape also started to come off especially from the edges and through the holes.

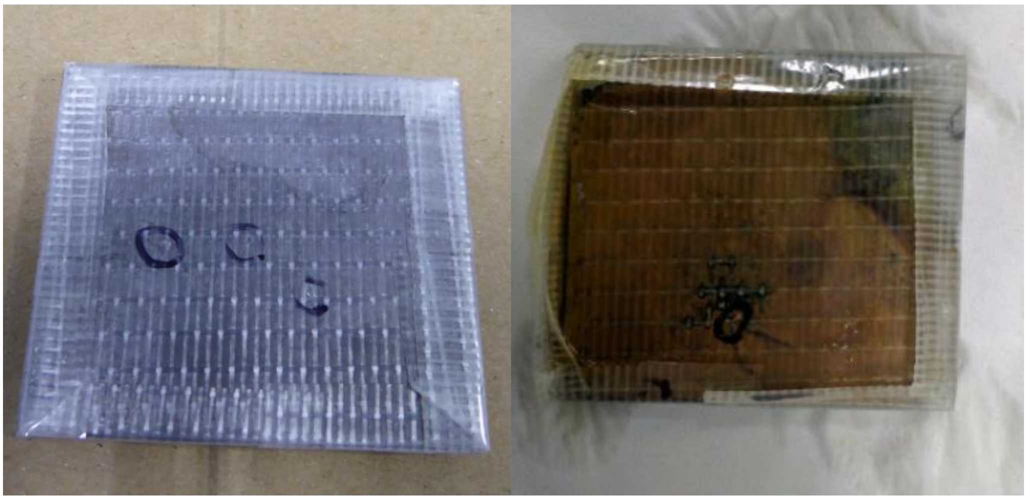
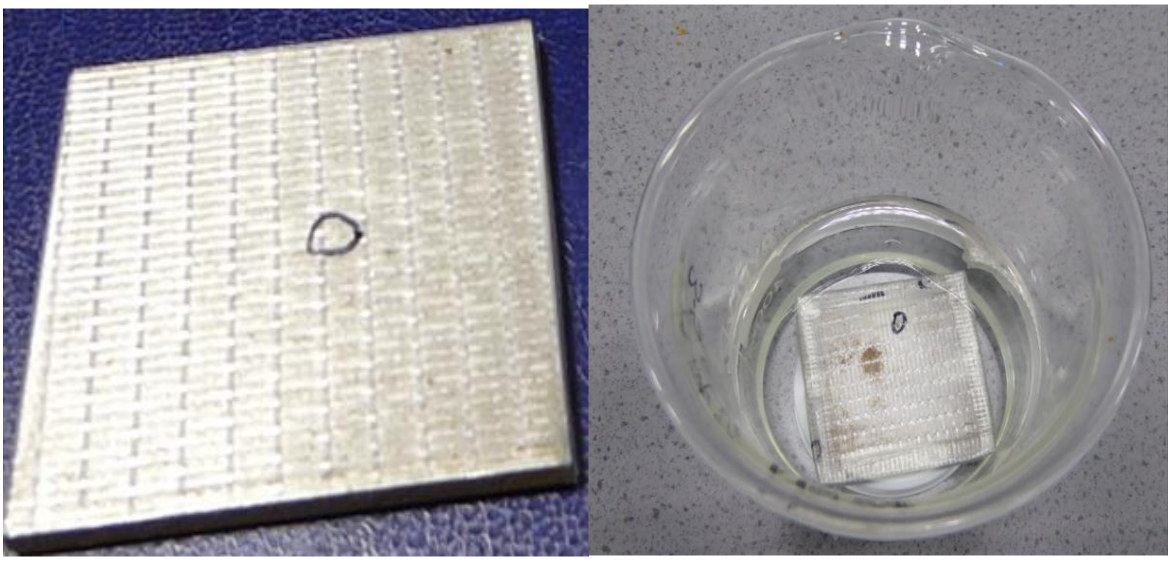


Figure 3-42 Pit formation Method 2 along with its outcome

Third Method

For the third method, another way was devised by combining two materials, the water-resistant tape and a lacquer solution. The same procedure as in the second method was followed but for the edges lacquer solution was used instead to make it less permeable. It was left dipped in 3.5%Nacl solution for two weeks. This experiment was more successful than the previous one as in this one, water didn't leak in from everywhere but just a couple of loose areas. Finally some progress was made in experiments of pit formation as some parts were not corroded heavily as shown in figure 3.43. A few images of the sample were taken with an optical microscope, shown in figure 3.44. This technique is also mentioned by [2] in a journal discussing varied methods on developing corrosion pits.



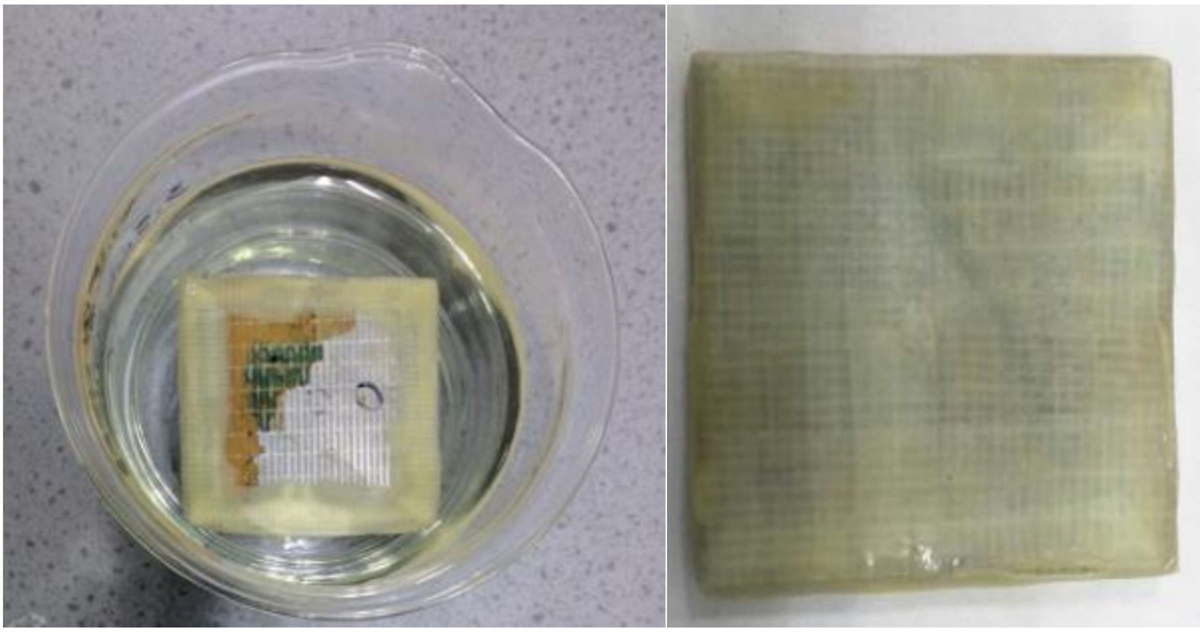


Figure 3-43 Pit formation Method 3 steps

The results showed the initiation of pit formation as shown in figure 3.44 below.

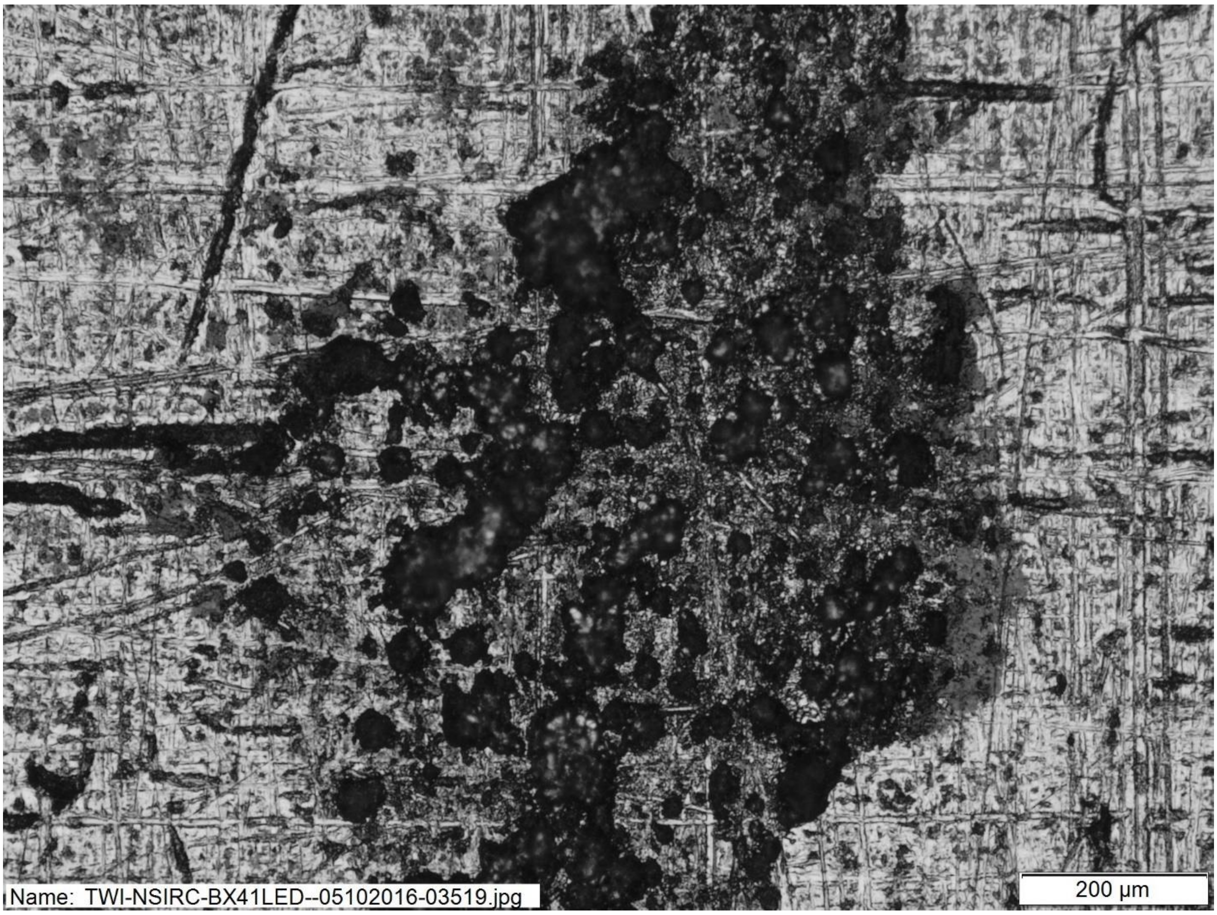


Figure 3-44 Pit formation Method 3 outcome on a sample

3.8.5 Sample 5: Real image – Rail axle

This sample has less corrosion on it, hence it was most suitable for the project. It will be able to capture different stages from early pitting to cracks. This specific sample proved to be is one of the major portions in the database hence it is discussed in more detail' in chapter 4. The setup of the experiment is shown in figure 3.45.

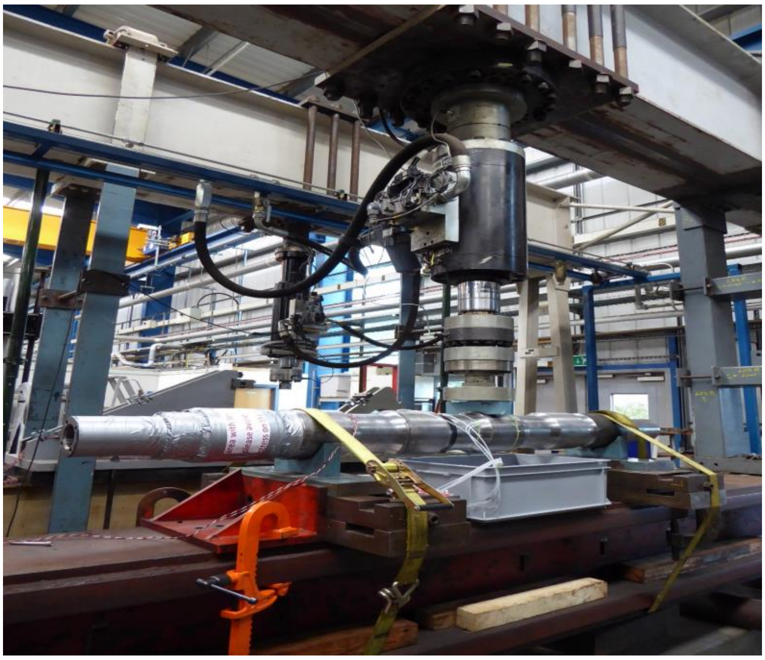


Figure 3-45 Pit formation Method 4 set up

The data collection system started to indicate some progress by working with real images. It shows a few pit formations in the early stages and the size can be seen by the scaled image in figure 3.46.

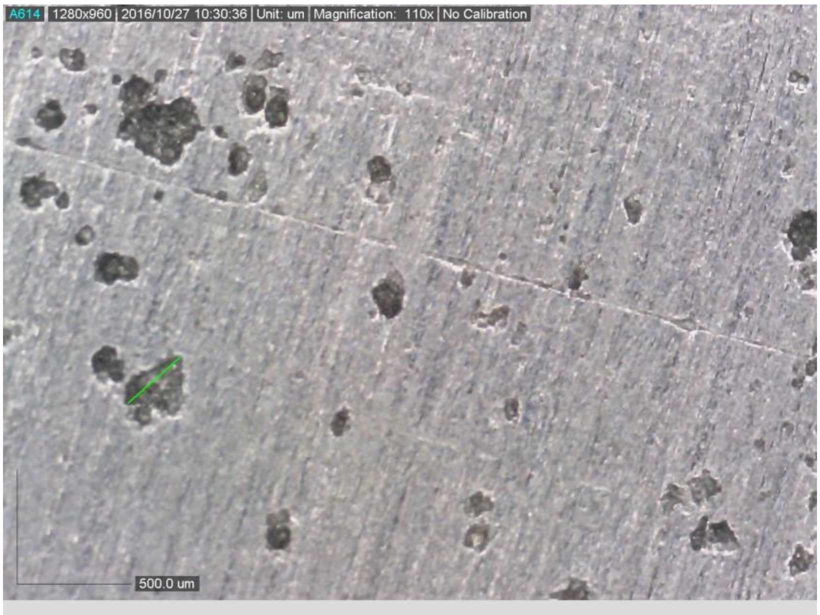


Figure 3-46 Pit formation Method 4 outcome of a sample

3.9 Data Collection System setup

For the initial system setup, as a one off, there were a few necessary materials and equipment requirements. They were needed to be selected appropriately, keeping the requirements of the project goal in perspective.

- Suitable camera was selected in order to take microscopic flaw images.
- Scanner was designed and selected that can hold the camera at a specific distance and can rotate around the axle.
- Sample pits were designed to experiment with the initial simple detection system.
- Cleaning product was chosen, as the surface of the sample needs to be prepared.

Once all of the above selections and upgrades were done then it was required to setup an image acquisition system, combining all these selected components together. Then there are a few operational pre-requisites that are required prior to the data collection each time. For the inspection of defects, the operating procedure, as shown in figure 3.47 consists of a few simple steps which are explained in the following sub-sections in detail.

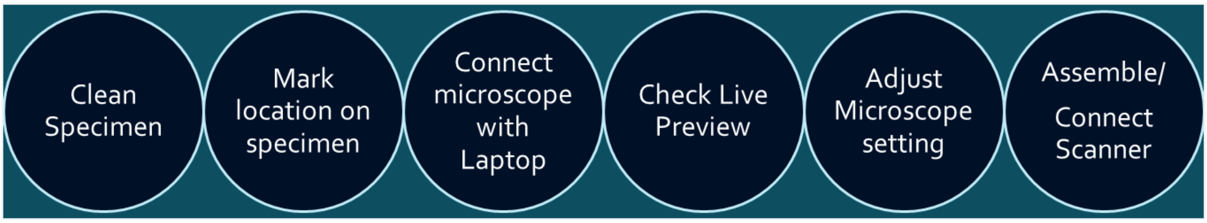

Clean Specimen
Mark location on specimen
Connect microscope with Laptop
Check Live Preview
Adjust Microscope setting
Assemble/ Connect Scanner

Figure 3-47 Devised operational protocol for data collection system

3.9.1 Clean the sample

This step is needed to capture exploitable images. In some cases, the component is covered in a layer of rust which needs removing in order to observe the presence of cracking and pitting. For rust removing, a chemical rust remover in the form of a gel is used due to its practicality and efficiency as discussed in section 3.5 earlier. The layer of product applied onto the specimen should be around 5mm thick in order to remove a sufficient amount of rust. It is then kept on the component for at least an hour in order for the cleaning process to be fully carried out. If the component is heavily rusted then it can be left overnight. But the gel could get hardened on the component. It is better to keep the component lubricated by moving the gel with a brush. Following the application of the rust remover, the area needs to be rinsed properly in order to get rid of the residue left by the reaction between the chemical and the rust.

3.9.2 Check live preview

Once the sample is cleaned and ready for collection, the microscope can be attached to the laptop via a USB cable. A live preview can be seen on the laptop screen, to check and ensure the connection between the microscope and the laptop. When the sample area is visible on the screen, it makes taking images much easier. As the effects of changing the microscope focus control is immediately observable on screen. This helps in taking images of correct orientation as the image will seem tilted otherwise.

3.9.3 Assemble the scanner

If the manual scanner needs to be used, then the links need to be assembled according to the diameter of the component. The microscope is placed onto the holder and its position is adjusted such that the microscope touches the surface of the component. It should be put as close as possible to the surface in order to obtain images at high magnification with appropriate resolution. The semi-automatic scanner also requires a bit of setup too.

3.9.4 Adjust the microscope settings

Before taking the images, it is important to note the microscope's settings such as the magnification (indicated on the laptop screen as well as on the knob of the microscope), and the location of the microscope on the component. It is also required to adjust the polariser of the microscope which would help with better lightning. Finally, pictures can be taken with the microscope's software in order to keep a record of any anomalies encountered during the inspection. The image number should be noted systematically, correlated with the location of the microscope on the axle, to keep a note of the image location on the sample. Comparing different cycle's flaw data throughout the experiment, can help keep a track on the growth of the pits or cracks.

3.9.5 Data setup system

The setup used for the image acquisition consists of three major components which includes; laptop with the software installed, microscope attached to the laptop via the USB port cable and, if required, the microscope holder with or without the scanner. The simple system setup without any support can be seen in figure 3.48 and system setup with a manual chain scanner can be seen in figure 3.49.

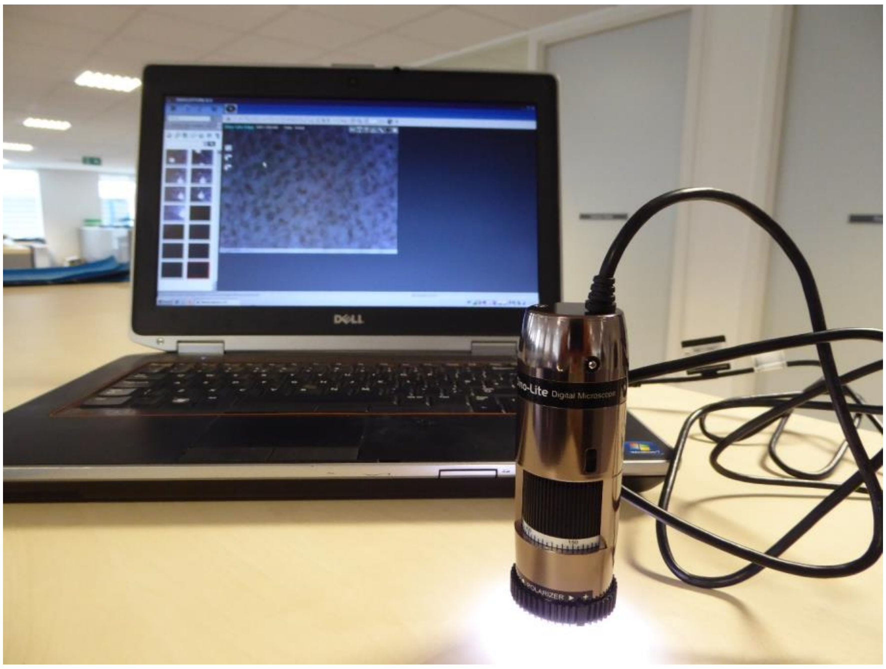


Figure 3-48 Setup of the data collection system without support

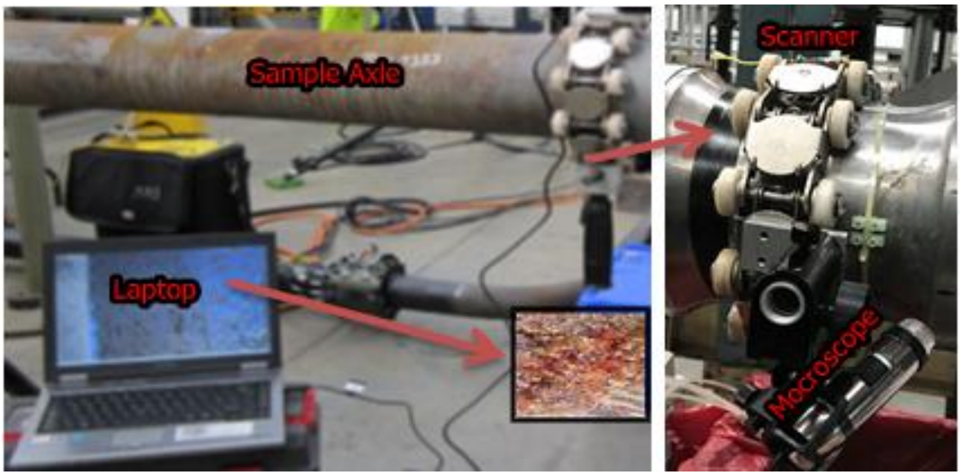
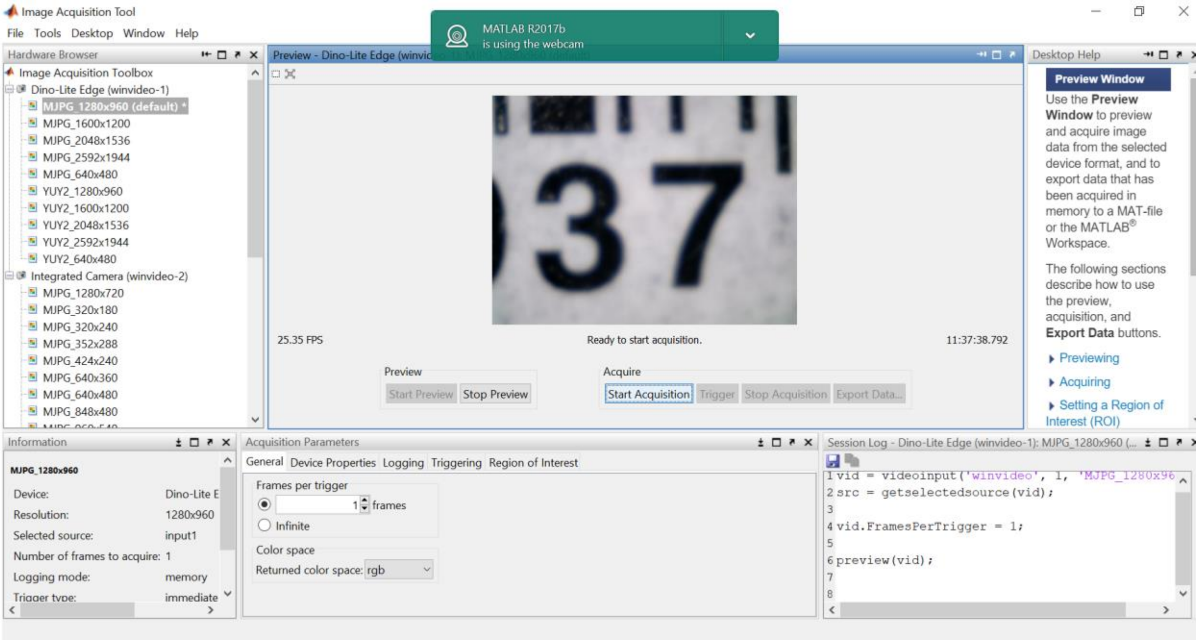


Figure 3-49 Setup pf the data collection system with manual scanner

This system needed to be improved during the project as discussed earlier, in order to enhance the image acquisition process. Hence the system for data collection has been greatly improvised and an updated scanner has been designed to take automatic images, where the microscope sits on a designed holder, as shown in figure 3.51. To acquire better contrast images, MPI may be performed on the sample, especially to detect cracks as discussed earlier.

To communicate between microscope and matlab, an interface was used which is shown in Figure 3.50. This enabled more control on the speed of data acquisition with reference to the scanner movement. It collects videos from the microscope directly and stores the data in the specified selected folder. This is an optional tool as data can be collected directly from the microscope as well.



25.35 FPS

Ready to start acquisition.

11:37:38.792

Preview Window

Use the **Preview Window** to preview and acquire image data from the selected device format, and to export data that has been acquired in memory to a MAT-file or the MATLAB® Workspace.

The following sections describe how to use the preview, acquisition, and **Export Data** buttons.

MJPG_1280x960

Device:	Dino-Lite E
Resolution:	1280x960
Selected source:	input1
Number of frames to acquire:	1
Logging mode:	memory
Trigger type:	immediate

Frames per trigger
● 1 frames
○ Infinite

Color space
Returned color space: rgb

```
vid = videoinput('winvideo', 1, 'MJPG_1280x96
src = getselectedsource(vid);

vid.FramesPerTrigger = 1;

preview(vid);


```

Figure 3-50 Data collection communication interface

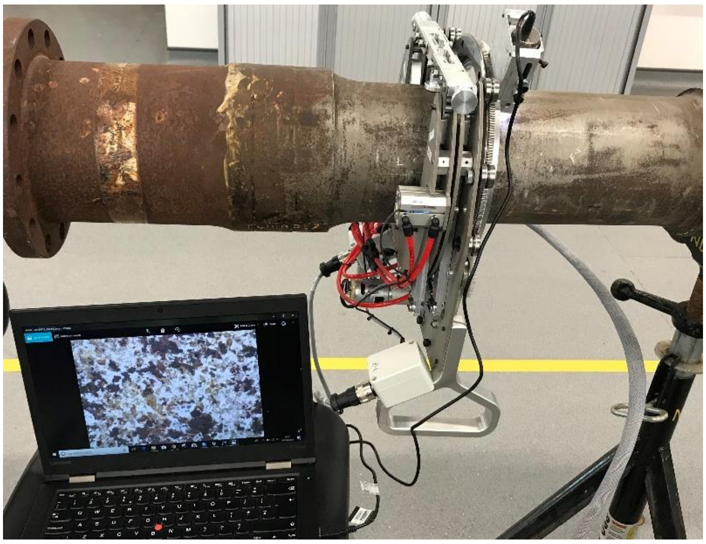


Figure 3-51 New data collection system setup with automatic scanner

The above setup enables communication with the microscope and collects data. The data is collected as video and converted in image format later on. Images are then post-processed with embedded algorithms to highlight specific features.

3.10 Initial Layout of Detection System

The development of an automatic detecting system is becoming a major requirement nowadays as it helps to manage resources more efficiently. For this project, the data acquired is extremely big for which computers are crucial, as large data sets require a lot of complex computations and extraction of quantitative information. Hence, a system architecture was needed to be designed that could depict the different stages of the DDS (defect detection system).

The description of a solution is also called a design [123]. It has also been argued that the combination of both conceptual and technical design, is the software design process [124]. Hence it is really important to think it through in the early stages. The design is then implemented and tested and if it doesn't work then another solution is suggested. The main design framework of the project contains four stages; Data acquisition, Image pre-processing, Feature extraction and Classification as indicated in figure 3.52.

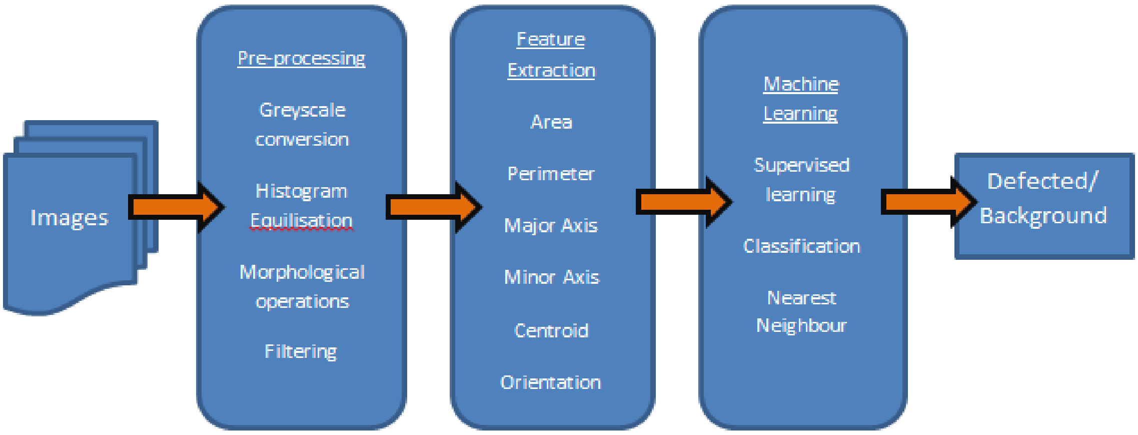


Figure 3-52 Basic stages in the detection system

Once the system has been setup, data has been collected then the images are passed into an image analysis system. The first part will include processing the images and finding the key areas-of-interest along with the flaw dimensions. The second part will include image segmentation based on key features for pits and cracks by using machine learning techniques and/or deep learning neural network. This system may be called the automatic defect detection system (ADDS).

Image analysis is meaningful when information is obtained from digital images using digital image processing techniques. This involves processing an image into some basic elements in order to extract useful statistical data. It can include tasks such as de-blurring, correcting perspective distortion, finding shapes, detecting edges, removing noise, feature extraction, counting objects, and measuring region and image properties of an object. It is a broad term that includes a range of techniques that generally fit into these subcategories: Image enhancement to remove noise, image segmentation to isolate regions and objects of interest, Morphological filtering to remove more noise, Region/Feature analysis to extract statistical data. In order to develop an advanced image analysis system, there are several steps to be taken, including:

1. ***Collection/acquisition of Data:*** The data collected currently consists of 2D images by using a 200 magnification microscopic camera. Other possibilities will be reviewed. These could be used with two lights; LED lights to brighten the dark objects and UV lights. The material of rail axle and pipelines is carbon steel for example A1N and A4T.

2. ***Standard API:*** The Standard API 579-1 part 6 is going to be followed for the procedure used in Non-Destructive Testing for the pits in the pipelines.

3. ***Image pre-conditioning:*** Digital images are prone to a variety of types of noise. In the pre-conditioning phase, noise will be defined. Then several ways will be used in order to remove the noise such as: Linear Filtering, Median Filtering, Adaptive Filtering or Morphological filtering.

4. ***Image enhancements*** include methods like edge detection, image enhancement, etc. Edge detection is the name for a set of mathematical methods that aim at identifying points in a digital image at which the image brightness changes sharply or, more formally, has discontinuities. The points at which image brightness changes sharply are typically organized into a set of curved line segments termed edges. Some of the Edge detection methods are; Roberts, Prewitt, Sobel and Canny.

5. ***Image segmentation:*** Image segmentation is the process of dividing an image into multiple parts. This is typically used to identify objects or other relevant information in digital images. There are many different ways to perform image segmentation, including: Thresholding methods such as Otsu and Histogram; Clustering methods such as Fuzzy (FCM) or K-means; Transform methods such as watershed segmentation, Texture methods such as texture filters.

6. ***Pattern recognition methods:*** A deep learning (DL) technique is a class of machine learning techniques that models hierarchical abstractions in input data with the help of multiple hidden layers. Machine learning methods are based on learning representations of data. For example, an image can be represented in many different ways but some representations are better than others at simplifying the learning task. Research in this area attempts to make better representations and create models to learn these representations that are loosely based on the working of a human brain. They are good at recognizing patterns so it may be possible to classify cracked and non-cracked images as well as pits or non-pitted directly from the original images by using Artificial Neural Network

(ANN). However, the method using only ANN without image processing will require more computation time, since the training images have so much information. Thus, it is required to include the image processing before applying ANN step in order to develop an effective process.

3.10.1 Image Pre-processing

Image pre-processing is concerned with preparing the image data for the main processing tasks, and will typically include restoring the effects of corruptions that have occurred during the acquisition or transmission of the images, reformatting the image data, and then enhancing the quality of the image for visual inspection and interpretation or and to segmentation of regions and objects in an image on the basis of similarity criteria that permit features to be extracted for interpretation and classification tasks. [125]

Grey-scale conversion: This is relatively straightforward and involves converting a true colour image, which has Red, Green, and Blue values for each pixel, and converts these to a grey-scale value, by eliminating the hue and saturation information while retaining the luminance. An example of a converted image is shown in figure 3.53 below.

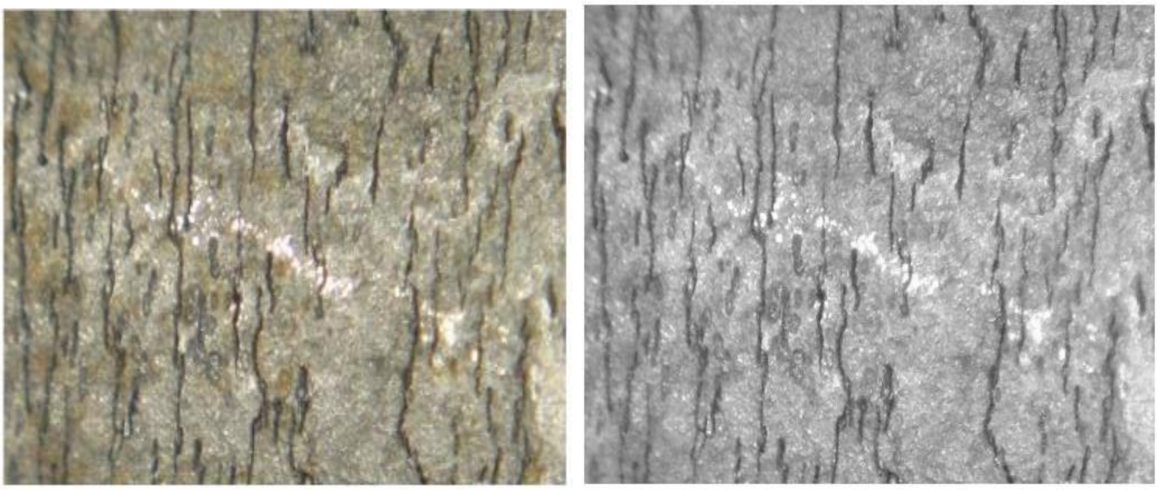


Figure 3-53 Image pre-processing step of conversion to grey-scale

Image enhancement: Improving the quality of the image for visual inspection and interpretation is the main goal of this task. This is particularly important for such images that tend to be complex and noisy, and interpretation of region or features that are important or of interest are enhanced for display or subsequent analysis [83] [126]. The histogram equalisation was performed for visual inspection only. Some image enhancement techniques that are appropriate have been displayed below.

Histogram Equalisation: This is one of several contrast changing operations and effectively spreads out the grey-level values along the total range of values in order to achieve higher contrast [69]. This is especially useful when an image has low contrast because of a small range of grey-level values, such that the background and foreground are bright at the same time, or else both are dark at the same time. Histogram equalisation involves constructing a histogram showing the distribution of grey-level values in an image.

Noise reduction: Images often get corrupted during the image acquisition or transmission [74]. It is an extremely important step to remove or reduce such noises in many applications because the performance of succeeding image processing tasks will depend on the data being uncorrupted as possible. The images get corrupted by noise when some of the pixels of the original image are replaced with new pixels having luminance values near or equal to the minimum or maximum of the allowable dynamic luminance range. In order to smooth the image there are three common techniques mean filter, Gaussian (low pass) filter, and median filter. All three use data from the neighbourhood of a pixel to generate the new pixel value which leads to the images becoming smoother and less noisy. But using this technique has a consequence that makes the image less sharp and blurred.

The mean filter simply replaces a pixel grey-value by the average value over its neighbourhood. Gaussian filter is similar to the mean filter, the difference being that the grey-level values of the pixels in the neighbourhood are weighted according to their distance from the centre of the neighbourhood. The weights follow the standard Gaussian distribution. Median filtering is quite popular because, for certain types of random noise, it provides excellent noise-reduction capabilities, with considerably less blurring than for the other techniques for a similar size of neighbourhood. The median filter replaces the grey-level value of the pixel at the centre of a neighbourhood by the median of the grey-levels of the pixels in the neighbourhood, it is basically a non-linear filter to achieve good result in many Image Processing applications [64]. Median filters are particularly effective in the presence of impulse noise, because of its appearance as white and black dots superimposed on an image.

Image segmentation: The edge-detectors that are commonly mentioned in the research are Prewitt, Sobel, Roberts and Canny. They are the most popularly used segmentation methods as they give fast and effective results depending on the desired goal. Different methods are applied for different applications and produce better results accordingly. An example of ‘Canny’ method, as it is one of the most powerful operators, is shown below in figure 3.54 to display the effect and importance of this step on an image.

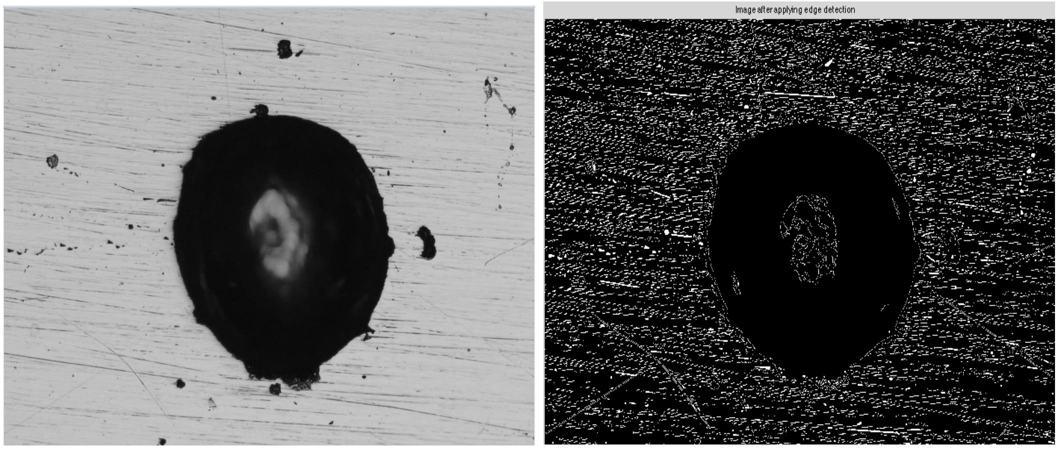


Figure 3-54 Canny operator being performed for image processing

3.10.2 Feature Extraction

Feature extraction is classified into the structural and textural features. The defects like pits and cracks do not have a well-defined structure. Pits may be more well-defined structurally so they can be defined by their area so for research purposes pits are taken as semi-elliptical shape objects while cracks as vertical thin lines (on rail axles, cracks tend to grow circumferentially and therefore the camera can be oriented to show these as vertical lines).

Further research will be done on the morphology of the defects to be able to identify and classify them more accurately. These defects are more identifiable through the difference in their colour and texture so more emphasis will be taken to extract these kinds of features in specific. So there could be three main features for the classifiers to be considered such as the morphology may be vertical thin straight lines for cracks and semi-elliptical shape for pits; colour on the basis that the defects will be darker than the background region; and texture needs to be explored further like they might be a fixed pattern of either the defect or the surface material which could be extracted. Some of the popular features that are extracted for image processing are as follows:

Area means: The area measures are appropriate for images containing segmented regions and can be derived directly from the boundary by filling in the region defined by the boundary and then counting the number of pixels in the region.

Grey-scale Histogram descriptors: These descriptors give measures of the distribution of grey-level values in a segmented region, or the whole image, and can be visualised as related to the shape of a gray-scale histogram for the grey-level value. The main descriptors are the mean, standard deviation, dispersion, mean square value or average energy, entropy, skewness and kurtosis [88].

Low spatial frequency components: The previous feature extraction techniques have been concerned with deriving grey-scale and spatial descriptors from the image. An alternative representation of the image data is the spatial frequency structure. In this representation, large low frequency components indicate the presence of larger structures in the image, while the high frequencies are related to the fines details and noise content of the image.

The most popular transform in Image Processing applications is probably the Discrete Cosine Transform (DCT) because they are fast algorithms, and dedicated hardware, to compute DCTs and only a small number of coefficients are used the DCT-based approach to image recognition is extremely fast compared to other methods [127]. The DCT frequency components have also been used by other researchers for feature representation [128].

3.10.3 Classification

This section introduces the general framework of the ANNs on the basis of which the next stage of the project will be designed. The basic terminologies have been discussed in the literature studies. This section will discuss the summary of the whole process and the some of the design issues to be considered.

ANNs are computer-based mathematical structures, which have their origins in biological Neural Networks. The artificial neurons are set in layers and interconnected with each other with weights, and this enables the ANN to process non-linear statistical data and model complex relationships between inputs and outputs. ANNs are generally considered a ‘black box’ approach to pattern analysis and classification since the model parameters are hard to interpret in terms of physical meaning [15].

Concept of programming by example or training is a very important property of the ANNs. The large number of weights between the nodes makes it difficult to pre-set them to obtain the correct result. Instead, the ANN is programmed using training data, and in the case of supervised learning, the correct target classification for the training data. Each time an input is presented, the network computes the output on the basis of the current weights, and then adjusts these systematically by an amount dependant on the difference between the output and the target output. This learning process continues iteratively until the ANN outputs value equal or close to the target, depending on the application [14].

Feed-forward Back-propagation ANNs are the most extensively used network to solve different kinds of problems across a varied area of applications [129]. ANN is capable of obtaining a new structure of internal connections that is appropriate for solving a determined task. Each layer is fully connected to the succeeding layer with weights, and input data Feeds-

Forward to the hidden layer and then the output layer. This architecture is commonly known as a Multi-Layered Perceptron (MLP), where the term multi-layered perceptron is used for networks which consist of an input layer, one or more hidden layers and an output layer. The output of the network is determined by the activation of the units in the output layer. The choice of choosing the activation function is one of the design decisions. Majority of ANN's use the sigmoid function as it is smooth, continuous, and monotonically increasing (derivative is always positive). Also, the sigmoid units bear a greater resemblance to real neurons than linear or threshold units.

The Back-Propagation aspect of this architecture refers to the procedure used to adjust the values of each node's weights during each iteration, until the training data generates the target outputs. In essence, the error between the ANN output and the target is propagated backwards through the ANN with the values of output layer nodes' weights being adjusted by an amount dependant on the size of the error, and often the gradient of the error (i.e., how quickly the error is being reduced at each iteration, the followed by the weight of the nodes in the hidden layer. As the architecture of this type of ANN involves many layers of nodes, consideration needs to be given to the number of hidden layers, and the number of nodes in the inputs layer, hidden layers, and output layer.

There is no theoretical limit on the number of hidden layers, most Pattern Recognition applications achieve very good classifications with 1 or 2 hidden layers, and extra layers tend not to improve the classification accuracy and incur large computation times. Some work has been done which indicates that a maximum of three hidden layers are required to solve problems of any complexity. The universal approximation theorem states that an Artificial Neural Network with one hidden layer can approximate any function with any desired accuracy, provided that it has enough neurons in the hidden layer and that the activation functions of the neurons are non-linear. However, for some functions the number of neurons needed in the hidden layer can be very large or even infinite.

The number of nodes in the input layer is effectively set by the number of input features that are being used for classification. Increasing the number of features can be beneficial, as the cost of increasing computing time, does not always lead to more accurate classifications, as the features may not be completely independent, or may actually be representing more

There are many techniques used to determine the optimal number of neurons in a hidden layer of an ANN. Some techniques try to increase and decrease the number of neurons and connections dynamically during the learning process, while other approaches use statically ANNs, with a different number of neurons each time. There is no easy way to determine the optimal number of hidden nodes without training using a range of numbers of hidden nodes and checking the classification accuracy achieved with each, remembering that too many nodes cause training to diverge, or lead to over fitting.

The number of nodes in the output layer is effectively set by the number of different output classification required. This is dependent on the design of the previous layers; normally it is a low number.

3.11 Image analysis Frameworks

Advancement in the field of machine learning implicates to deal and tackle with immense amounts of data for training, learning and testing the systems, especially when dealing with image and video datasets. Graphical processing unit (GPU) is essential for running large machine learning frameworks plus memory and space requirements increase when dealing with images as compared to normal datasets. It is worth noting that all these DL frameworks are undergoing constant development with active contributions from researchers and the open-source community, and therefore the study results and conclusions from the reported comparative studies may have changed.

3.11.1 MATLAB

It is a high performance language yet very easy to use and has an extensive set of useful in-built functions [10] for image processing specifically called the Image Processing Toolbox. With MATLAB and the IP toolbox combined together, image-processing operations can be written in a more compact and clear manner, thus providing an ideal software prototyping environment for the solution of image processing problems.

For deep learning purposes, it provides a toolbox, Deep Learning Toolbox. with which only simple commands are needed to create and interconnect layers of deep neural network. It includes many examples for implementation as well as pre-trained networks. It allows to import pre-trained networks from ONNX, TensorFlow-Keras, and Caffe into MATLAB. Tools like Image labeller have been introduced last year which helps to label the data in an easier way. It also has an ‘ImageDataStore’ function that can be used to import data from image collections that are too large to fit in memory.

3.11.2 Nvidia CUDA graphics card

GPUs is a requirement for big machine learning frameworks [130]. The reason is that massive datasets are used for training, learning and testing such systems. Especially with the use of image and video datasets, the specification requirement increases quite high as compared to normal datasets. With the CUDA technology developed by Nvidia, it allows parallel computing, utilizing the processing power of video cards, which can increase computation speed dramatically. As neural networks are inherently parallel algorithms, taking advantage of the ability of GPUs to perform a large number of calculations in parallel allows training of networks with good performance.

Google has been working on tensor processing unit (TPU) since 2016, as a proprietary, non-commercial artificial intelligence (AI) accelerator application-specific integrated specific circuit (ASIC). It is a chip specifically designed to work with and accelerate TensorFlow computations. There are other cloud computing solutions available for deep learning offered as a service by companies like Microzosft (Azure), Amazon (AWS), Google (AutoML Vision), and so on. One could also “rent GPUs in the Cloud” from these companies for training their deep learning models.

3.11.3 Cluster computing

This facility allows to run analyses requiring significant computational resources in a batch environment but it is not suitable for software requiring user interaction or a graphical user interface. It is configured to provide a single ‘head node’ and 20 ‘worker nodes’. The head node provides 8 cores and is intended for job management. Each worker node consists of 24 cores with 512GB RAM.

The SLURM workload manager is used to manage the worker nodes and the remote execution of individual analyses. But the users are not able to directly connect to the worker nodes, instead jobs are submitted to a queue and are distributed to worker nodes by the Slurm workload manger. When cores become available on worker nodes, Slurm selects the highest priority job from the queue that will fit within the available resources. The priority of individual jobs is based on the users’ base priority and the number of jobs they have recently submitted. Each analysis must have a batch script file. The file contains information about the job such as what executable to run, the number of cores required, and any other information required for Slurm to manage the analysis. Typically all files required by an analysis, along with the batch script file, are placed in the same directory. Then a terminal connection is opened, change the working directory to the directory where the files are located, then submit job to the queue using the sbatch command. In order for it to work, Matlab was needed to be installed and re-scripting had to be done according to the slurm requirements.

3.11.4 Caffe

It was developed at the Berkeley AI Research (BAIR) center and the Berkeley Vision and Learning Center (BVLC) at the University of California, Berkeley with “expression, speed, and modularity in mind” [130]. It is considered to be an easy-to-deploy production platform developed exclusively for DL-based computer vision systems and is believed to be one of the fastest ConvNet or CNN implementations available with an ability to process over 60 million images per day.

3.11.5 TensorFlow

TensorFlow, originally developed by researchers and engineers working on the Google Brain

Team, is a platform-independent open source library that uses data flow graphs for numerical computation and is mainly designed for developing and implementing deep neural network models [131]. One major advantage of TensorFlow that vastly increased its popularity among DL researchers and companies is its ability to deploy computation to one or more CPUs/GPUs on a variety of systems and devices through a single application programming interface (API). Based on a comparative study of Theano, Torch, Neon, and TensorFlow DL frameworks, [132] concluded that TensorFlow, although a very flexible framework, is not as competitive as other studied frameworks in terms of its performance on a single GPU.

3.11.6 Keras

Keras is a high-level Python DL library and API capable of running on top of TensorFlow, CNTK, or Theano as the backend. It is well known, among both budding DL researchers and experienced ones, for its ease-of-use (minimal programming) and ability to allow fast prototyping. Like other open-source DL software frameworks, Keras is built on the guiding principles of user-friendliness, modularity, and extensibility.

3.11.7 Conclusion

The programming language chosen for the project implementation is MATLAB. The image processing and deep learning toolbox are useful tools for the project.

Python and tensorFlow were applied for deep learning unet classifier model in order to compare with matlab time efficiency. Also used slurm through WinScp and Putty for cluster computing tasks to improve the training runtime of the models.

Nvidia GTX970 graphics card with 8GB of RAM is used for the network training. Experiments were performed using MATLAB, and ran on the same Nvidia graphics card. This work had to face delimitations of RAM and GPU. Later on in the project, a cluster was setup as the desktop kept on crashing or giving memory error as the data involved is massive. An

This project has been run and tested on the Mac platforms and Windows operating systems that have MATLAB installed.

3.12 Chapter Summary

This chapter discusses methods and materials to setup the data collection system. For early research stage different viable and existing NDT techniques were explored in order to establish the best possible method and procedure for detection of pits and micro-cracks within corrosion. The existing NDT method is a visual examination (possibly with a photographic record) and then manual counting and pit assessment. After the investigations, it was concluded that the data will be gathered by using microscopic visual inspection NDT technique with/without MPI.

After selecting the appropriate NDT method to gather data, the next step was to setup a system for surface inspection. It discusses full details of the chosen methods of data collection by also giving reasons as to why the specific method was selected. Listed all the pre-requisites of the software experiment that need to be done before running it through the image analysis tool. It consists of all the steps taken such as sample selection, thorough sample preparation, appropriate camera selection and proper hardware setup. The main architecture of the project design is presented in this chapter. It also discusses existing frameworks that are used when dealing with image analysis that include MATLAB and TensorFlow.

In the next chapter, data sites and labelling will be discussed in length, including specific requirements for ground truth and metrics that will be used for evaluating the system based on ground truth.

Chapter 4
Data Creation and Ground truth labelling

Image collection is the first and foremost step of the defect detection system. It discusses the diversity of the database from multiple sources explaining in detail about the data origin. It also includes creating a ground truth database which is essential in order to analyse the performance of the algorithms. This is the first main contribution as there are no existing pixel-wise labelled database for a microscopic pit data from rail axles.

4.1 Introduction

This chapter discusses the acquired and labelled data which is the first main contribution as there are no existing pixel-wise labelled database for a microscopic pit data from rail axles as per knowledge. After discussing the data collection design, setup and protocols in the previous chapter, this one focuses on the data itself. It explains the data origin giving details about the experimental setup, observations and findings of the data source. This gives a clear picture of the data acquired with full details. Once the images have been collected, they need to be labelled for quantitative evaluation as well as for training supervised models. In order to achieve a diverse and in-depth database for the defect detection system,

- Planned experiments throughout to acquire data from multiple sources (4.2)
- Started with laboratory trials conducted at TWI (4.3)
- Then prepared for workshop investigations with experts in Polimi (4.4)
- Performed validation trials on site 1 (4.5)
- Carried out Inspection on site 2 (4.6)
- Decided cut-off value based on workshop investigations (4.7)
- Devised a labelling procedure (4.7)
- Labelled the data which is also known as ground truth (4.7)
- Created database consisting of data images and their labelled images (4.8)
- Made selection about machine learning and deep learning models being multi-class or multi-label (4.9)
- Considered important indicators to evaluate the performance of detection system. They are based on computational comparisons between processed/predicted image vs the ground truth giving quantitative performance results (4.10)

Once all the material and equipment have been setup, and the main architecture of the defect detection system (DDS) has been decided, as discussed in the previous chapter, an image database is required for image analysis for the DDS. In order to do that, data needs to be collected either in the form of images or videos. Multiple sources of data have been included to give more depth and diversity to the database.

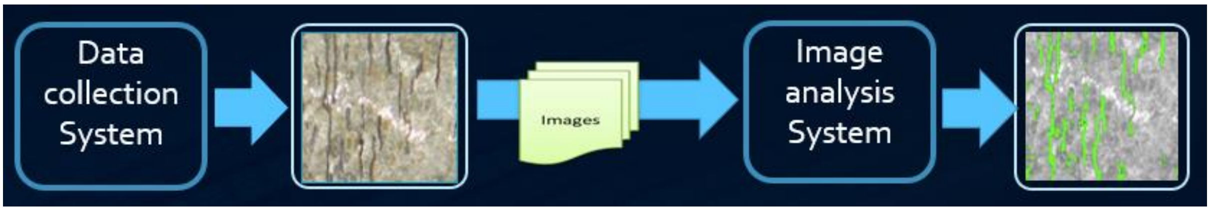


Figure 4-1 Basic framework of detection system

As can be seen in figure 4.1, image collection is the input to the detection system hence the images collected are needed to be carefully acquired. Data creation, collection (plus improvement) of 3000 plus images and labelling for pixel-wise pit data set of 141 million data points (attained from 115 images) and 100 image-wise classification has been acquired. Later on, this chapter describes the data labelling requirements and procedure. One of the biggest challenges of the project was to label the data. It requires a lot of the expert's time in order to label each and every micro-flaw with so much detail and correctness. It is a subjective matter even among the experts, highly time consuming and often really complex. Hence a set of 115 images were chosen that best represented the whole dataset including varied sources and components, mixed lighting conditions, density coverage on the image, number of flaws present and sizes of the flaw. A cut-off size was needed to be determined as illustrated by figure 4.2.

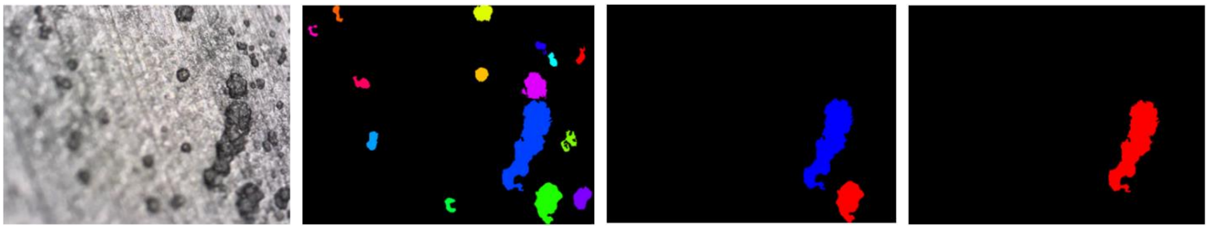
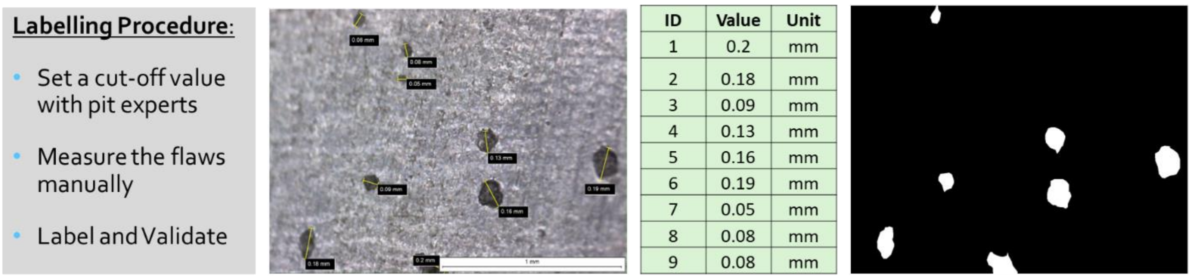


Figure 4-2 Showing varied cut-off thresholds of the flaw size to be detected

Labelling procedure was devised in order to create ground truth as shown in figure 4.3 and figure 4.4. This labelled data is used for measuring the performance of the DDS, which is discussed in the end of this chapter.



Labelling Procedure:

- Set a cut-off value with pit experts
- Measure the flaws manually
- Label and Validate

ID	Value	Unit
1	0.2	mm
2	0.18	mm
3	0.09	mm
4	0.13	mm
5	0.16	mm
6	0.19	mm
7	0.05	mm
8	0.08	mm
9	0.08	mm

Figure 4-3 Showing the labelling procedure of the ground truth illustrated with an example

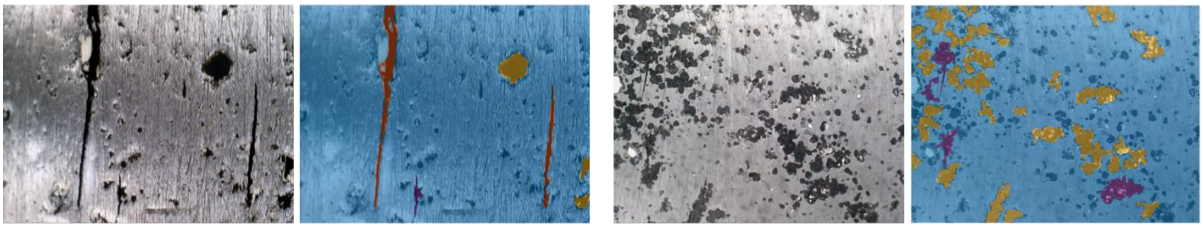


Figure 4-4 Showing an illustration of the ground truth after labelling on sample images

4.2 Data Sources

Real data is of crucial importance especially if there is a specifically aimed industry problem that needs to be solved. When computer-aided data (images) are created and used in such cases, they do not tend to cover the limitations of the real-life scenarios of the problem. Hence, even if some of the results in the existing literature research might show promise, they still remain theoretical in value. Pits that are formed under controlled environment are more well-defined and hence easier for the analysis to be conducted on them. While the real sample images deal with effects such as environment, grinding marks, scratches and other such factors. Hence, in comparison there are more complexities to deal with and therefore relatively more challenges to be analysed. As the thesis presented here is part of the industry-led RAAI project as an application, it was necessary to collect real data as a requirement. But also because there are no datasets available of such specifications.

For classification purposes especially intelligent classifiers, the more data acquired, the better, as this increases the depth as well as the confidence of the database. Similar kinds of data can increase the confidence level of the database, while gathering it from different samples gives more depth and diversity to the database. Therefore a considerable amount of work was carried out on acquiring the knowledge of the flaws to be detected, how to measure them and how to improve the data collection system. This of course also required knowledge of the failure mechanisms resulting from the incipient flaws.

As discussed in the previous chapter, the microscope was upgraded to improve image resolution and quality, and also the scanner was automated to improve data handling and ensure 100% coverage. This chapter is all about creating and improving the database. The next chapters are about making improvements to the image analysis system to speed up from manual analysis which will be discussed in the later chapters of the thesis. All images are taken with due courtesy of TWI Cambridge. Initially, there was more focus to capture pit images, as this represents the earlier stage in the corrosion fatigue process.

The next four sections discuss the datasets gathered from different sites and samples; entailing the setup, observation and findings. The data sources include the trials at TWI, investigations in co-operation with Applied Inspection Ltd at two site locations and the validation at Politecnico de Milano.

4.3 Laboratory trials at TWI

The test samples were two cracks developed in central radii by corrosion fatigue in an axle at TWI by applying 3-point bending test. The objective was to observe cracks generated by corrosion fatigue. A water circulation system was put in place in order to ensure corrosion on the axle. When the tests were interrupted, the water system was stopped in order to allow clearance and image clarity for the microscope tests.

4.3.1 Setup

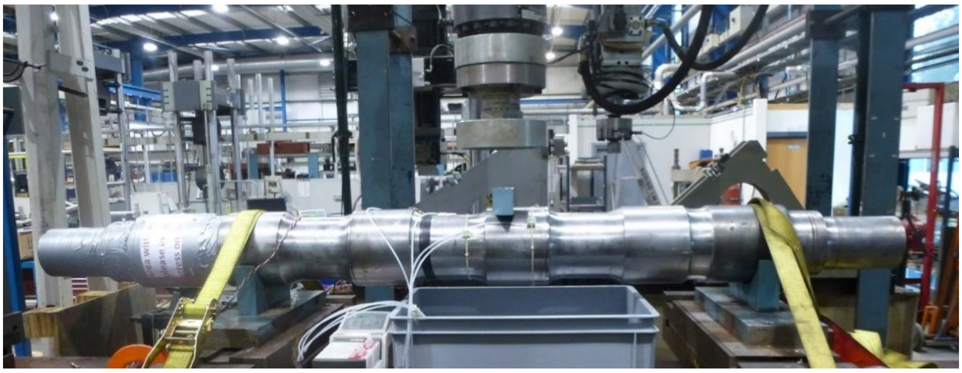


Figure 4-5 Axle set-up on fatigue rig and water circulation for lab collected data

Figure 4.5 and Figure 4.6 display the tests carried out on the axle. The tests were carried out on the 6 o'clock and 12 o'clock positions for the axle.

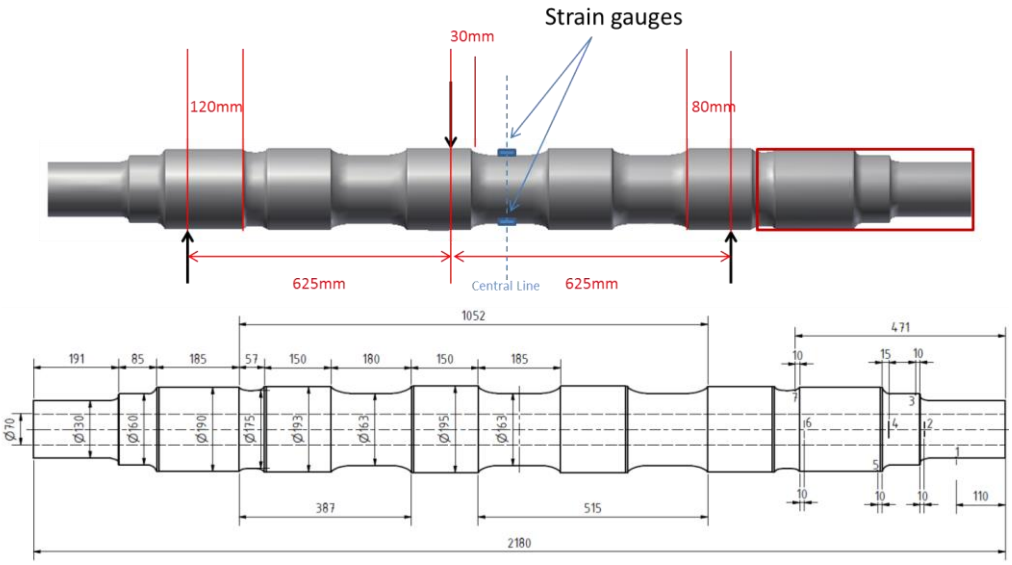


Figure 4-6 Location of the strain gauges for the lab tests

The tests were carried out throughout the fatigue tests and contained cracks at two sites, one large one resulting from the coalescence of a series of cracks, and many micro-cracks which had not propagated. This testing carried on for several months. There were a few testing issues with the axle:

- Water flow difficult to control – uneven corrosion
- Limited access to crack area (under rig) made inspection difficult during test
- Use of manual scanner made position of camera difficult to reproduce

- Limited scanned area for images meant sometimes there was no overlap, even while scanning in 5mm grid.
- 3 point bend is not truly representative of axle fatigue

4.3.2 Observations

Images from previous cycles were stitched together to match the area of the crack locations to locate the earlier development of these cracks. To note the axial position of the flaw on the component, tracks were alphabetized from track A to track E.

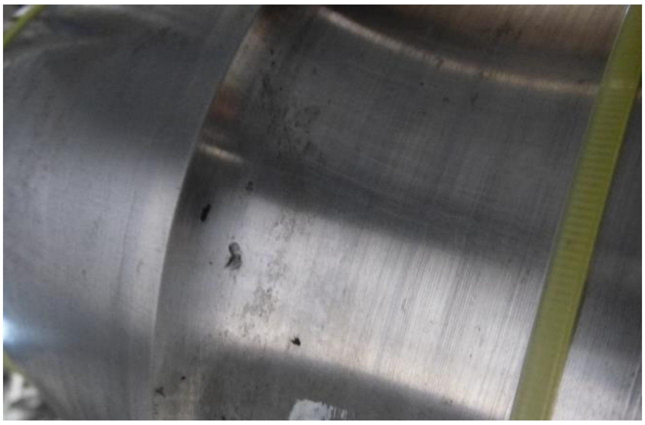


Figure 4-7 Sample axle that needs to have track markings to monitor flaw positions

1st crack at 12 o’clock

The tests were interrupted at regular intervals of 10000 or 25000 cycles in order to monitor the crack initiation and crack propagation as shown in table 4.1. The crack size shown is equal to the length of the crack times the depth of the crack.

Table 4-1 Lab data Bending testing log for 1st crack at 12 o’clock

Stop No.	Date	Load min. (kN)	Load max. (kN)	Cycles	Status	Crack size (mm)
0	19/10/2016	38	380	-	Test Start	-
1	23/10/2016	38	380	250,000	Stopped for crack checking, no crack	-
2	30/10/2016	38	380	500,000	Stopped for crack checking, no crack	-
3	04/11/2016	38	380	750,000	Stopped for crack checking, no crack	-
4	07/11/2016	38	380	1,000,000	Stopped for crack checking, no crack; load range increased to 20kN-450kN	-
5	13/11/2016	20	450	1,250,000	Stopped for crack checking, no crack	-

6	20/11/2016	20	450	1,406,118	Position limit triggered, no crack	-
7	25/11/2016	20	450	1,750,000	Stopped for crack checking, crack initiation found at the transition part	?
8	04/12/2016	20	450	2,137,567	Stopped for crack checking	?
9	12/12/2016	20	450	2,187,567	Stopped for crack checking, crack depth reach 2mm, test terminated	(30 to 40) x 2

A crack of around 1mm can be seen in figure 4.8 showing example Image B403-2016 after 1250Kcycles, at the 5th iteration, where the load was increased after 1M. There is a magnified image of the crack shown along the side. The crack was traced in track B from checking the data log book.

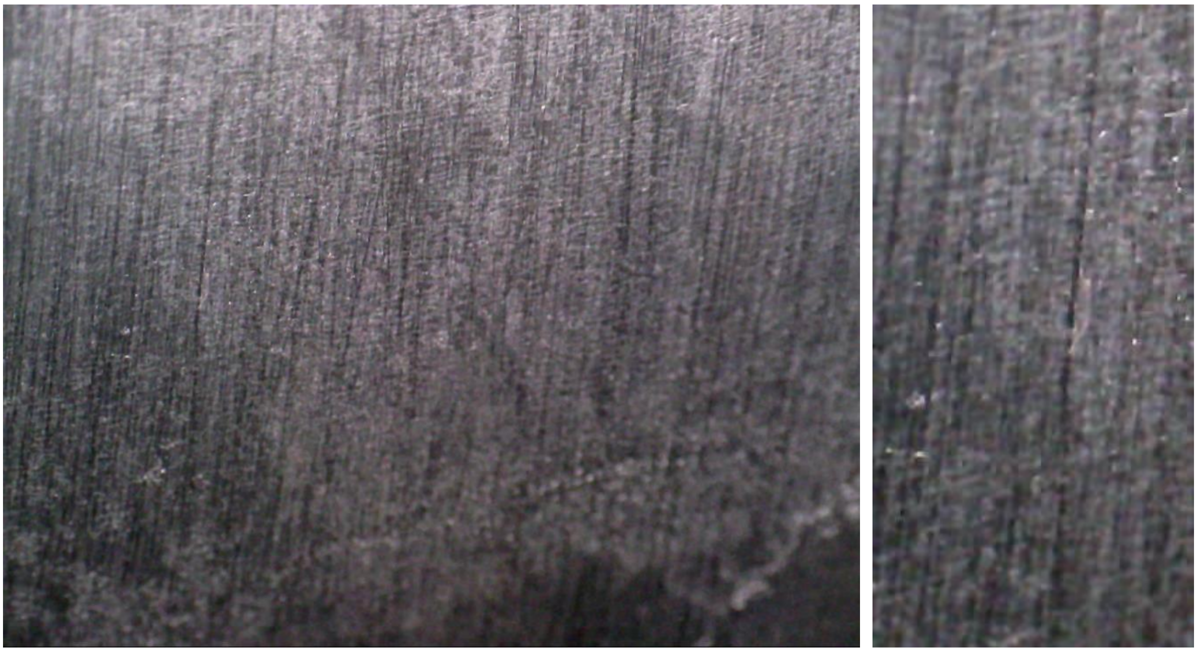


Figure 4-8 Image of 1st crack at 12 o'clock at 5th iteration

The frame images, along the duration of the test for this particular flaw, can be seen in figures 4.9-4.11 below, which show the flaw at other cycles subsequent to the 5th iteration. Image C055-2016 at 6th iteration, Image C442-2016 at 7th iteration, Image C510-106 at 8th iteration.

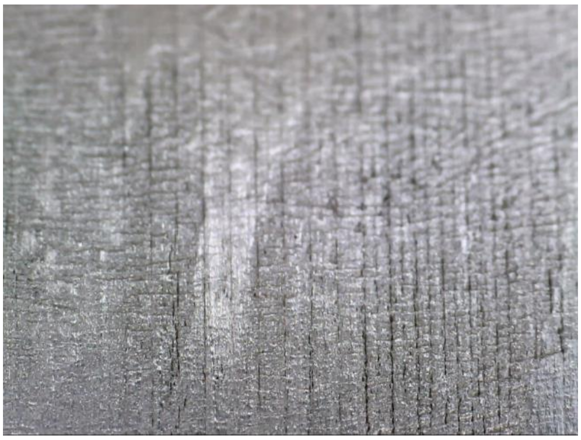


Figure 4-9 Crack image at 6th iteration from same area

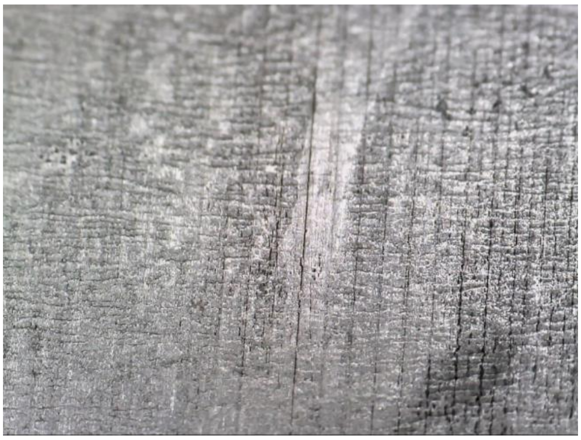


Figure 4-10 Crack image at 7th iteration from same area

By stitching the image frames together, the full length crack in the 9th cycle can be seen as shown in figure 4.12. Images of the final cracks were also taken with an iPhone camera to capture the whole crack in a single frame. It can be seen that both the microscopic-stitched images and the mobile camera one are the same crack.

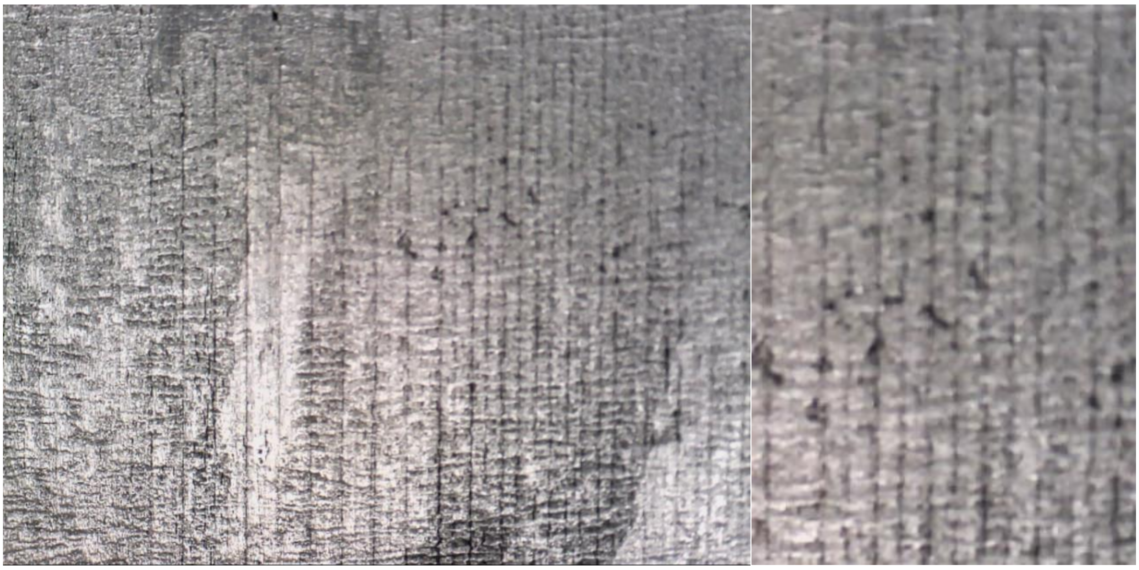


Figure 4-11 Crack image at 8th iteration from same area

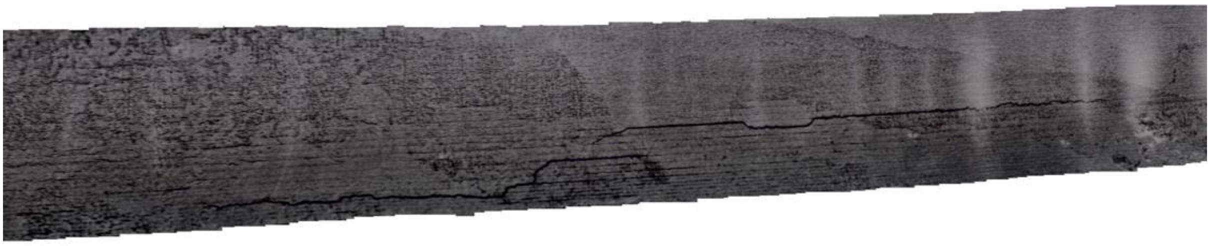


Figure 4-12 Crack image at 9th iteration showing stitched image (including the specific area)

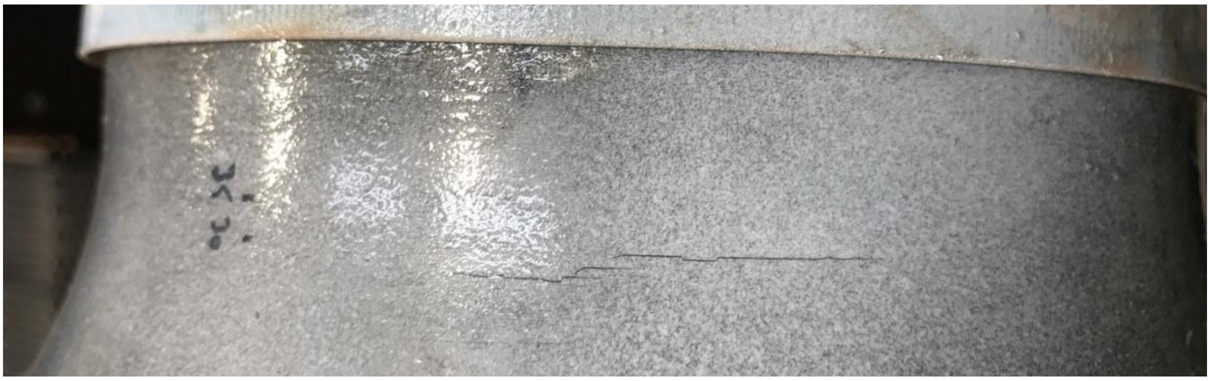


Figure 4-13 Crack Image taken with a mobile camera

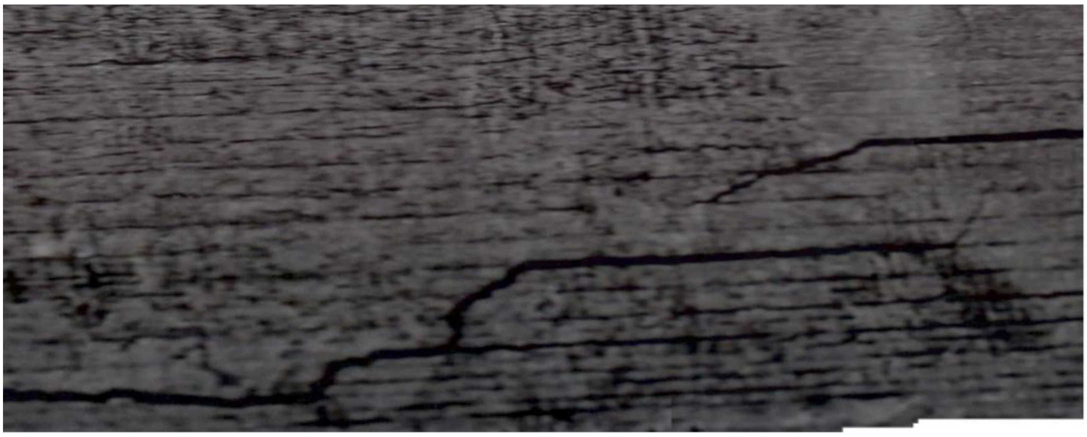


Figure 4-14 Magnified Crack image at 9th iteration

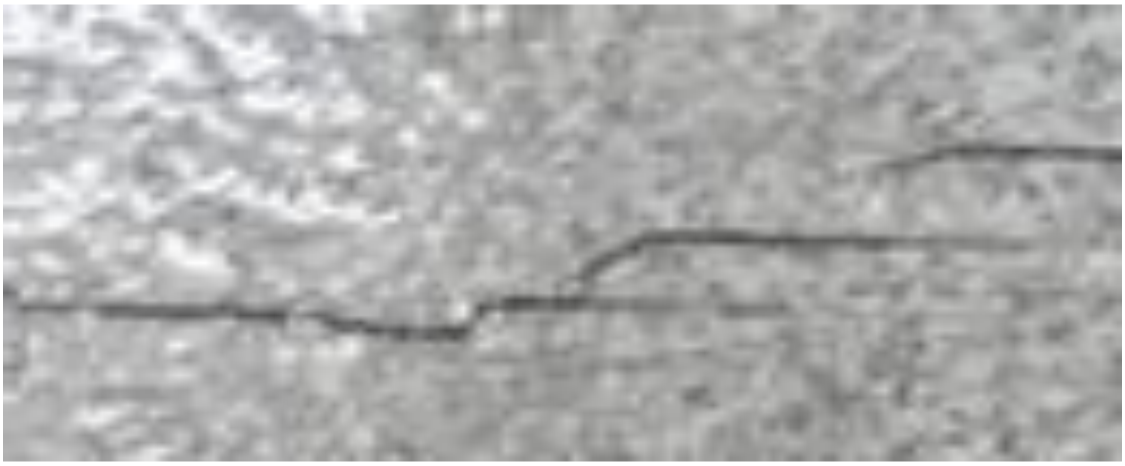


Figure 4-15 Magnified crack image at 9th iteration showing improvement after applying MPI

2nd crack at 6 o'clock

For the rail axle test on the other side, it was loaded for 600,000 cycles in total. The testing log is shown in Table4.2. This crack was at 180 degree to the 1st crack. Example image after 250K cycles, C722, with the Final Crack 600Kcycles, A934 are shown in figure 4.16 and 4.17.

Table 4-2 Lab data Bending testing log for 2st crack at 6 o'clock

Stop No.	Date	Load min. (kN)	Load max. (kN)	Cycles	Status
0	23/01/2017	20	450	-	Test Start for the other side
1	26/01/2017	20	450	250,000	Stopped for crack checking, no crack
2	06/02/2017	20	450	500,000	Stopped for crack checking, with 1 or 2mm crack depth
3	10/02/2017	20	450	600,000	Stopped for crack checking, crack is big enough so the test is complete

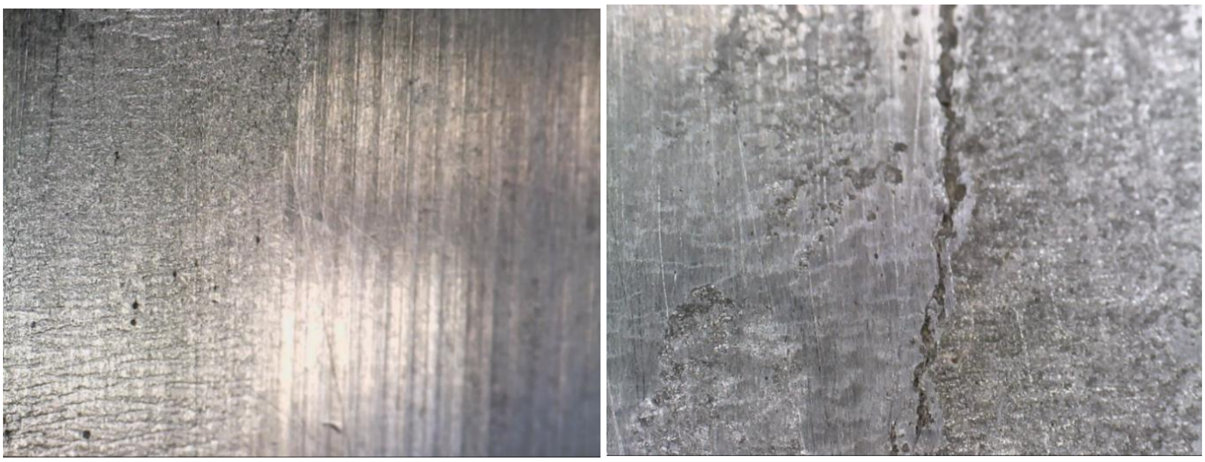


Figure 4-16 Image of 2nd crack at 6 o'clock (a) at 2rd iteration; (b) at 3rd iteration

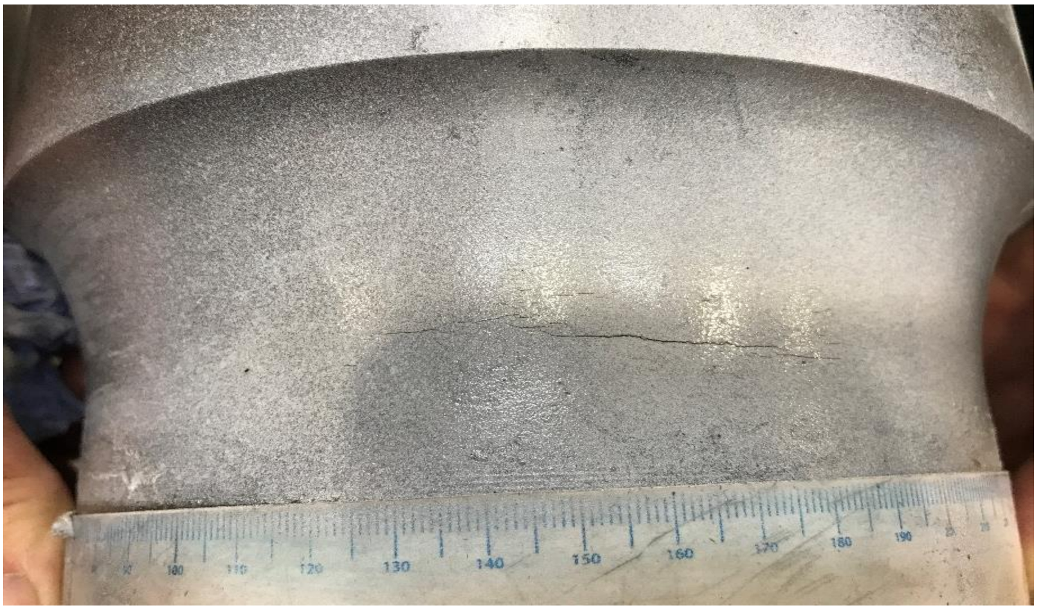


Figure 4-17 Image taken with a mobile of the 2nd crack after using MPI

4.3.3 Findings

- The degree of cleaning influenced the appearance (contrast relies on oxide remaining)
- Cracks visible at 0.3mm length at low magnification.

There were a few points that were learnt from this data case which were previously not considered:

- Cracks could start in machining lines rather than corrosion pits which makes them difficult to distinguish. Some of the earliest sign of cracks were ignored due to this reason, that they were just grinding marks. But after checking with Eddy current and confirmation of presence of cracks there, the cracks were then considered as cracks.
- Using MPI enhances the contrast of the images especially for the purpose of crack detection. Maybe the detail of the crack is not that clear but the deeper the crack the more visually it can be seen as illustrated through figure 4.18.

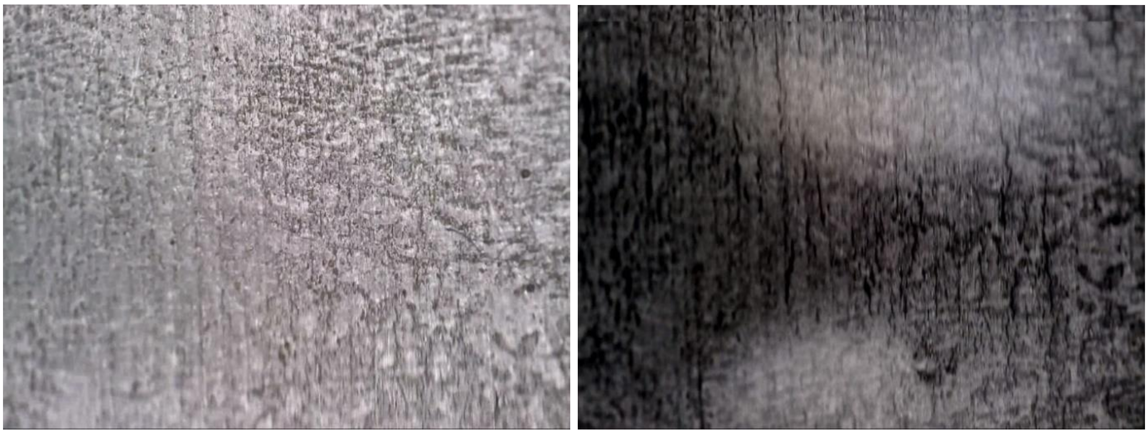


Figure 4-18 Improvements in crack imaging by MPI ink without contrast paint

4.4 Workshop investigation-1

During this visit it was possible to compare different axle preparations (DeOx gel, MPI, polishing, grinding) and to investigate them in different ways (different zoom levels, microscope resolutions), keeping in perspective that the images will be used as an input to the classification defect system. Then the output of this system, the measured crack or pit lengths, will be used for the estimation of residual axle's lifetime for the RAAI project. This workshop helped to inspect a set of axles, courtesy of Alstom with different surface conditions, some being more corroded than others, and some having cracks. Different combinations of microscope magnification and surface preparation was investigated.

4.4.1 Setup

The axle samples investigated were a high speed train (Pendolino), Trams (T1, T2 T3 and T4) and Freight (F9, F8, F7 and F3). The first step was to look through the axle sets visually and roughly outline the interesting areas. Then the next step was to check with these areas the microscope and mark the locations using paper and tape with a scale. Then the areas were examined with different magnifications of the microscope. After that, a comparison was made of different surface preparations and use of MPI.

The different surface preparation conditions during investigations were as follows:

- Initial conditions: Axle surface as it is just after being removed from service;
- After rust removal: Axle surface after rust removal by means of one or more applications of DeOx gel;
- After grinding: Axle surface grinded by means of a grinder;
- After air polishing: Axle surface polished by means of an air-polisher.

Axle's surface treatments when all carried out are performed in this order. In some cases, depending on the results, just some of these treatments are applied.

Different digital microscopes have different resolution which are as follows:

- Useful ratios to get defect sizes in millimetres after extracting them in pixel are, @60 it is 200px/mm and @180 it is 600px/mm.
- TWI's microscope has a higher definition than the other microscope used by Polimi which are, @60 it is 132px/mm and @180 it is 540px/mm.

4.4.2 Observations

Pendolino

Images were taken of the Pendolino axle shown in figure 4.19 and it was observed that the corrosion present was very light, with a maximum pit size of 0.4mm and no cracks visible. The axle was therefore in the pre-cracking part of its life in corrosion fatigue.

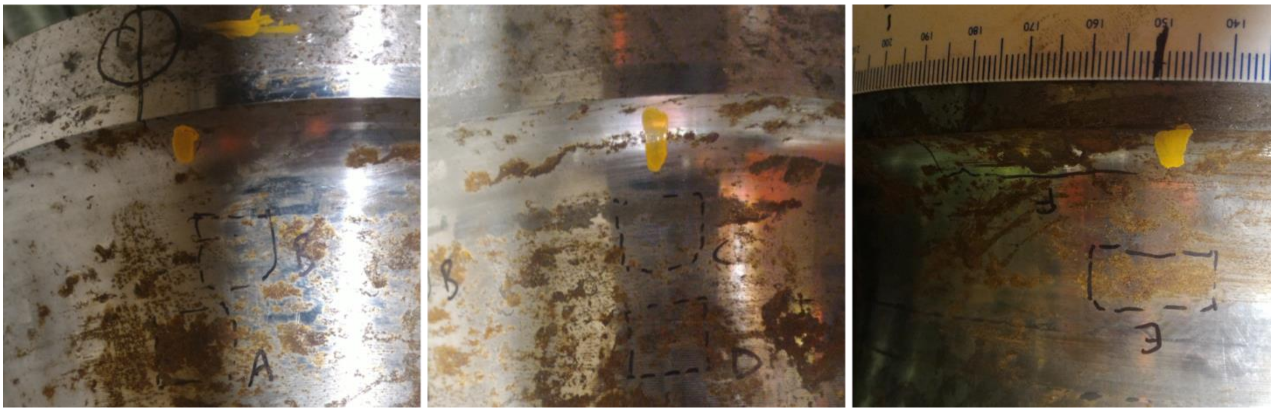


Figure 4-19 Data collection from the site sources along with marked areas

The next line of action was that all axles be shot-blasted to remove as much corrosion as possible, any visible areas of pitting be marked, pitted areas to be EC tested, MPI tested using a yoke and microscope examination to check for any corrosion fatigue cracking, Air polishing and chemical clean were shown to be effective at preparing the surface.

Trams

Images were taken using different tram axle sets and observed the following:

- Tram-T1: Cracked and corroded
- Tram-T2: cracked and corroded
- Tram-T3: Corrosion and cracks in the wheel-seat radii, and corrosion in the journal radius.
- Tram-T4: corrosion in the drive seat radius

The suggested line of action was to first apply De/Ox on all corroded areas, inspect the localised areas with Eddy Current, MPI the areas and use black ink only, examine with the microscope. An air tool could be used for polishing out in any areas of light corrosion, and then the surface re-examined.

Freight

Images were taken of the Freight_and observed that there was severe corrosion and some cracks (which were detected by EC beforehand). Initially the microscope was not able to detect cracks on the original surface because the surface corrosion obscured the detail. But after DeOx and MPI, cracks could be detected of the order of 0.4mm long.

4.4.3 Findings

- Cracks of the order of around 0.08mm could be detected with 180-200 magnification.
- Cracks of the order of 0.2mm could be detected with 60 magnification.
- It was deduced from discussion with the experts here, that the detection unit of 0.2-0.3mm is considered as significant damage to be looked and investigated. Hence, this measurement could be used for labelling the ground truth images.
- It was evident that using the higher magnification gave more depth (clarity) to the outline of the detected flaw.
- It can be seen that surface preparation effects the image outcome and flaw detection.
- The surfaces prepared by the polisher caused reflections and some difficulty of viewing the surface flaws.
- MPI by using black ink without contrast paint and viewed in residual field, considerably enhanced the crack detection capability
- Eddy current might be helpful in indicating the presence of a flaw with its location. But to find the measurements of the individual cracks is a bit difficult.
- There might be times when it is not possible to detect either pits or cracks due to reasons like bad surface cleaning, ineffectiveness of MPI and/or poor microscope

resolution. An example is the Freight sample, where without DeOx and MPI the defect was not detected.

4.4 Validation trials in Politecnico De Milano (polimi)

Polimi have been carrying out crack propagation tests on rail axles for many years. The purpose of the system is to estimate the remaining life of the axle given the presence of corrosion fatigue. It does this by detecting microscopically small cracks which appear originating from corrosion pits in the corrosion fatigue process. Then the life is estimated from the average length of the cracks detected. This measurement is normally done using replicas of the cracks and a travelling microscope.

The results from the microscope will be compared with results from the same area, which means area with the same fatigue history, taken with a precision travelling microscope. The test areas were from a sample designed specifically for fatigue tests. This had a thinned area to induce failure. From previous work the axle was known to have a life of 10million cycles, and had been fatigued for 3 million cycles (1/3 life) at the centre of the fatigued area. At areas remote from the centre of the thinned area, less of the life had been used, as it was at a lower stress level.

4.5.1 Setup

The validation tests took place in PoliMi where two axles were prepared for the tests. Figure 4.20 displays the axles used for the trials.

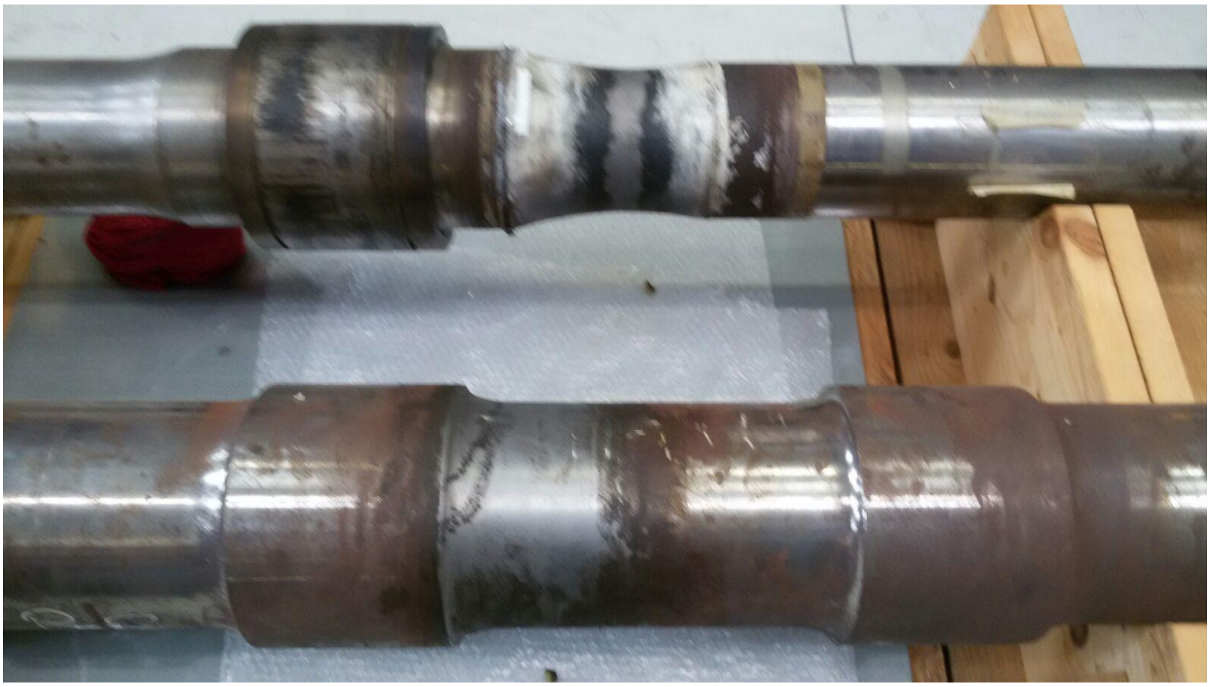


Figure 4-20 Axles used for the validation trials in PoliMi

The axle on the top of the image is the axle designed for the corrosion fatigue tests in PoliMi; the corroded area was cleaned prior to the validation tests. It is required to clean the area before placing the microscope as corrosion product; other residue can fill the pits, and cracks. The second axle in figure 4.20 is a Pendolino axle, which should contain corrosion. The axle was cleaned prior to the validation tests. The location of the cracking is in the radii due to the loading and corrosion conditions.

The axle was set up for testing as shown in figure 4.21 along with the microscope, oxide removal was done using the cleaning gel, the scanner was set up and data was recorded (around 1000 images and part of the circumference) at three longitudinal positions along the thinned area of the axle in order to compare different stress levels. The figure also indicates the flaw positions on the axle using a measuring tape. The data was converted into individual images and the image analysis program used to output lengths of indications.

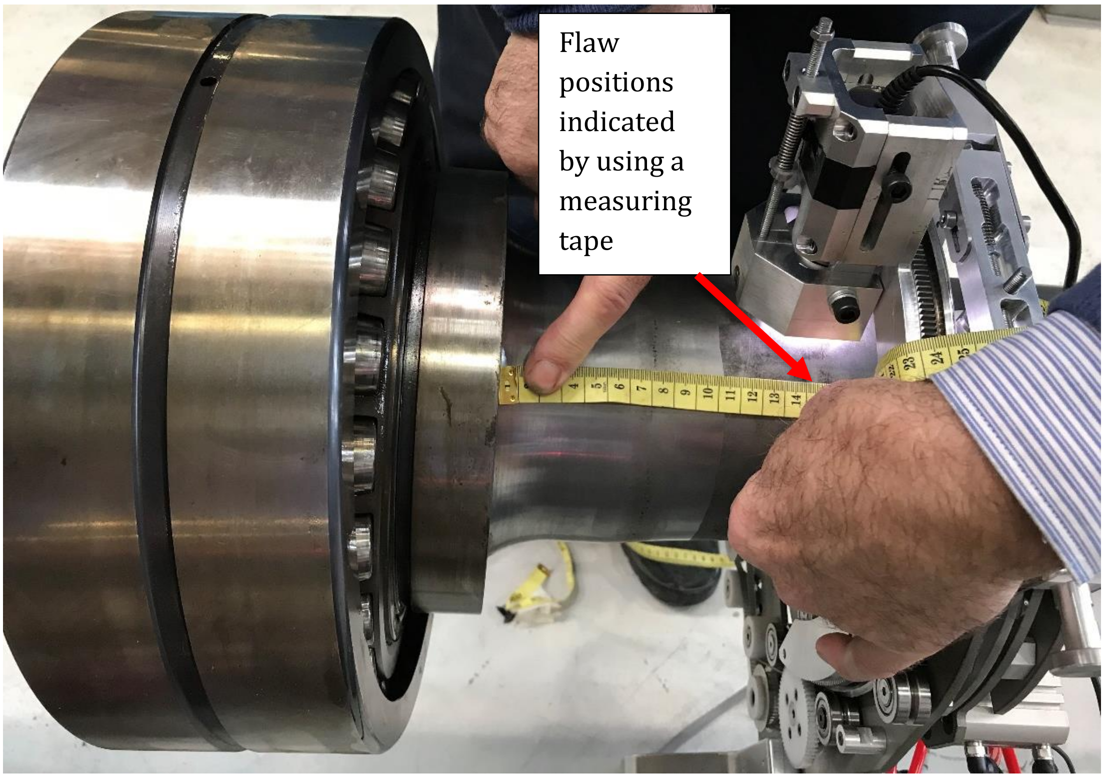


Figure 4-21 Axles used for the validation trials in PoliMi

4.5.2 Observation

Prior knowledge or experience has shown that if a very small threshold has been selected, a level of noise will also be selected in addition to the targeted features. Two thresholds were selected for the testing and validating purposes; 0.2mm at higher magnification and 0.6mm at lower mag. Three typical images from the three positions on the axle are shown in Figure 4.22, 4.23 and 4.24 side by side with their resultant processed images, showing the cracks and pits picked from the threshold process.

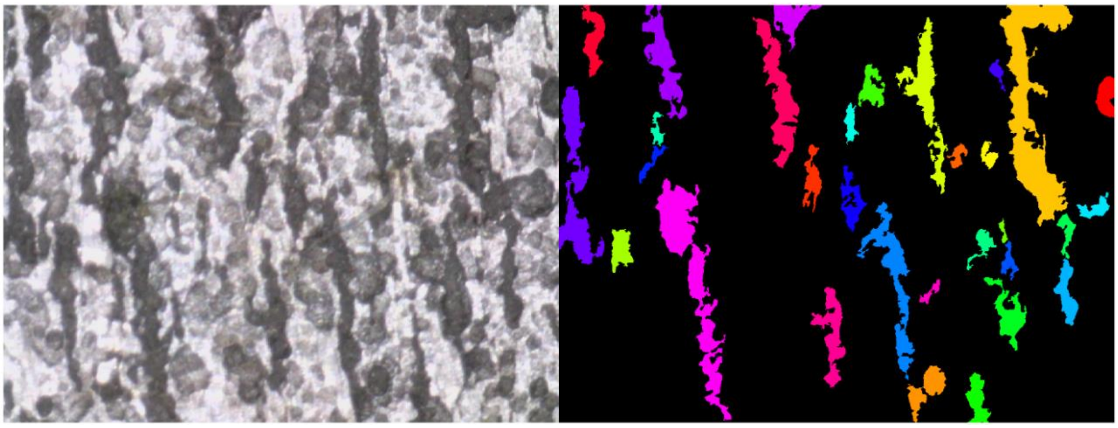


Figure 4-22 Image collected at position 140mm along with its processed outcome image

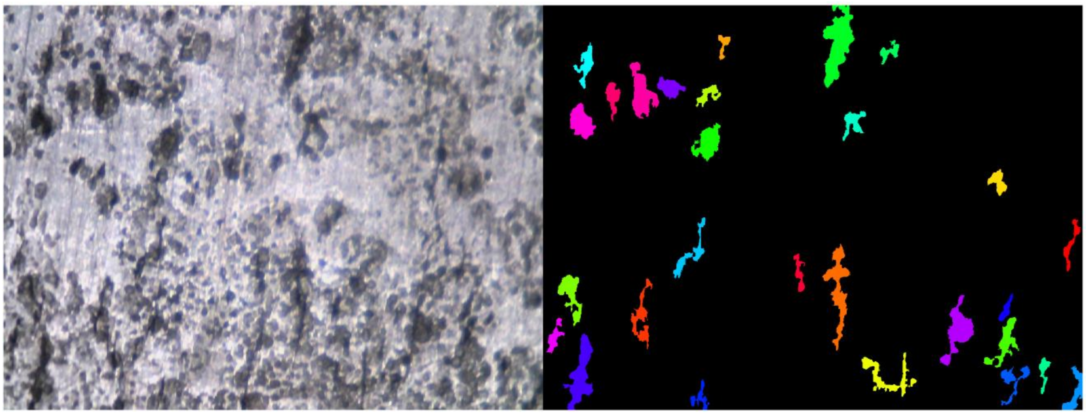


Figure 4-23 Image collected at position 210mm along with its processed outcome image

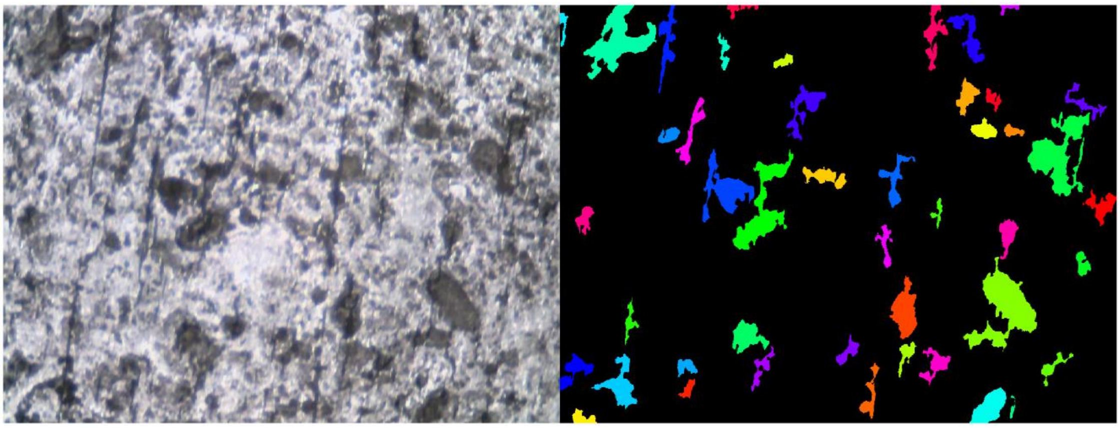


Figure 4-24 Image collected at position 240mm along with its processed outcome image

The summary of crack lengths for two different positions at both High and Low magnification with their respective Indication lengths are shown in Table 4.3

The FOV (Field of View) of Low mag = 4.9*3.6 while the FOV (Field of View) of High mag = 2*1.5. The high mag is restricted by the FOV of (2*1.5), hence the longest crack attained by the image will be dependent on this factor. The maximum size detected could be slightly longer than 1.5mm if the crack has grown diagonally. To summarise, it has data from two typical areas (Position 14cm is in the most cracked area at the centre, and Position 24 is at the edge), and two magnification settings of the microscope.

Table 4-3 Crack lengths produced from detection system with verified manual counting

				Manual Counting		Image analysis results		
Position	Mag	cut-off size (mm)	Frame #	# of cracks	Longest (mm)	# of Flaws (incl. pits)	Longest (mm)	Average (mm)
14 cm	High	0.2mm	F3	14	1.7	14	1.7	0.613
24cm	High	0.2mm	F1038	6	0.5-1.0	13	0.875	0.341
14 cm	Low	0.6mm	F167	52	2.5-3.0	47	2.809	1.05
24 cm	Low	0.6mm	F1043	none	-	3	1.34	0.95

Crack size measurements were done by using the precision travelling microscope, to be able to validate the image processing results. The longest crack lengths measured in the middle of the hourglass, at position 14cm at low magnification, is measured to be 2.5-3 mm. The average crack lengths should be around 0.4mm (400 microns) while with a minimum cut-off criteria of 0.6mm, the average length is supposed to be around 1 mm.

With a cut-off value set, the longest detected crack length will be same but the average lengths will change depending on the size set in the image analysis system. If the value is too low, then it detects a lot of noises as cracks, while if it's too big then it doesn't detect some of the flaws.

Table 4.4 shows the results, showing validation for position 14. The average length, from the image analysis system produced at position 14cm was 1.05mm and the microscope validated with it being 1mm. The longest length, from the image analysis at position 14cm was 2.809mm and the microscope validated with it being between 2.5-3mm.

Table 4-4 Verification of average and longest crack length at position 14cm

				Microscope measurements		Image analysis measurements		
Position	Mag	cut-off size (mm)	Frame #	Longest (mm)	Average (mm)	# of Flaws (incl. pits)	Longest (mm)	Average (mm)
14 cm	Low	0.6	F167	2.5-3	1	47	2.809	1.05

4.5.3 Findings & Validation

This data set provided a validation to the results produced by the image analysis system. So the post-processing method provides a quick way of highlighting, sizing and counting specific features. However, this is only possible with cleaned samples as corrosion and other residue can create false call. The method of cleaning should not create additional indications such as scratch marks or grinding marks which could be counted as cracks.

The software does also reduce the level of skill the operator as the algorithm set a standard for the desired defects to be counted. However good quality images are necessary for analysis as blurred images cannot be easily analysed.

4.6 On-site Inspection-2

This data visit was to perform in-situ inspection on the rail axles with the whole upgraded setup. Applied Inspection were carrying out inspections of a number of axles of ballast trains as shown in Figure 4.25. These rail axles had suspected corrosion on the outer radius of the journals. The location is shown in Figure 4.26. An opportunity was taken to use the RAAI corrosion inspection system for additional inspection of the corrosion to check for cracking.

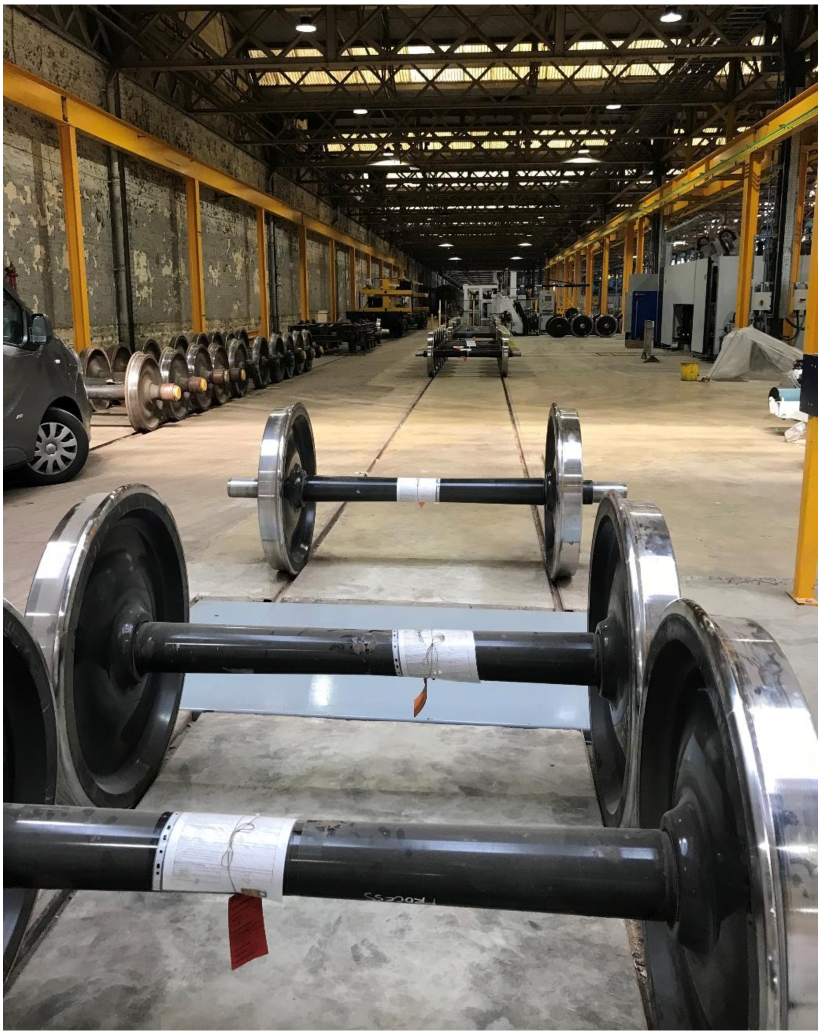


Figure 4-25 Site images of Freight (ballast train) axles in depot for inspection

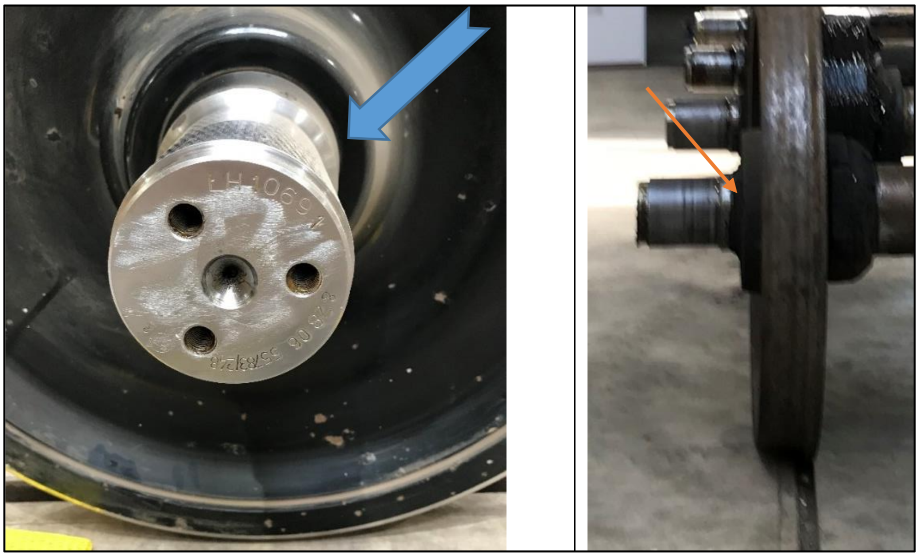


Figure 4-26 Site image showing location of suspected corrosion on axle

4.6.1 Setup

The inspection had to be carried out with the scanner mounted on the journal. This was of a much smaller diameter than that for which the scanner had been designed. To cope with this, two sleeves were made to fit the journal and make the diameter larger. These are shown fitted in Figure 4.27. The scanner was then mounted and the suspect area scanned as shown in Figure 4.28.

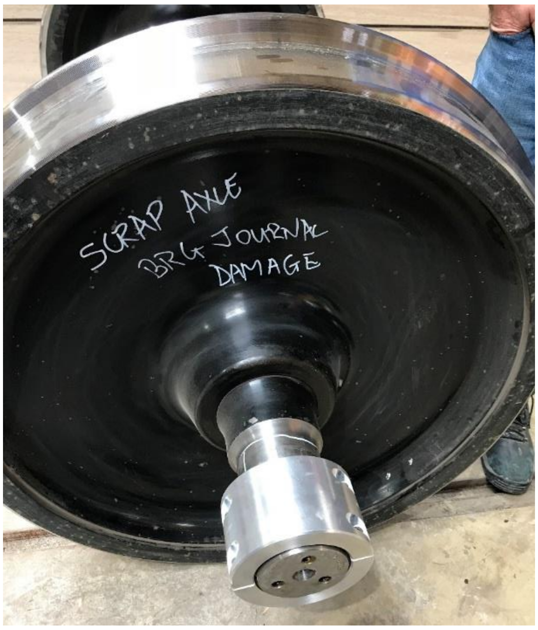


Figure 4-27 Site image of Sleeve to adapt to journal diameter

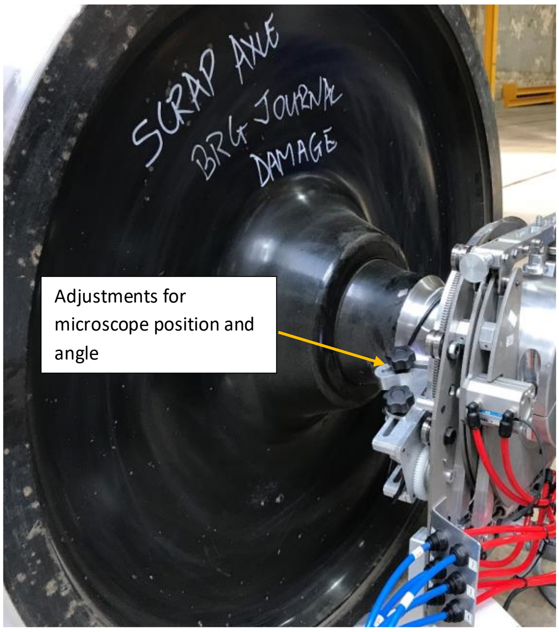


Figure 4-28 Site image of Sleeve to adapt to journal diameter with the scanner mounted

4.6.2 Observations

The area to be inspected had been cleaned by a finishing air tool. It was found that the normal RAAI procedure for cleaning would have required a number of applications of the chemical gel to remove the corrosion product. Areas of 4 axles were scanned with the system.

The corrosion appeared to have different patterns within each area. One pattern of the corrosion on the surface was that of areas, around 5mm wide by up to 20mm long. Within these patches, the corroded surface was continuous rather than exhibiting isolated pits. The difference between a finished area and a corroded area as described above is shown in Figure 4.29. There were some areas where minor pitting in the finished areas could be observed as seen in Figure 4.30. One possible crack indication from a pit was observed shown in figure 4.31, but it was concluded that this was from the machining marks.

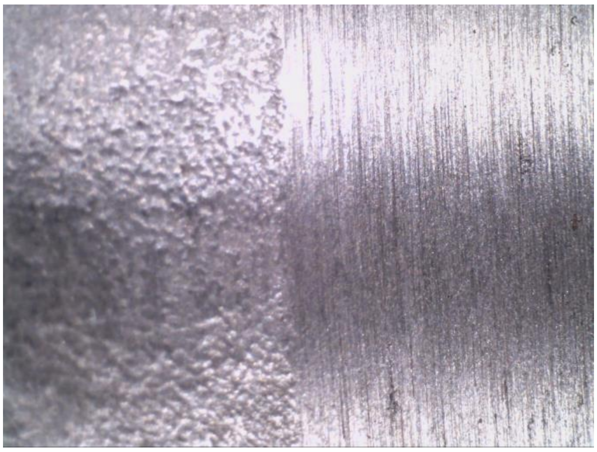


Figure 4-29 Site image showing finished and corroded areas

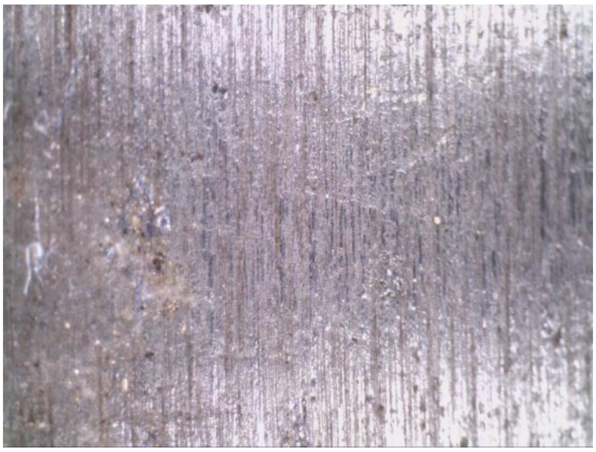


Figure 4-30 Site image showing Pits within the finished area

MPI was used to enhance any possible crack signals. A strong indication was found at the edge of the corroded area which can be seen in Figure 4.28, but it was concluded that this was a geometrical indication, as it followed the corrosion edge very closely.

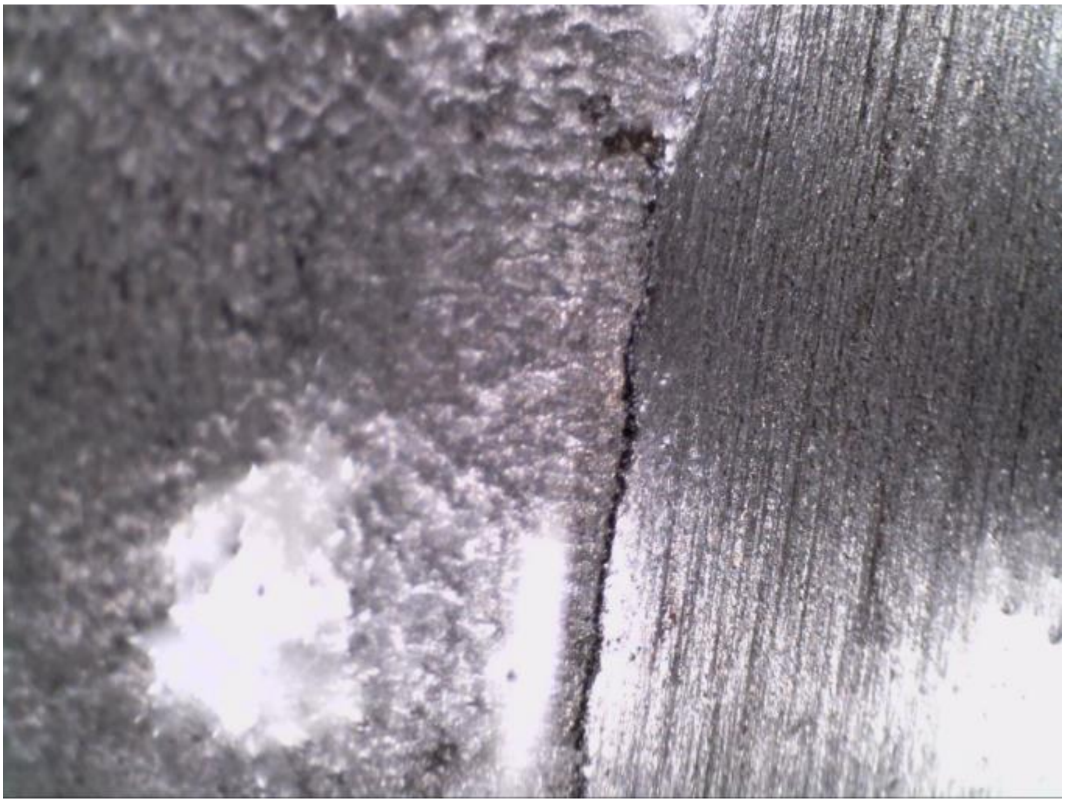


Figure 4-31 Site image indication at interface between corroded and finished areas

4.6.3 Findings

Therefore no reportable cracks were observed in the areas inspected, so the life estimation software could not be used (although an estimate of life can be made given no cracks were observed above a certain size). It is possible that as these were freight wagons with irregular usage the corrosion growth was quicker than crack growth in this case, so any cracks created were corroded away.

4.7 Ground truth

Ground truth (GT) or reference data forms the basis for performance evaluation in image processing and computer vision [133]. Having a reliable ground truth for the image data is essential to evaluate algorithms for the image analysis. An important question to consider is that whether a quantitative performance evaluation is required. If evaluation is required, then goals need to be established for the ground truth data. For instance, the purpose of the application requiring the ground truth data, the measurement indicators that will be used to access quality and success, and conclusions that must be made from the ground truth data in terms of accuracy and performance.

In simple terms, ground truth is the best possible knowledge of the data. It can be described in other ways for example, the gold standard, the ideal expected result, knowledge of the truth concerning a specific case, information via direct observation in contrast to inference. It is used in computing models to approve or disapprove research hypotheses. In the case of the cracks and pits in this work, no perfect knowledge of the real situation is easily available, so the ground truth is built up by expert elicitation of a selection of images. Its result for a model image analysis system is that the predicted output from the system is the same as the labelled ground truth (i.e the expert analysis), which would be 100% accuracy. However, the ground truth itself will be subject to inaccuracies particularly in this case. Nevertheless ideally, the model image analysis should be to be as close as possible to the ground truth and the system can be evaluated on this basis.

Predicted output from the model = Labelled ground truth

For both supervised and unsupervised classification, GT is used in order to compare and analyse algorithm's performance using metrics indicators. But for supervised machine learning, it also serves as an input data to the classifier, on the basis of which the model learns to classify. Problem arises when there is no ground truth information for a particular case problem. As in this case, ground truth didn't exist. Specifically, in this work for rail axles with microscopic pit and crack flaws, there was no pixel-wise labelled data available and therefore it was produced.

This section discusses the steps that were taken for ground truth, which are as follows:

- Know the cut-off size of the flaw
- Calibrate the flaws based on the selected size
- Label the images

4.7.1 Flaw Criteria based on Size & Magnification

Labelling is closely related to finding the root-cause of the task. Some ML scenarios may involve a subjective measurement where it is difficult to define an underlying objective truth, for example, an opinion that needs to be automated. In such cases, the ML model will be limited in its performance as it is dependent on the training data provided. This research scenario is similar in this regard.

The first step to the ground truth was to set the required size of the flaw essential for the project. This was a difficult task as inspection is a subjective matter and hence it was important to get a sense of the diverse opinion on this matter. It was realised that it needs to be more quantified. Same images were given to different pit assessment experts and they were asked to mark what they would like the system to detect as a flaw and the feedback results were quite different from each other. From this, it was evident that there needs to be a standard minimum size, to be able to quantitatively detect the flaw.

As discussed earlier, for the life prediction algorithm used by Polimi, for a Pit (p) the minimum size required is 0.5 mm or greater and for a crack (c) it should be 0.3 mm or more. If more conservative approach needs to be considered then, for a pit, it will be 0.2 mm or greater and for a crack it should be 0.1 mm or more, at the high magnification of the microscope as shown in Table 4.5.

Table 4-5 Initial complicated flaw size cut-off criteria for image labelling

Required size(mm)		Mag	FOV (mm)		# of Pixels	
Pit	Crack		x	y	Pit	Crack
0.5	0.2	60	6.5	4.8	100	40
		200	2	1.5	320	128
0.2	0.1	60	6.5	4.8	40	20
		200	2	1.5	128	64

A few calculations to establish the number of pixels in each case are considered here. These were done on the basis of the specifications of the camera being used. Images are taken at two magnification ranges with resolution of 960*1280; either at low mag which is around 60-80 or high mag which is around 180-220. For example, to find the exact number of pixels @80mag with FOV= 3.6*4.9 => 266pixel/mm and 261pixel/mm. There are two approaches for the size; one is a bit more conservative than the other. So as an example, if the picture has been taken @80 and we want to extract p>=0.5mm and c>=0.2 then the pixels equal p>=133 pixels and c>=53.2 pixels. By these calculations, the approximate size, needed to be eliminated from the image, was known.

The next step was to decide the magnification to be used as it seems that with a higher magnification, a more clear view of the sample is obtainable and the flaw is more pronounced while with the lower magnification, more area is covered because the field of view is bigger. As an example, an image with a few pits and also a crack, was taken at both magnifications (Figures 4.29 below). It can be seen in the figure that at a higher magnification the pit-to-crack flaw is more clearly visible. In short, the best way to detect a small flaw is to capture it at a higher magnification. So the higher the sensitivity to be detected, the higher the magnification required. However, this means that the field of view is smaller, so more scanning, more images and more processing is required. It will be necessary in practical situations to compromise these issues.

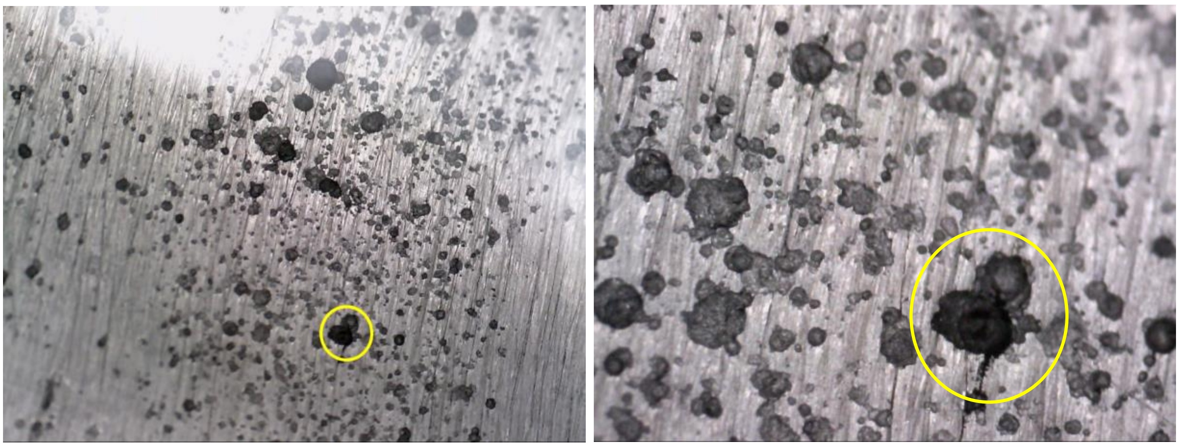


Figure 4-32 Images showing the same area at different magnifications

Hence, it can be seen in figure 4.32 that at same view at different magnifications, a crack is visible at the higher mag but at the lower mag it looks part of the pit edge but not so much as a definite crack. The solution might be to take pictures at higher mag only as these show better results with the Image processing.

Table 4-6 Final flaw size cut-off criteria for image labelling

Required size(mm)		Mag	FOV (mm)		# of Pixels	
Pit	Crack		x	y	Pit	Crack
0.3	0.3	60	6.5	4.8	64	64
0.1	0.1	200	2	1.5	64	64

To conclude, the initial flaw size criteria was quite complicated that required to develop separate processing algorithms for different flaws. But flaw size criteria was changed to make it more standardised for an image shown in Table 4.6.

For a pit, the minimum size became 0.3mm from 0.2-0.5mm. For a crack, it became 0.1 and 0.2. It picks roughly around 64 pixels, which if its high magnification it is 0.1 mm and if it's low magnification then it is 0.3 mm. In this case, the size is same for both the flaws, cracks and pits which makes it independent of the kind of flaw and the magnification of the microscope. Hence, no complicated calculations were required any more. In the end, just produce the flaw measurements for both magnifications, as the cut-off value is same. For pit and crack both, the thresholds selected for the testing and validating purposes were; at higher magnification, the minimum detectable size is 0.1mm and at lower magnification, the minimum detectable size is 0.3mm.

4.7.2 Calibration measurements

Once the criteria were selected, it became easier to compare the performance indicators of different image processing methods that were being examined. Hence this is a crucial stepping-stone in the ground-truth labelling. This step involved marking the flaws on the image by making measurements as shown in figure 4.33. These measurements will be used, to pixel-wise label the images. Two tools were used for this purpose; one is the camera's own software tool and the other is an open source measuring tool called 'Image Measure'.

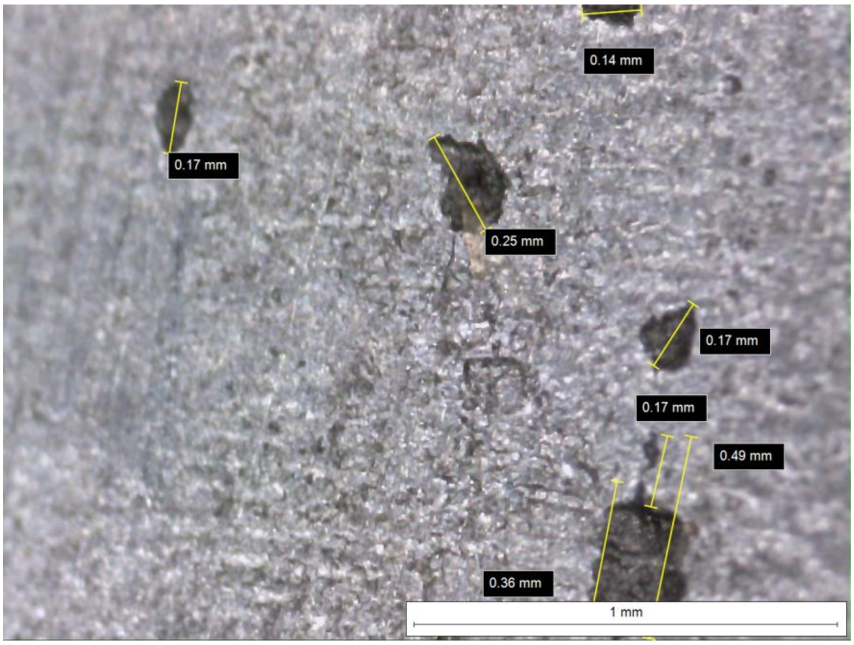


Figure 4-33 Manual measurement step to produce ground truth

4.7.3 Labelling Procedures

The first step towards this task was to measure the flaws. This was done by marking on the images attained which helped to determine the sizes of the defects and on that basis draw or label the ground truth images. Following are the different methods that were experimented for the ground truth images to be labelled:

4.7.3.1 Manual marking

This was the initial method used for labelling a shown in figure 4.34 but it didn't seem accurate on a pixel-level. For it to be used as the standard basis for comparison, it lacked the required accuracy for the microscopic measurements.

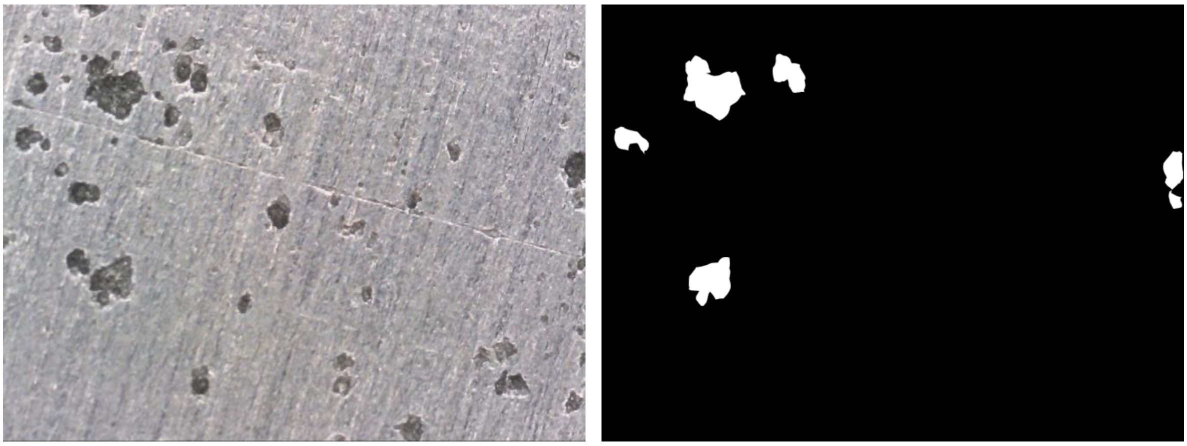
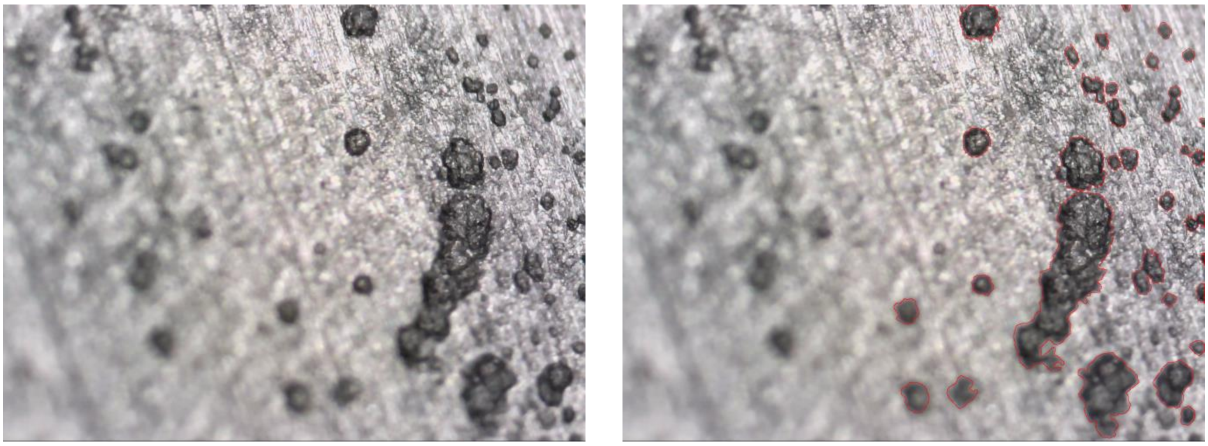


Figure 4-34 Showing original image and ground truth image after manual marking

4.7.3.2 Photoshop marking- Labelling method1

The image was marked by using Photoshop tool as shown in figure 4.35. This was better than the previous labelling system but research for a better labeller went on side by side.



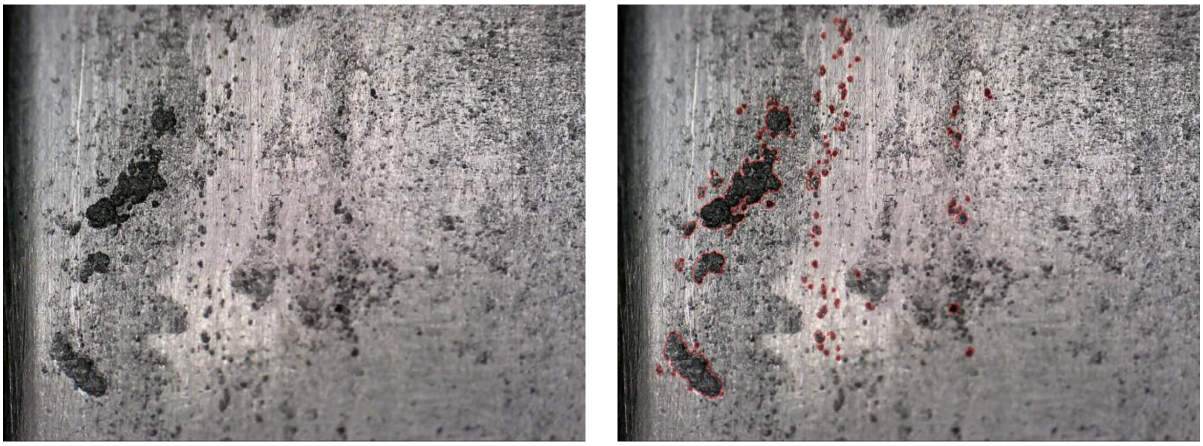


Figure 4-35 Showing original image and ground truth after Photoshop marking

4.7.3.3 Selected Image labeller method –Labelling method 2

This was a more suitable labelling method used. The Image Labeller app provides an easy way to mark pixel-wise region of interest (ROI) labels on an image. This was performed by manually labelling each image frame from an image collection. This was the next step after doing the measurements; the exact dimensions of the flaw are done so they are marked accordingly. Hence these are based on the agreed upon size criteria of the flaw. They were marked on the basis of three criteria features which are; Background in BLUE, Pits in YELLOW, Cracks in RED and Pit-2-crack in PURPLE. The GUI of the MATLAB application can be seen in figure 4.36 that shows Image_A614-2016 being labelled depending on the set criteria. It can be seen in figure 4.37 that the image is labelled using all four categories using the Image Labeller application.

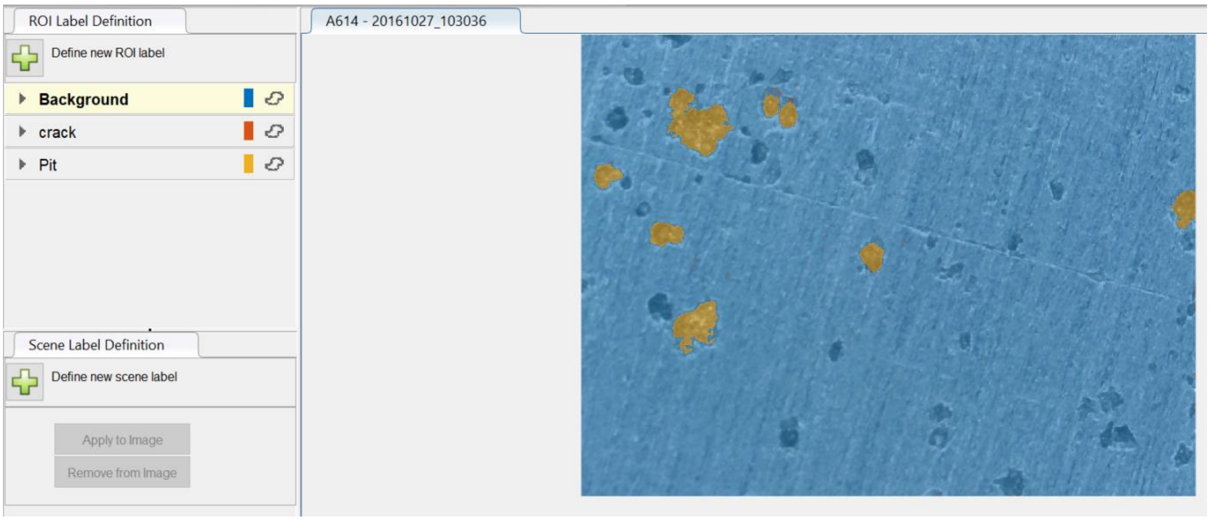


Figure 4-36 GUI of Image Labeller showing an image being labelled

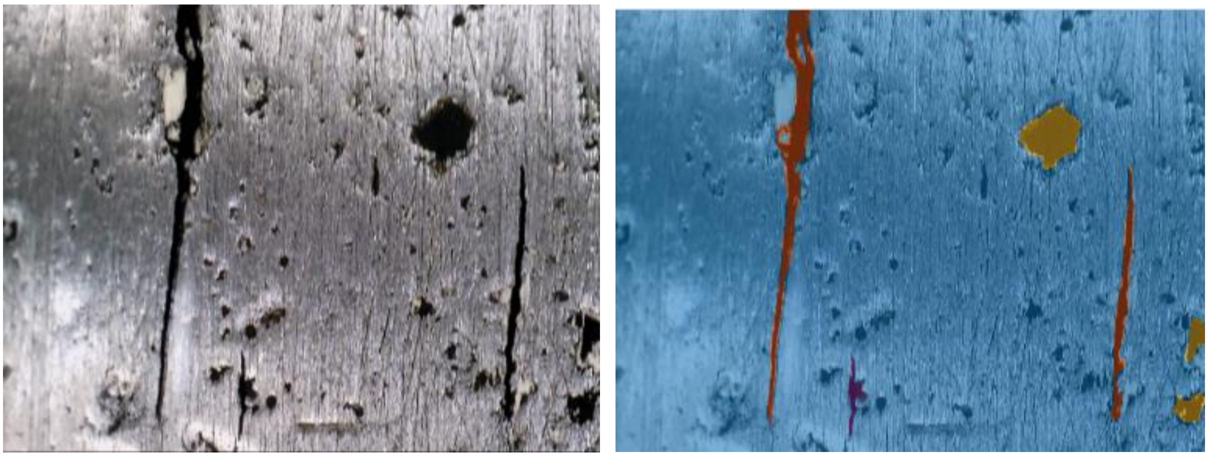


Figure 4-37 Image showing pits, cracks and pit-to-cracks marked by Image Labeller

4.7.3.4 Selected Method – Json –Labelling method 3

To save the outcomes of localisation and segmentation, a reliable and stable storage module was required. JavaScript Object Notation (Json) application has also been used for the labelling, as it saves extra information about the labelled image, which is more useful for deep learning. Labelled data was created by using this method for the keras model classifier for the pixel-wise deep learning model by using images with cracks.

4.7.4 Conclusion

This ground truth step has proved to be one of the biggest challenges of the research work. It is the standard against which the performance of the applied methods will be compared depicting an expert's opinion. This section discussed the steps that were taken for ground truth which can be summarised as follows:

1. Go through the database and select right data
2. Manually identify, the pits and the cracks, into separate flaws
3. Decide the cut-off value of the flaw size (section 4.7.1)
4. Count the flaws
5. Calibrate to measure the lengths and widths of the flaws (discussed in section 4.7.2)
6. Validate them with experts
7. Label them by marking (discussed in section 4.7.3)

In the first step, calibration measurements as shown in figure 4.38, were made on the basis of the flaw size selected which is around 0.1mm for higher magnification and 0.3 for lower magnification.

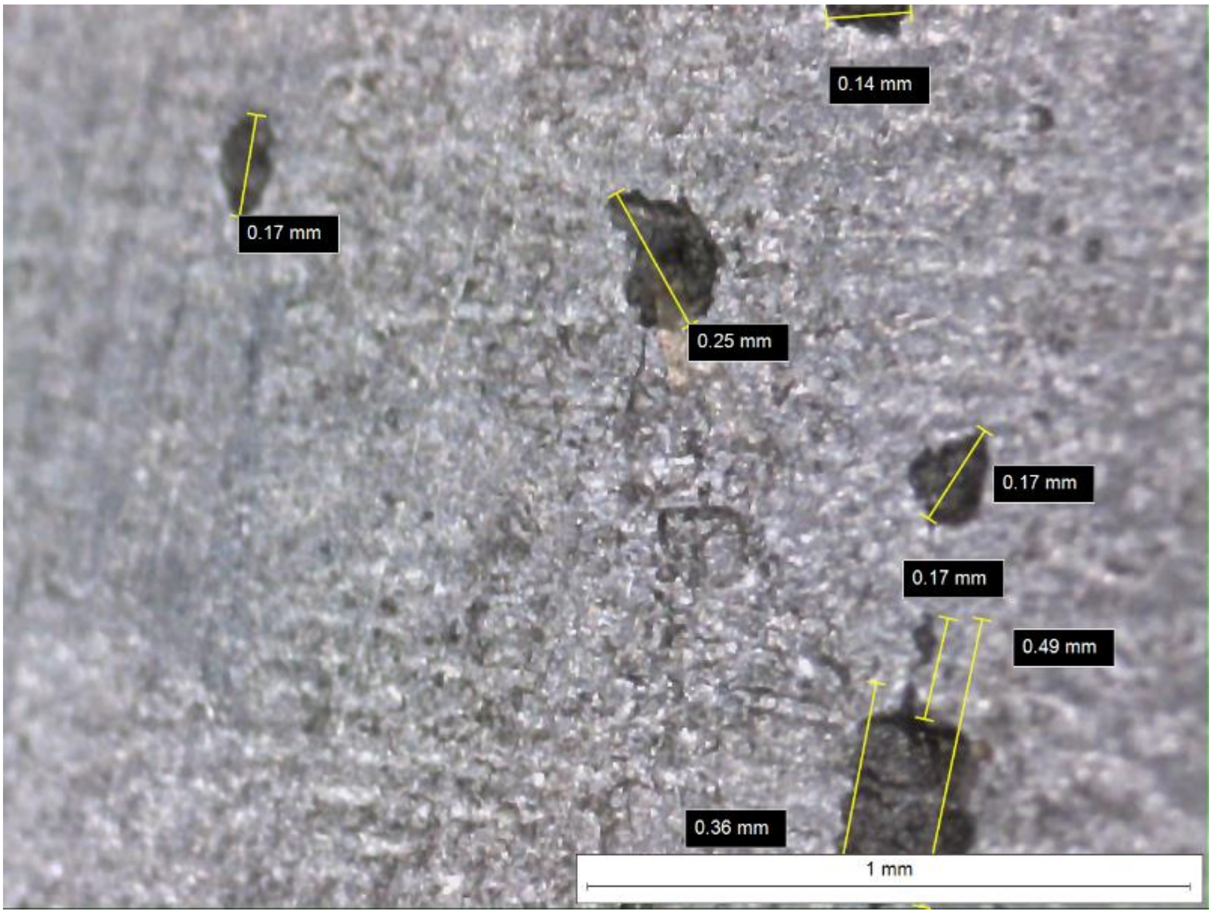


Figure 4-38 Example of a higher magnification image being measured

The next step is to label the images. Different methods were tried like manual marking, Photoshop marking and in the end Image labeller tool was selected. This has been shown in figure 4.39 using ImageB514-2016 as sample. The next step is to perform the metrics evaluation. To find efficiency of an algorithm, there are some measures that can be used to determine and analyse its performance. This is done by comparing the predicted data against the ground truth for evaluation.

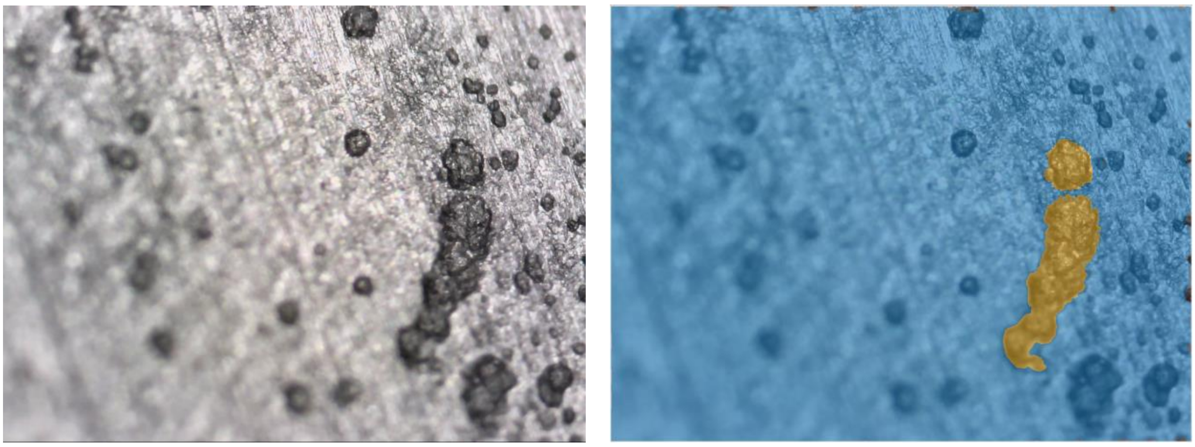


Figure 4-39 Sample image showing labelling marked by Image Labeller

In the next two sections of the chapter, it will discuss the database structure used that consists of both data and ground truth and then it will discuss important metrics that have been applied to evaluate the performance of the algorithms.

4.8 DATABASE STRUCTURE

Nowadays, many pre-trained networks are available but for a model to learn correctly, it requires data. Hence, assuming the right data is used, the size of the training set and/or ground truth data is the key to its accuracy. As the data grows larger, it improves the performance of the model. Supervised algorithms require response data i.e. manually-annotated ground truth to be fed into the classifier models. This is the challenging part as it requires massive amounts of correctly labelled data. For this research, flaw images were collected. Total of around 3000 images were collected but many of them had to be discarded due to the bad quality of the images, which includes blurriness and bad lighting. As mentioned before, the data collection process was gradually improved during the course which has helped to gather better quality images for later phase of the research.

Multiple options for labelling were considered before choosing the 'Image Labeller'. Amount of labelled data was difficult in our case. Labelling rust and no-rust is easier in comparison to classifying pits and cracks as usually there are pits present when cracks are present. Another point is that labelling pixel-wise is more time consuming with each picture taking an hour minimum. In short, labelling is more complex in the specific project case. Labelled data consists of 141,312,000 (141 million) data points for training and testing the models. This is a very high number of data inputs hence it required powerful computational devices to run such models.

The pit set 'p1' consists of 115 images that have pit flaw/s. This p1 data set has been used throughout the classification for comparison purposes, implemented IP, ML and DL techniques. Since this is pixel-wise labelled hence the size of observations as input are, 115 times the image size. The resolution size of the images collected is 1280 times 960 which gives 1,228,800 pixels per image. So in total this means 141,312,000 many training & testing data into the classifier. This is a lot of data that requires a lot of computer processing and storage hence it had a lot of computational and hardware challenges to start with.

To apply performance measures, a pixel-wise labelled dataset of 20 crack-only images and 115 pit-only images, have been created. These images can be classified into 'flaw' or 'not flaw' main classes. For further specificity, flaws can be placed into four general classes which are as follows:

a) Cracks only
b) Pits only
c) Pit-2-crack
d) Background

The structure of the data sets for pixel-wise classification is a very simple one which contains; Pits only and Cracks only. For the image-wise classification for deep learning it contains two models; Pit or not-pit model, and crack or not-crack model.

For the data collected, following shows the condition of the structure being inspected, from the different sources:

- TWI – progress of flaws from pits to pit-to-crack to cracks; consists of both pits and cracks
- Site visit 1 – used for experimenting with MPI with ink or no-ink for better images & flaw size discussion; consists of pits and few cracks
- Polimi- majorly used for validation; consists of more crack images
- Site Visit 2- for an on-site visit to diverse the data; consists of pits, no crack found

4.9 Selection between multi-class and multi-label (for ML & DL)

The image segmentation method used can count and locate flaw/s but cannot classify the type of flaw. For that, deep learning has been used in order to predict the flaw type in the image.

The DDS can be designed in the two possible design models. This section discusses about the labelling and classification of the pixels and to select one of the methods for the output response. Multi-label models means a pattern that can belong to more than one-class. While multi-class means that the

- Classes are mutually exclusive
- Treated as OVR (one vs rest) case
- Binary classification is simply multi-class classification with 2-labels

For this project, in the broader sense, there are two major classes:

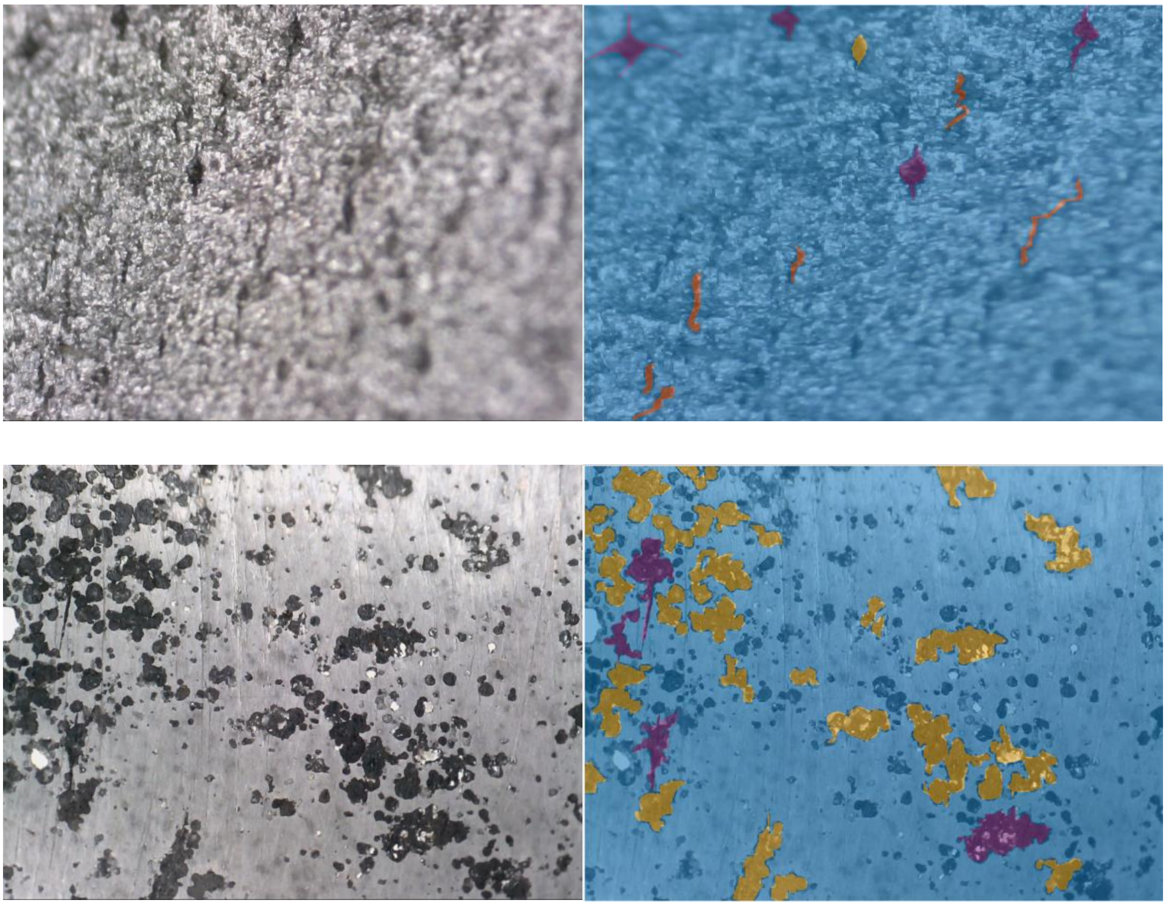
1. Flaw: this consists of Pit, crack and both (Pit2crack i.e. crack initiating from a pit)
2. Non-flaw: background

But we want to be able to differentiate between the flaws like when the flaw is a pit, crack or both. The reason to do this is that when the surface has a crack, then it is much more important that it should be detected.

For pixel-wise classification, the problem is with pixels we want to label as both; so I have added it as a separate label so a pixel can only belong to 1-class i.e a multi-class classification solution. The reason why I want to add a separate label is that there are some pits/cracks which belong to both the classes otherwise the labelling will be highly challenging and time consuming as it will have to follow certain set of rules such as,

1. If a crack is seen initiating from within a pit from one side then label it half way across as crack and rest as pit
2. If crack seen from both the ends then all of it is a crack

So instead of applying this, in case the object has ‘both’ then we just label it as ‘both’ which shows that it is still important as it has a crack but the inspector can double check it. For the images shown in figure 4.10, Red = crack, Purple= Both, Yellow=Pit, Blue = background



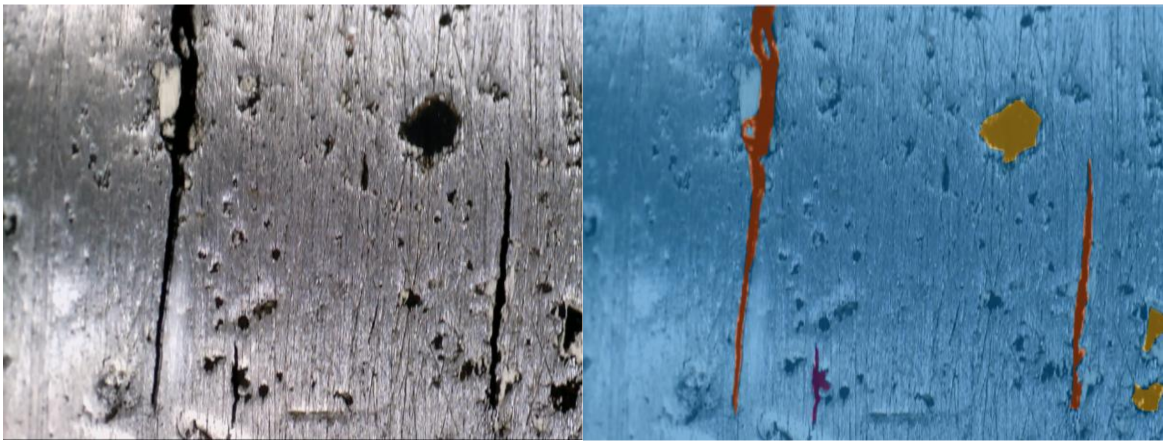


Figure 4-40 Image labeller showing ground truth marked with separate classes

There are a few different solutions possible for this problem.

Table 4-7 Defect labelling complexity showing different classes in the table

Pixel #	Defect type	Defect labelled
1	Has a crack only	Crack
2	Just background	Background
3	Has crack and pit both	Both
4	Has a pit only	Pit

- In simple terms or broader view just considering whether a pixel has a flaw or is just a background, considering the above pixel values as shown in table 4-7, the output will be as shown in table 4-8

Table 4-8 Apply binary classification, flaw vs background

Pixel #	Defect
1	1
2	0
3	1
4	1

- And if we want to apply two separate system; one for pits and one for cracks then the output will be as in the table 4-9 and table4-10 for pit and crack respectively.,

Table 4-9 Pit Model after applying two separate models for solving the problem

SYSTEM 1	
Pixel #	**Pit**
1	0
2	0
3	1
4	1

Table 4-10 Crack Model after applying two separate models for solving the problem

SYSTEM 2	
Pixel #	**Crack**
1	1
2	0
3	1
4	0

- If its multi-label then one pixel could belong to two columns at a time so the output will be as shown in table 4-11

Table 4-11 Apply multi-model solution to the problem

Pixel #	Pit	Crack	Background
1	0	1	0
2	0	0	1
3	1	1	0
4	1	0	0

- If its multi-class then all the columns need to be mutually exclusive of each other i.e. one pixel can only belong to one class, so output will be as shown in table 4-12

Table 4-12 Apply multi-class solution to the problem

Pixel #	Pit	Crack	Background	Both
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1

4	1	0	0	0

Hence it can be seen that the difference with the multi-label and multi-class is the 3rd row in table 4-11 and table 4-12, which is the pixel associated with both a pit as well as a crack. To clarify this point, if we consider sample 7, as shown in figure 4.41 below and think of object-wise classification (for simplicity) then the output we be, as shown in table 4.13. The method used is that the background is separate [all zeros], pits are separate, if pit then '1' else '0'. Similarly for cracks, if crack then '1' else '0'.

Table 4-13 Apply the multi-class method on an example image

Method 1 – Multi-Label	**Method 2 = Multi-class**
Crack = 10	Crack = 6
Pit = 5	Pit = 1
	Both = 4
Background = rest	Background = rest

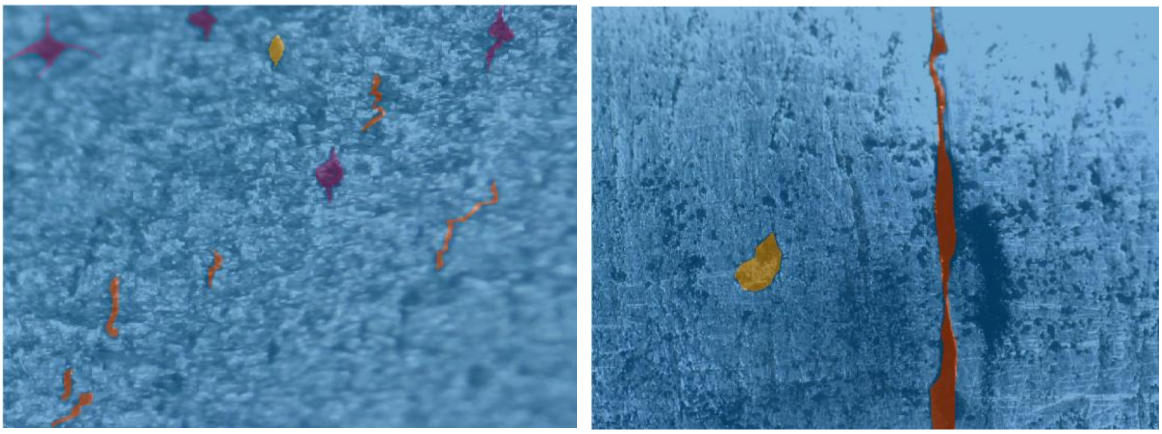


Figure 4-41 Images showing labelled data by using multi-class model solution

4.10 Performance Metrics to quantitatively evaluate Models

The performance of an algorithm can be quantitatively evaluated by comparing the resultant processed or predicted image to the labelled ground truth image (GT), in terms of measurable metrics. All following metrics are based on four major measures; True Positive (TP), True Negative (TN), False Positive (FP), and False negative FN.

Where,

Black colour = background
White colour = detected flaw
TP = True Positive -> It's a flaw and it's predicted as a flaw; HIT
TN = True Negative -> It's a background and it's predicted as background;
FP = False Positive -> It's a background but it's predicted as flaw
FN = False Negative ->It's a flaw but it's predicted as background; MISS
A = GT image
B = Predicted image

Multiple metrics have been implemented, that will be seen in the tables of the next three chapters later on. In total there are **sixteen** metrics that have been used for in-depth analysis for all methods. For image segmentation, **five** additional metrics have been added such as CV, RI, VOI, BE and GCE that are able to measure more in-depth performance. For machine learning and deep learning, all the above have been applied plus a few metrics for visual display have been added. Confusion matrix (CM), Scatter plot and Area under the curve (AUC) for ML and Intersection of Union (IoU) for DL. Metrics that have been used in this thesis, are mentioned by stating their definition or formula in this section, to get an overview of all performance indicators implemented throughout. These will just be mentioned by their abbreviations in the following chapters especially in tables when discussing the results of the implementations.

4.10.1 Accuracy (Acc)

The pixel accuracy is a simple metric that describes the fraction of all pixels that were correctly predicted, from the total amount of predicted pixels. Higher the value of accuracy means that the system performed better. The pixel accuracy is then calculated as,

$$Accuracy = (TP + TN)/(TP + TN + FP + FN)$$

4.10.2 Sensitivity (TPR)

Also known by Sensitivity, TPR, Recall or Lift Numerator. Higher value of sensitivity reflects better system performance. It is calculated as,

$$Sensitivity = TP/\ (TP + FN)$$

4.10.3 Specificity (TNR)

It is also known as True Negative Rate or TNR. Higher the value of specificity result means the system performed better. It is calculated as,

$$Specificity = TN/(TN + FP)$$

4.10.4 Precision (PPV)

It is also known as Positive predictive value (ppv). When the value of Precision is high, it reflects that the system performance is also high. It is calculated as,

$$ppv = TP / (TP + FP)$$

4.10.5 Negative predictive value (NPV)

Higher value of NPV reflects better system performance. Negative predictive value is calculated as,

$$npv = TN/(FN + TN)$$

4.10.6 Fallout (FPR)

It is also known as False Positive Rate, Fall-out, (1-specificity), FPR. Lower value of fallout reflects better system performance. It is calculated as,

$$fpr = FP/(FP + TN)$$

4.10.7 False discovery rate (FDR)

When the value of FDR is lower, it reflects that the system performed better. It can be calculated as,

$$fdr = FP/(FP + TP)$$

4.10.8 Miss rate (FNR)

It is also known as FNR, False negative rate, Miss-rate, (1-sensitivity). Of course, the less the system misses, the better it's performance. It is calculated as,

$$fnr = FN/(FN + TP)$$

4.10.9 F-score (f1)

The F-score considers both the precision and the recall in order to calculate the score. If the score is 1 then it is the perfect score, so the higher the better. The F1 score defines the harmonic mean of precision and recall, defined as

$$f1 = 2 * TP / (2 * TP + FP + FN)$$

4.10.10 Root mean square error (RMSE)

This is a value which calculate errors in the system so obviously the lower this value is, the better system performance it depicts. It is calculated as,

$$rmse = sqrt(sum(A - B).\hat{}2)/length(A)$$

4.10.11 Matthews Correlation Coefficient (MCC)

Higher value of MCC reflects better system performance. Matthews Correlation Coefficient can be defined as,

$$mcc = ((TP * TN) - (FP * FN))/sqrt((TP + FP) * (TP + FN) * (TN + FP) * (TN + FN))$$

<u>For Image segmentation</u>

For segmentation specifically, other indicators are also useful. If the segmentation result is more similar to the GT, RI and CV will be higher but VoI, GCE, and BDE will be smaller.

4.10.12 Covering (CV)

It is an overlap measure that can be also used to evaluate the segmentation effect. The higher the value of CV the better the system has performed. It can be calculated as,

$$cv = TP/(TP + FP + FN)$$

4.10.13 Rand Index (RI)

It is also known as Probabilistic Rand Index (PRI). It is a similarity measure that counts the fraction of pairs of pixels whose labels are consistent between the computed segmentation and the corresponding GT segmentation. Hence, a higher value will reflect that it has more similarity with each other.

4.10.14 Global Consistency Error (GCE)

It computes the degree to which two segmentations are mutually consistent. It is used to compute an error value, hence a lower value will mean better system performance.

4.10.15 Variation of Information (VoI)

It is a similarity measure that is always used to measure the distance between two segmentations in terms of their average conditional entropy. Lower the value of VoI depicts less variation depicting better system performance.

4.10.16 Boundary Displacement Error (BDE)

It is an error measure that is used to measure the average displacement error of boundary pixels between two segmentations so when the value is low it shows the system performed better.

4.10.17 Elapsed time

It is the time taken by an algorithm to run the whole program. It will show more efficiency if takes less execution time.

For Machine learning

Evaluations that have been used are variations of the four major metrics indicators. These following metrics are just for display purposes to make the outcomes more clear for machine learning performance analysis.

4.10.18 Confusion Matrix (CM)

It is used to measure the type of errors produced by a classifier. The component of a confusion matrix is True Positive (TR), True Negative (TN), False Positive (FP) and False Negative (FN).

4.10.19 ROC curve

Another measure commonly used by machine learning experts is Area Under the ROC Curve. The ROC stands for Receiver Operating Characteristic (ROC). It measures the trade-off between true positive TR and false positive FR rates that means a plot between Precision and Recall values. Area Under the ROC Curve (AUC) is the total area under the curve out of 1. The higher the AUC, the more likely the classifier to predict more true positives.

For Deep Learning

The central metrics for deep learning are precision and recall, already calculated previously.

4.10.20 Intersection over Union (IoU)

The Intersection over Union (IU) score measures the accuracy of an object detector, by calculating the similarity between the region that was predicted by the model and the ground truth. Hence higher value shows better performance by the system. This metric has been implemented for visual display of individual test images outcome for deep learning only as shown in figure 4.42. It is defined as,

$$IoU \ = (Area\, of\ Overlap)/(Area\, of\ Union)$$

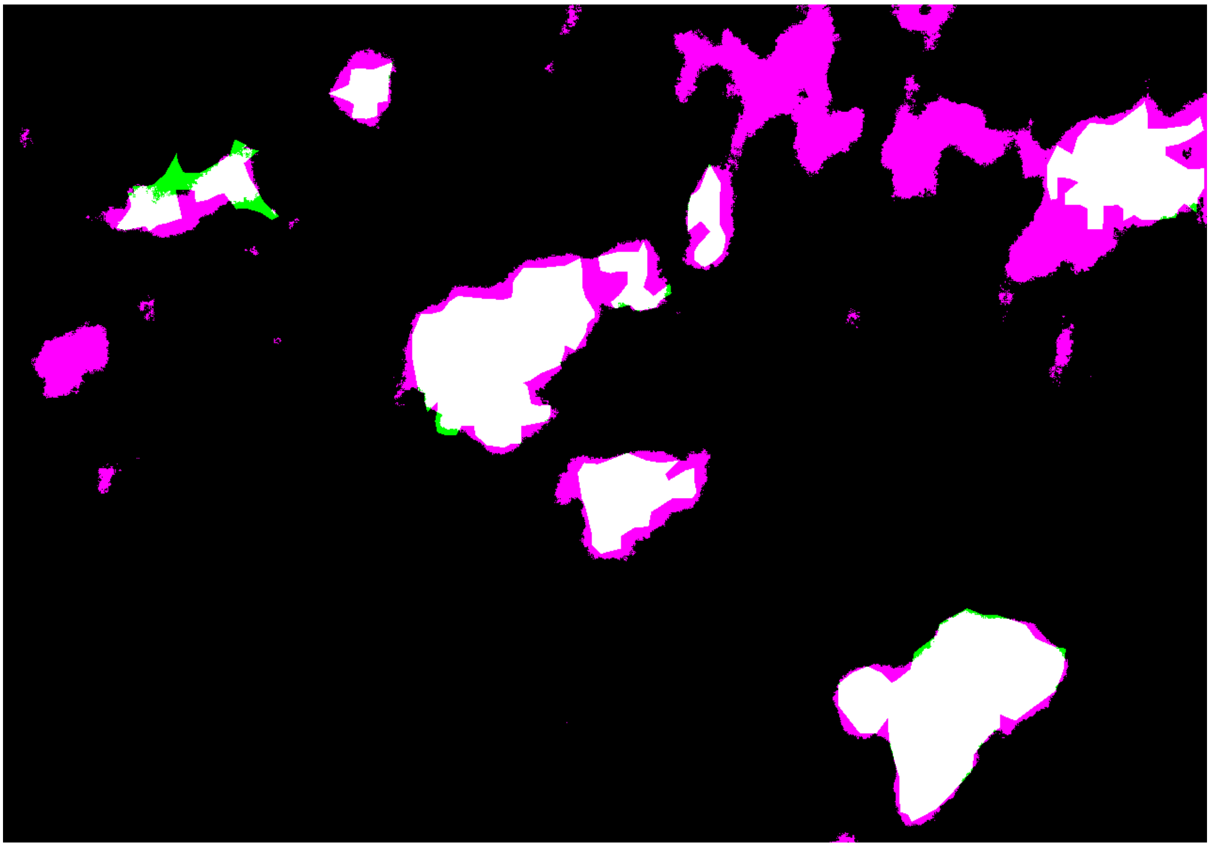


Figure 4-42 Example display of IoU performance indicator for deep learning evaluation

4.11 Chapter summary

This chapter discusses database that consists of original image captured and the labelled ground truth image. It mentions the data sources by explaining the setup of the experiment, observations and the findings. Some of the key findings from these experiments are summarised here. The degree of cleaning, influences the appearance as the contrast relies on oxide remaining. Flaws have bene seen at 0.2mm length with low magnification and 0.08mm at higher magnification, sometimes even smaller than these values. Cracks could start in machining lines which makes them difficult to distinguish. Some of the earliest sign of cracks were ignored due to this reason that they were just grinding marks. But after checking with Eddy current and confirmation of presence of cracks there, those machining marks were then considered as cracks. Using MPI enhances the contrast of the images especially for the purpose of crack detection. It was deduced from discussion with the inspection operators, that the detection unit of 0.2-0.3mm is considered as significant damage to be looked and investigated. Hence, this measurement will be used for labelling the ground truth images. It showed evidence that using the higher magnification gave more depth (clarity) to the outline of the detected flaw. Eddy current might be helpful in indicating the presence of a flaw with its location. But to find measurements of the individual cracks is difficult. There might be times when it is not possible to detect either pits or cracks due to reasons such as bad surface cleaning, ineffectiveness of MPI and/or poor microscope resolution. The Polimi data provided validation to the results produced by the image analysis system. So the image-processing method implemented provided a quick way of highlighting, sizing and counting specific features using on-site data. However, this is only possible with cleaned samples as corrosion and other residue can create false call. The method of cleaning should not create additional indications such as scratch marks or grinding marks which could be counted as cracks. The software also reduces the skill level requirement of an operator, as the algorithm sets a standard for the desired defects to be counted.

Then it lists the steps taken for creating a labelled database that includes setting flaw size criteria, calibrating and procedures used for labelling images. With the labelled ground truth, the performance of the system can now be quantitatively measured, as to how well the system has processed the image. Multiple metrics have been implemented, that will be seen in the tables of the next three chapters later on. In total there are sixteen metrics that have been used for in-depth analysis for all methods. For image segmentation, five additional metrics have been added such as CV, RI, VOI, BE and GCE that are able to measure more in-depth performance. For machine learning and deep learning, all the above have been applied plus a few metrics for visual display have been added. Confusion matrix (CM), Scatter plot and Area under the curve (AUC) for ML and Intersection of Union (IoU) for DL. The next chapter, discusses the algorithms that implement image processing with a step-by-step explanation of each of the task including visual aids.

Chapter 5
Unsupervised methods implementation, comparison & evaluation

This chapter discusses three main unsupervised image segmentation algorithms implemented for this research, such as watershed, Morphological and Gaussian based FCM clustering, after citing the initial trials and its results. For each of them, it shows individual steps along with visual outcomes and also presents the effects of variation of flaw-size parameter. Later on, flaw detection results are shown along with measurements and then evaluation of these methods is implemented on the basis of multiple metrics to quantify their performances. In the end, a comparison to one of the state-of the art segmentation methods is made vs the findings of this research.

5.1 INTRODUCTION

This chapter discusses unsupervised learning methods which is the second main contribution to the research. After discussing the research design, system setup, data and ground truth in the previous chapters, from now onwards the final phase of the research design starts, which is all about image analysis. Three methods of unsupervised learning have been implemented for this study which are based on watershed-based, Gaussian-based FCM clustering and morphological-based filtering, to investigate, make comparisons and evaluate on the basis of performance indicators. Then a state-of-the-art method has also been implemented on the same dataset to evaluate the performance of our research against a bench mark. This will enable to select the one with better performance for this specific industry problem dataset. This chapter consists of the following:

- Implementation of the initial trials based on some basic functionalities (section 5.2)
- Then for each of the three algorithms (watershed-based, morphological and gaussian) it explains main steps in the method along with visual aids (section 5.3-5.5)
- Next, compares all three method's flaw measurements using same sample image (section 5.6)
- In the end, a state-of-the-art (SOTA) method is implemented that is compared and evaluated with other implemented unsupervised methods, using the same dataset, to choose the best performer (section 5.7)

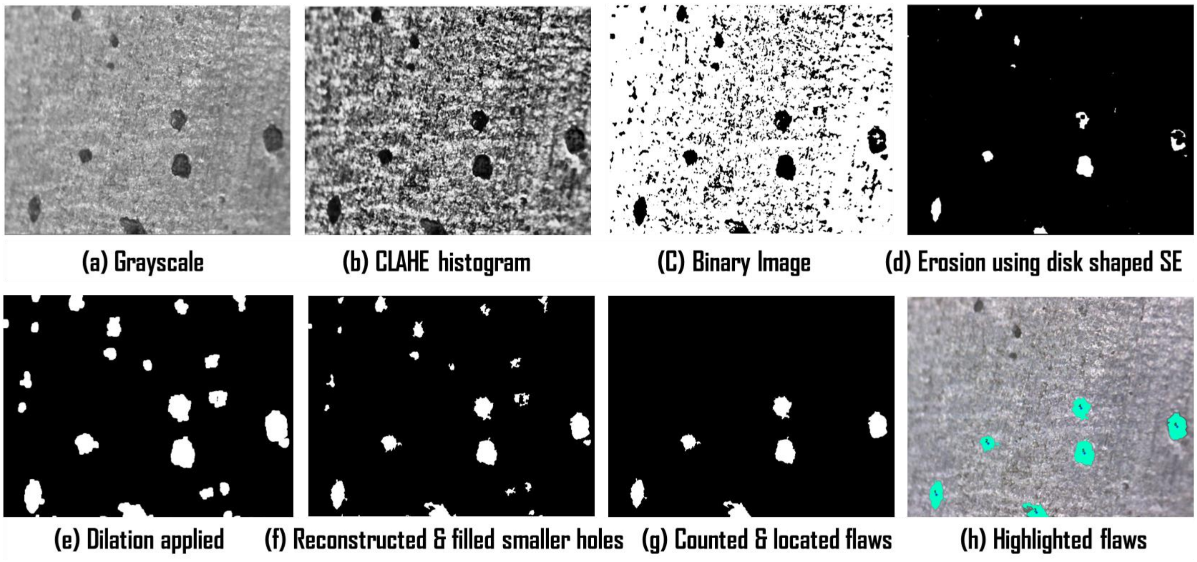

(a) Grayscale
(b) CLAHE histogram
(C) Binary Image
(d) Erosion using disk shaped SE
(e) Dilation applied
(f) Reconstructed & filled smaller holes
(g) Counted & located flaws
(h) Highlighted flaws

Figure 5-1 Example illustration of unsupervised learning steps

To easily compare between different algorithm's performances, two images have been selected and used throughout this chapter. This includes display outcomes of each step of each algorithm as shown in figure 5.1, plus measurements of each flaws in the image and the results table saved as an excel file after applying performance metrics. Less complicated images are chosen so that an easily detectable visual can be displayed. ImageA186-2016 is a bit of a complex image that consists of unusual shaped pits and also has surface scratch marks whereas ImageB919-2016 is a visually clearer image to easily decipher results. The results show the exact location of the defects of pits or cracks in the image, the exact number of flaws as well as they give the measurements of each flaw. It produces an excel file with all the measurements for each pit counted.

Implementation of all algorithms, except the initial trials, follow three major steps which include; pre-processing, segmentation and region extraction. The outcomes consist of the resultant segmented image along with two excel files. One has the flaw measurements such as flaw length and area etc and the other has the performance measures such as accuracy, recall and precision etc.

As researched in chapter 2, image segmentation can be used to distinguish objects in an image. There are two approaches to apply segmentation which are region-based and boundary-based. Its main task is to assign labels to every pixel in an image in a way that the pixels that share same labels have some commonality of characteristics. They are created around the discontinuity and similarity of image intensity values, where the image is partitioned based on abrupt changes in intensity value such as edges. The resultant segmented image is a set of segments that cover the entire image otherwise it can also produce a set of contours extracted from the image.

There are multiple factors involved in order to improve an image for segmentation tasks; first of all the task that needs to be achieved, suitability of the method to the task, the sequence of the methods applied as well as of course the image quality. Applying a combination of multiple methods improves the process but the trick is in the correct suitability of the sequence of the methods applied according to the target.

Edge detection has been commonly used in this perspective where the points at which image brightness changes sharply are typically organised into a set of curved line segments called edges. They can be detected using various edge detectors such as Sobel, Prewitt, Roberts, LoG (Laplacian of Gaussian) and Canny. Another popular method is the Watershed transform, which is recognised as a powerful morphological tool used in image segmentation because of its simplicity, speed and complete division of the image [134]. It is grouped under region-based segmentation approaches, where it provides closed contours even when the regions have weak boundaries and low contrast.

5.2 Initial Trials of image processing

Image processing is a very vast subject, and there are a lot of existing researched methods that can be used in laying the foundation. Implementing an algorithm that works in this particular case was the challenging task. So the first stage in the initial phase, was to research on the topic whereas the second stage was to apply them to check their performance with the available data at the initial stage.

This initial version has a combination of different image enhancements and edge detection techniques that are mentioned in the literature review but these have been implemented here as individual steps as shown in figure 5.2. Common edge detection techniques were compared such as Laplacian; Roberts; Sobel; Laplacian of Gaussian, zero-cross; Canny; Prewitt filters for the initial trials. These techniques try to find the edges of the objects of an image, for example by labelling them with white colour. Filling those white shapes will show where the actual objects are located, while blurred objects, which do not have sharp edges, will stay black or give a discontinued line of white dots.

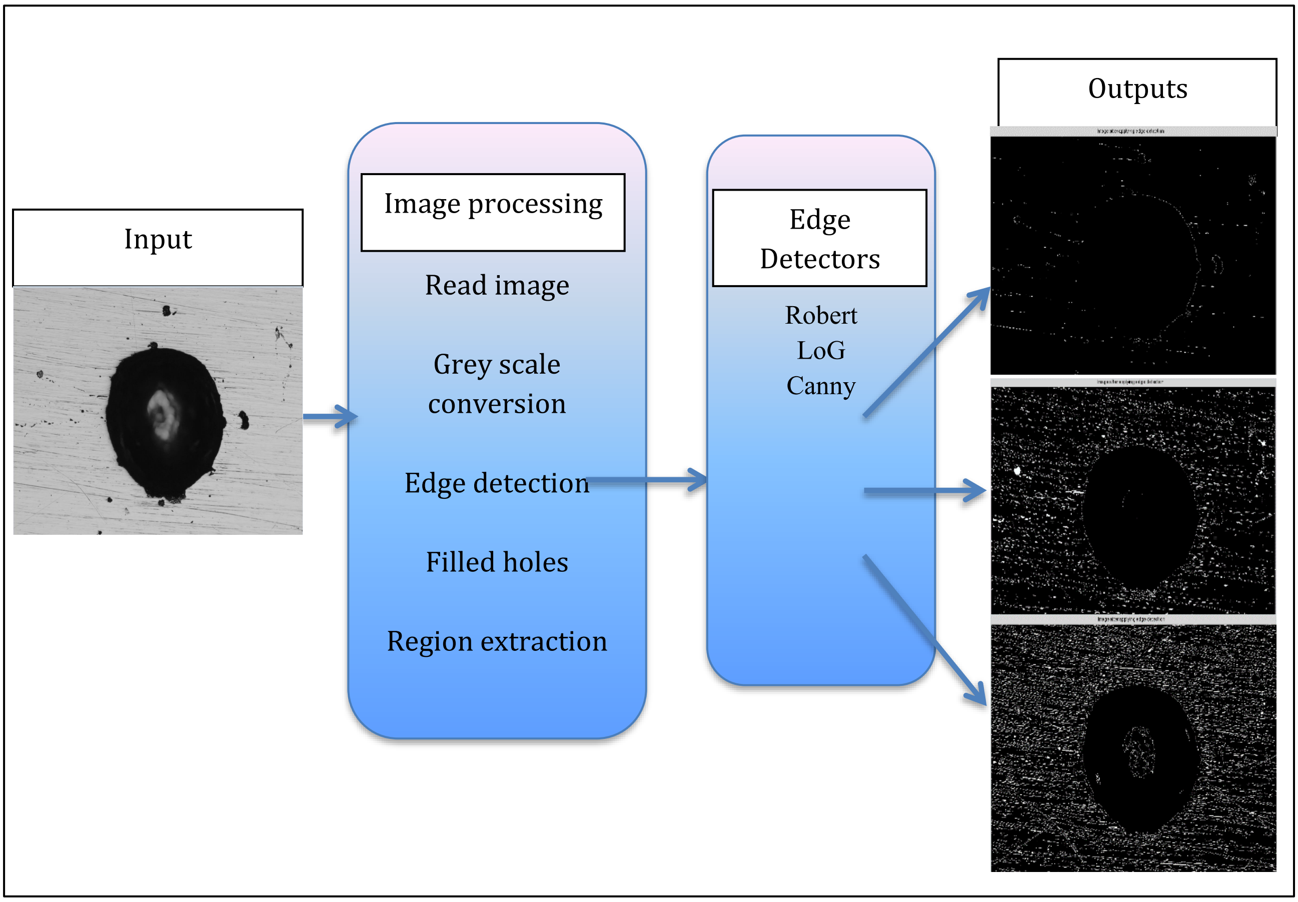


Figure 5-2 Block diagram of the initial algorithm of unsupervised algorithm

The main task was to perform segmentation that included steps such as; eliminate small objects from binary image by using a technique that works by removing all connected components that have fewer than ‘P’ pixels, fill the holes in the produced image and in the end apply feature extraction to get object’s centroid, lengths and other interesting features. Many varied images that consist of pipeline, a carbon steel block, real images as well as images from the standard, were used for testing purposes. Some images contain pits and some contain cracks. The main focus is on the pits, as not a lot of work has been done in this area in comparison to cracks.

5.2.1 API standard grade 4 pitting as sample

As an initial test sample, into an edge detector, an industry API-579 standard of grade 4 level pitting, as seen in figure 5.3, was used. The reason was to use a very basic image which is easy to verify by manual counting plus to get a rough idea of pit assessment. The detector used as an example is the Robert edge detector and the result is shown in figure 5.4.

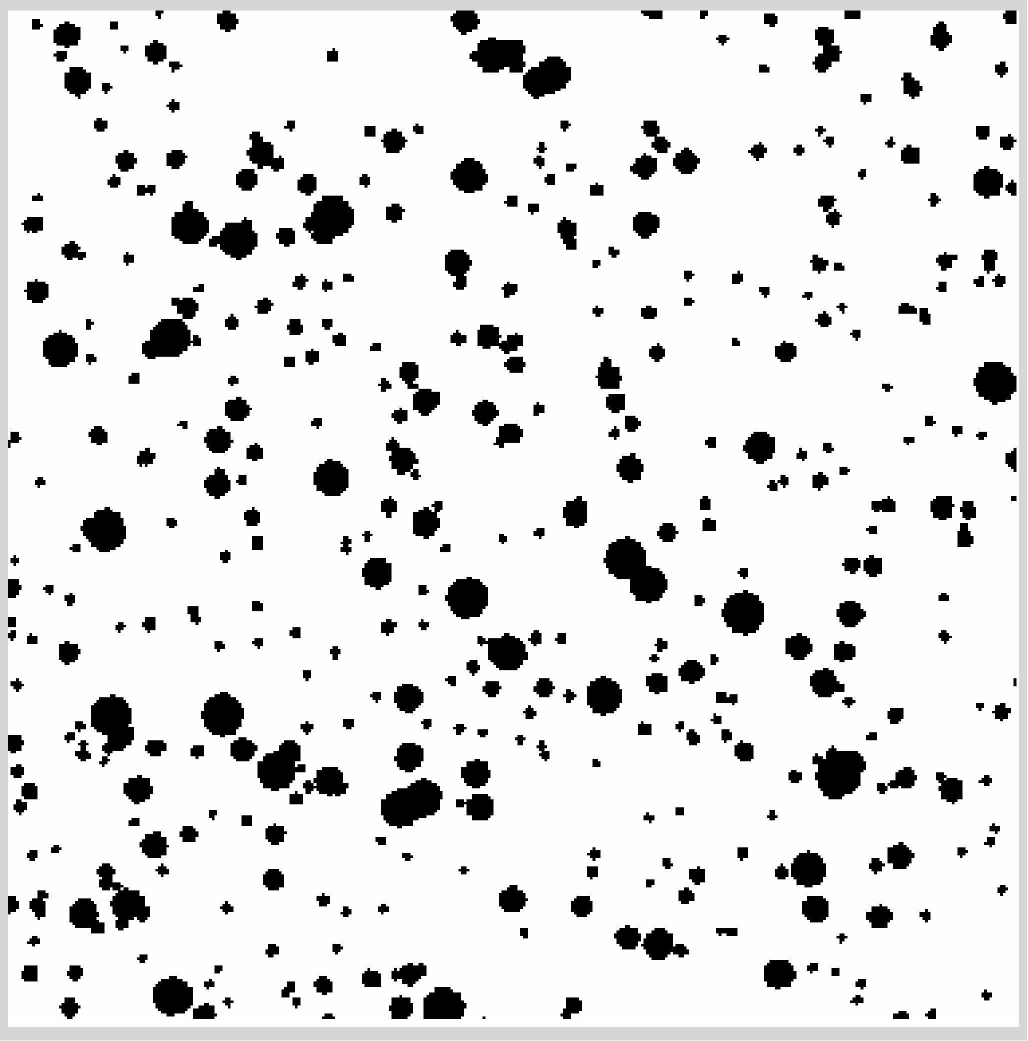


Figure 5-3 Sample image of API standard used for testing initial algorithm

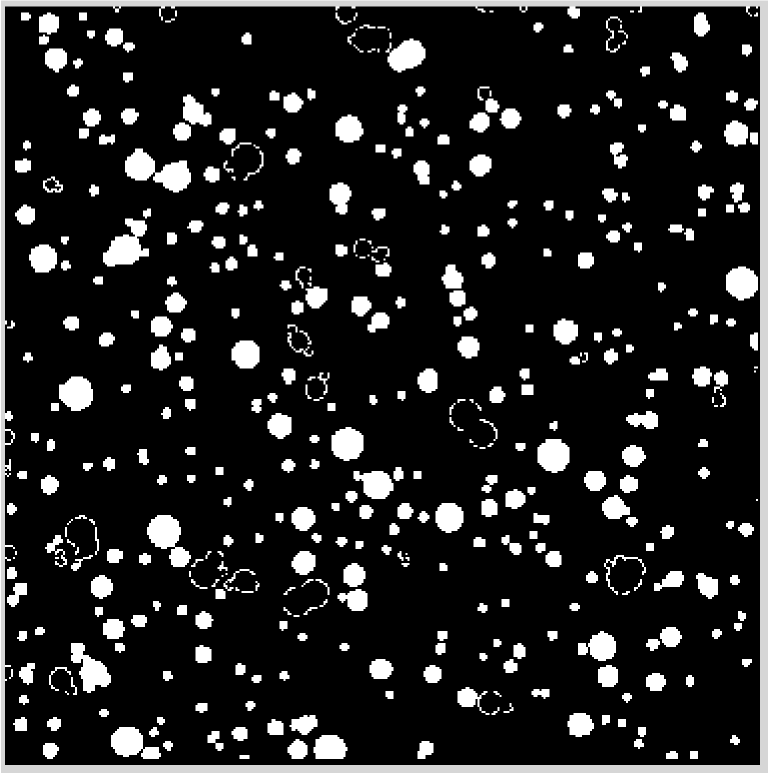


Figure 5-4 Result of API standard image after applying 'Robert' edge detector

5.2.2 Single pit and multiple pits as samples

It is interesting to see the results produced, for different edge detectors for the same image input. All of the above mentioned edge detectors were tested. It can be seen that each of the detectors can be served for different purposes. For example, some are more sensitive to the edges as compared to the others. But sometimes less sensitivity is required so that only the prominent edges in the image are picked.

For sample Image-08 that consists of a single pit shown in figure 5.5, figures from 5.6 to figure 5.8 show result outcomes of Robert, Log and Canny detectors and illustrates the fact that Canny is the one which has picked more edges of the pit which is in the centre of the image and hence it can be seen that the shape of the pit is more prominent in the resultant image.

For sample Image-09 that consists of multiple pits shown in figure 5.9, figures from 5.10 to figure 5.12 show result outcomes from Robert, Log and Canny detectors; once again highlighting that Canny has picked more edges but in this case it can be observed that if only subtle edge detection is required then either Robert or Log would be better options as Canny has a lot going on the resultant image which might not be suitable for some applications and requirements.

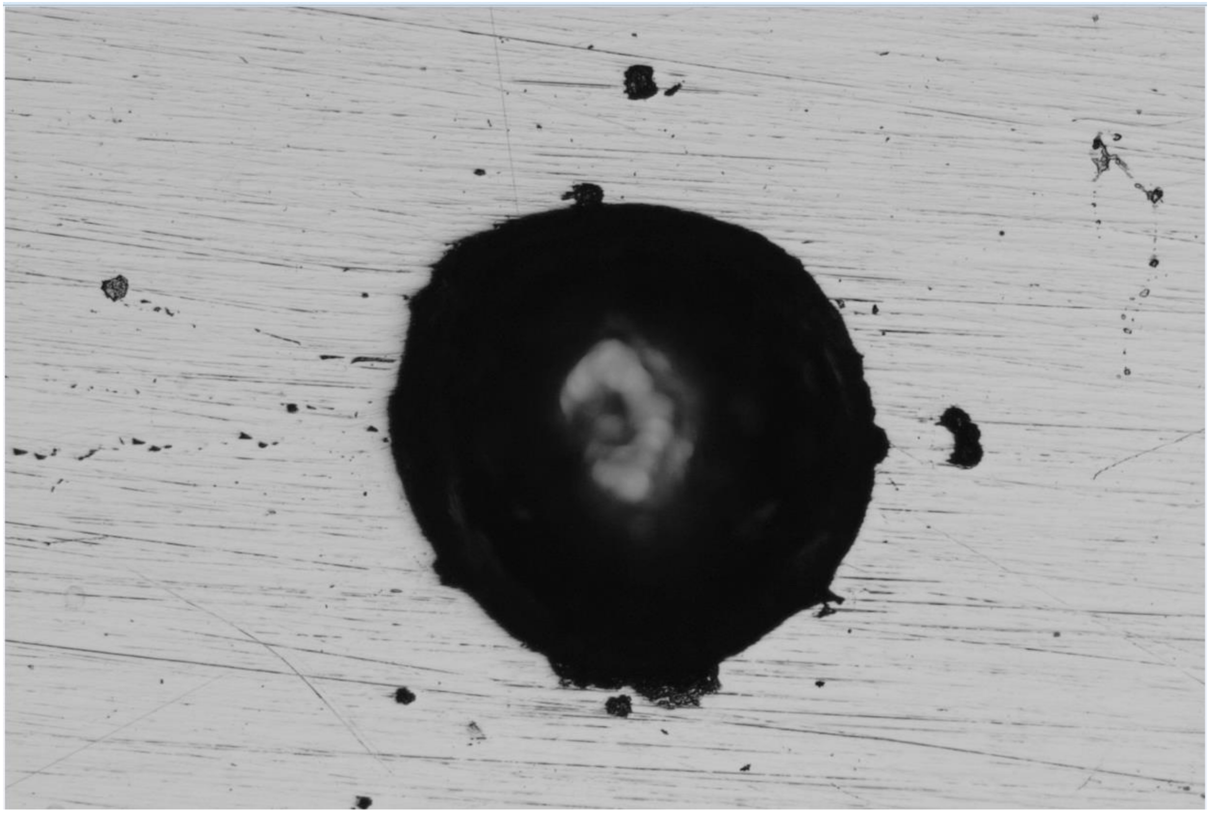


Figure 5-5 Sample test image used consisting of a single pit

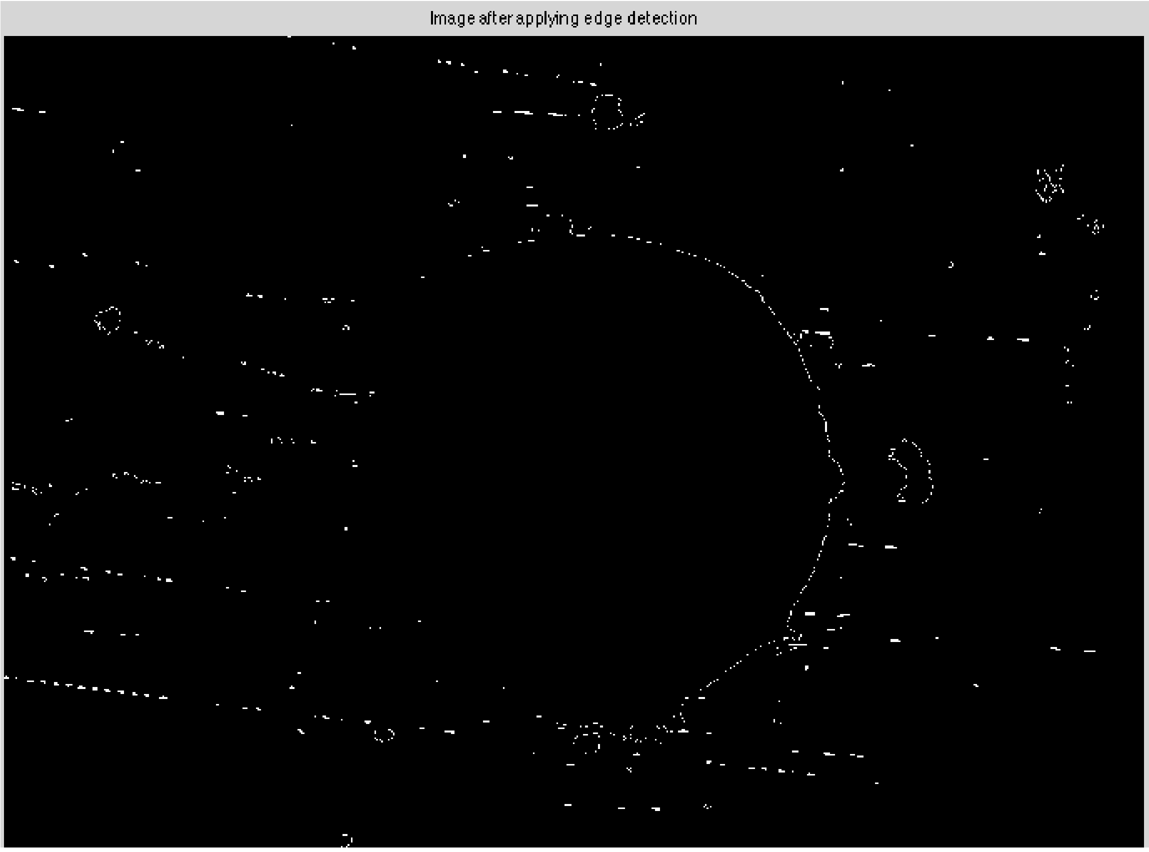


Figure 5-6 Result of single pit sample image after applying 'Robert' edge detector

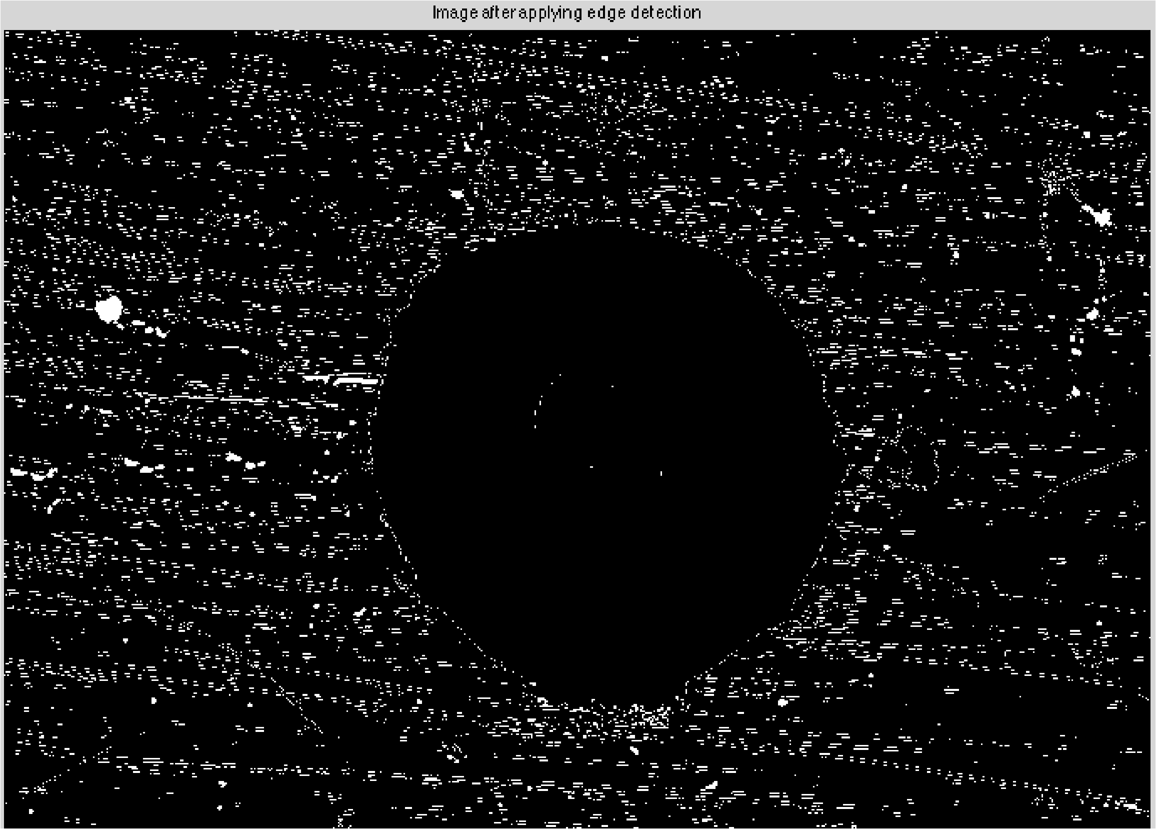


Figure 5-7 Result of single pit sample image after applying ‘Log’ edge detector

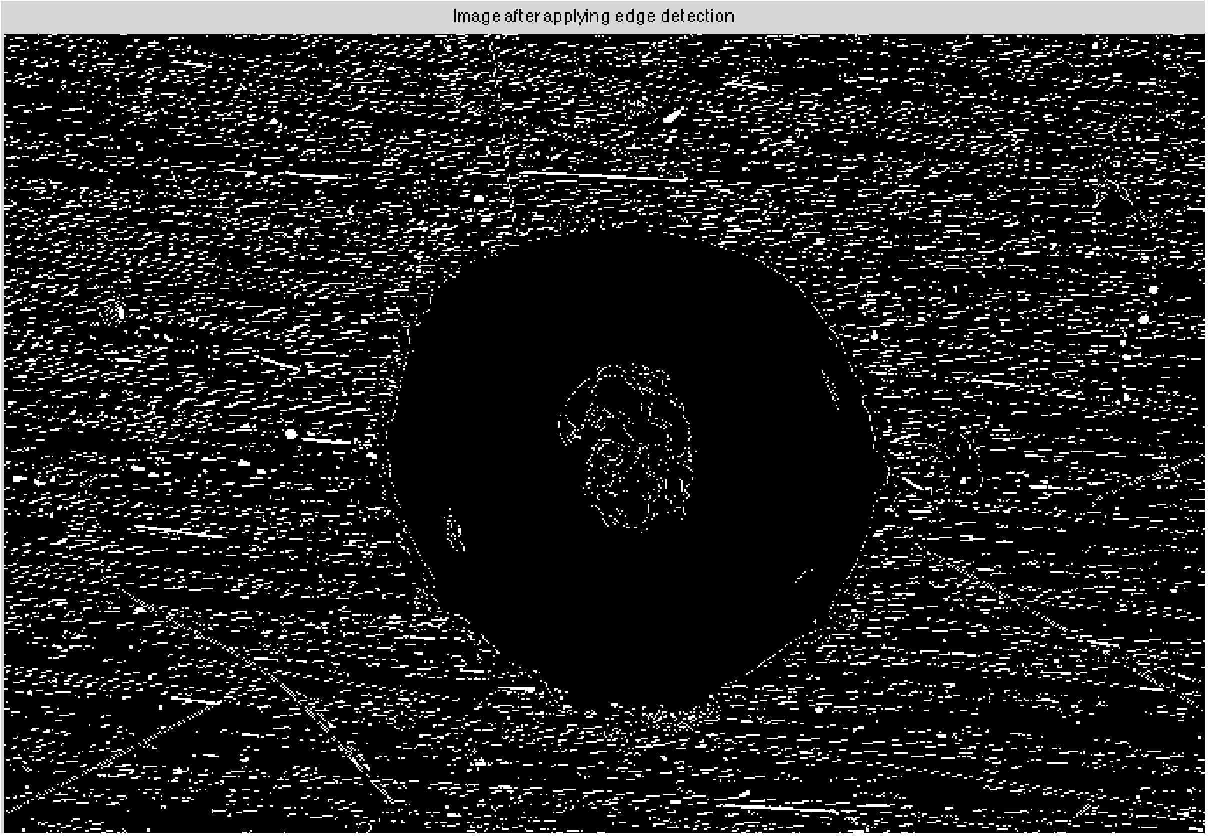


Figure 5-8 Result of single pit sample image after applying ‘Canny’ edge detector

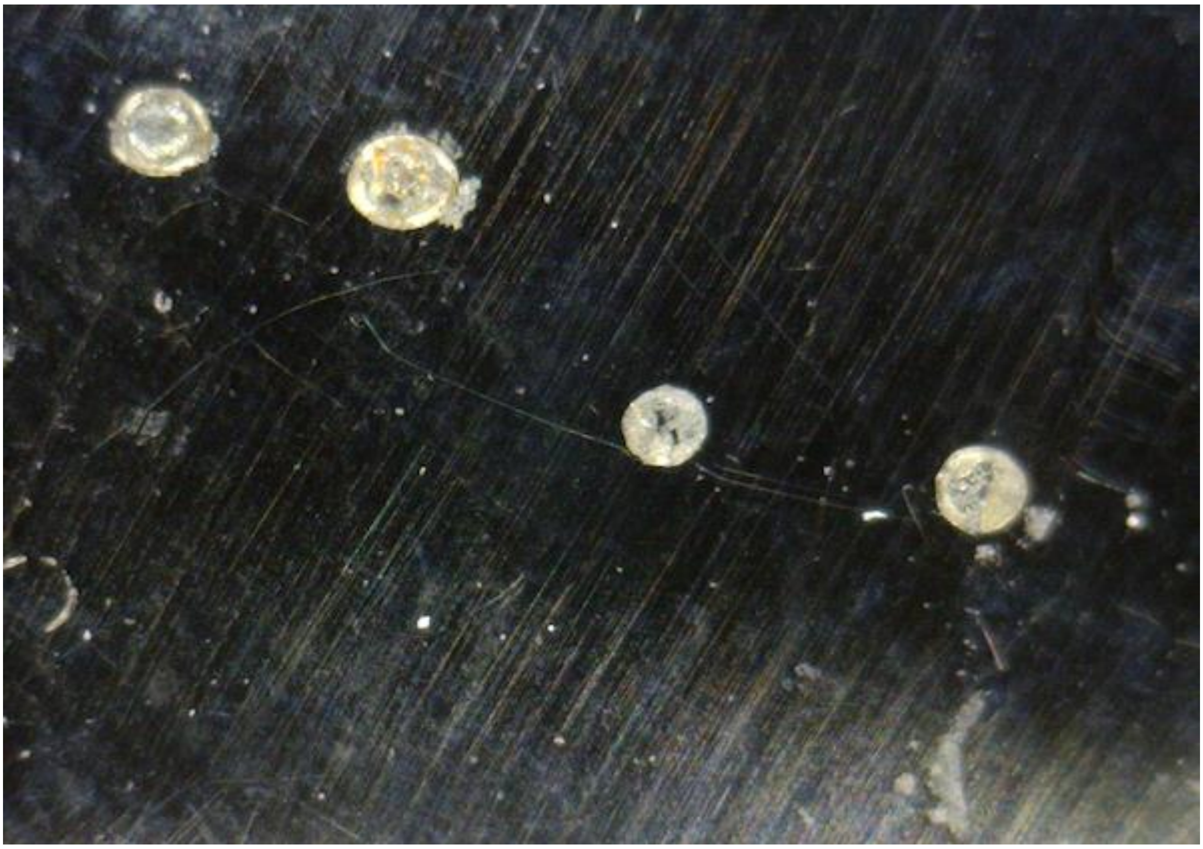


Figure 5-9 Sample test image used consisting of multiple pits

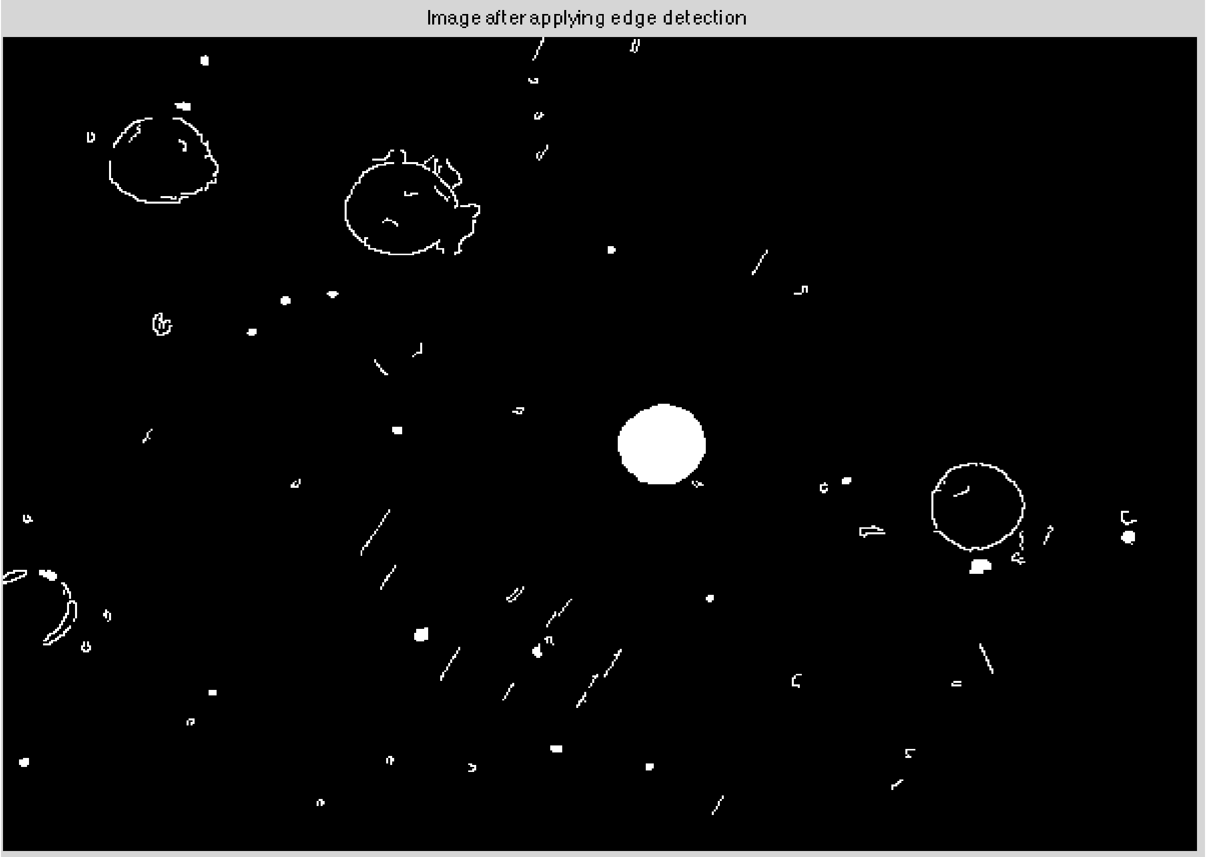


Figure 5-10 Result of multiple pit sample image after applying 'Robert' edge detector

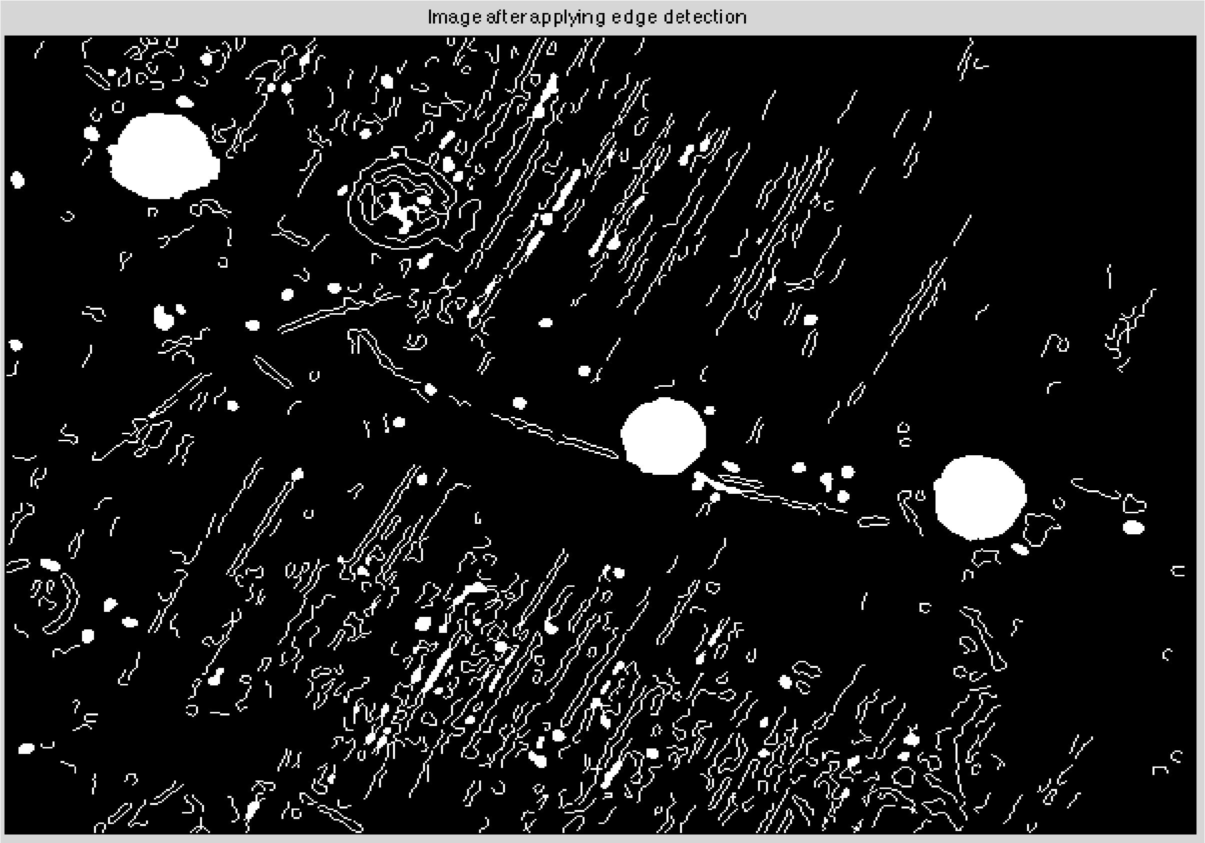


Figure 5-11 Result of multiple pit sample image after applying ‘Log’ edge detector

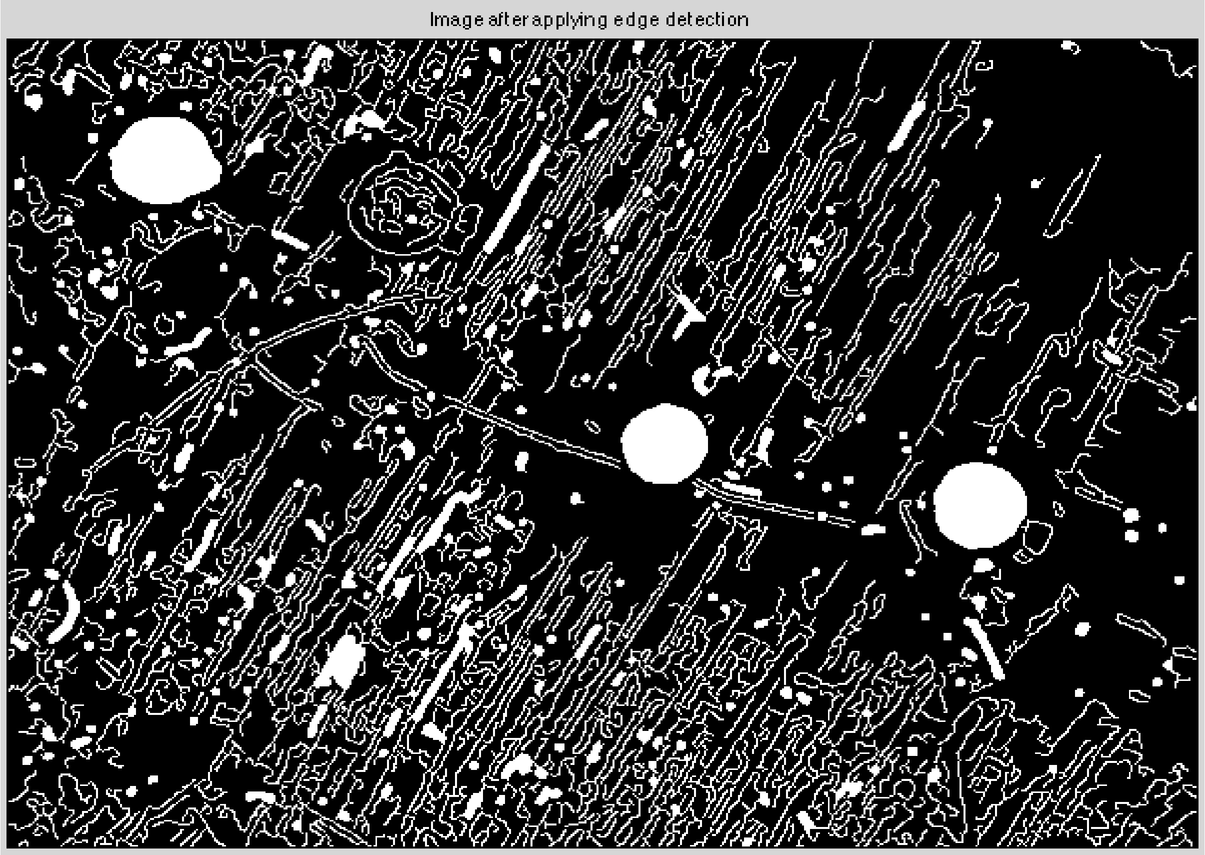


Figure 5-12 Result of multiple pit sample image after applying ‘Canny’ edge detector

5.2.3 Real image from a rail axle component as a sample

The sample in this test is a real image taken from a rail axle component shown in figure 5.13. The results for this image show that from Robert shown in figure 5.14, Sobel shown in figure 5.15, LoG shown in figure 5.16 and Canny shown in figure 5.17 detectors; once again Canny is the one that has picked the edges with more sensitivity. Although LoG detector also shows some promise as it is better than the Robert and Sobel. Hence, Canny is a better choice as it shows more sensitivity by clear indication of the pits, which is required by the application. After edge detection, thresholding was applied along with filling the holes, to get the acquired pits.

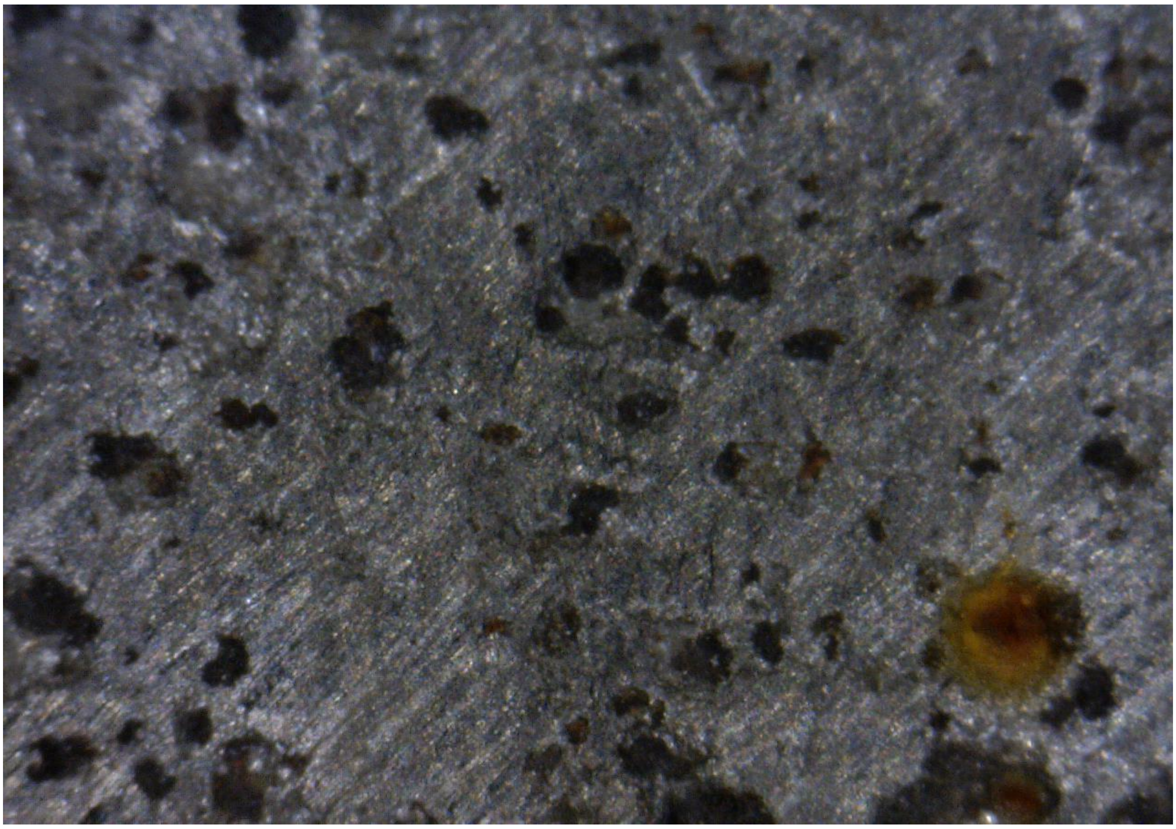


Figure 5-13 Real image from a rail axle component used as a test sample

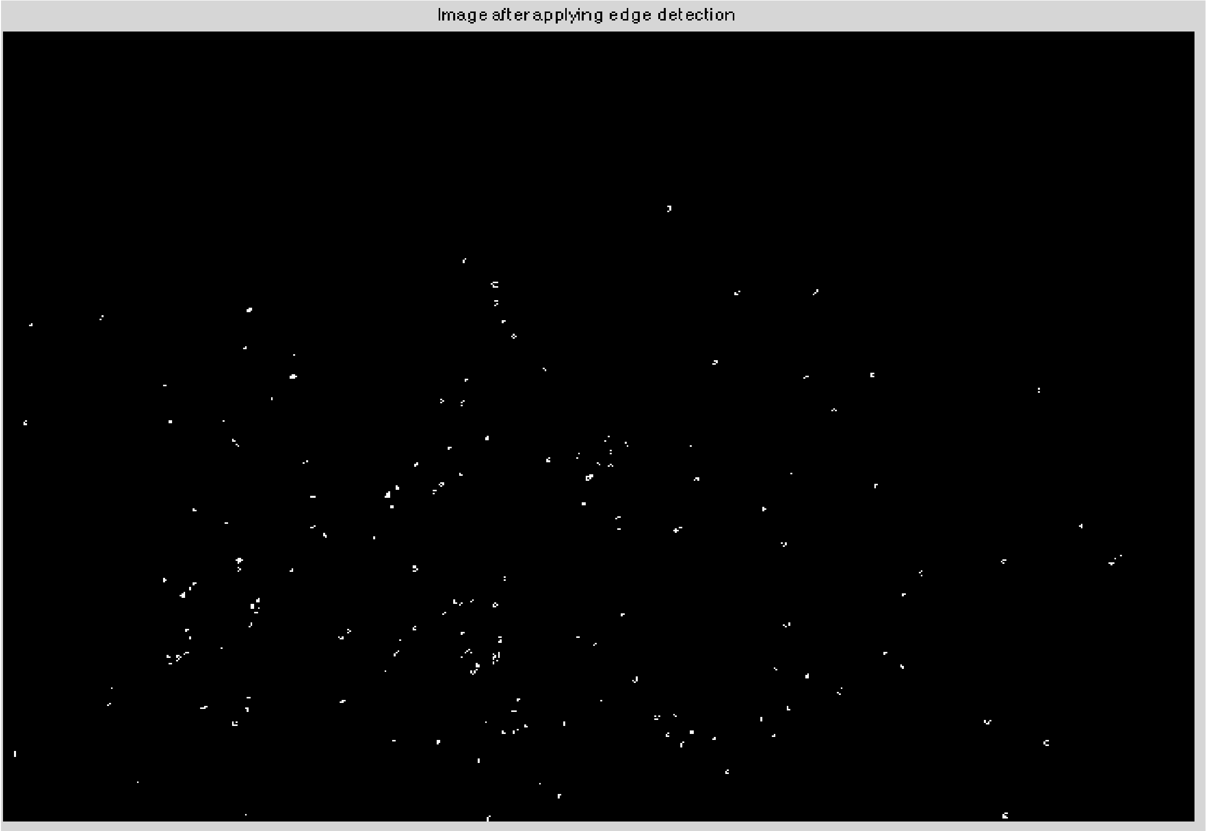


Figure 5-14 Result of real original axle image after applying 'Roberts' edge detector

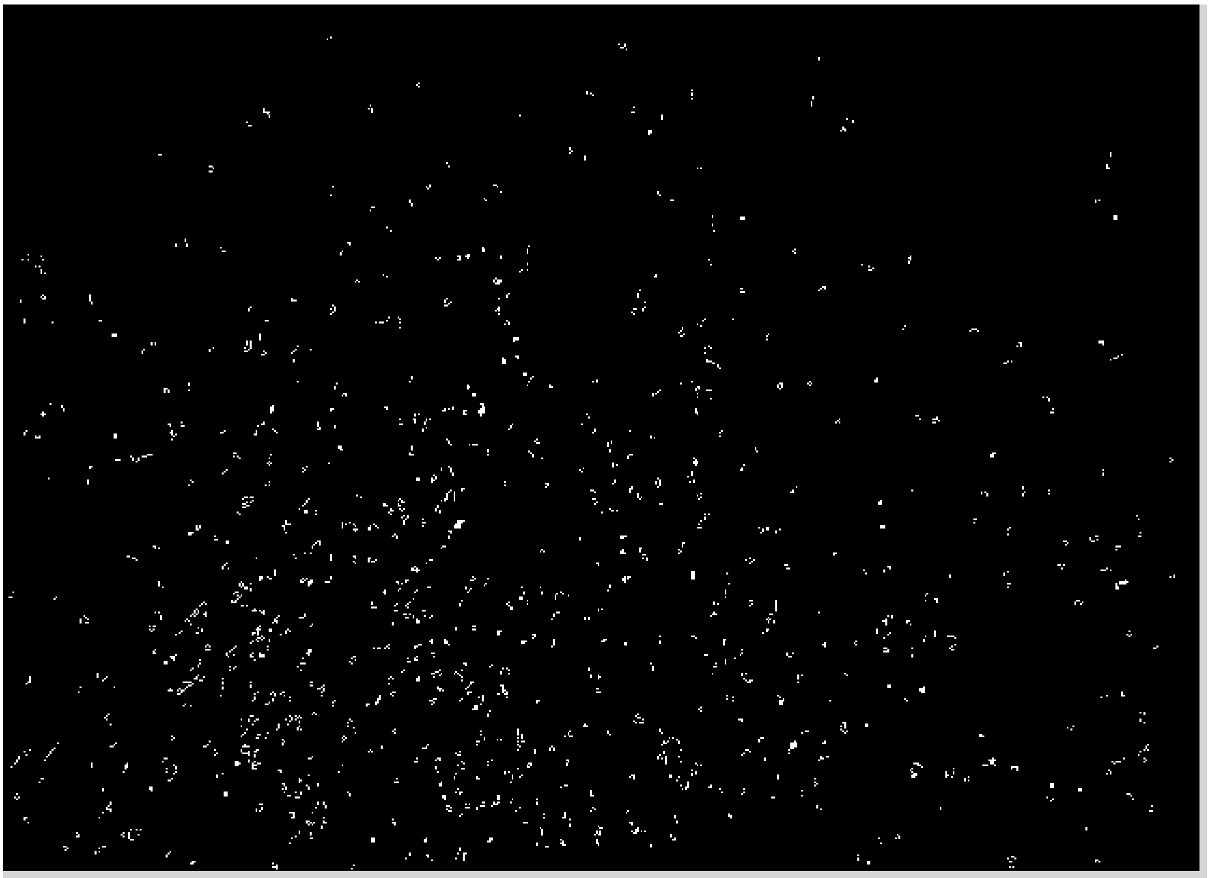


Figure 5-15 Result of real original axle image after applying 'Sobel' edge detector

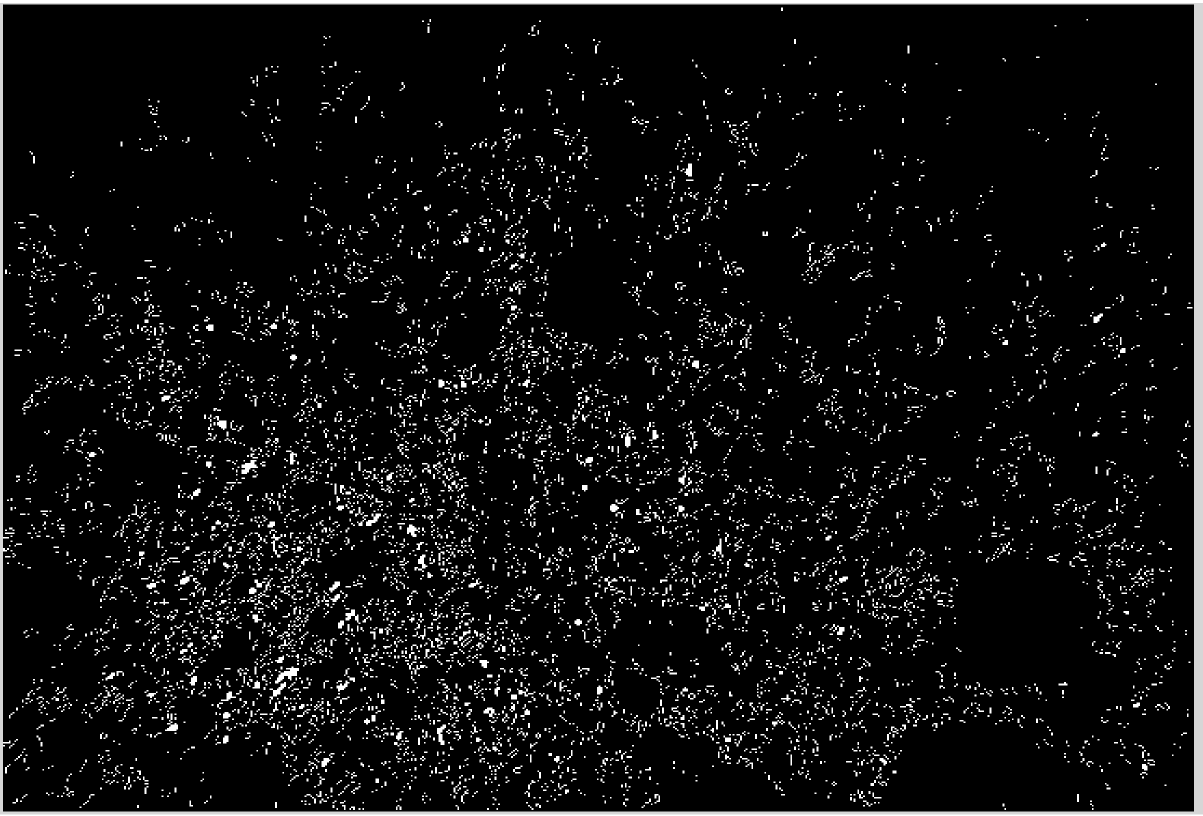


Figure 5-16 Result of real original axle image after applying ‘Log’ edge detector

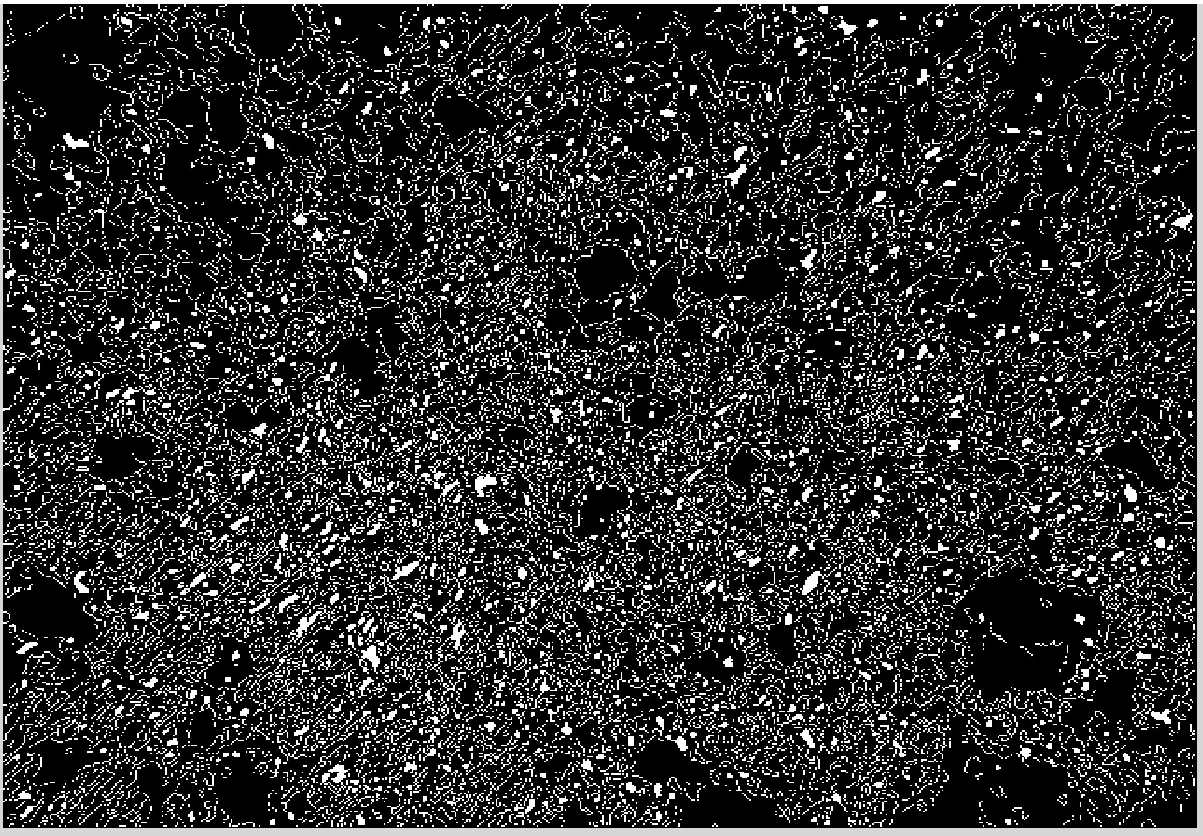


Figure 5-17 Result of real original axle image after applying ‘Canny’ edge detector

5.2.4 Initial algorithm drafted

Grade1 pitting Image is the image that has been verified by the existing method of counting the pits. It gives the number of pits which is 47 in this case, after applying processing. The picture in figure 5.18 and figure 5.19, is covering an area =150*150mm. It can be observed and measured by the figure that the pits are of the same lengths and areas. It is an image from the standard to check and test my algorithms for simplicity.

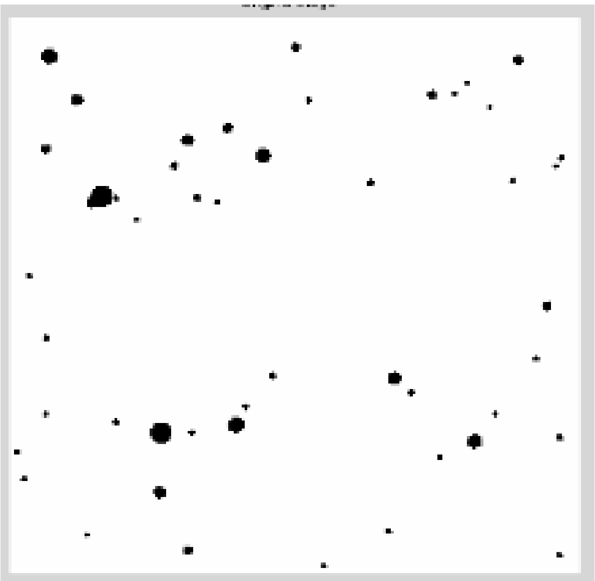


Figure 5-18 API standard image used for testing

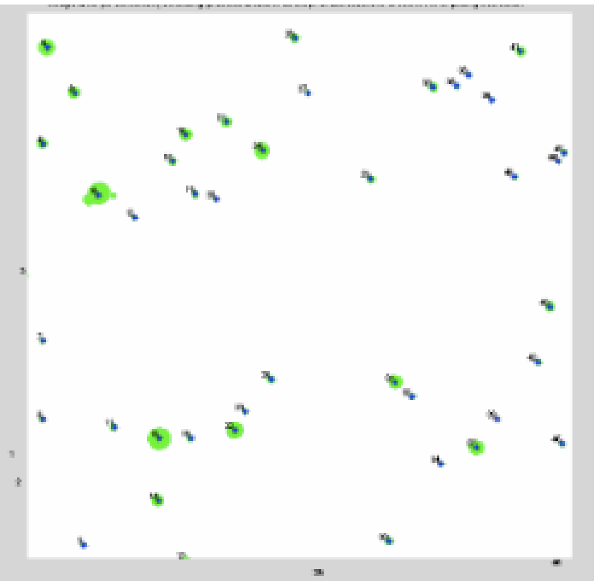


Figure 5-19 Result of API image after applying initial algorithm

The proposed method of image processing has been performed on many varied images that consist of real images as well as images from the standard for testing purposes. Some images contain pits and some contain cracks and pits. The main focus is on the pits for the initial testing. These images have been taken in TWI from components like rail axle, pipeline, and carbon steel block as discussed.

5.2.5 Conclusion

For these trials performed, following points and conclusions were drafted out:

1. Around 20 varied images were taken as samples for the algorithm. Some were partially successful and gave results; some gave incorrect results which could be seen by looking at the image and/or manually counting them. The summarised results can be seen in a tabular form in table 5.1 for some of the images tested.
2. From the above point it is clear that data needs to be massively increased for the next stages of the research, in order to add more depth to the database.

Table 5-1 Outcomes of the initial algorithm along with the manual counting numbers.

S.N	Image Name	Type	Initial algo	Manual Counted
1	Image1-A042	Bmp	260	75-80
2	Image2-grade1	Bmp	47	47
3	Image3-grade2	Bmp	125	121
4	Image4-grade3	Bmp	185	184 (+1 to 4)
5	Image5-grade4	Bmp	380	370
6	Image6-mc1	png	420	83
7	Image7-A614	Bmp	265	30-35
8	Image8-M1	Png	used	
9	Image9- msample	png	used	

3. Three steps were used to know whether the result given by the algorithm is correct: algorithm runs successfully without any errors; produces a result; and is verified by using the existing method which is manually counting the flaws.
4. From the above point it can be seen that the algorithm is still in its development phase and needs a lot of improvisation.
5. The verification has been done as shown in Table 5.1 by the current existing method of counting each of the pits or cracks in the image, using a field operator.
6. Considering the above statement, it was realised at this point that these verified numbers are just a rough estimate as it differed from one operator to the other. To solve this problem maybe a size threshold can be set for selection criteria, which links to the next point.
7. Discussions with field experts needed to be organised regarding the topic of defect size, which ones to count in and which ones to ignore, that is how big or small the defect size should be, for it to be included for selection.

5.3 WATERSHED-BASED ALGORITHM IMPLEMENTATION FOR IMAGE SEGMENTATION

For image segmentation it is an effective tool that deals with the object's boundary and finds local changes. It is based on watershed transform which targets to search for a border between two objects called catchment basins. It can be understood by the following concept that if water falls into these basins, level of the water rises until the neighbour basins share the same level. Hence, the outcome of the algorithm is a hierarchy of catchment basins [135].

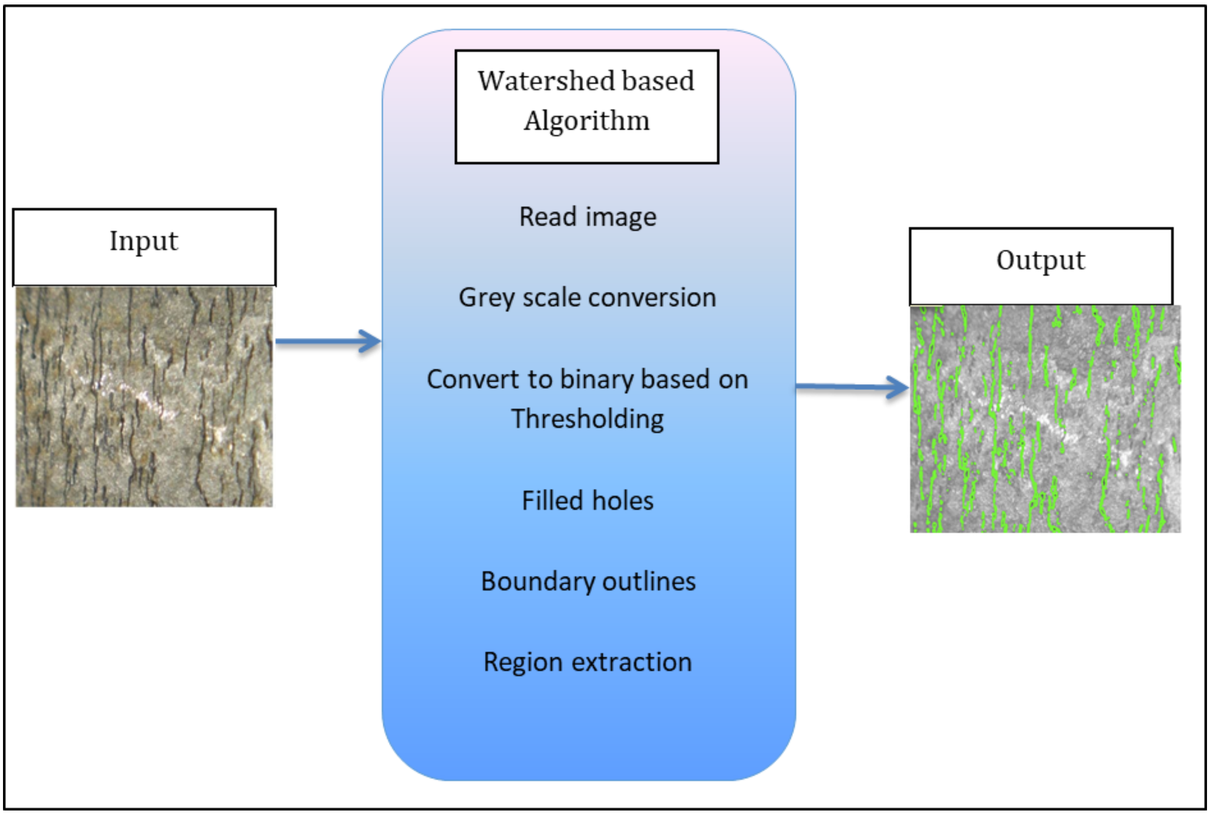


Figure 5-20 Block diagram of Watershed Algorithm

The algorithm basically works around the watershed concept and later on selects features based on length and intensity of the detected object. It starts by reading an RGB image as an input data. By the end, counts and make measurements of regions of interest (ROI) and then classifies pixels into defects/background by displaying the detected flaws in a different colour to the background for visual ease. This can be seen in a block diagram in figure 5.20. The main flow of the algorithm is discussed in this section, which includes some of the following steps:

1. Read the RGB image;
2. Convert it to grey scale;
3. Get its histogram;
4. Set a threshold value;
5. Threshold the image to get a binary image of class logical by choosing either ‘<’ or ‘>’ sign to select bright objects or dark objects;
6. Fill in the holes to get rid of small background pixels within the binary image;
7. Label each object to make measurements of it;
8. Apply pseudo random color labels to distinguish between them easily;
9. Extract features of the detected objects;
10. Then save the interesting measurement results like Area and length of each of the objects in an excel file for all the images.

11. Display the processed image, as a visual outcome of the algorithm.

5.3.1 Conversion from RGB to Grey scale values

After reading in a test sample, Image-06 shown in figure 5.21, the next step is the conversion from colour to grey scale shown in figure 5.22 by using rgb2gray. It converts RGB values to grey scale values by forming a weighted sum of the R, G, and B components, 0.2989 * R + 0.5870 * G + 0.1140 * B. It converts by eliminating the hue and saturation information while keeping the luminance.

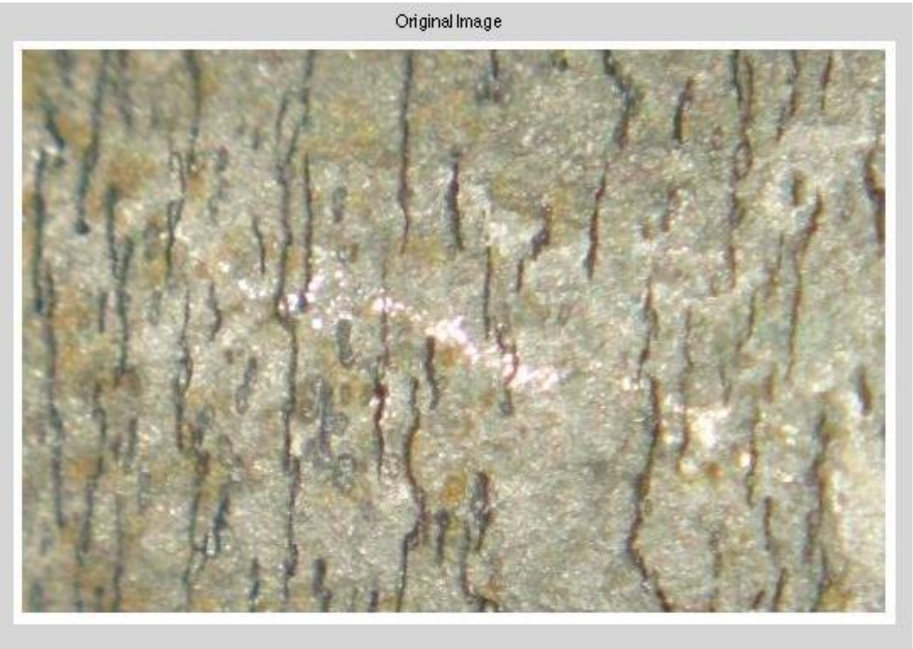


Figure 5-21 Test sample used for showing Watershed algorithm implementation

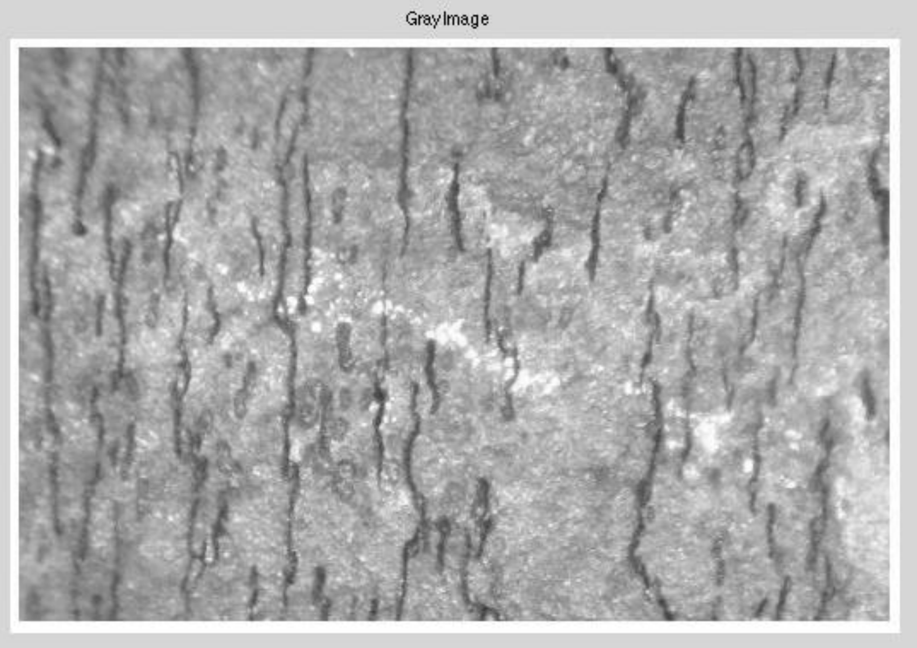


Figure 5-22 Watershed algorithm outcome after converting to grey scale

5.3.2 Normalised thresholding applied

Applied normalisation as thresholding as shown in figure 5.24 below. The range is equal to the maximum range for the data type, so for 8-bit images the maximum range is 0-255. Note that normalization of RGB images is not supported, so the image is converted into grey scale first, as done in the previous step. This task is basically like enhancing the image contrast by stretching the histogram as shown in figure 5.23. Main step is to find the minimum and maximum values in the image and then equalize the image by swapping the minimum value to '0' and the maximum value to '255' to get a normalized threshold value.

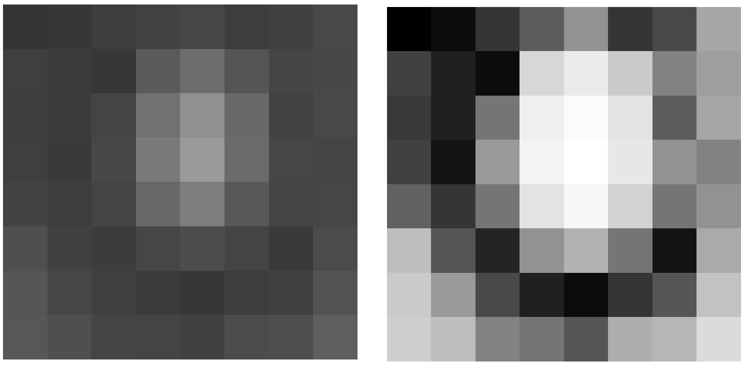


Figure 5-23 Showing the effect of applying normalised thresholding

5.3.3 Thresholding based on normalized threshold value

Then apply threshold to this value with the original image, to get a binary image as shown in figure 5.24. If the pixel value of the original image is less than the normalized threshold value, change it to value '1', this is done in order to choose darker objects from the image.

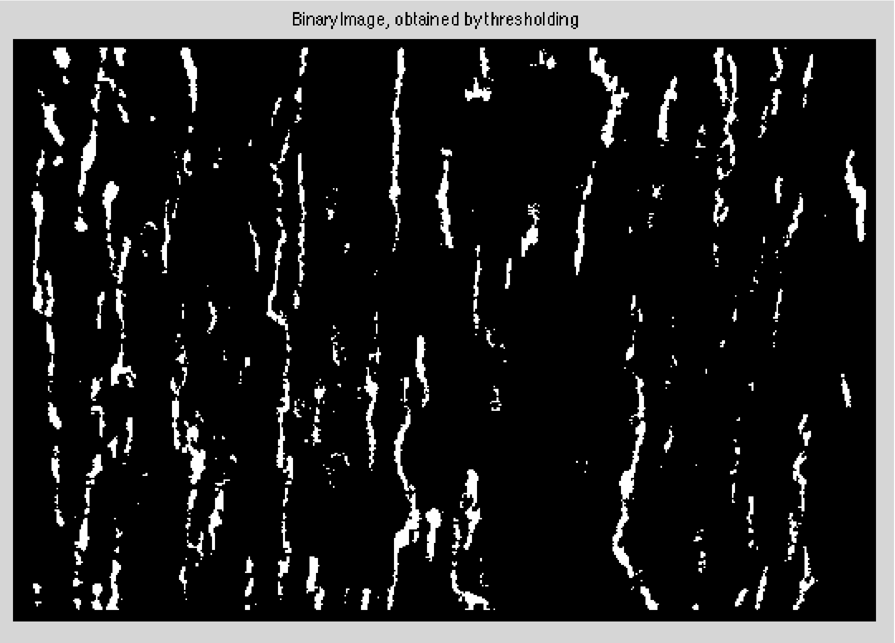


Figure 5-24 Watershed algorithm outcome after applying thresholding

5.3.4 Fill the small holes

This step is to fill regions and holes in the binary input image. In this syntax, a hole is defined as a set of background pixels that cannot be reached by filling in the background from the edge of the image or hole is a ‘dark’ region surrounded by ‘bright’ regions. In simpler terms, ‘0’s surrounded by ‘1’s is a hole.

5.3.5 Trace the boundaries

This step is used to trace the exterior boundaries of objects, as well as boundaries of holes inside these objects, in the binary image where ‘1’s pixel values belong to an object and ‘0’ valued pixels present the background. The output of the boundaries outlines applied is shown in figure 5.25 below. It descends into the outermost objects, also called parents, and traces their children that is the objects completely enclosed by the parents.

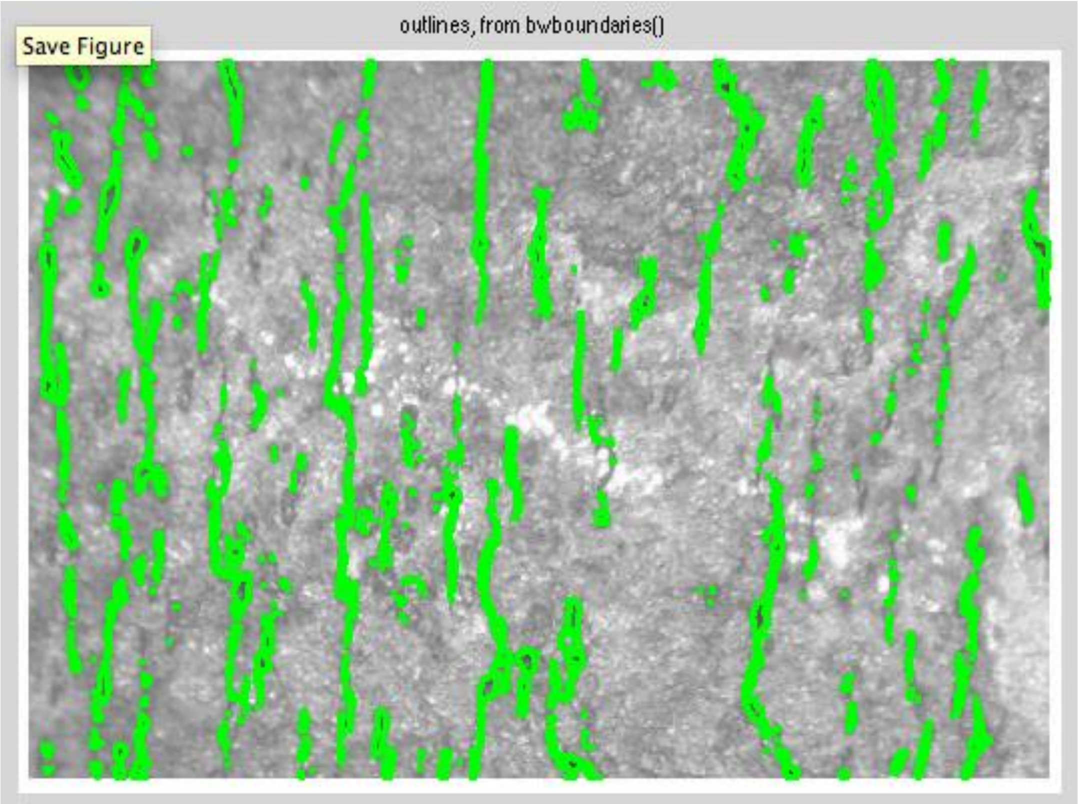
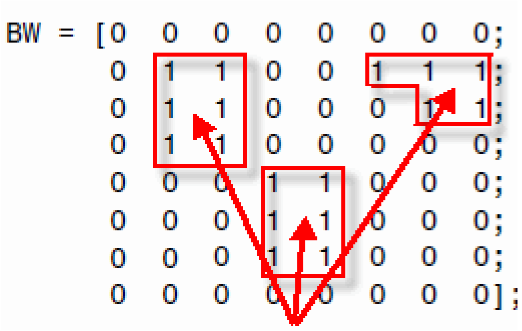


Figure 5-25 Watershed algorithm outcome after tracing the object boundaries

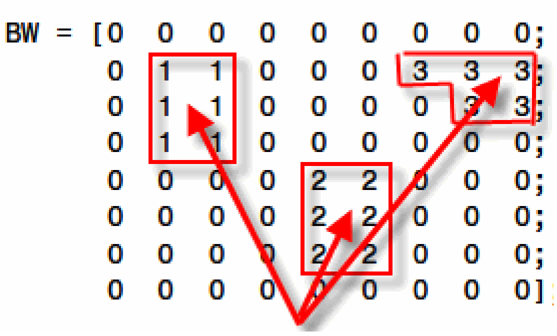
5.3.6 Label detected objects in the binary image

To label the binary image to be able to count and measure the detected objects, connected components labelling procedure [136] is used. It goes through the image and based on pixel connectivity of neighbour size, groups its pixels into components. A connected component is a set of pixels that form a connected group [137]. This means that all pixels in a connected component share similar pixel values and are in some way connected with each other. So in a binary image, this will mean the clusters of ‘1’s among the background values of ‘0’s. Once all groups have been determined, each group is labelled with a unique number. For example, the binary image in figure 5.26 &5.27 has three connected components.



```
BW = [0  0  0  0  0  0  0  0;
      0  1  1  0  0  1  1  1;
      0  1  1  0  0  0  1  1;
      0  1  1  0  0  0  0  0;
      0  0  0  1  1  0  0  0;
      0  0  0  1  1  0  0  0;
      0  0  0  1  1  0  0  0;
      0  0  0  0  0  0  0  0];
```

Figure 5-26 Showing Connected components in a binary image [138]



```
BW = [0  0  0  0  0  0  0  0  0;
      0  1  1  0  0  0  3  3  3;
      0  1  1  0  0  0  0  3  3;
      0  1  1  0  0  0  0  0  0;
      0  0  0  0  2  2  0  0  0;
      0  0  0  0  2  2  0  0  0;
      0  0  0  0  2  2  0  0  0;
      0  0  0  0  0  0  0  0  0]
```

Figure 5-27 Showing Labelled connected component in an image [138]

Extracting and labelling of various disjoint and connected components in an image is central to many automated image analysis applications. In the resultant image, the pixels labelled as ‘0’ are the background and the pixels labelled as ‘1’ make up one object; the pixels labelled as value ‘2’ make up the second object; and so on. For better visual impact, they can be later changed to colour labelled components, where each label identifies each object in the label matrix by mapping it to a pseudo random colour (or follows a colour map). This quantifies the number of detected defects in the images.

5.3.7 Extracting properties from the detected objects

It is one of the central steps that improve the overall performance of the automated inspection system by extracting meaningful features from an image. Thus enabling the use of many pattern recognition and classification techniques to be applied to the resulting output [139]. Hence, this proved to be a crucial step which measures the detected object properties, for the black and white labelled image provided from the previous step.

One of these particular values is a ‘centriod’ which is the centre of the mass. This is represented by (x,y) locations of where the middle of each detected object is located. Centroid is just one of the properties. There are many other interesting features like area, perimeter, eccentricity, average intensity of the object, major (length) and minor (width) axis lengths of the objects, that can be used for quantitative measurements. These results are saved in an excel file and is used for further specific data extraction.

5.3.8 Gathering specified information from the extracted properties

Once the objects have been detected and their properties have been extracted along with their measurements, they can be used to further improve the results. Considering the size criteria of the objects detected, as discussed in chapter 4, the minimum cut-off value is set to the values 0.1mm for high magnification and 0.3mm for low magnification. Now the major axis length feature comes in handy, as it is used to remove all objects smaller than these specified values, which acts as a post processing step.

These values were established to be the cut-off size in the late phases of the research. But for the initial phases, different combinations of area, major axis length and average intensity were applied to explore the result outcomes. This can be seen by showing a few set of visual examples (from figure 5.27 to 5.38) and tables (from table 5.2 to 5.5) to elaborate the point.

Test Sample B514-2016

This image contains many small pits that can be ignored and a few pits of the interested size. After applying region extraction in the previous step, the number of detected flaws counted are, flaws= 441 as shown in figures 5.28. This figure, a) shows the test sample Image B514-2016, b) shows the resultant Watershed Binary image, c) displays traced object boundaries on the watershed applied image, d) shows the outcome of Colour-labelled connected components on the watershed image. Then figure 5.29 shows histogram of the gray scale version of the image.

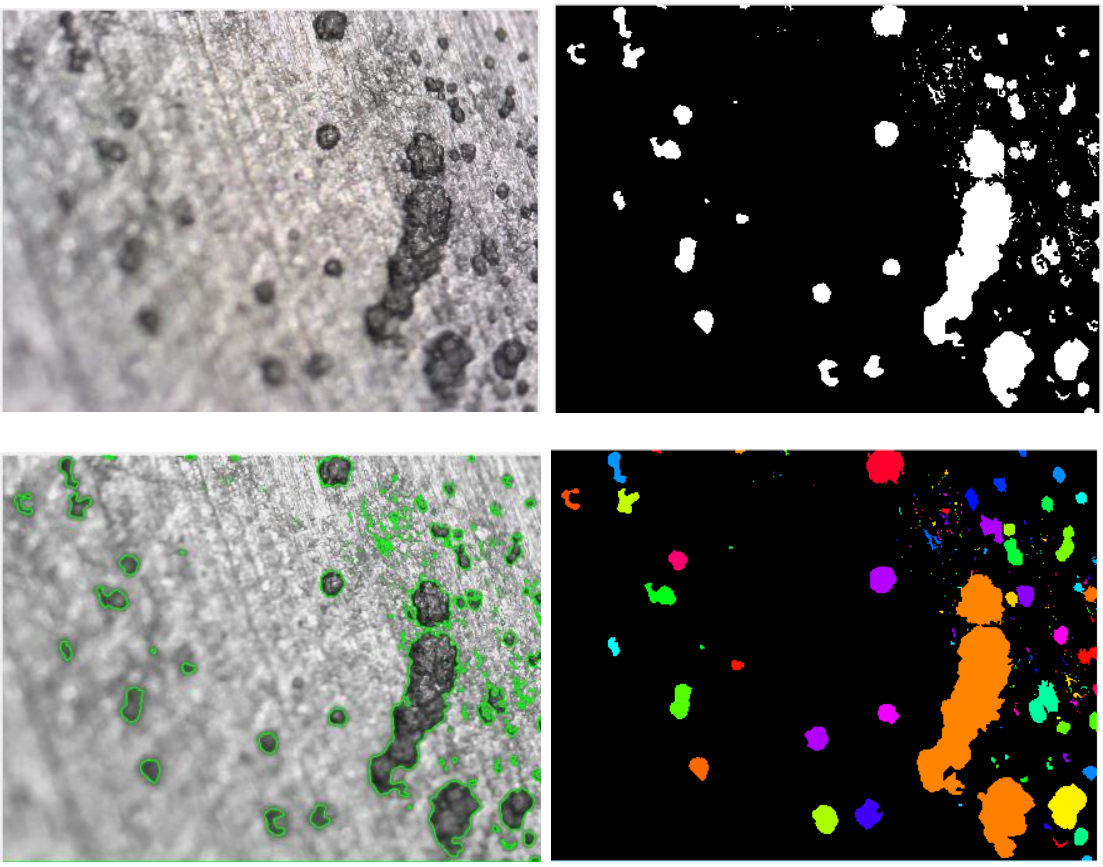


Figure 5-28 Watershed outcome on B514-16 applying labelled connected components

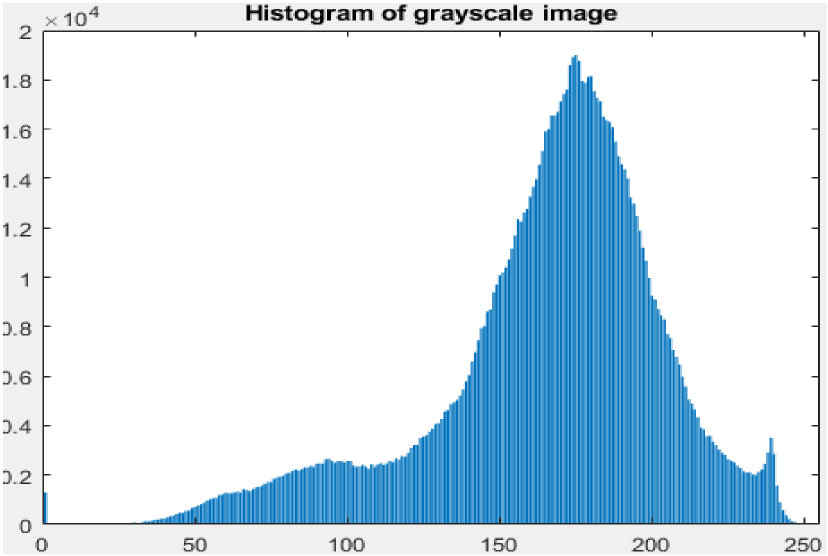
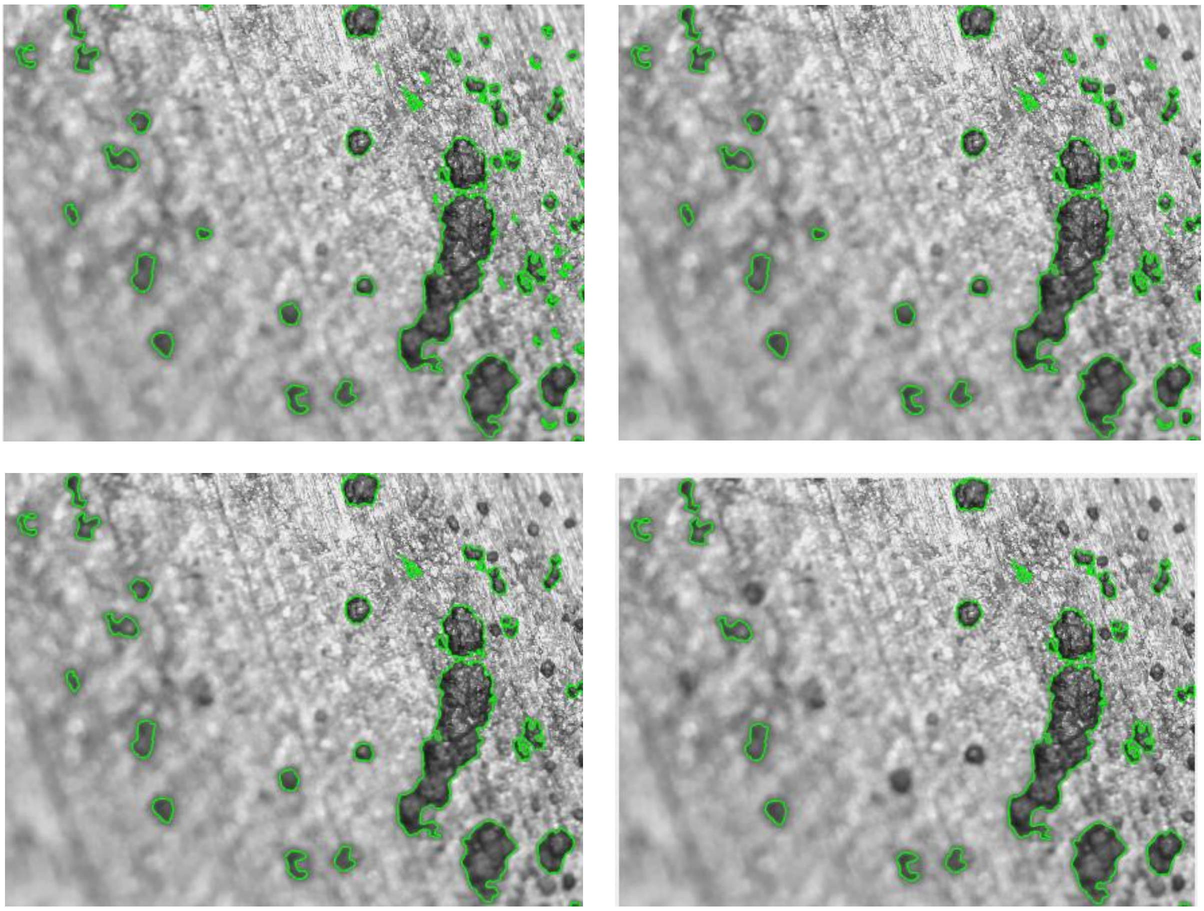


Figure 5-29 Watershed outcome on B514-16 showing histogram of the grey scale image

After the initial stage of segmentation, different lengths of the object detected were tested, which in return produces different number of flaws picked. Table 5.2, shows different outcomes of the number of flaws based on the object length restrictions. The figure 5.30, shows visual watershed results with same test sample, Image B514-2016 with different cut-off lengths producing and detecting different number of pits.

Table 5-2 Watershed on B514-16 showing different outcomes based on cut-off length

Image #	Flaw Length (pixels)	Number of Flaws
B514-L1	>= 20	57
	>= 30	39
	>= 40	29
	>= 50	22
	>= 128	2
	>= 320	1



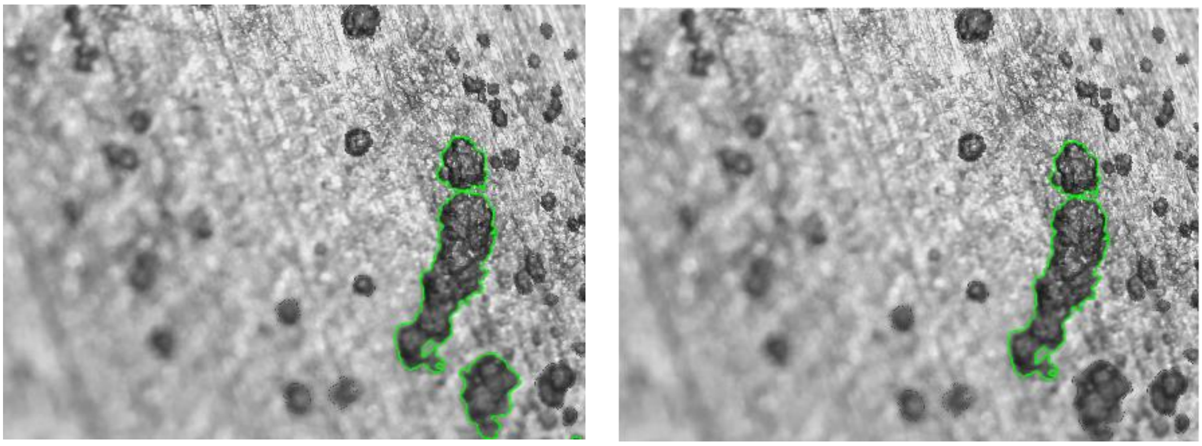


Figure 5-30 Watershed results on B514-16 showing varied outcomes based on length

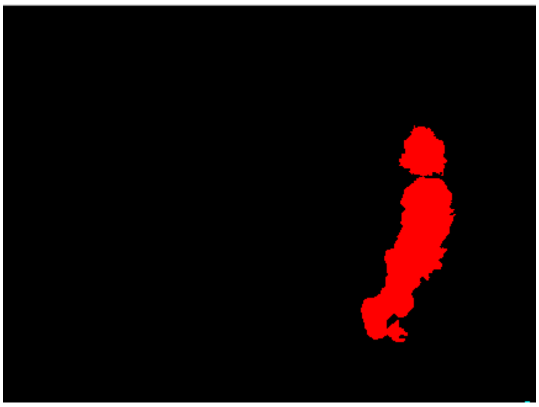


Figure 5-31 Watershed coloured-labelled outcome with length cut-off set to 0.5mm

From previous figure, the watershed coloured-labelled outcome can be seen in figure 5.31 with length greater than or equal to 320 pixels, which equals to 0.5mm at 200 magnification, showing that only a single pit was detected on this cut-off criteria.

Test Sample B005-L2

It is an interesting image as it contains a crack, possibly one or, if it appears disjoint in 1-2 places in the middle, then possible two or three cracks but it has scratches that need to be ignored. After applying this algorithm, the resultant figures are shown from figure 5.31 to 5.33. Figure 5.32 shows, a) the test sample Image B005-L2, b) the resultant Watershed Binary image, c) display of traced object boundaries on the watershed applied image, d) the outcome of Colour-labelled connected components on the watershed image. Then figure 5.33 shows histogram of the gray scale version of the image.

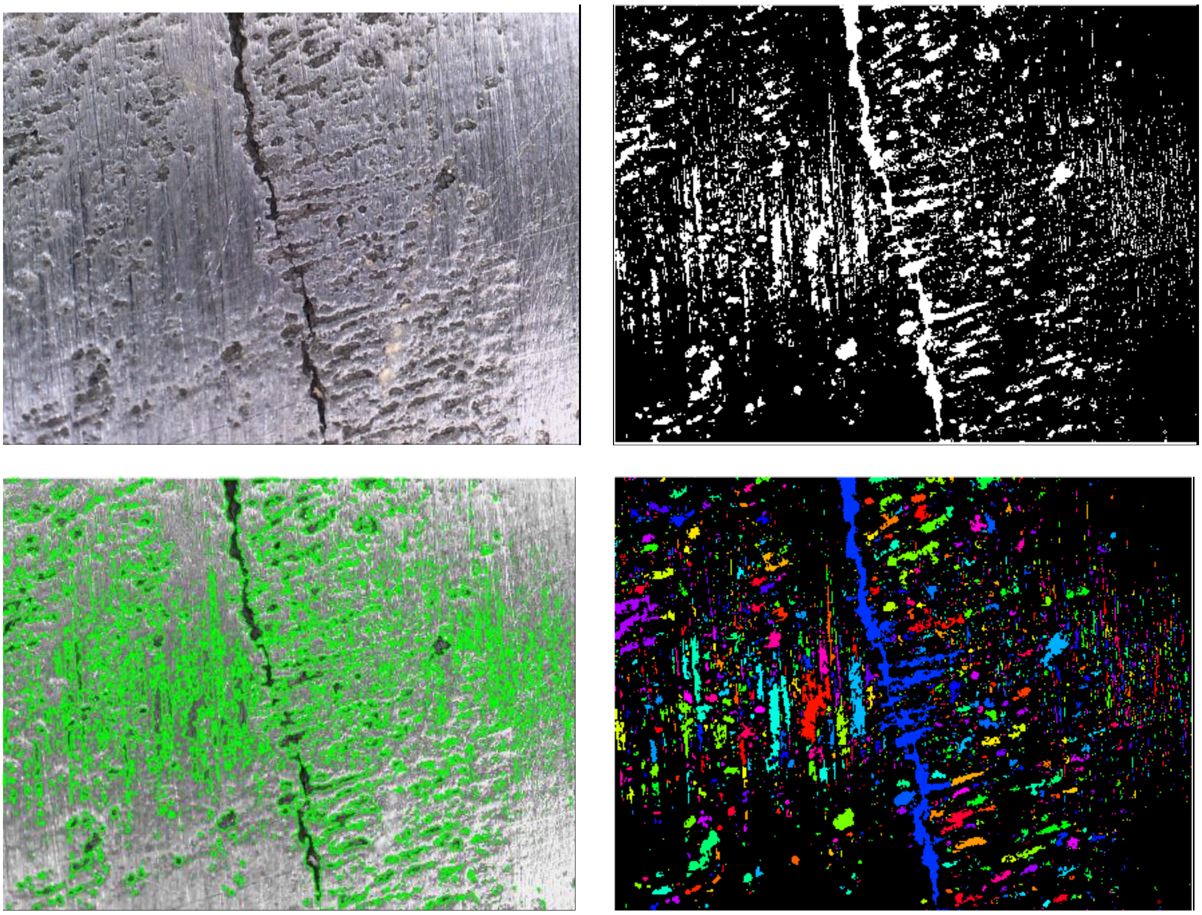


Figure 5-32 Watershed outcome on B005-L2 after applying labelled connected components

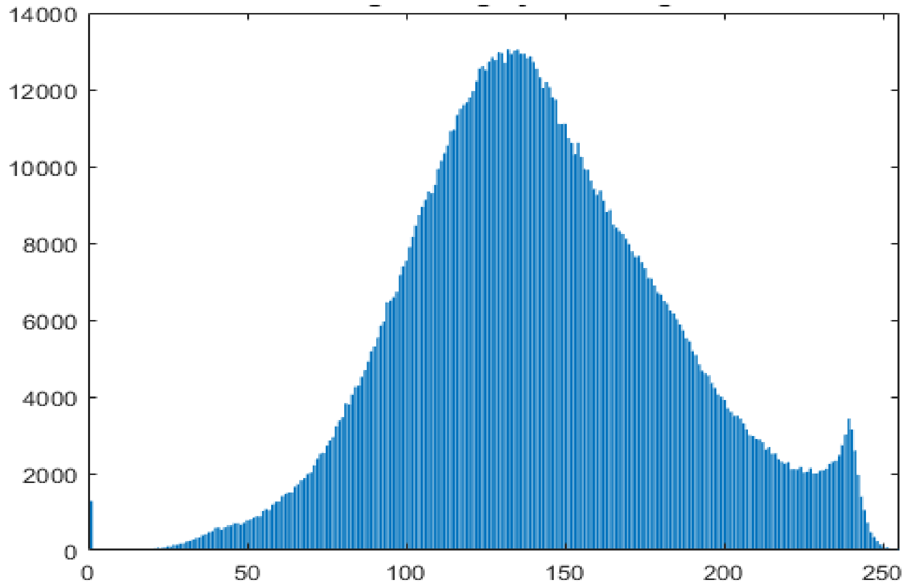


Figure 5-33 Watershed outcome on B005-L2 showing histogram of the grey scale image

After the initial stage of segmentation, different lengths of the object detected were tested, which in return produces different number of flaws picked. Table 5.3, shows different outcomes of the number of flaws based on the object length restrictions. The figure 5.34, shows visual watershed results with same test sample, Image B005-L2 with different cut-off lengths producing and detecting different number of cracks.

Table 5-3 Watershed on B005_L2 showing different outcomes based on cut-off length

Image #	Length	Mean Intensity	# of Cracks
B005-L2	>=40	-	99
	>=250	-	1
	>=100	-	2

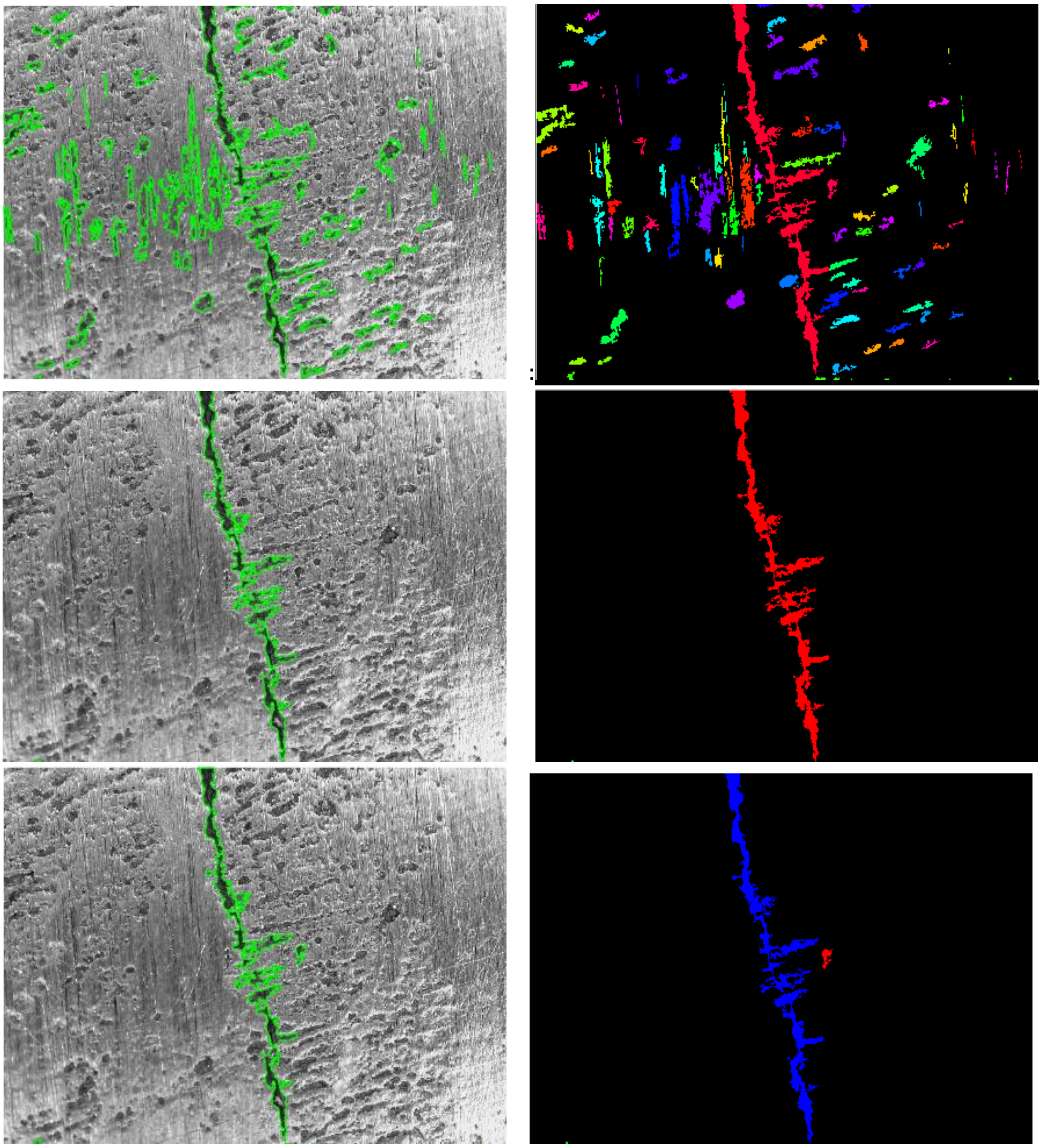


Figure 5-34 Watershed results on B0005-L2 showing varied outcomes based on length

From previous figure, the watershed coloured-labelled outcome can be visually seen in figure 5.34, with different lengths showing that a single crack, two cracks and multiple cracks were detected on specific cut-off values.

Test Sample B009-L2

It is an interesting image as it shows 2 cracks. The figure 5.35, a) shows the test sample Image B009-L2, b) displays traced object boundaries on the watershed image.

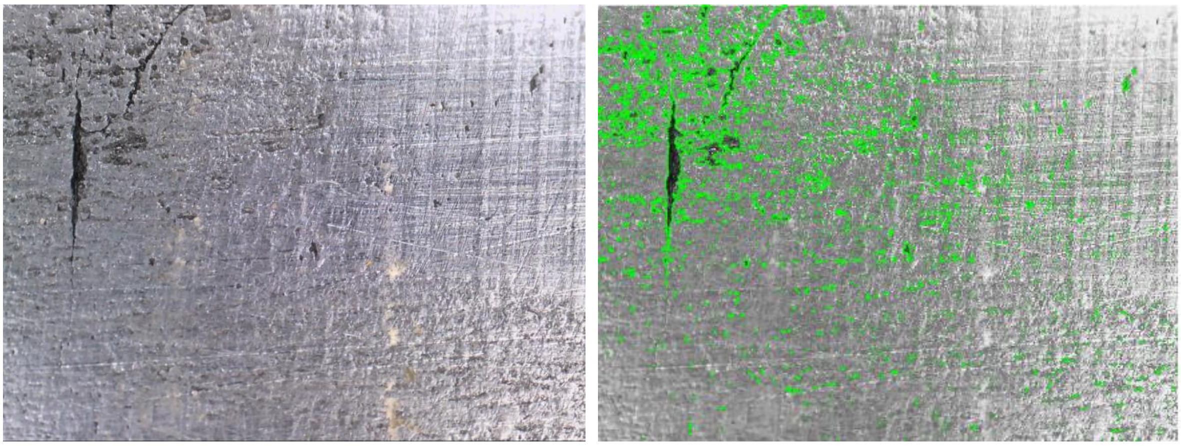
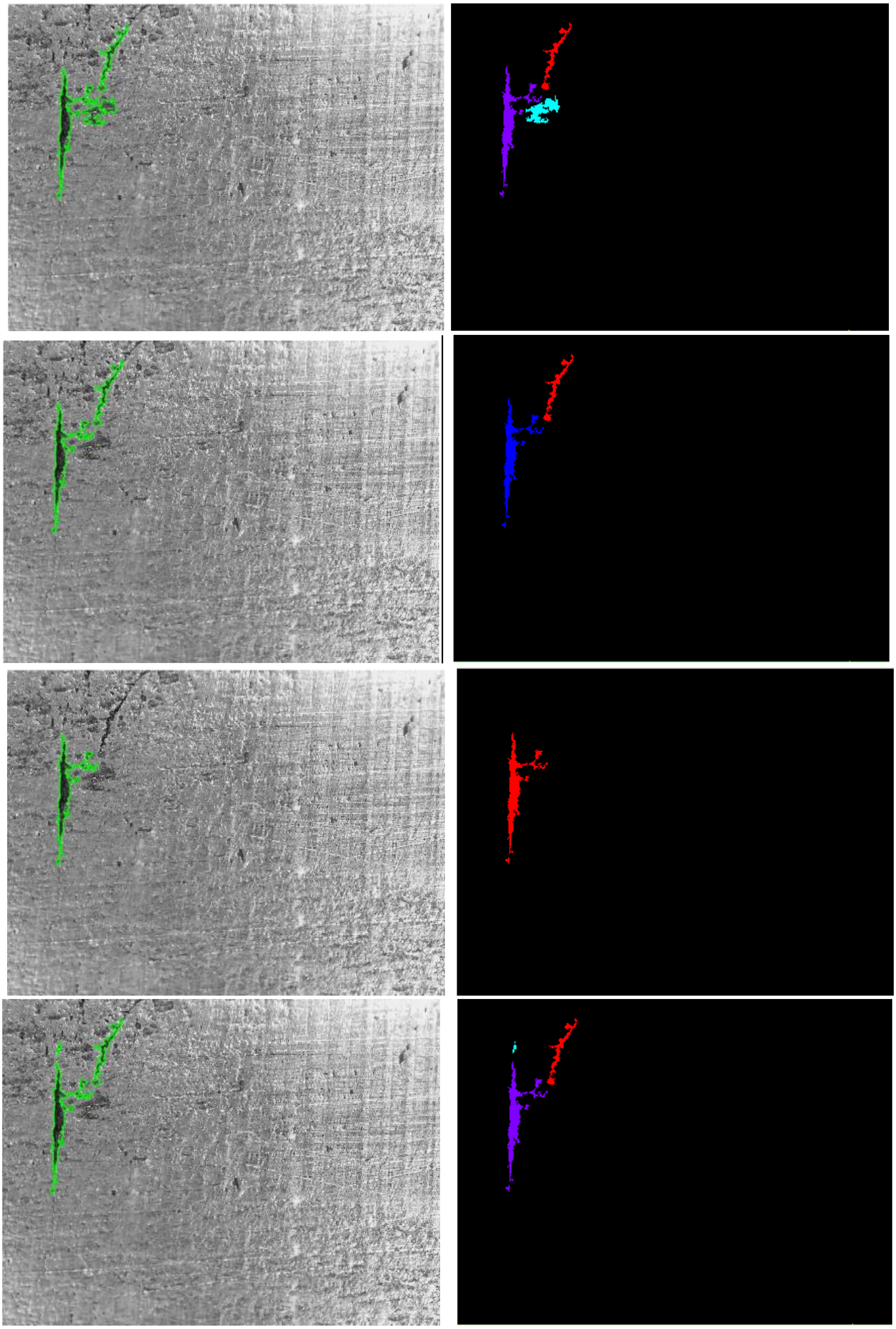


Figure 5-35 Watershed outcome on B009-L2 displaying traced object boundaries

Table 5.4, shows different outcomes of the number of flaws based on the object length restrictions. The figure 5.36, shows visual watershed results with same test sample, Image B009-L2 with different cut-off lengths producing and detecting different number of cracks.

Table 5-4 Watershed on B009-L2 showing different outcomes based on cut-off length

Image #	Pixel Length	Mean Intensity	# of Cracks
B009-L2	>=100	-	3
	>=200	-	2
	>=250	-	1
	>=100	70	3
	>=40	70	7



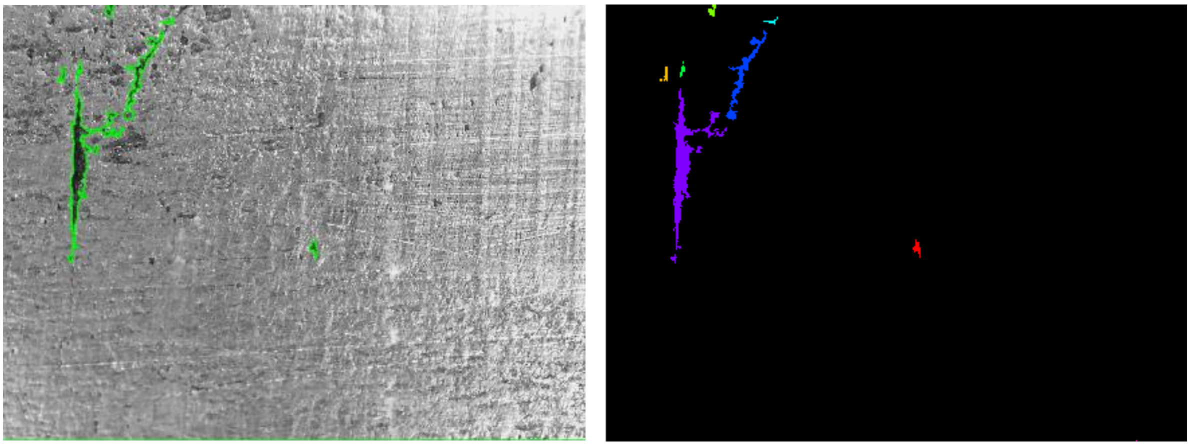


Figure 5-36 Watershed results on B009-L2 showing varied outcomes based on length

Test Sample B918-L1

This image contains clear good-sized pits that should be picked after surface noise removal. After applying region extraction in the previous step, the figure 5.37, a) shows the test sample Image B514-2016, b) shows the resultant Watershed Binary image, c) displays traced object boundaries on the watershed applied image, d) shows the outcome of Colour-labelled connected components on the watershed image. Then figure 5.38 shows histogram of the gray scale version of the image.

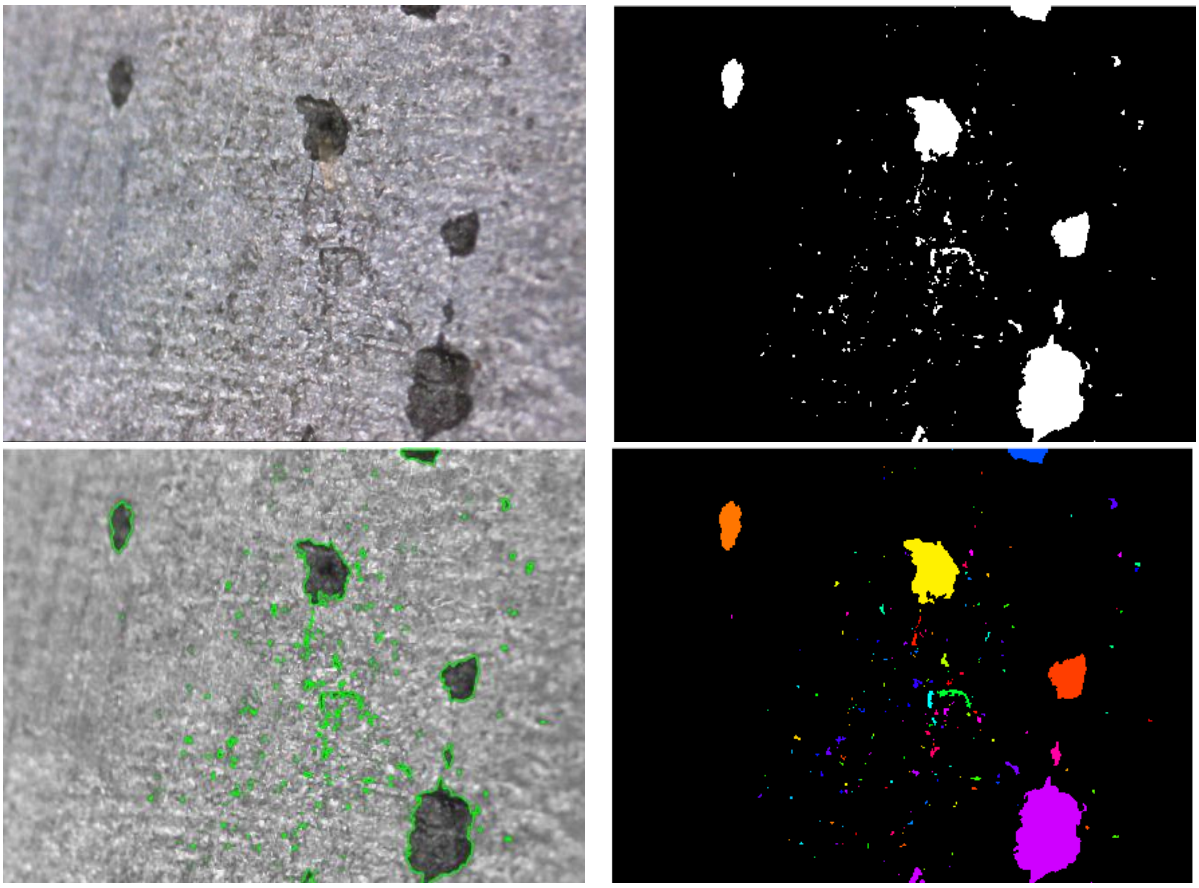


Figure 5-37 Watershed outcome on B918-L1 after applying labelled connected components

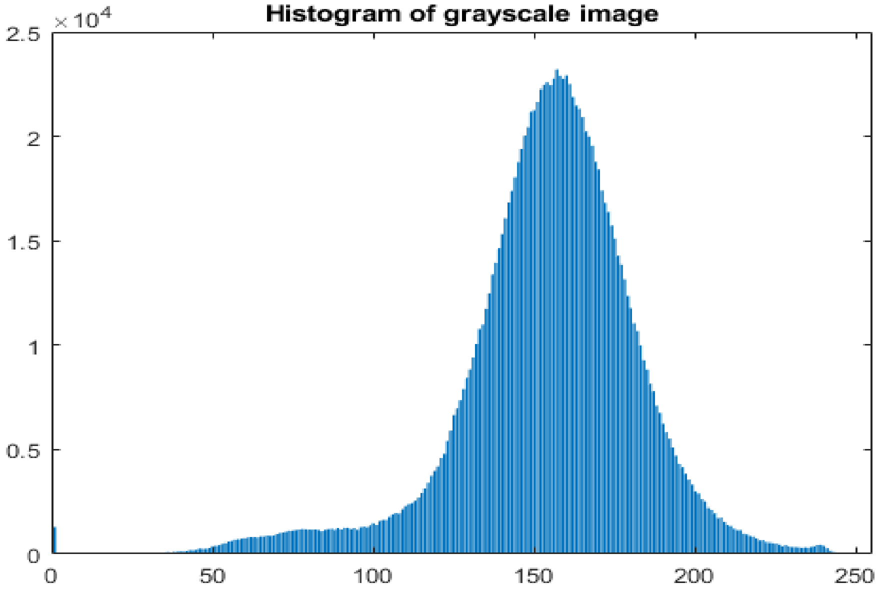
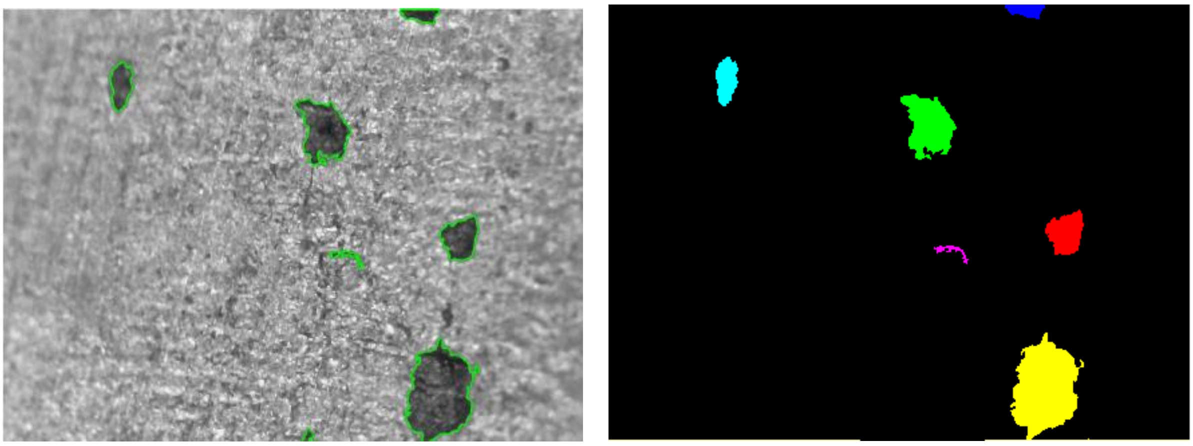


Figure 5-38 Watershed outcome on B918-L1 showing histogram of the grey scale image

Table 5.5, shows different outcomes of the number of flaws based on the object length restrictions. The figure 5.39, shows visual watershed results with same test sample, Image B918-L1 with different cut-off lengths producing and detecting different number of pits.

Table 5-5 Watershed on B918-L1 showing different outcomes based on cut-off length

Image #	Length	Mean Intensity	# of Pits/cracks
B918-L1	64	-	6
	128	-	2
	320	-	1



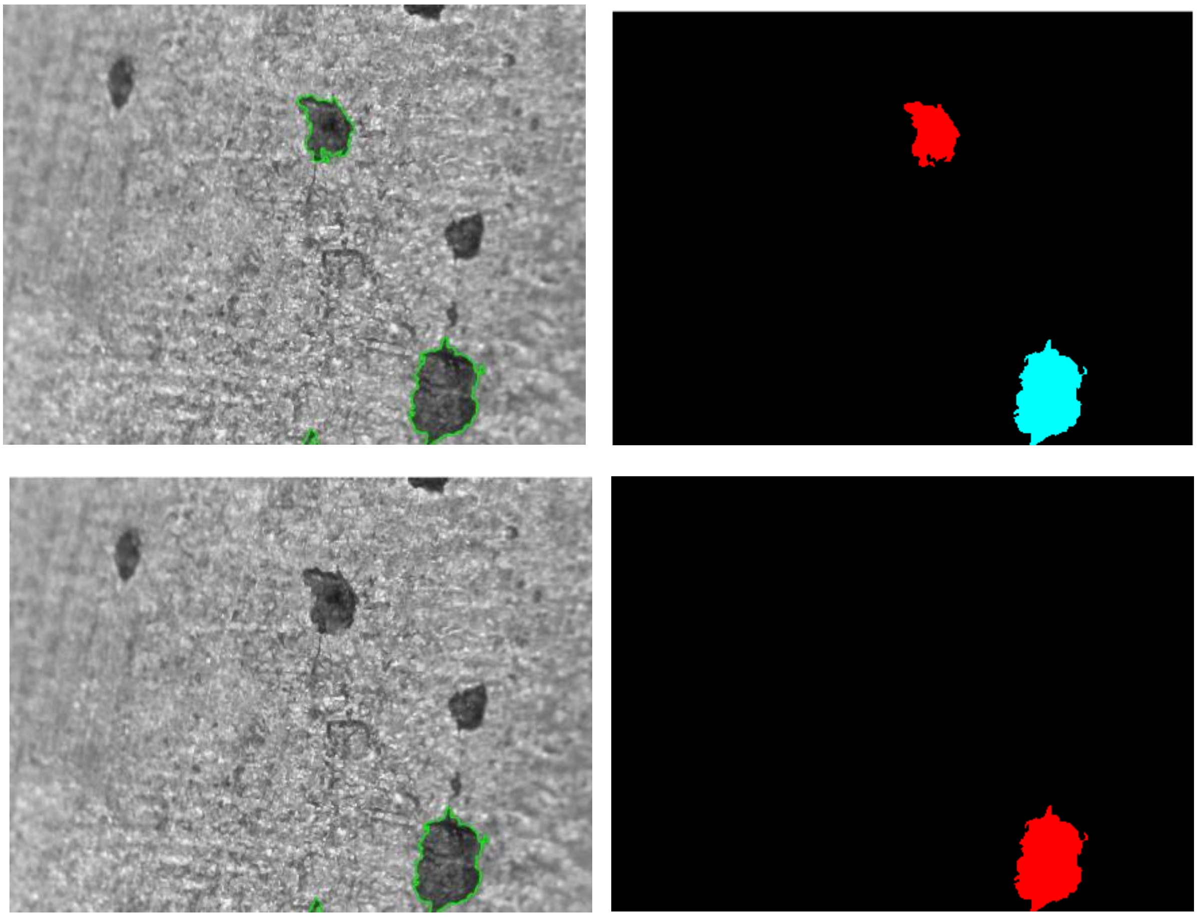


Figure 5-39 Watershed results on B918-L1 showing varied outcomes based on length

5.3.9 Conclusion

It has been able to detect and count the number of flaws, for both pits and cracks, with good shape extraction especially boundary details, as seen from numerous examples displayed above. This is the easiet yet effective method that fits the requirement smartly. It works better when the images are taken at a high magnification as they have clear edges and it is also affected by lighting conditions and data handling.

5.4 MORPHOLOGICAL-BASED ALGORITHM IMPLEMENTATION FOR IMAGE SEGMENTATION

Mathematical morphology [140] [141] is a popular methodology that is used for image analysis, smoothing, segmentation, edge detection, thinning, shape analysis and coding [142], where it examines the geometry of an image directly in the spatial domain. The advantage of mathematical morphology is its capability to identify and/or enhance features of specific shape and orientation in the data [77]. Morphology operates under the control of a structuring element, where regions can be reshaped or morphed in various ways. Image regions are the light and dark portions of an image and structuring element (SE) can be thought of as a parameter to the morphological operation, which is based on the shape that needs to be extracted. The most fundamental operations are morphological dilation and erosion.

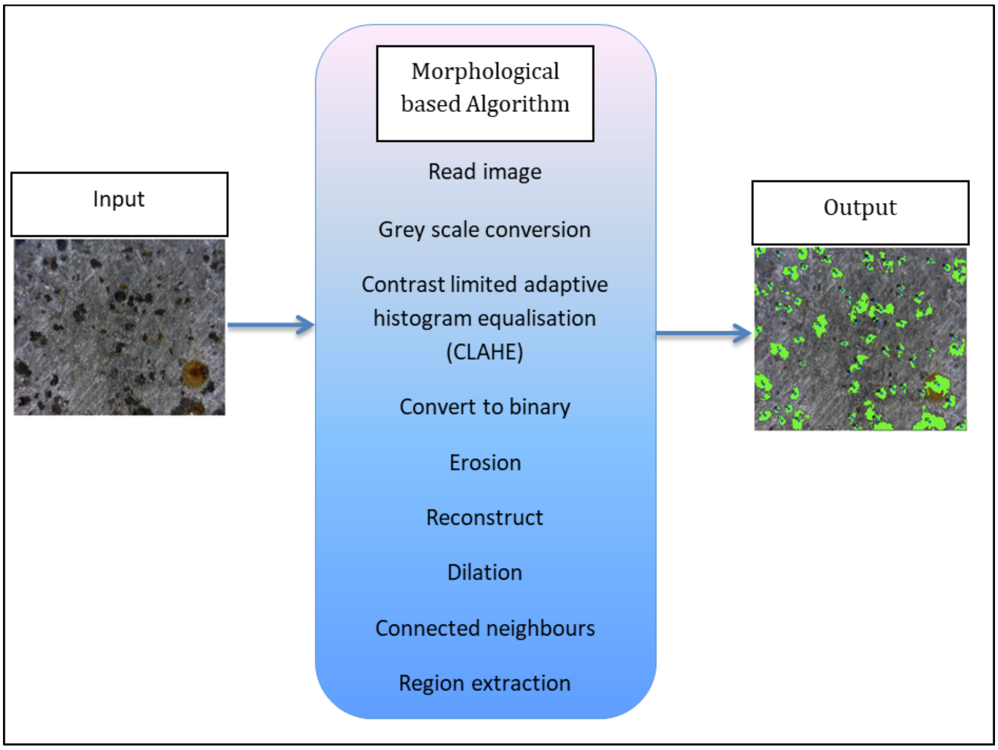


Figure 5-40 Block diagram of Morphological algorithm

Similar to the previous algorithms, it takes an RGB image as an input and displays the processed image in the end along with an excel file for the pit's counts, measurements and performance. This can be seen in a block diagram in figure 5.40.

5.4.1 Design and structure the implementation steps

Some of the steps will be explained further and some have already been explained in the previous algorithm steps. The main flow of the algorithm consists of the following steps:

1. Read the RGB image;
2. Setup the parameters
3. Add an outside border;
4. Convert it to grey scale;
5. Apply Contrast Limited adaptive Histogram Equalisation
6. Used Kittler Minimum Error Thresholding to get an optimal threshold value
7. Converted to binary image by using the threshold value above
8. Created SE1
9. Applied Erosion based on SE1 and then Reconstruction
10. Fill the holes

11. Performed dilation based on SE1 and then reconstructed again.
12. Inversed the image
13. Computed the regional maxima
14. Created SE2
15. Performed closing based on SE2 and then erosion.
16. Removed small objects with a ‘p’ value
17. Subtract to apply correction
18. Thresholding applied by using ‘Tfm’ value
19. Delete border
20. Connected the components to label each object to make measurements of it;
21. Apply pseudo random color labels to distinguish between them easily;
22. Extract features of the detected objects;
23. Then save the interesting measurement results like Area and length of each of the objects in an excel file for all the images.
24. Display the visual processed images
25. Assessment steps

5.4.2 Setup the parameters

Main parameters used in the algorithm are as follows:

- R1: erosion radius number one, first cleaning of the background to remove the noise (value close to a pit radius)
- R2: erosion radius number two, cleaning of the borders of the markers obtained by the first opening/closing step (R2<R1)
- Tfm: threshold referring to the histogram computation allowing to remove false markers (the real markers have to highest grey levels on the displayed histogram and should ultimately be the only one to remain in order to have a satisfying count)
- C: parameter useful to add a white border to the original image in order to be able to compute markers situated at the image edges, corresponding to the colour of the border
- sc_high: scale factor for high magnification conversion to mm units
- sc_low: scaling factor for low mag to get parameters converted into mm.

5.4.3 Conversion from RGB to Greyscale values

Performs conversion from RGB shown in figure 5.41 to grey scale as shown in figure 5.42.

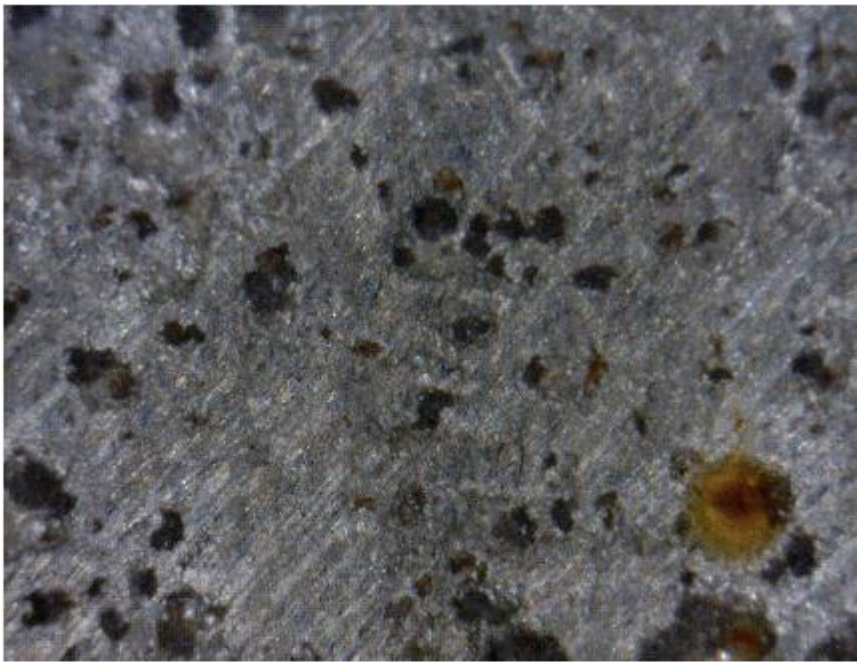


Figure 5-41 Test sample used for showing Morphological algorithm implementation

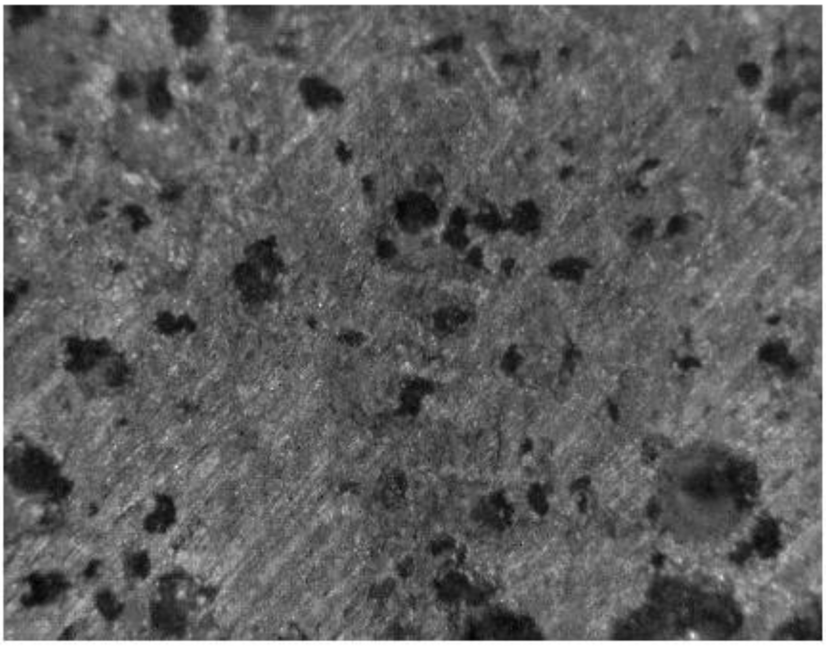


Figure 5-42 Morphological algorithm outcome showing conversion from RGB to grey scale

5.4.4 Apply Contrast Limited adaptive Histogram Equalisation (CLAHE)

Performed image enhancement based on the grey level intensities of the pixels. To enhance the contrast of the grey scale image by transforming the values using (CLAHE) Contrast Limited adaptive Histogram Equalisation [125]. CLAHE operates on small regions in the image, called tiles, rather than the whole image. It has greatly improved the image, as shown in figure 5.43.

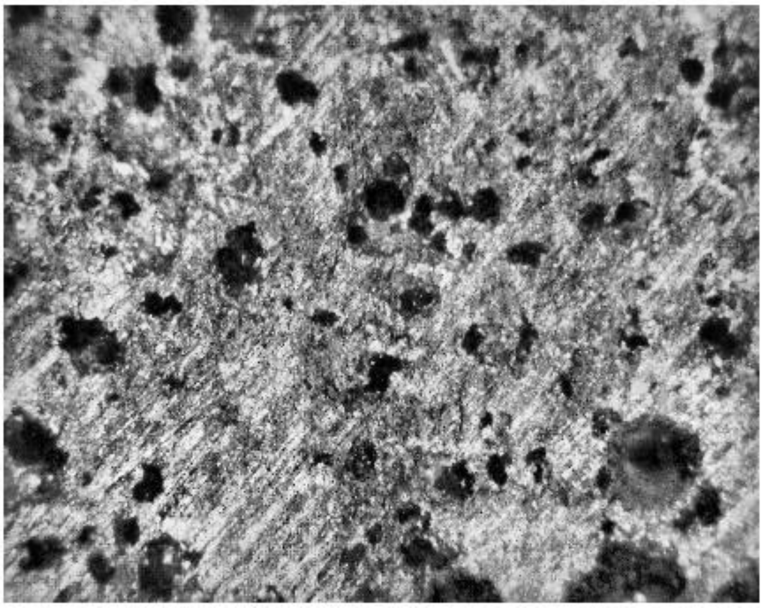


Figure 5-43 Morphological algorithm outcome after applying CLAHE

5.4.5 Kittler Minimum Error Thresholding

Kittler Minimum Error Thresholding [143] was used to compute an optimal threshold. It is a function computing a special method to access the threshold value based on a repartition of pixel grey level value on both side of an evolving threshold value.

5.4.6 Conversion to binary image

Edge detection: brightness discontinuities i.e. intensity changes. After that converted to black and white scale shown in figure 5.44.

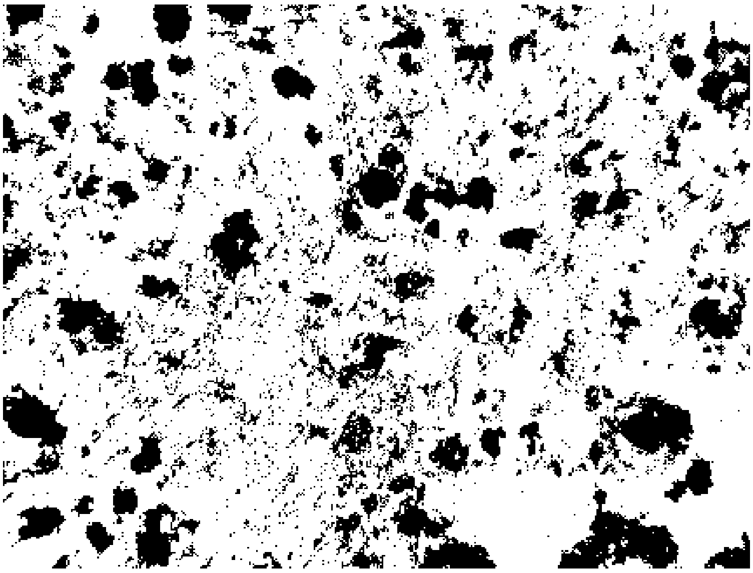


Figure 5-44 Morphological algorithm outcome after conversion to black & white

5.4.7 Morphological operations performed on the binary image based on SE1

Morphological filtering are based on shapes. They perform shrinking and/or expanding operations with regard to a certain structuring element (SE). A morphological structuring element (SE) called ‘strel’ was used to create a disk-shaped SE. The shape chosen depends on the type of feature being dilated. The morphological operators applied were erode, reconstruct and dilate.

5.4.8 Morphological operation performed based on SE2

Performed compliment to the previous resultant image and computed the regional maxima value. Then another SE was created for cleaning up the markers. This was followed by closing operation and then erosion.

5.4.9 Removed small objects

In this step white areas smaller than ‘p’ pixels were deleted from the binary image. Then computed extended max value so anything greater than ‘tfm’ value was deleted in the image.

5.4.10 Delete border and label the connected components

Connected the neighbours by applying the regional analysis methods which compute the regional maxima of the Image using specified connectivity as shown in figure 5.45 and coloured labelling in figure 5.46.

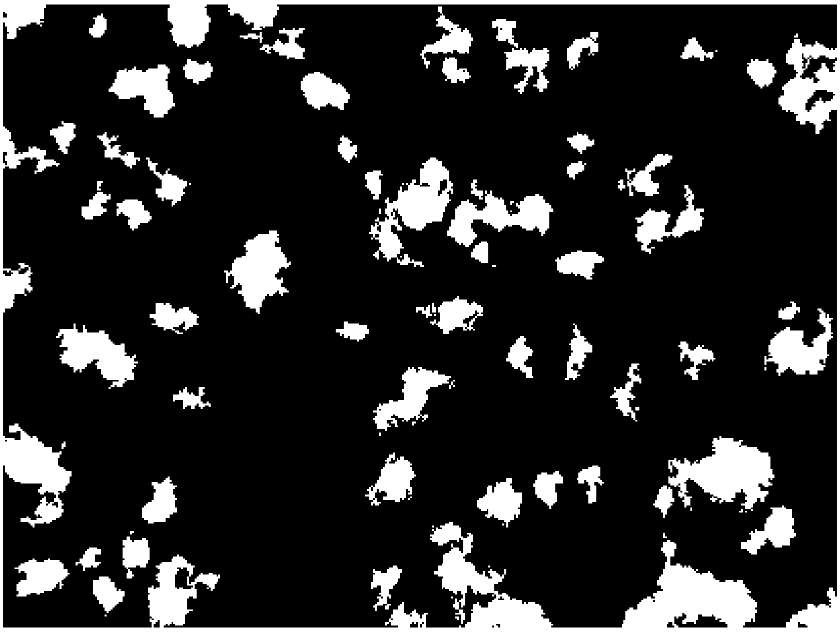


Figure 5-45 Morphological algorithm outcome after applying connecting neighbours

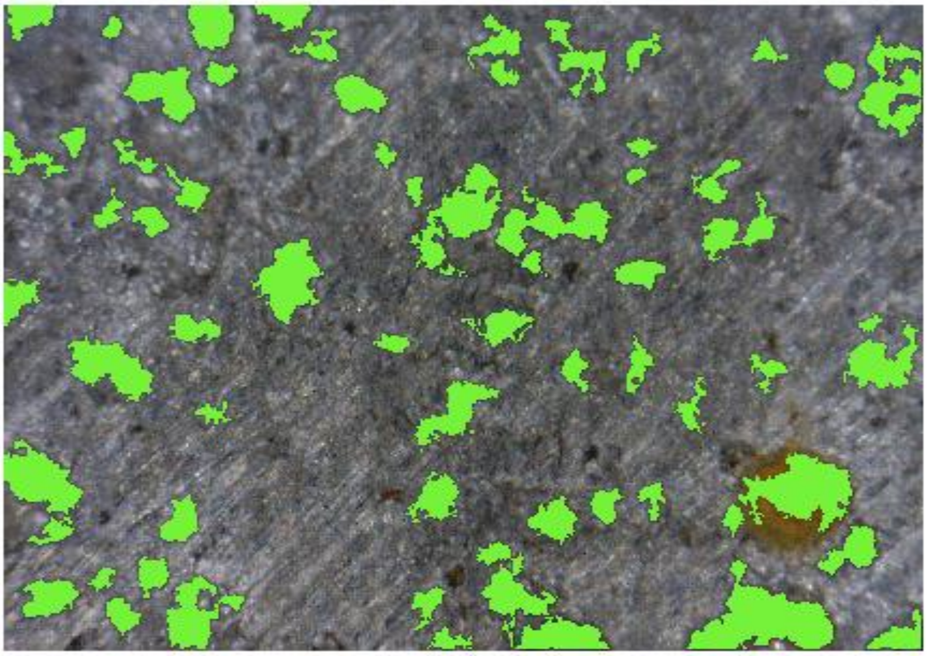


Figure 5-46 Morphological algorithm outcome after applying Region extraction

5.4.11 Count the number of flaws and display it on the image

In the end, displayed the image with the pits counted on the image as can be seen in figure 5.47 below along with an excel file.

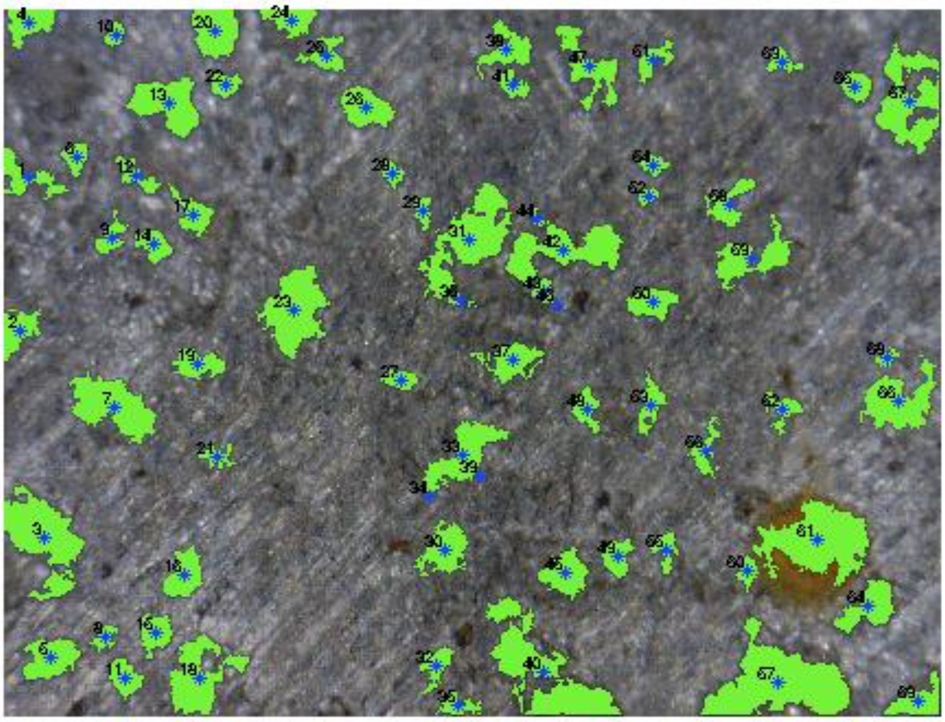


Figure 5-47 Morphological algorithm outcome showing final result with flaw counts

Isolate regions/objects from the background depending on its similar properties to identify relevant information such as its length, area, eccentricity and average intensity.

5.4.12 Nearest Neighbour

This step gives the list of objects nearest or closest to each other. The results are then displayed and also the results are saved in a file. This enabled to find the nearest neighbour to a pit entering the centroid matrix as an input.

5.4.13 Assessment steps applied based around API standard for 2D images

The following steps have not been included, when comparing the performance of different algorithms. There are many additional steps to perform 2D-Pit assessment based on the industry standard [9], some of the mains ones are as follows:

First assessment: refers to the first step of the API standard where it allows the user to know whether the proximity of two pits is considered to be dangerous. The result shows a danger scale, if danger it shows ‘1’ and if otherwise then shows ‘0’.

Second assessment: This step is performed in order to re-dimension the LTA (Local thin area) of each pit or pit couple and display the result. The input is the output of first assessment, plus other values. In the end shows horizontal length, vertical length, new centre of LTA abscissa, new centre of LTA ordinate, origin abscissa of the new LTA, origin ordinate of the new LTA. In short, if first assessment passed then, it shows the value of the corresponding rectangle, otherwise shows it for the pit of interest.

Third assessment (a): computes for the second time the distance between the LTAs (being either isolated pits or new rectangle areas) referring to the API standard second step. The result is a corrected set of values from second assessment only keeping the necessary LTAs from the new results. Third assessment (b): computes the array containing the final results and enables to display the final LTAs

5.4.14 Varying parameter ‘p’ values to change flaw size detected

For the initial phases, different combinations of ‘p’ value were applied to explore the result outcomes. This can be seen by showing a few set of visual examples from figure 5.48 to 5.53 and tabular sheet from table 5.6 to 5.8 to elaborate the point. This ‘p’ equals a value (threshold) on the basis of which an object is removed if the number of pixels is less than value of p. This is done in the end to remove extra small objects in which we are not interested like for example a pit roughly smaller than 0.5mm but in terms of pixel area. There are two other values with which further comparisons can be made, R1 and R2. For these sets of comparisons they are R1 = 8 and R2=2. It can see in the table below, the comparison of removing ‘p’ pixels from the same image and the effect it has on the performance measures.

The good point about removing the unwanted pixel area later on in the processing, is that it does not dilate the shape of the pits.

Test Sample B514-2016

This image contains many small pits that can be ignored and a few pits of the interested size as previously discussed in watershed implementation of the same image. The figure 5.48, a) shows the test sample Image B514-2016, b) shows the resultant Morphological Binary image.

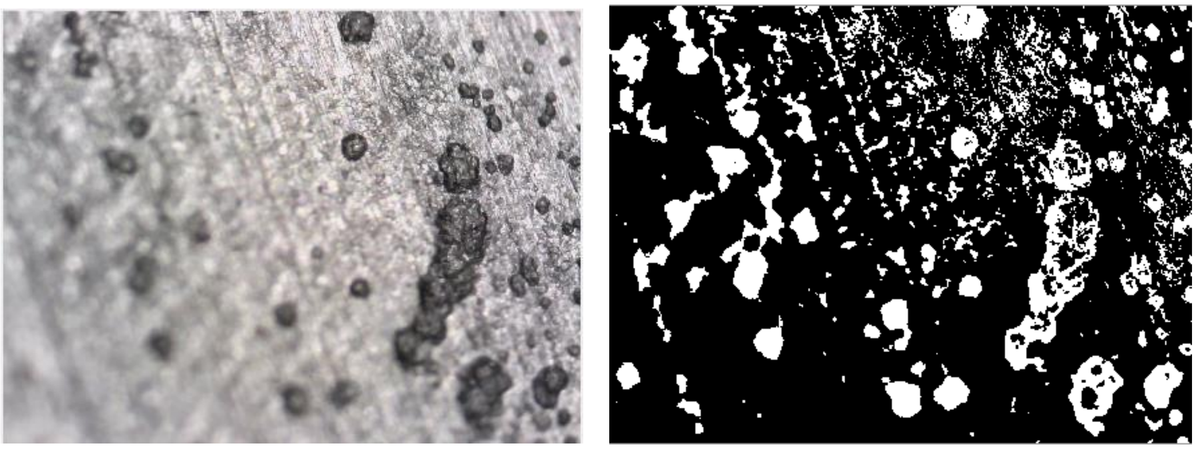


Figure 5-48 Morphological outcome on B514-16 showing resultant binary image

Table 5.6, shows different outcomes of the number of flaws based on the object length restrictions. Visual morphological result in figure 5.49 on same test Image B514-2016 with cut-off length value of 7000 shows that it picks 3 pits.

Table 5-6 Morphological on B514-L1 showing effect of variation in the cut-off size

Image #	Length	Mean Intensity	# of Pits/cracks
B514-L1	2500	-	Too many
	5000	-	Too many
	7000	-	3

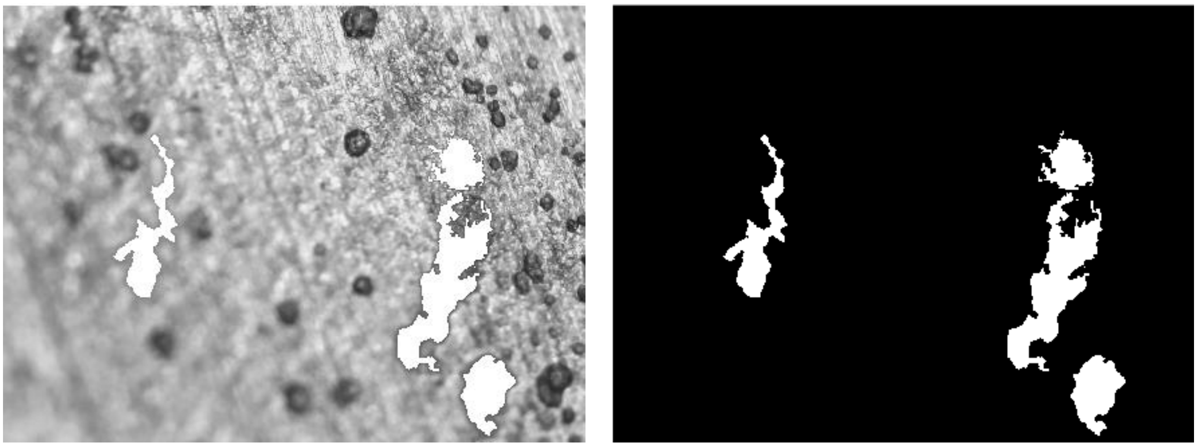


Figure 5-49 Morphological outcome on B514-16 with 7000 cut-off length

Test Sample B918-L1

This image contains clear good-sized pits that should be picked after surface noise removal. After applying region extraction in the previous step, the figure 5.50, a) shows the test sample Image B514-2016 and b) shows the resultant Morphological Binary image

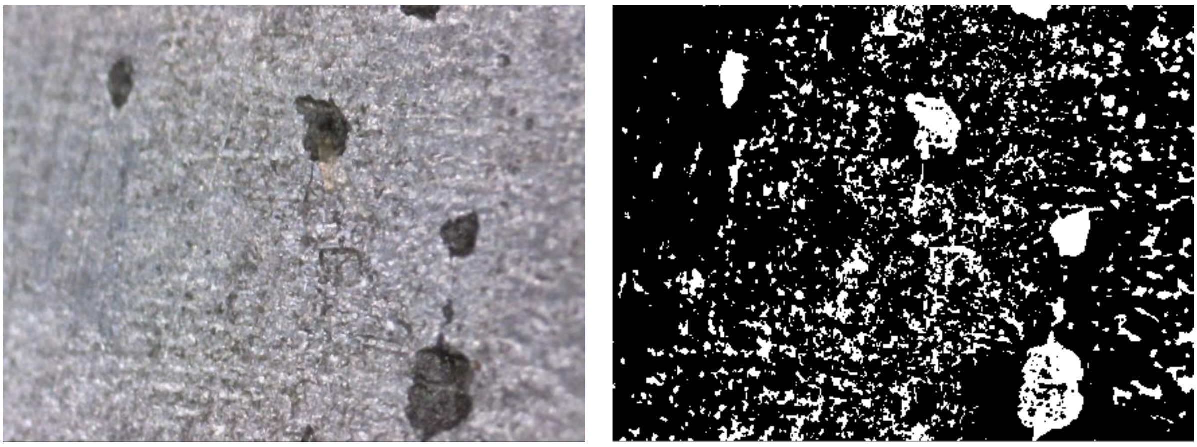


Figure 5-50 Morphological outcome on B918-L1 showing resultant binary image

Table 5.7, shows different outcomes of the number of flaws based on the object length restrictions. The figure 5.51, shows visual morphological results with same test sample, Image B918-L1 with different cut-off lengths producing and detecting different number of pits.

Table 5-7 Morphological on B918-L1 showing effect of variation in the cut-off size

Image #	Length	Mean Intensity	# of Pits/cracks
B918-L1	2500	-	5
	5000	-	3
	7000	-	2

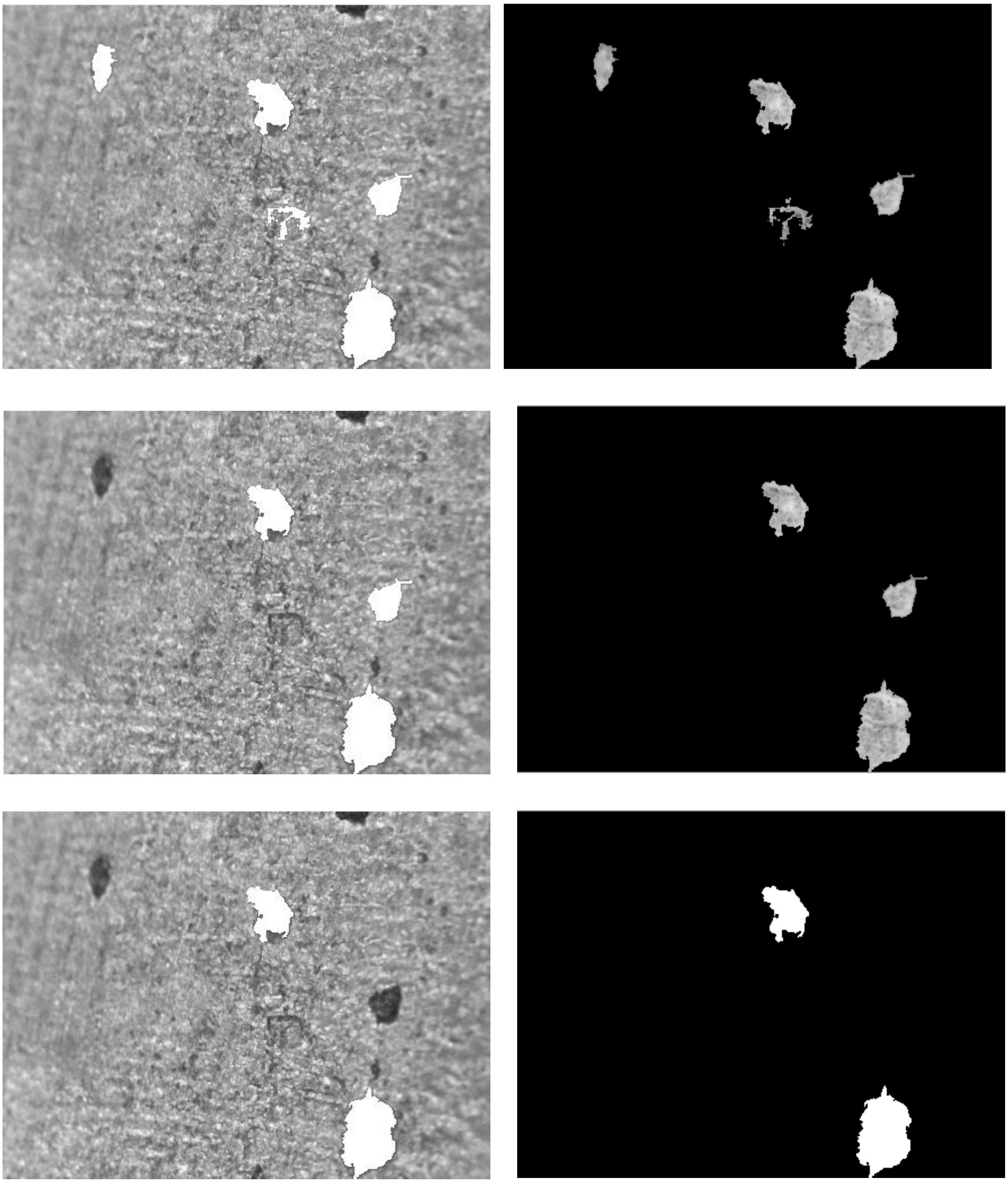


Figure 5-51 Morphological results on B918-L1 showing varied outcomes based on length

Test Sample C278-L1

This image, as shown in figure 5.52 along with its ground truth, contains clear contrast good-sized two to three pits that should be picked after surface noise removal where a) shows the test sample Image C278-L1, b) shows the labelled ground truth of the image

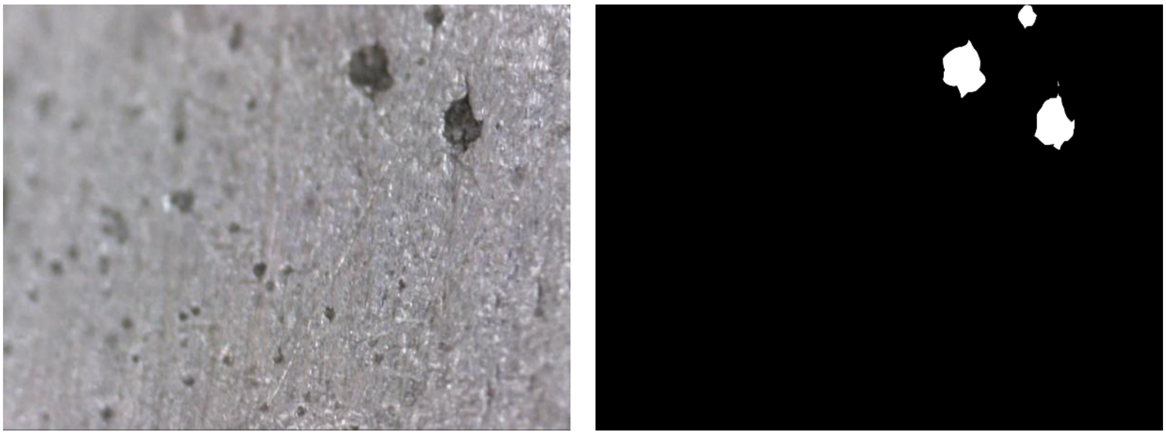
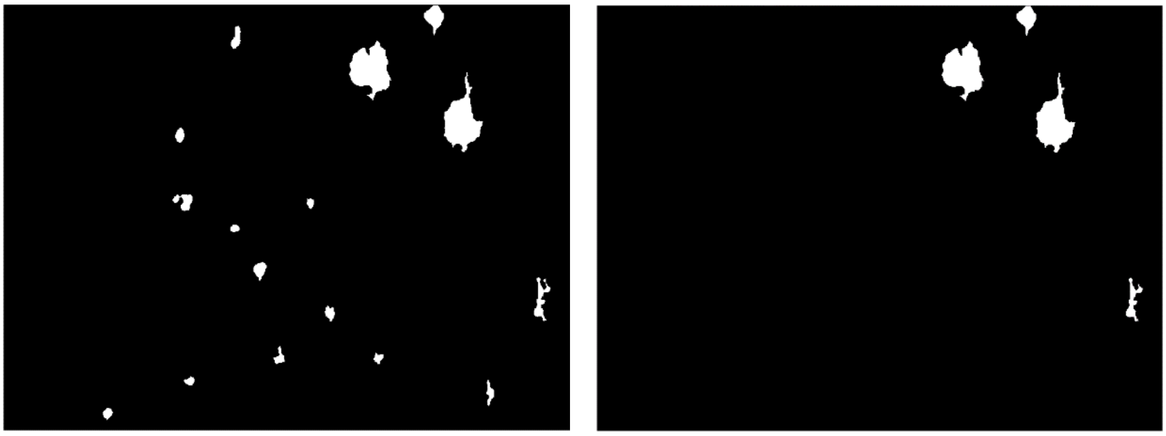


Figure 5-52 Morphological sample test image along with its ground truth image

Table 5.8, shows different outcomes of the number of flaws based on the object length restrictions. The figure 5.53, shows visual morphological results with same test sample, Image C278-L1 with different cut-off lengths producing and detecting different number of pits.

Table 5-8 Morphological algorithm for image C278-L1 showing variation effect in cut-off size

Image #	Length	Mean Intensity	# of Pits/cracks
C278-L1	1000	-	4
	1500	-	3
	2000	-	2



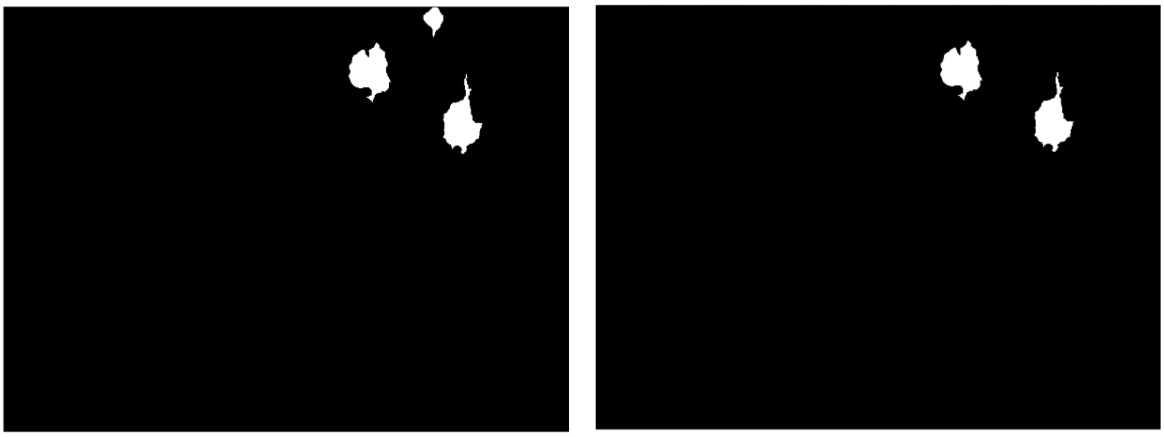


Figure 5-53 Morphological results on C278-L1 showing varied outcomes based on length

5.4.15 Conclusion

To determine the structuring element shape, the flaw being classified needs to be understood. It can be described based on the knowledge of the operators and the data collected. The assumption made in this study is that flaw indications are of darker colour in comparison to the background. The second assumption is regarding the type of flaw, if it's a pit then it is likely to have an elliptical shape and if it's a crack then it has an elongated shape. The shape information is extremely effective for detecting flaws which do not have a high contrast with the background.

Many adjustments with the parameters were made in the earlier versions of this algorithm. The algorithm basically works on the shape extraction, around the morphological structuring element from the image. But it also includes many other steps which improves the performance of segmentation which are mentioned in the sub-sections above. This algorithm also attempts to apply Pit Assessment, based on the industry API-579 standard. Although, since the camera used was 2D and the assessment requires depth information, it is only considered as an attempt towards the assessment based on 2D information, which is shown as a last step of the algorithm.

5.5 Gaussian-based FCM clustering Algorithm implementation for image segmentation

Clustering is one of the common image segmentation methods [53] [81]. FCM stands for fuzzy c-means, which is an unsupervised segmentation algorithm that is based on the concept of searching cluster centres. This is done by iteratively adjusting their position and evaluation of an objective function. This iterative optimisation of the FCM algorithm is basically a local searching method. This searching is used to minimise the distance among the image pixels in corresponding clusters and maximise the distance between cluster centres.

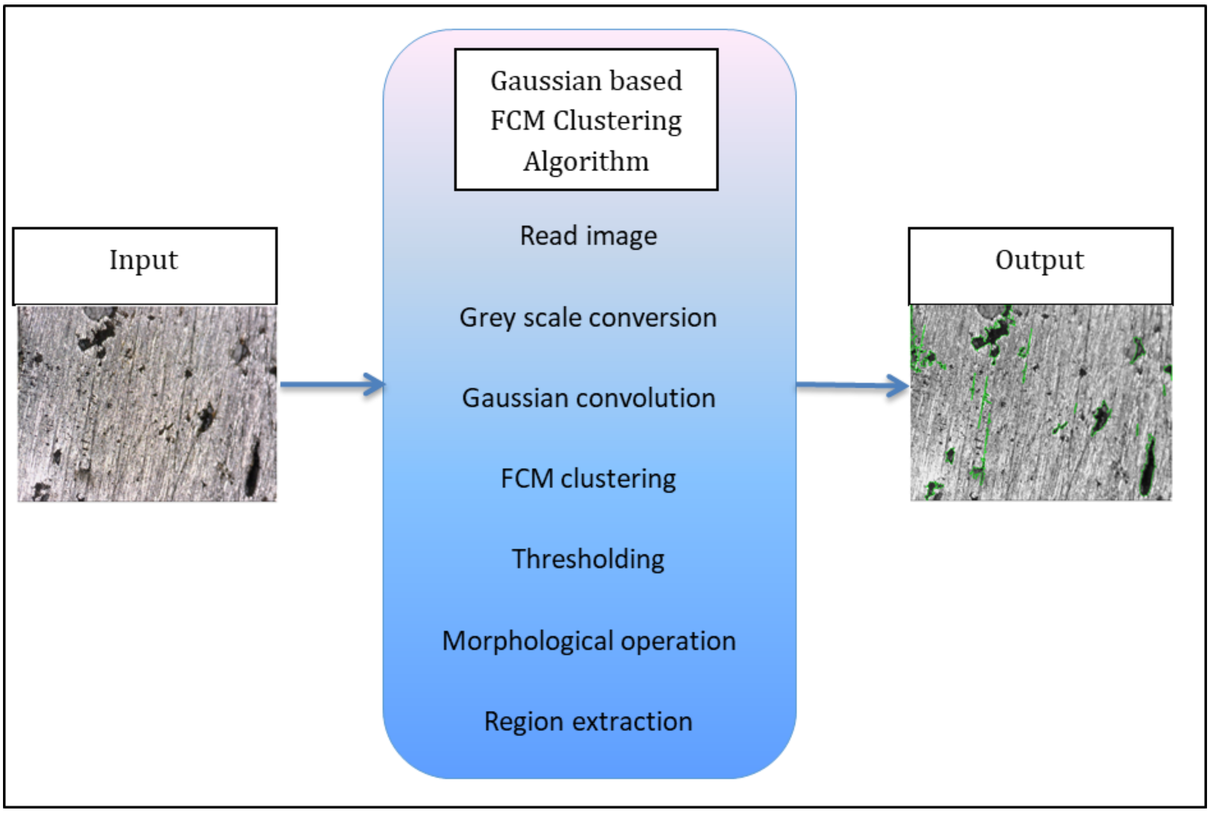


Figure 5-54 Block diagram of FCM based Gaussian Algorithm

The algorithm basically works around FCM clustering and also applies other segmentation techniques. It is a combination of a few techniques explained in the sub-sections. As in the previous algorithm, it starts by reading an RGB image as input. By the end, counts and make measurements of regions of interest (ROI) and then classifies pixels into defects/background by displaying the detected flaws in a different colour to the background. This can be seen in a block diagram in figure 5.54 The original images used as examples are real images as Image A186-2016 is shown in figure 5.55. The main flow of the algorithm is discussed in this section, which includes some of the following steps:

1. Read the RGB image;
2. Convert it to grey scale;
3. Apply Gaussian convolution
4. FCM clustering used
5. Thresholding the image
6. Apply morphological operations;
7. Label each object to make measurements of it;
8. Apply pseudo random color labels to distinguish between them easily;
9. Extract features of the detected objects;
10. Then save the interesting measurement results like Area and length of each of the

objects in an excel file for all the images.

11. Display the visual processed images.

5.5.1 Convert original image to grey scale

After reading in the test sample, Image-A186-2016, the next step is the conversion from colour to grey scale as shown in figure 5.56 by using rgb2gray. It converts RGB values to grey scale values by forming a weighted sum of the R, G, and B components, 0.2989 * R + 0.5870 * G + 0.1140 * B. It converts by eliminating the hue and saturation information while keeping the luminance.

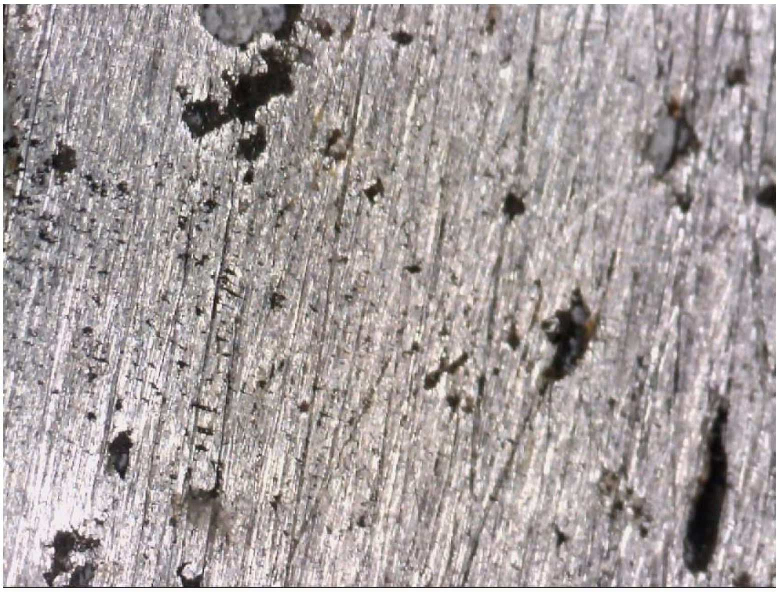


Figure 5-55 Test sample used for showing Gaussian algorithm implementation

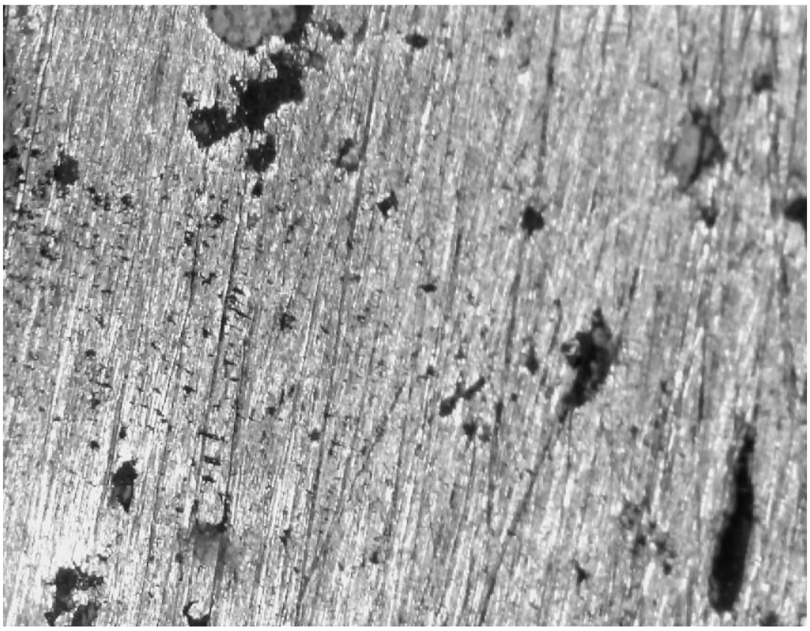


Figure 5-56 FCM Gaussian algorithm outcome after converting to grey scale

5.5.2 Apply Gaussian convolution for smoothing

A suitable mask has been calculated, and then Gaussian smoothing was performed using standard convolution methods, the resultant outcome shown in figure 5.57. The result of Gaussian smoothing is a blurred image that removes detail and noise [64]. It uses a different kernel to a mean filter, which represents a bell-shaped hump. The Gaussian outputs a weighted average of each pixel's neighbourhood, with the average weighted more towards the value of the central pixels unlike the mean filter's uniformly weighted average. Because of this, a Gaussian provides gentler smoothing and preserves edges better than a similarly sized mean filter.

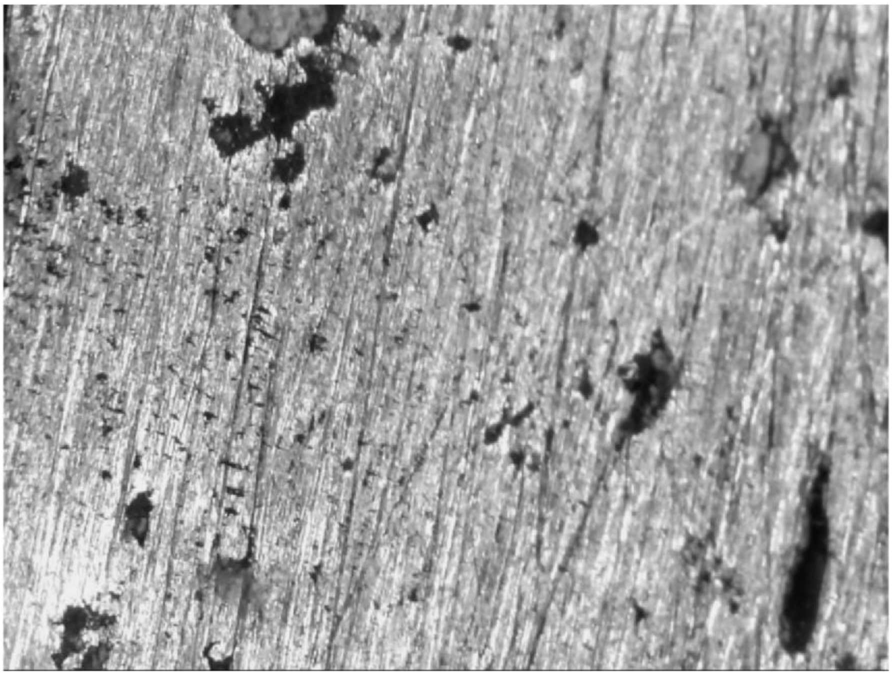


Figure 5-57 FCM Gaussian algorithm outcome after Gaussian smoothing

Mainly the reason for using Gaussian as a smoothing filter is due to its frequency response. Most convolution-based smoothing filters act as low-pass frequency filters, which means that they remove high spatial frequency components from an image.

5.5.3 FCM clustering used

It is one of the methods of clustering in which one piece of data may belong to two or more clusters. This is in contrast to k-means clustering which assigns them to a single cluster only. FCM allows the pixels to belong to multiple classes with varying degrees of membership. It can expose pixels in terms of grey value level so as that it can show hierarchical position of related defects by grey value. This grey scale based FCM clustering not only effectively suppresses noise interference but also rectifies wrong classification of pixels pretty easily. The outcome of this is shown in figure 5.58. Then a thresholding was applied to the binary image as shown in figure 5.59.

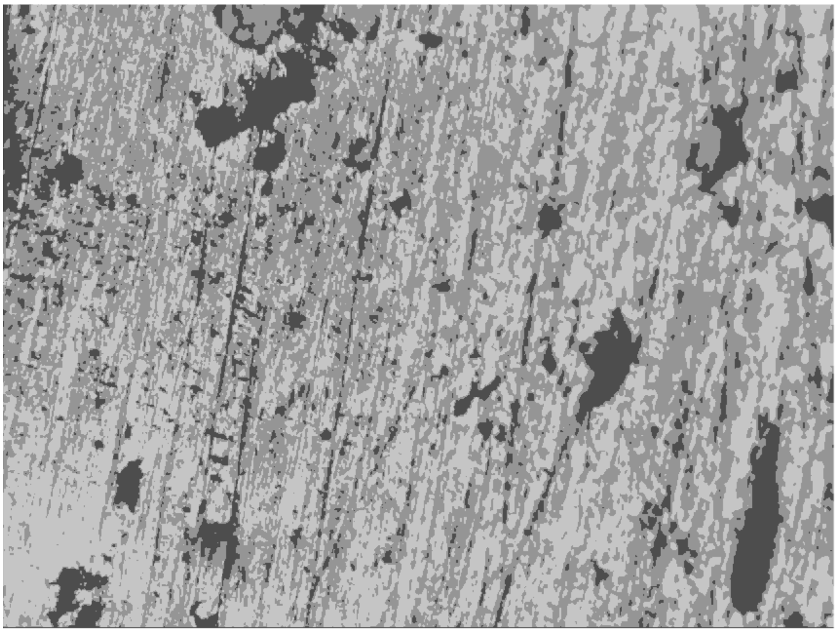


Figure 5-58 FCM Gaussian algorithm outcome applying FCM Clustering

5.5.4 Thresholding the image

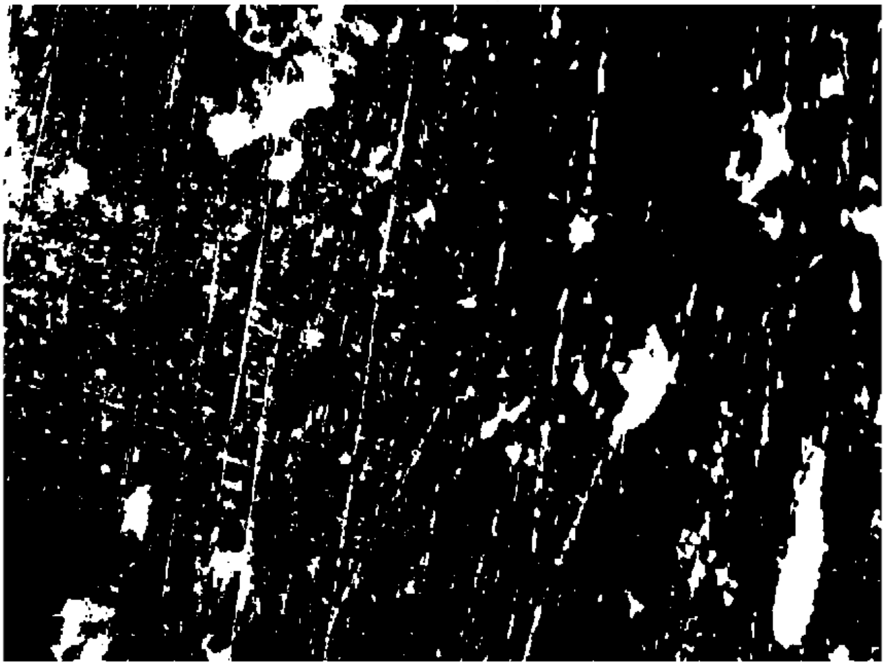


Figure 5-59 FCM Gaussian algorithm outcome after applying thresholding

5.5.5 Apply morphological operations

Morphological operations are applied on the basis of ‘strel’ SE, which is disk-shaped as this will resemble to elliptically shaped pits. An opening operation is performed, consisting of erosion and dilation, and then closing is performed on a different SE, p2. These are performed to remove noises and/or unimportant areas. The outcome of which is shown in figure 5.60.

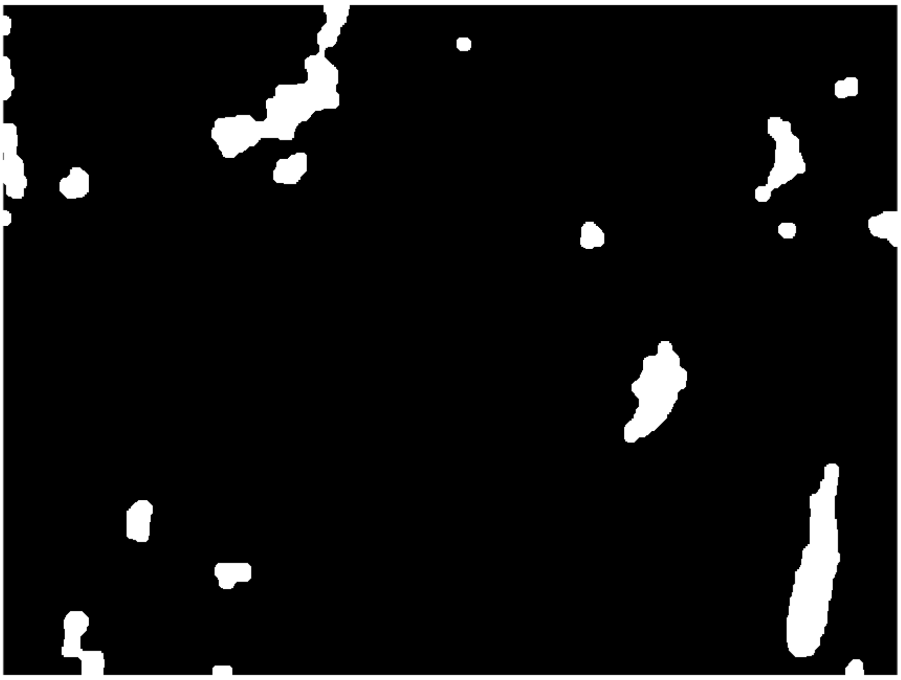


Figure 5-60 FCM Gaussian algorithm outcome after morphological operations

5.5.6 Label each object to make measurements of it

Then each object is labelled as shown in figure 5.61 after applying labelled connected component and then applied pseudo random colour labels to the image as shown in figure 5.62, to distinguish between them easily.

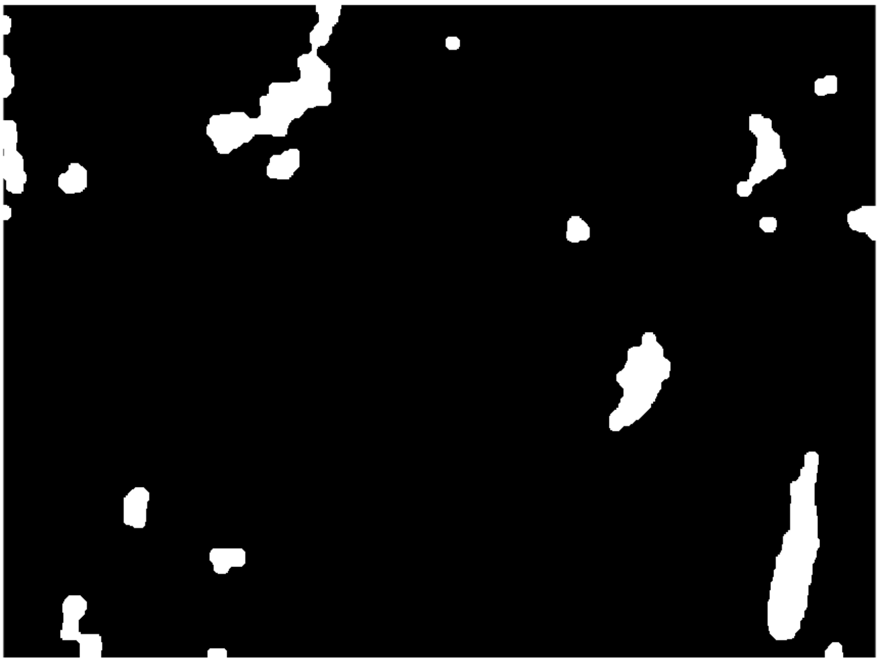


Figure 5-61 FCM Gaussian algorithm outcome after labelled connected components

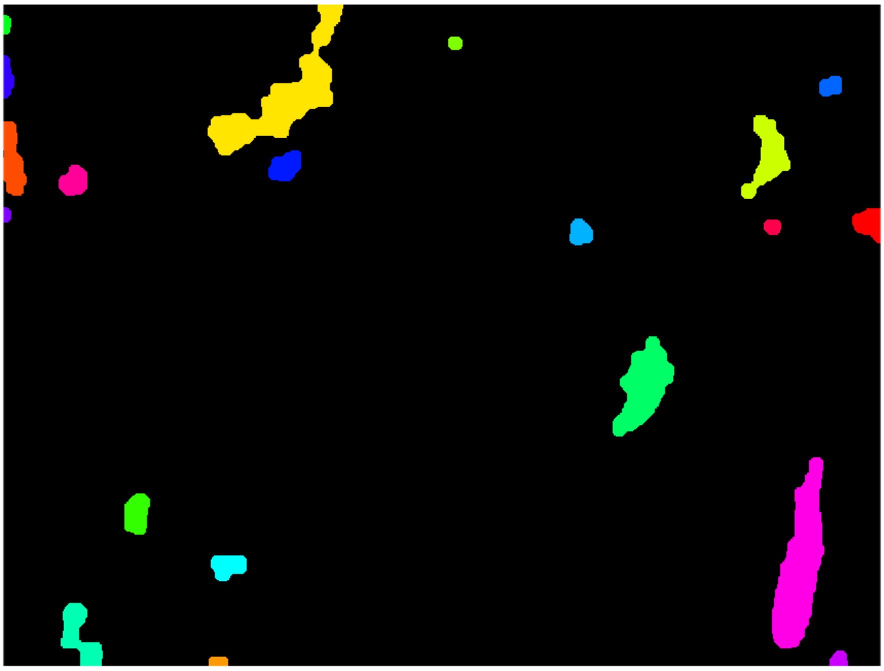


Figure 5-62 FCM Gaussian outcome after applying coloured labelled component

5.5.7 Extract features of the detected objects

Then save the interesting measurement results like Area and length of each of the objects in an excel file for all the images. Display the visual processed images in figure 5.63.

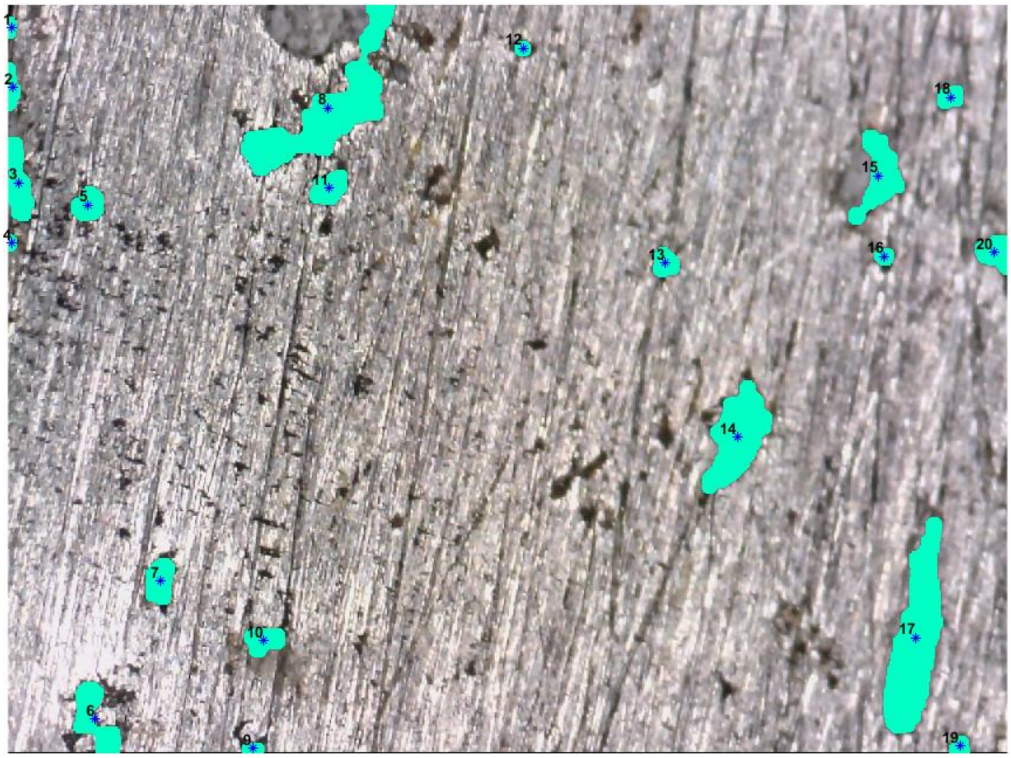


Figure 5-63 FCM Gaussian outcome showing number of flaws counted

5.5.8 Varying SE values to change flaw size detected

For the initial phases, different combinations of ‘set of p’ value were applied to explore the result outcomes. This can be seen by showing a few set of visual examples as shown in figure 5.64 and tables 5-9, to elaborate the point.

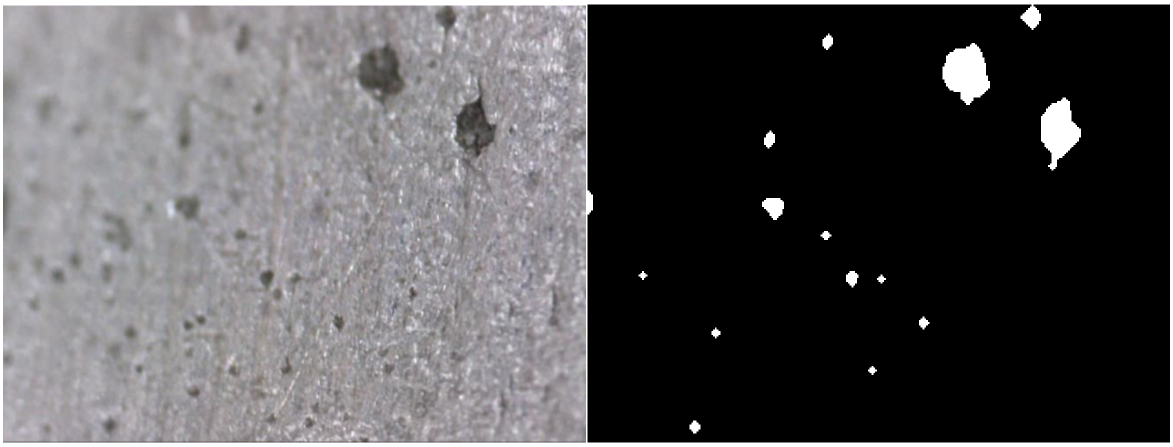


Figure 5-65 FCM Gaussian outcome on B514-16 showing resultant binary image

The sample test image consists of clear contrast good-sized two to three pits that should be picked after surface noise removal, where figure 5.65, a) shows the test sample Image C278-L1 and b) shows the resultant FCM Gaussian Binary image

Table 5-9 Gaussian on B514-L1 showing effect of variation in the cut-off size

Image #	Parameter 1	Parameter 2	# of Pits / cracks
C278-L1	2	3	(First one) 15 pits + shape ok
	1	1	>15-30 pits
	2	4	15 pits but shape more near to gt
	4	4	4 pits but shape too dilated

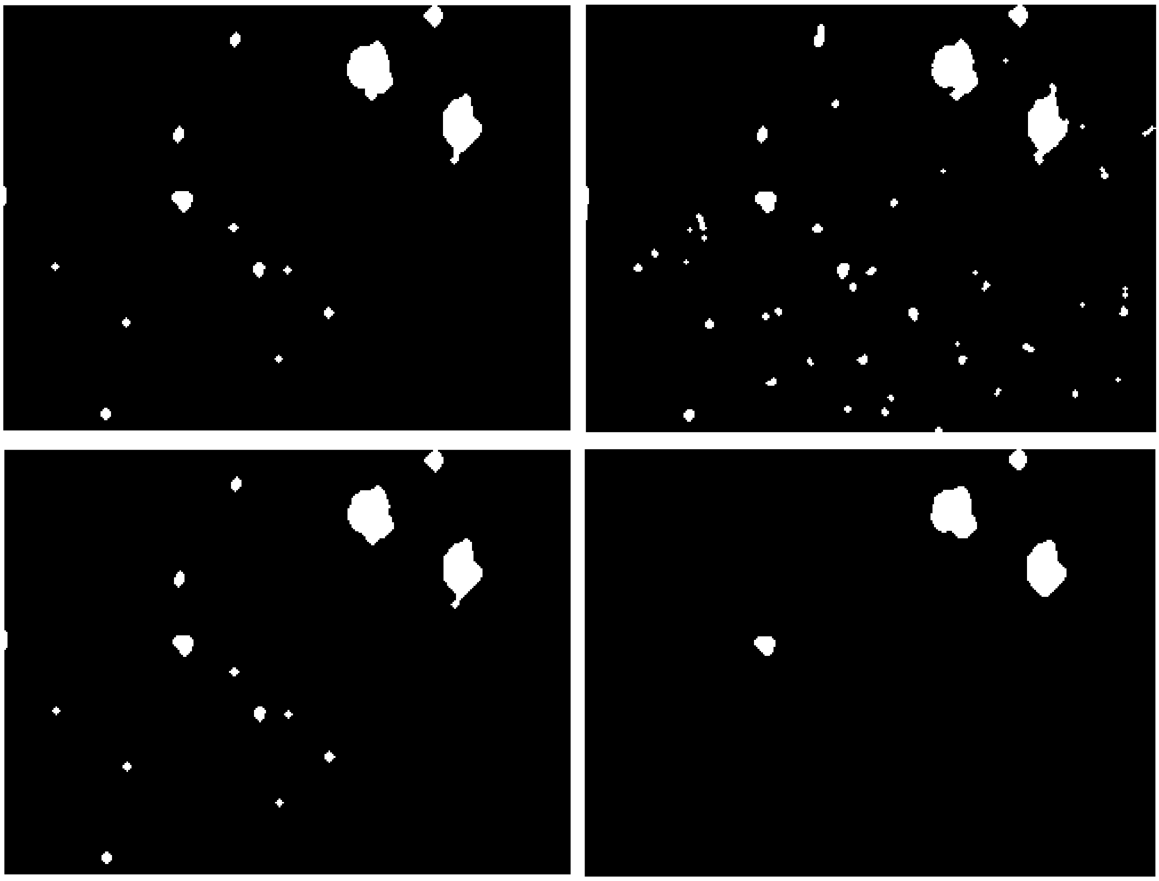


Figure 5-66 Gaussian results on C278-L1 results showing varied outcomes based on length

5.5.9 Conclusion

In this algorithm, the morphological operators used have not performed that well and hence the shape of the outcomes get distorted in that step. It is built around LoG edge with the watershed algorithm to generate the final segmentation results with less over segmentation. The mask of LoG is set as 5x5. The mask can be modified for obtaining a better segmentation result. The end result is to highlight edges.

5.6 COMPARISON BASED ON FLAW MEASUREMENTS USING SAME IMAGE

The example image taken is B919-2016 for looking into the results in detail as shown in figure 5.66. The resolution of the image is 1280x960 pixels which has been taken at a higher magnification with an image size of 2x1.5mm. As discussed in chapter 4, interested flaw size is set as, for high mag it is any object equal or greater than 0.1mm and for low mag the value needs to be equal to or greater than 0.3mm. The flaw measurements are performed on all the above three algorithms and then comparisons were made with the labelled ground truth as shown in figure 5.68. The procedure mentioned in chapter 4, was followed for the ground truth labelling by first measuring it as shown in figure 5.67, then labelling it according to size criteria and in the end, object features were extracted.

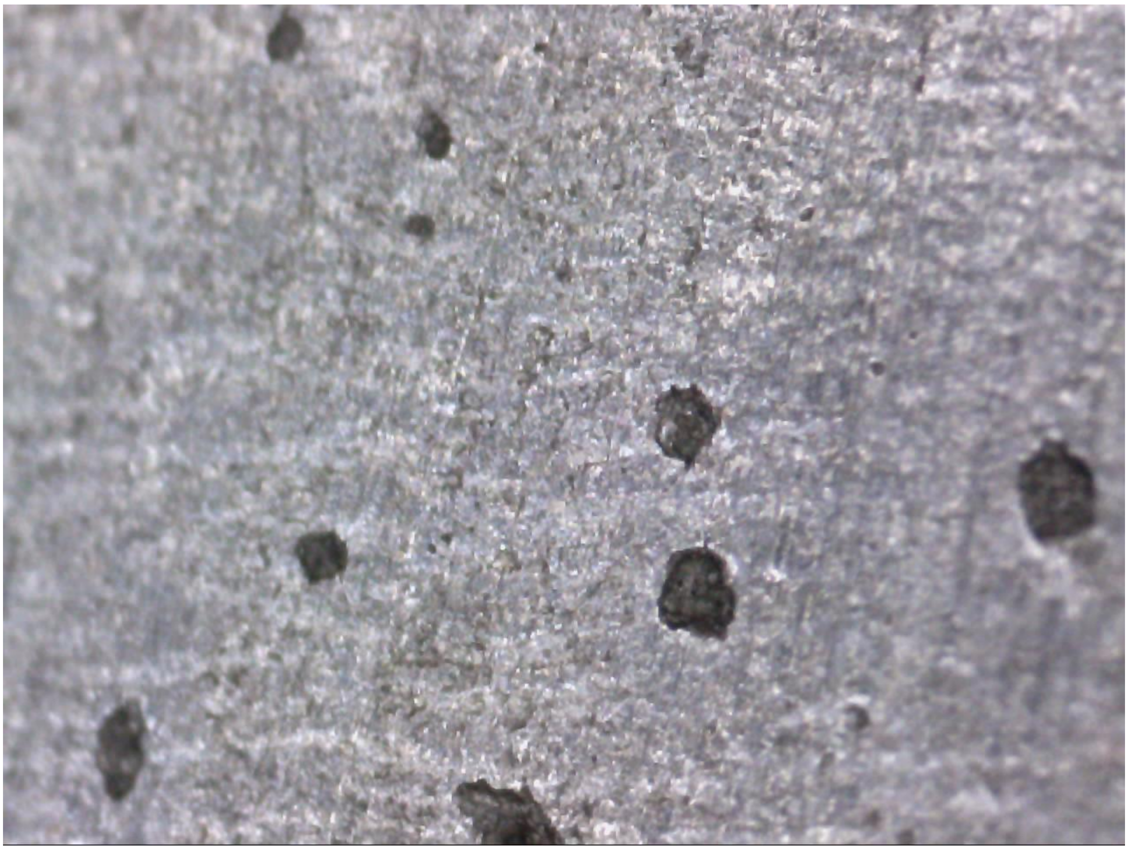


Figure 5-67 Test image to illustrate Flaw measurement by all unsupervised learning methods

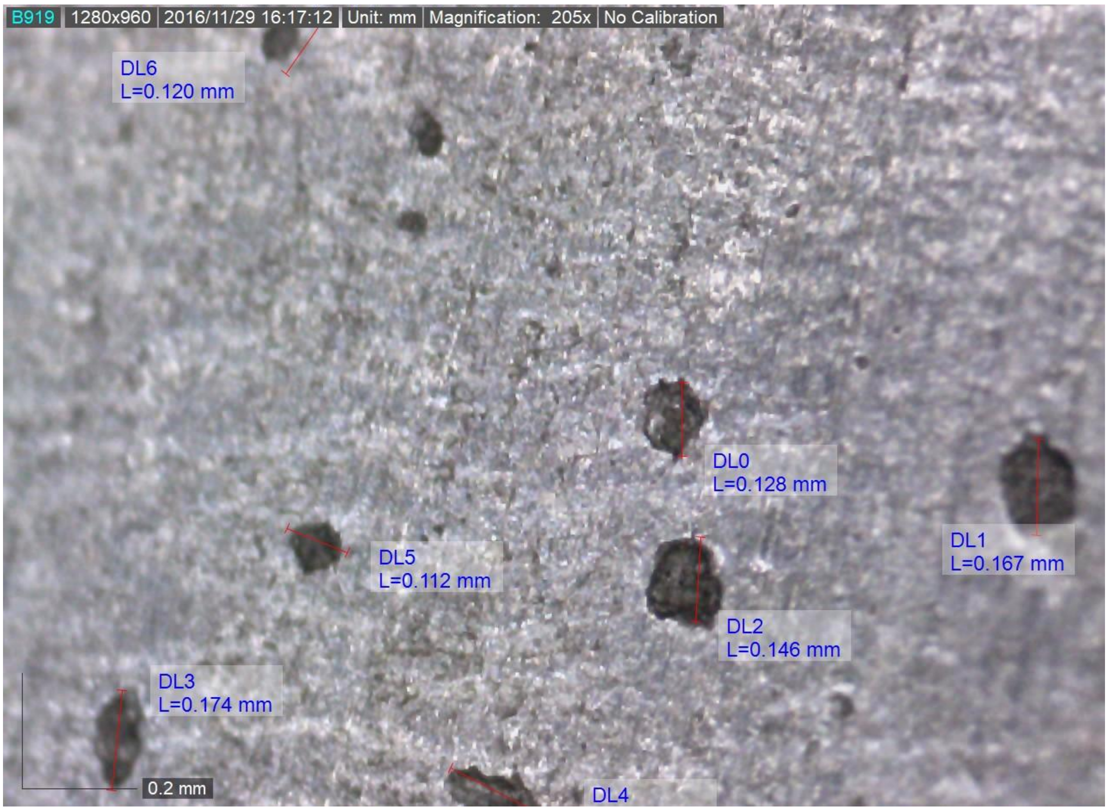


Figure 5-68 Manual measurements made for labelling the test image

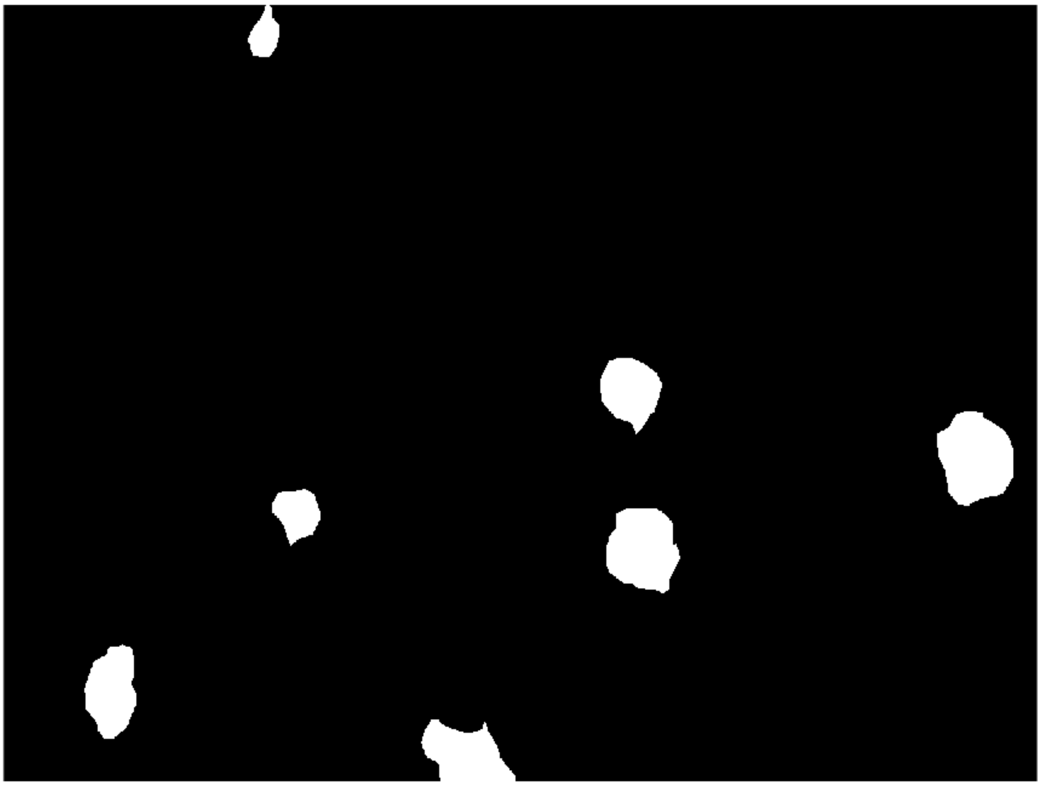


Figure 5-69 Ground Truth of the test image for evaluation of unsupervised methods

5.6.1 Watershed based algorithm's flaw measurements using same image

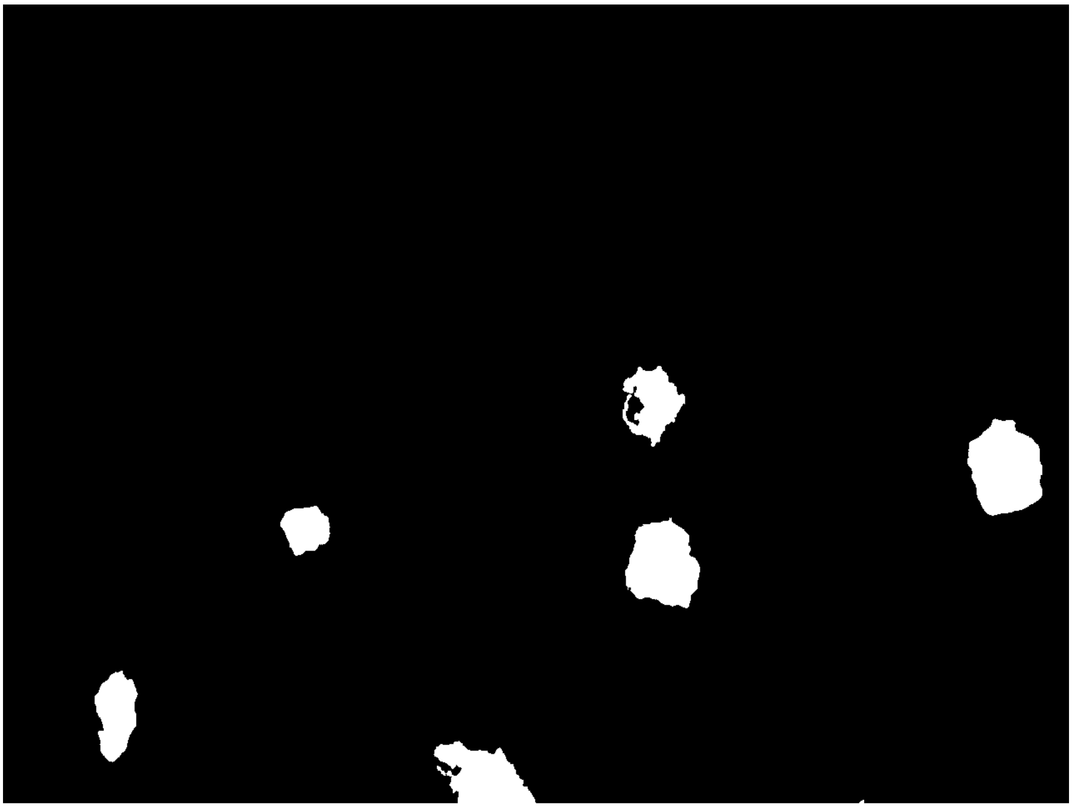


Figure 5-70 Flaw measurement detected by Watershed algorithm on same image

Based on the Watershed based algorithm's flaw measurements, the results outcome is displayed visually in figure 5.69 and is also displayed in tabular form, Table 5.10 showing individual lengths, area, intensity and eccentricity of each flaw, detected by the watershed algorithm.

Table 5-10 Watershed results showing flaw measurements of the defects detected

IPv1-Flaw	Mean Intensity	Length(mm)	Length(pix)	Area(mm)	Area(pix)	Eccentricity
1	61.34652748	1.053161512	658.225945	11.0352	6897	0.991310388
2	81.04939852	0.172893886	108.0586787	6.2512	3907	0.8974722
3	72.97797877	0.093258042	58.28627596	4.0688	2543	0.222829774
4	78.38183143	0.144138324	90.08645242	6.5872	4117	0.661905168
5	66.81519096	0.169183909	105.7399429	11.1856	6991	0.574092009
6	71.60310137	0.182179536	113.8622099	12.4848	7803	0.629688135

5.6.2 Morphological based algorithm's flaw measurements using same image

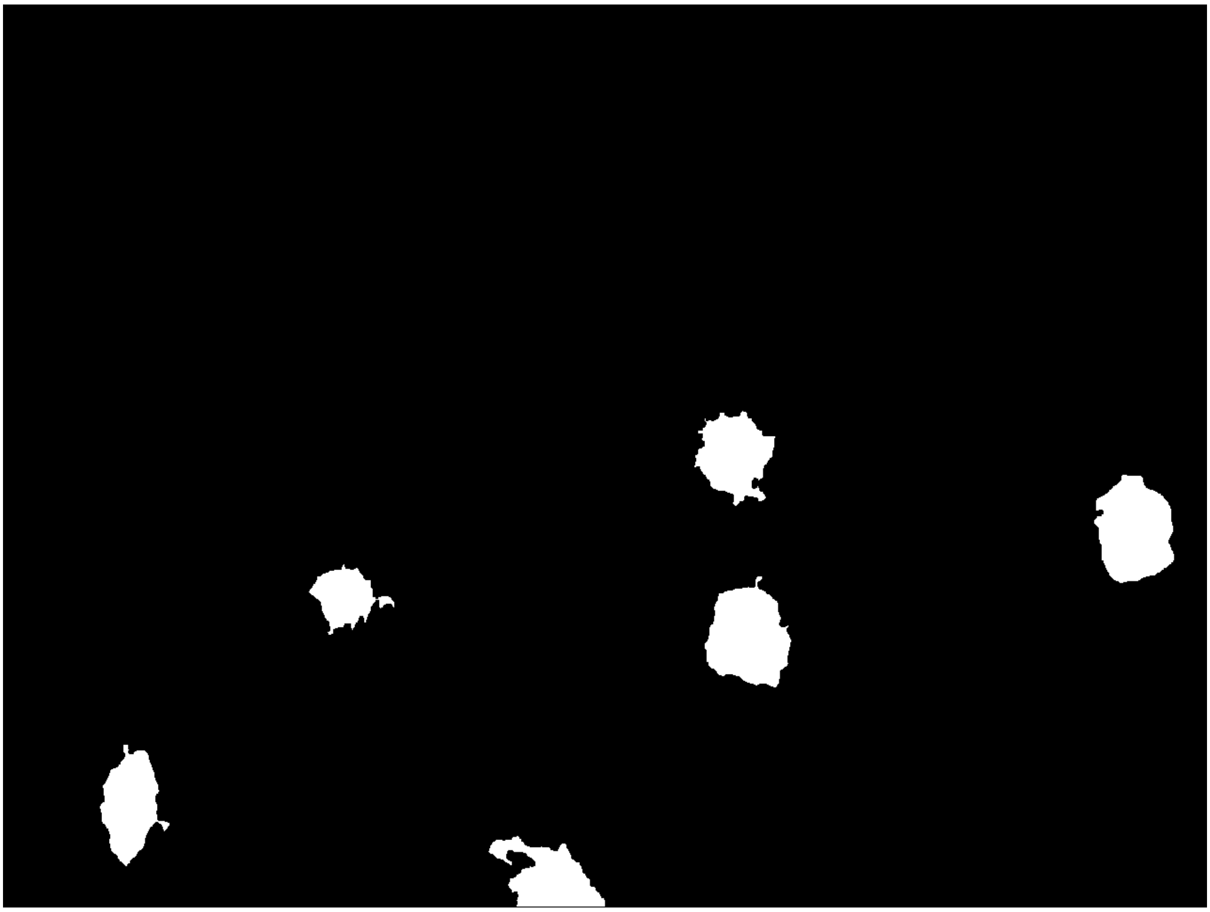


Figure 5-71 Flaw measurement detected by Morphological Method on same image

Based on the Morphological based algorithm's flaw measurements, the results outcome is displayed visually in figure 5.70 and is also displayed in tabular form, Table 5.11 showing individual lengths, area, intensity and eccentricity of each flaw, detected by the watershed algorithm.

Table 5-11 Morphological results showing flaw measurements of the defects detected

IPv2-Flaw	Mean Intensity	Length(mm)	Length(pix)	Area(mm)	Area(pix)	Eccentricity
1	90.08144	0.188316	117.6973	8.448	5280	0.863335
2	83.27575	0.11649	72.80651	5.2048	3253	0.511668
3	72.10783	0.189221	118.2634	8.072	5045	0.825841
4	88.92521	0.147452	92.15748	8.6	5375	0.534466
5	68.46525	0.17337	108.3563	11.648	7280	0.583798
6	70.37529	0.181242	113.2763	11.5536	7221	0.685359

5.6.3 FCM Gaussian based algorithm’s flaw measurements using same image

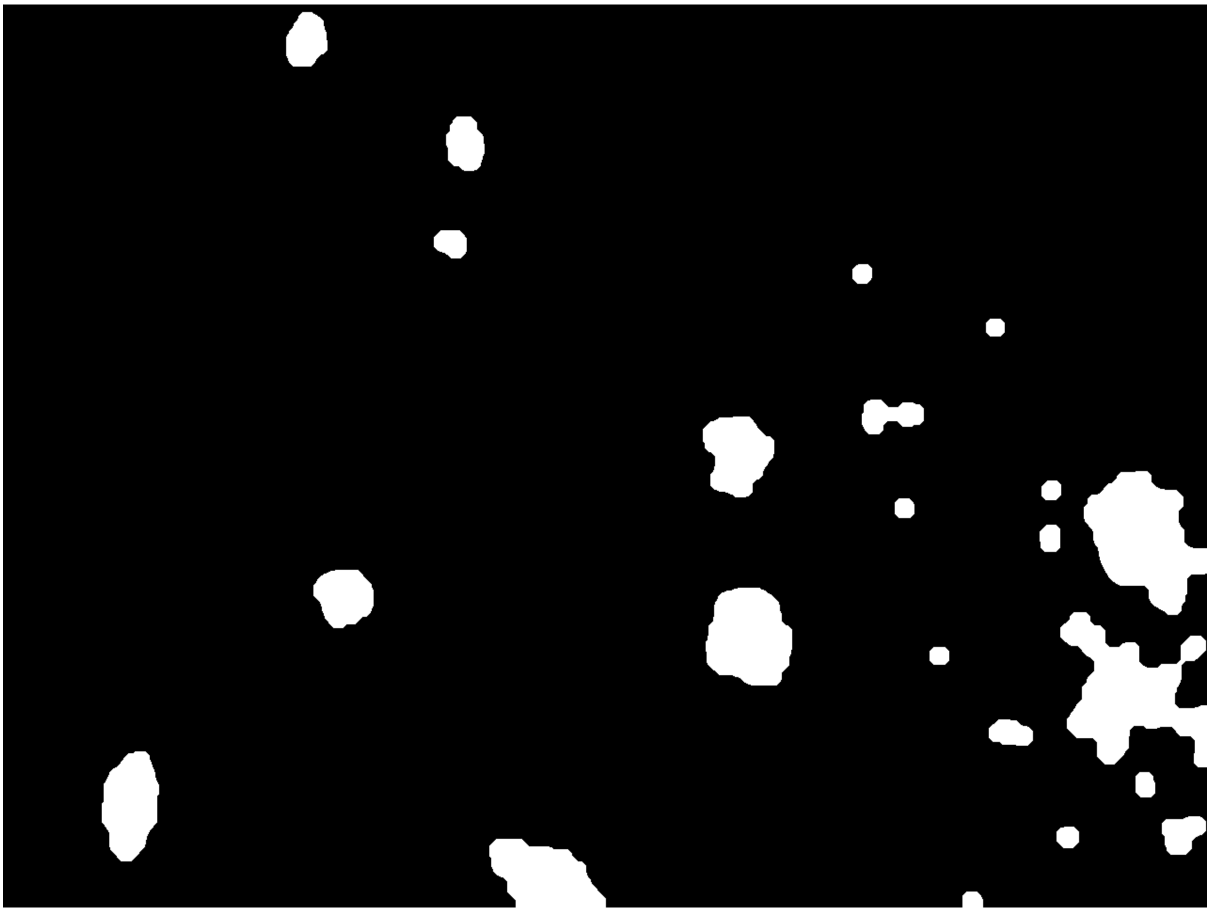


Figure 5-72 Flaw measurement detected by Gaussian based algorithm on same image

Based on the FCM Gaussian based algorithm’s flaw measurements, the results outcome is displayed visually in figure 5.71 and is also displayed in tabular form, Table 5.12 showing individual lengths, area, intensity and eccentricity of each flaw, detected by the watershed algorithm.

Table 5-12 Gaussian algorithm results showing flaw measurements of defects detected

IPv3-Flaw	Mean Intensity	Length(mm)	Length(pix)	Area(mm)	Area(pix)	Eccentricity
19	125	0.282	176	21	12954	0.6
20	89	0.250	156	19	12070	0.8
6	78	0.204	127	10	6330	0.9
1	90	0.186	116	8	5287	0.9
8	71	0.173	108	12	7602	0.5
7	85	0.141	88	8	4698	0.6
10	121	0.123	77	3	1643	0.9
3	80	0.103	64	5	3035	0.3
2	96	0.096	60	3	1988	0.7
5	91	0.096	60	3	1888	0.7
22	129	0.081	51	2	1412	0.7
15	125	0.079	50	2	1047	0.8
4	96	0.060	38	1	854	0.6
16	125	0.050	31	1	569	0.7
21	128	0.047	29	1	502	0.7
18	122	0.040	25	1	470	0.3
13	104	0.038	24	1	340	0.6
17	126	0.037	23	1	385	0.4
11	124	0.037	23	1	395	0.2
9	124	0.036	23	1	384	0.3
12	127	0.036	23	1	359	0.4
14	119	0.034	21	1	340	0.3

5.6.4 Ground truth's extracted flaw measurements of same image

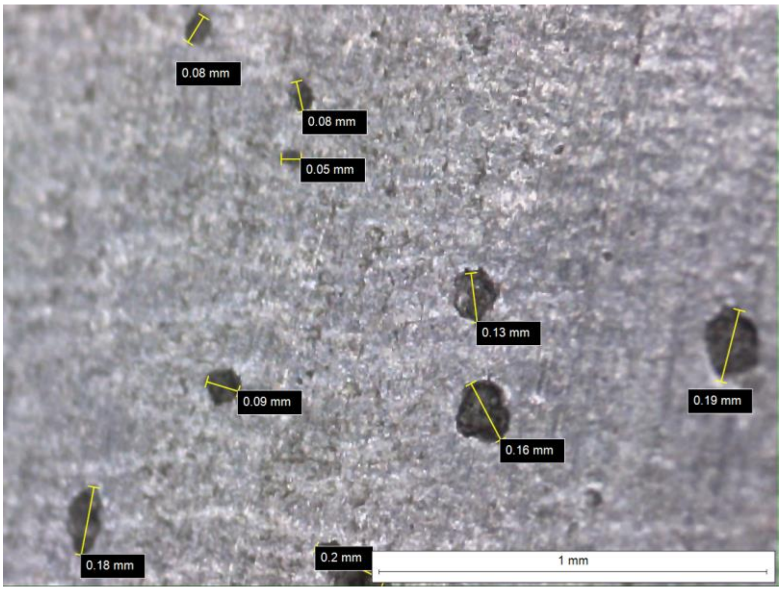


Figure 5-73 Ground truth's extracted flaw measurements of same test image

Based on the measured and marked flaw measurements, as shown in figure 5.72, of the ground truth of same image, the extracted outcome is displayed visually in figure 5.73 and is also displayed in tabular form, Table 5.13 showing individual lengths, area, intensity and eccentricity of each flaw, detected by the watershed algorithm. This process of extracting features from the labelled ground truth is done in order to compare its dimensions with the rest of the algorithms implemented.

Table 5-13 Dimensions of labelled ground truth to compare with the rest of the algorithms

Object Type	Unit of Measurement	ID	Value	Unit
Line	Length	23	0.2	mm
Line	Length	24	0.18	mm
Line	Length	25	0.09	mm
Line	Length	26	0.16	mm
Line	Length	27	0.19	mm
Line	Length	28	0.13	mm
Line	Length	29	0.05	mm
Line	Length	30	0.08	mm
Line	Length	31	0.08	mm

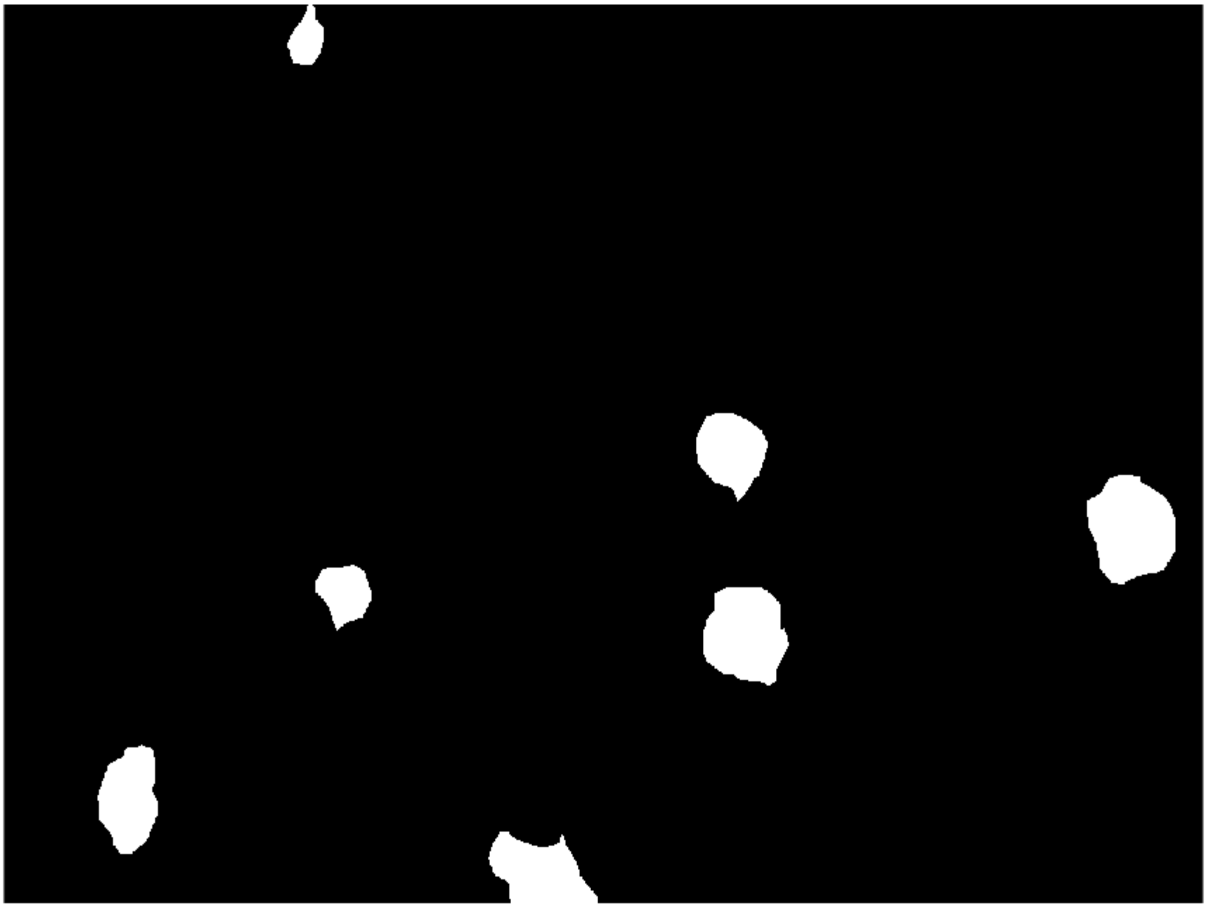


Figure 5-74 Ground truth of same test image

5.6.5 Conclusion

It can be seen from table 5.14 that the Morphological algorithm's measurements are more nearer to the labelled ground truth information. Watershed's results show an edge boundary problem for one of the flaws detected at the edge, otherwise it is closer to the actual values.

Table 5-14 Comparison of all unsupervised algorithms with the ground truth

Flaw	Watershed Length (mm)	Morphological Length (mm)	Gaussian Length-(mm)	Manual Length (mm)
1	1.053	0.189	0.282	0.2
2	0.182	0.188	0.250	0.19
3	0.173	0.181	0.204	0.18
4	0.169	0.173	0.186	0.16
5	0.144	0.147	0.173	0.13
6	0.093	0.116	0.141	0.09
7			0.123	0.08
8			0.103	0.08

This object at boundary problem can be solved by removing the black line at the bottom from the original images. When measuring for the ground truth, the Pits which have a more elliptical shape, are darker in colour and seem very close to 0.1mm, get selected as well, 0.8mm being the smallest value to be selected under such exceptions.

5.7 Evaluation with a state-of-the-art algorithm using same dataset

Results are based on the average performance on the full dataset, with same images, for each algorithm implemented. Visual outcomes are shown by using same image, B919-2016, as shown in figure 5.74. This image is used just to show main individual steps of each algorithm. The metrics used are the same ones already discussed in chapter 4 in much detail that also include the specific image segmentation metrics such as CV, RI, VOI, BE and GCE. Each subsection includes a table of the performance metrics applied on full dataset consisting of same images. Then it plots the result using confusion matrix and also displays main individual steps performed by each of the algorithm. Later, in the conclusion to this section, it shows a comparison table between all four of algorithm based on both, individual as well as average performances of these algorithms.

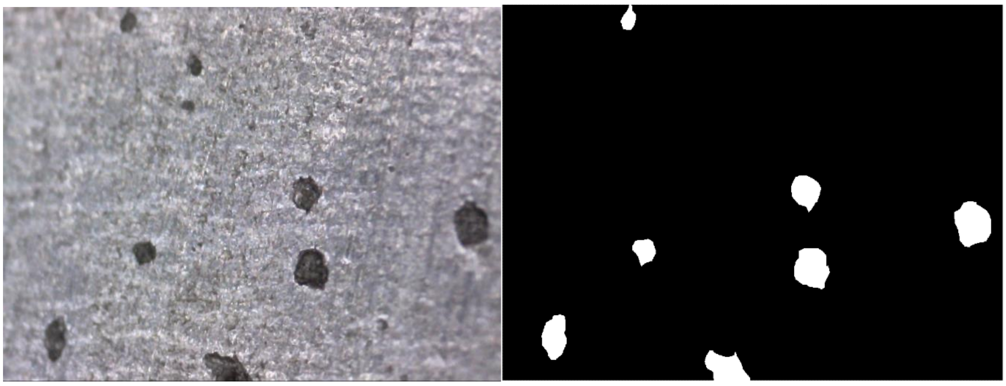


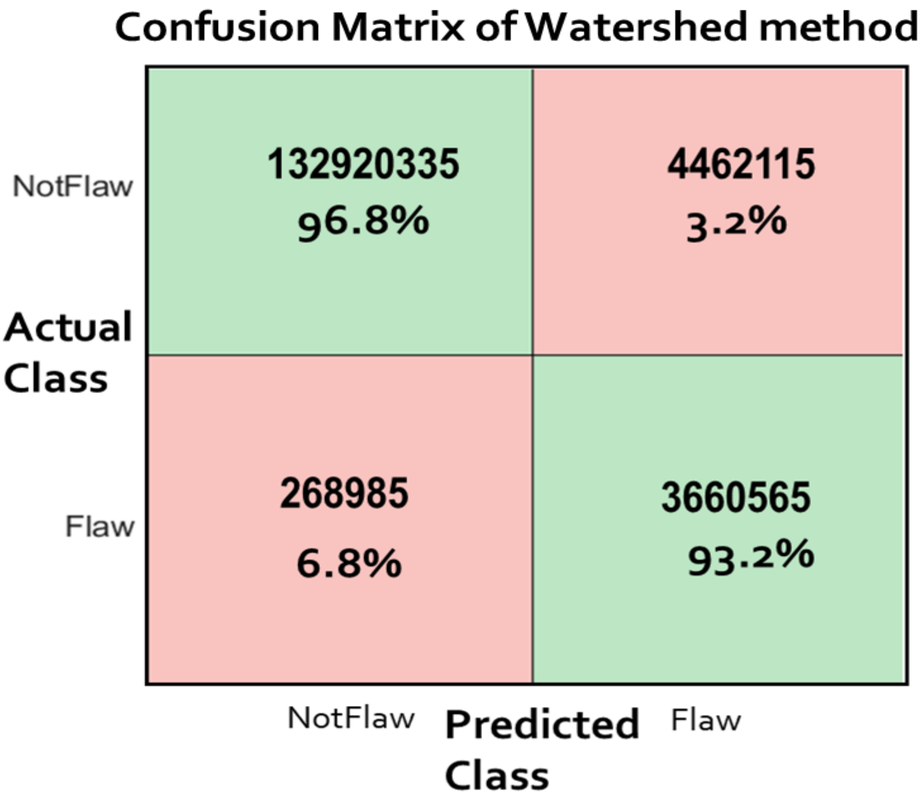
Figure 5-75 Test image & its ground truth to evaluate implemented unsupervised methods

5.7.1 Watershed performance results using same dataset

The performance results of watershed method on all data can be seen through a table showing all metrics as shown in Table 5.15 and can also be highlighted with a confusion matrix in figure 5.75. Individual visual results of the algorithm can be seen in figure 5.76.

Table 5-15 Watershed performance indicators results using same Dataset

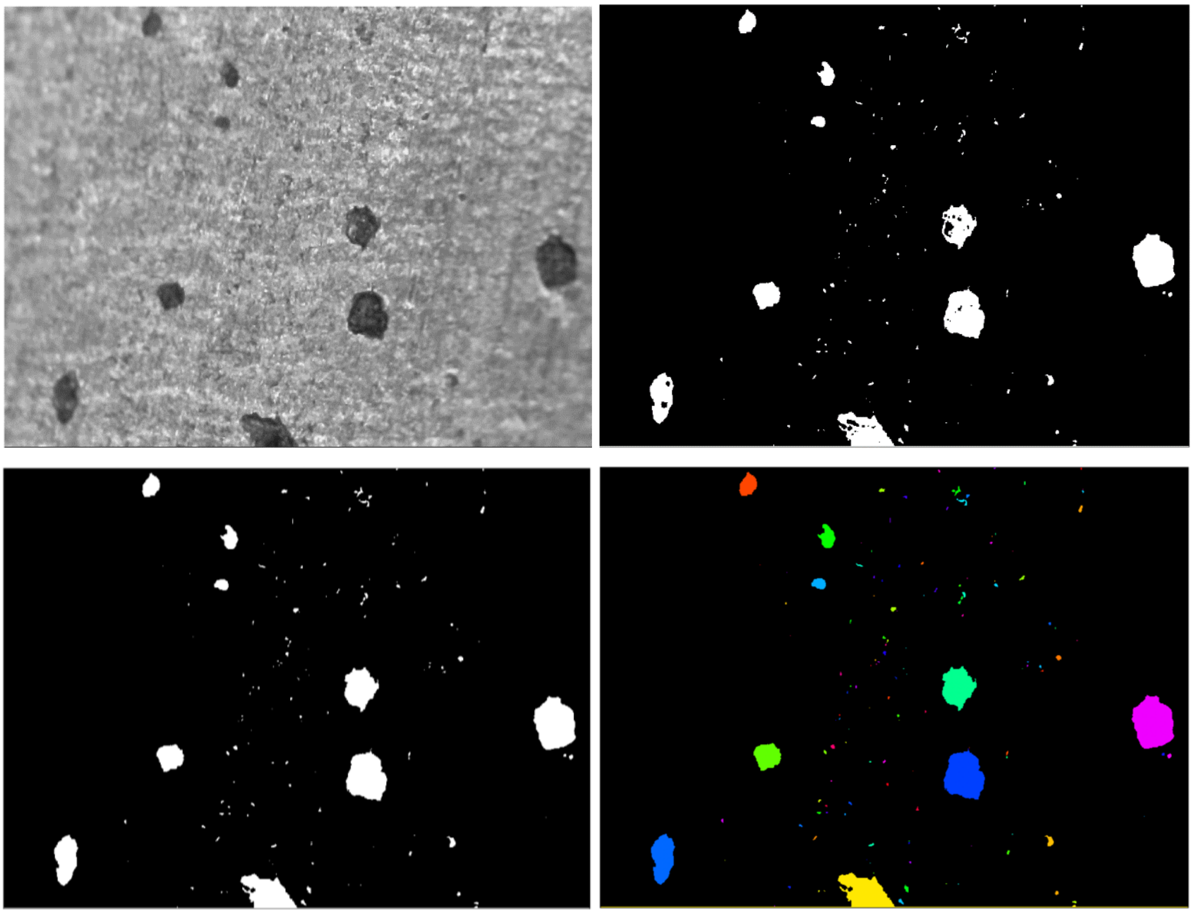
Metrics name	Watershed
True Positives ↑	3660565
True Negatives ↑	132920335
False Positives ↑	4462115
False Negatives ↑	268985
Recall/TPR ↑	0.931548142
TNR↑	0.967520488
Precision↑	0.450659758
Accuracy ↑	0.966520182
NPV↑	0.997980431
f1↑	0.607450239
RMSE↓	0.029672852
MCC↑	0.635104967
CV↑	0.436214387
RI↑	0.935925086
VOI↓	0.394679969
GCE↓	0.018750518
BE↓	68.57215118



Confusion Matrix of Watershed method

		Predicted Class	
		NotFlaw	Flaw
Actual Class	NotFlaw	132920335 96.8%	4462115 3.2%
	Flaw	268985 6.8%	3660565 93.2%

Figure 5-76 Watershed outcome of full dataset displayed by confusion matrix



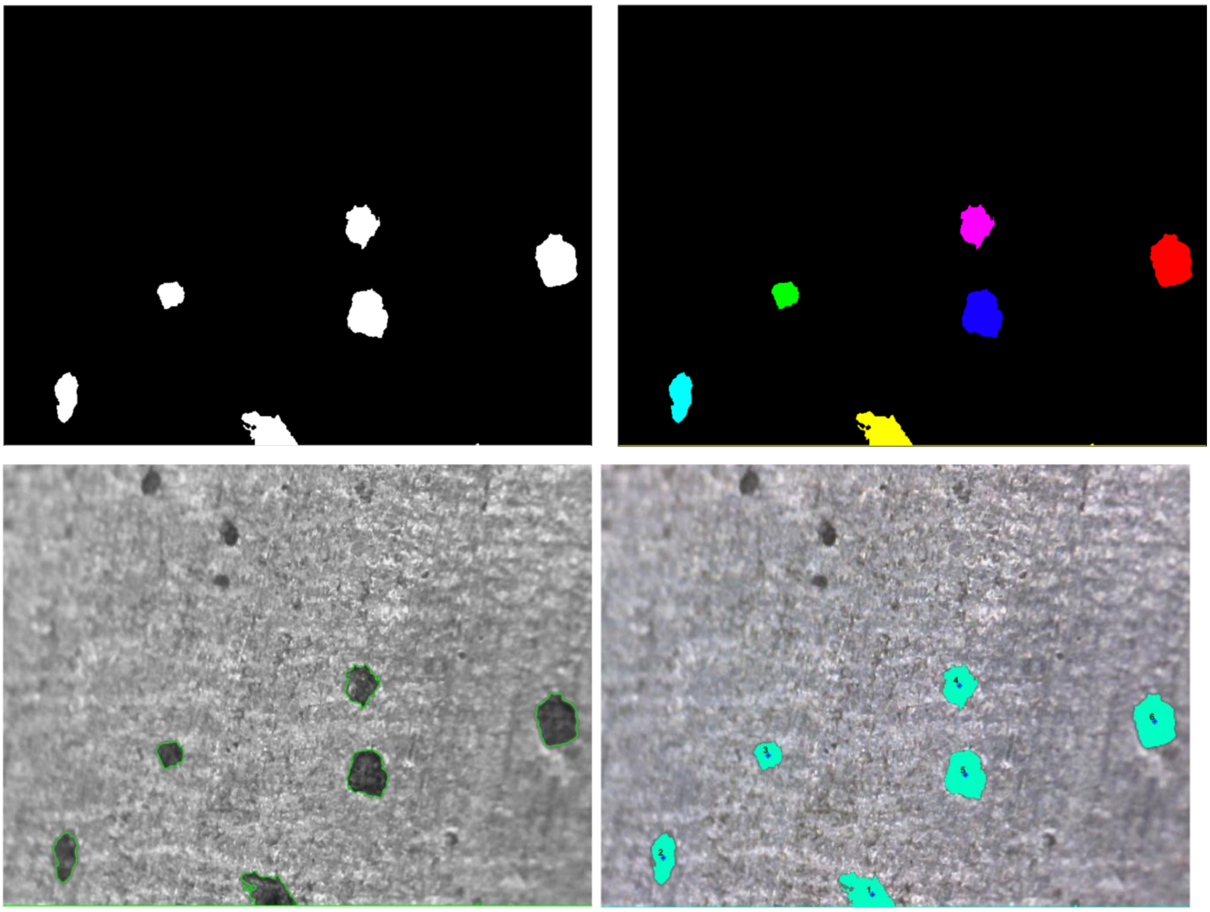


Figure 5-77 Watershed results displaying individual steps on same image

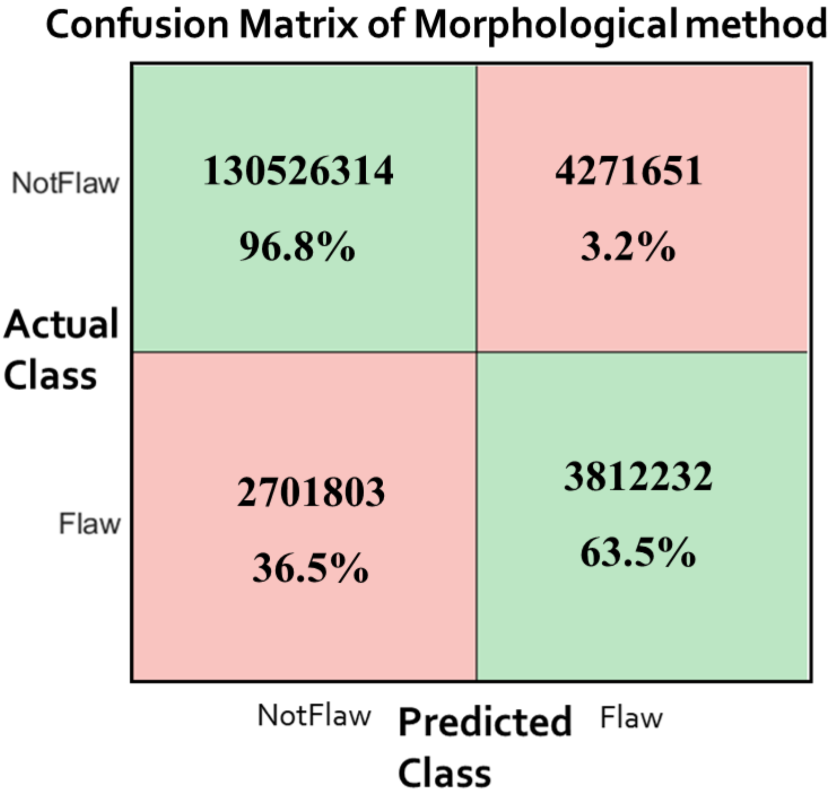
5.7.2 Morphological results displaying individual steps using same dataset

The performance results of morphological method on all data can be seen through a table showing all metrics as shown in Table 5.16 and can also be highlighted with a confusion matrix in figure 5.77. Individual visual results of the algorithm can be seen in figure 5.78.

Table 5-16 Morphological performance indicators results using same Dataset

Metrics name	Morphological
True Positives ↑	3812232
True Negatives ↑	130526314
False Positives ↑	4271651
False Negatives ↑	2701803
Recall/TPR ↑	0.634950793
Specificity/TNR ↑	0.968208186
Precision↑	0.496357341
Accuracy↑	0.950652075
NPV↑	0.979225284
f1↑	0.489944722

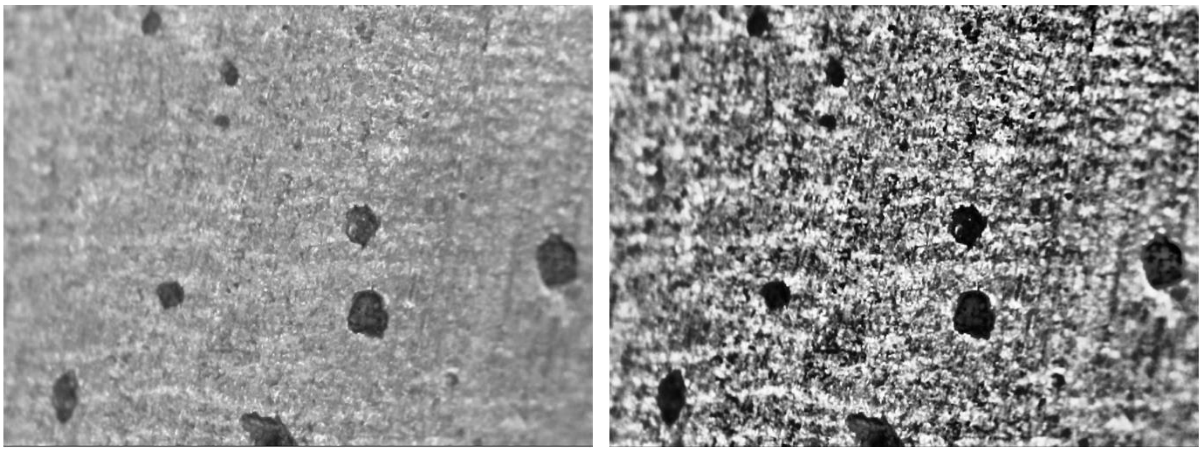
RMSE↓	0.032936835
MCC↑	0.518695071
CV↑	0.349932574
RI↑	0.924338159
VOI↓	0.410286651
GCE↓	0.030300853
BE↓	121.3262869

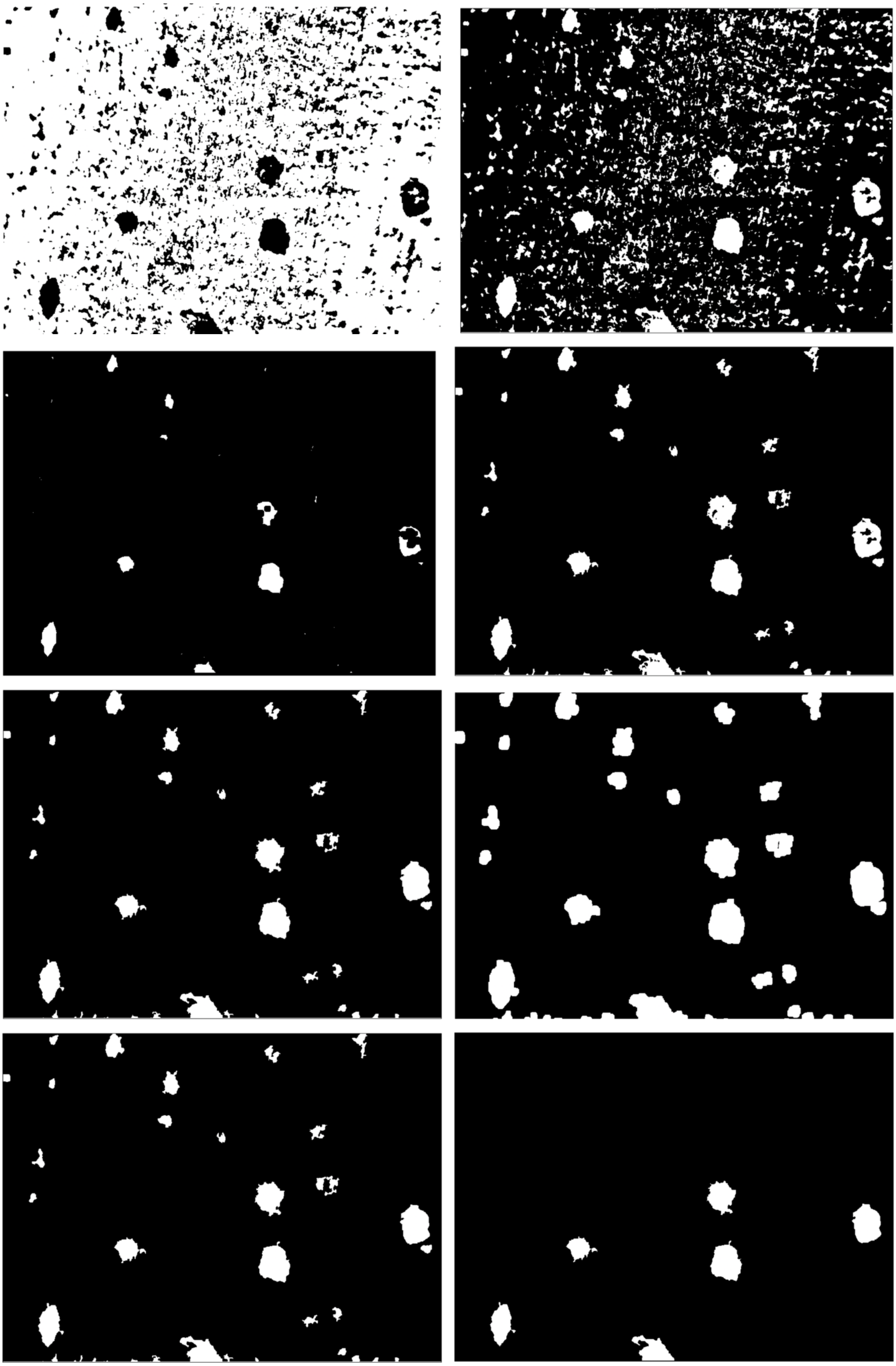


Confusion Matrix of Morphological method

Actual Class		Predicted Class	
		NotFlaw	Flaw
	NotFlaw	130526314 96.8%	4271651 3.2%
	Flaw	2701803 36.5%	3812232 63.5%

Figure 5-78 Morphological outcome of full dataset displayed by confusion matrix





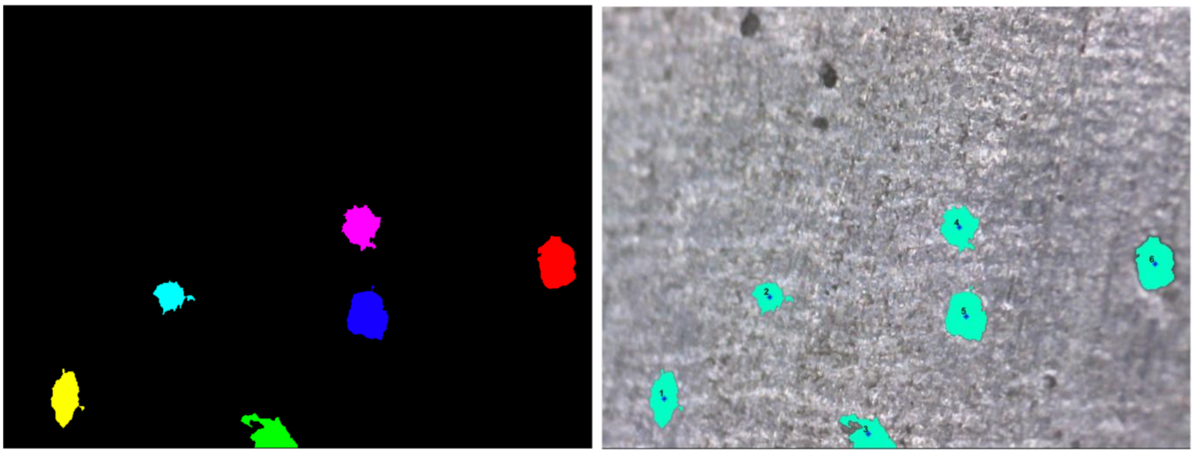


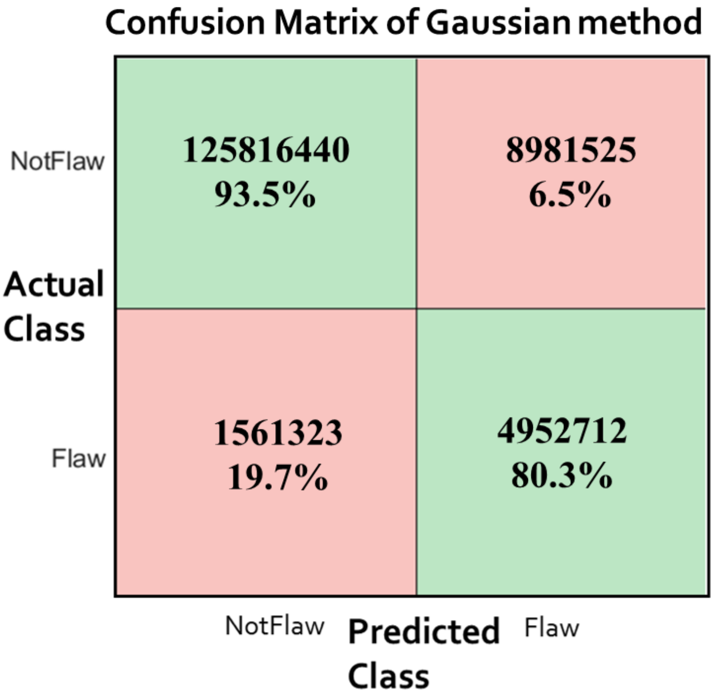
Figure 5-79 Morphological results displaying individual steps on same image

5.7.3 Gaussian based FCM clustering results using same dataset

The performance results of gaussian method on all data can be seen through a table showing all metrics as shown in Table 5.17 and can also be highlighted with a confusion matrix in figure 5.79. Individual visual results of the algorithm can be seen in figure 5.80.

Table 5-17 FCM Gaussian performance indicators results using same Dataset

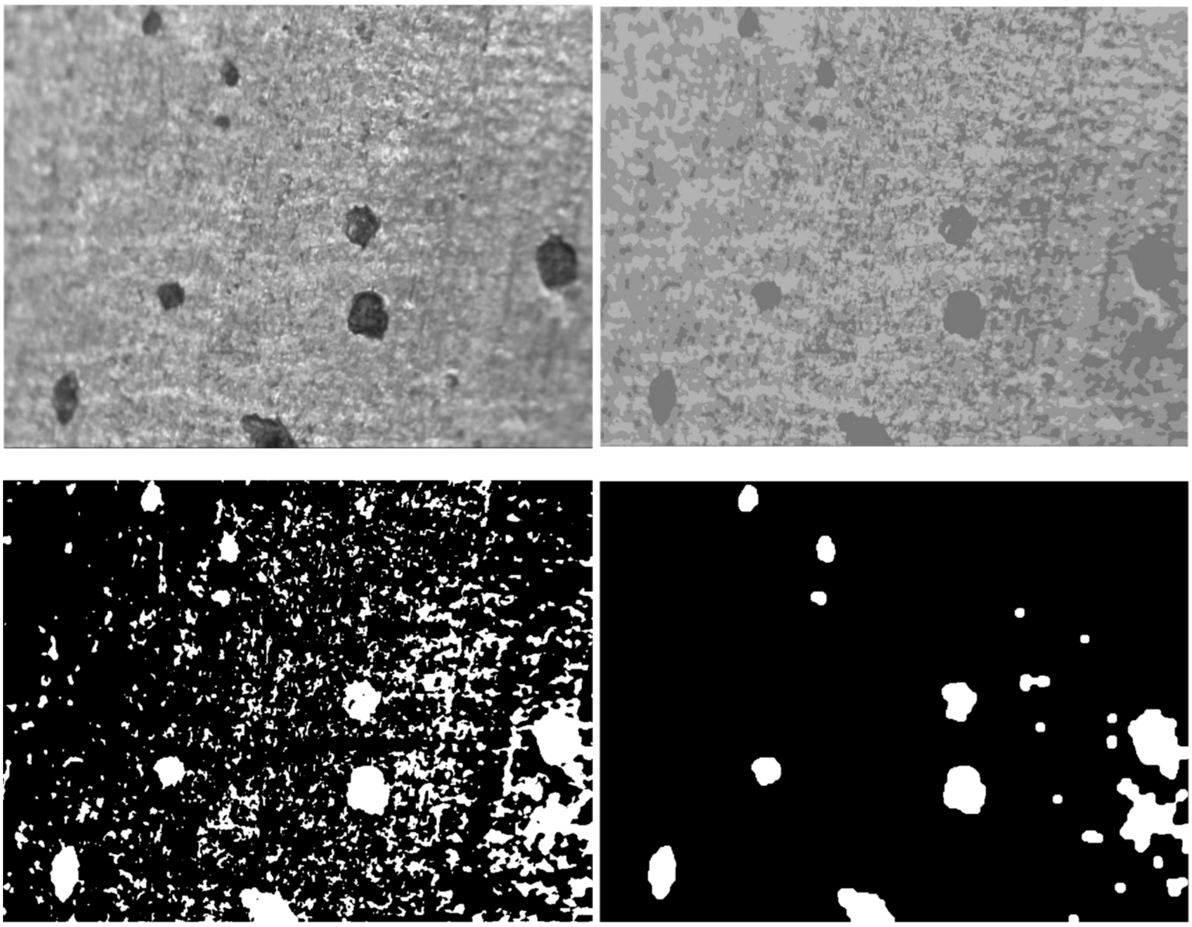
Metrics name	Gaussian
True Positives ↑	4952712
True Negatives ↑	125816440
False Positives ↑	8981525
False Negatives ↑	1561323
Recall/TPR ↑	0.80346251
Specificity/TNR ↑	0.934658029
Precision↑	0.43235867
Accuracy↑	0.925393116
NPV↑	0.987780845
f1↑	0.482997974
RMSE↓	0.06130606
MCC↑	0.508932789
CV↑	0.36220493
RI↑	0.874837156
VOI↓	0.58453533
GCE↓	0.028545977
BE↓	104.8260835



Confusion Matrix of Gaussian method

		Predicted Class	
		NotFlaw	Flaw
Actual Class	NotFlaw	125816440 93.5%	8981525 6.5%
	Flaw	1561323 19.7%	4952712 80.3%

Figure 5-80 FCM Gaussian outcome of full dataset displayed by confusion matrix



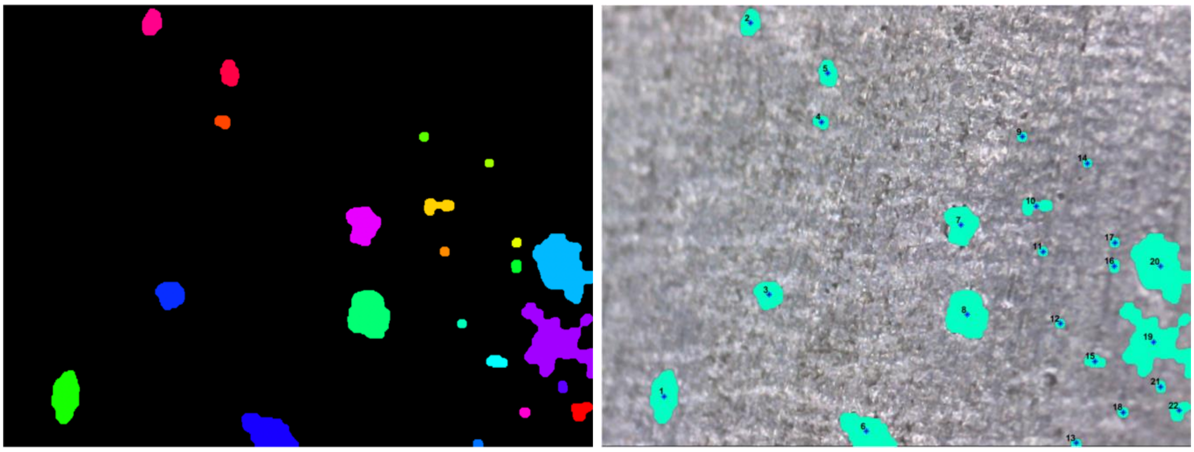


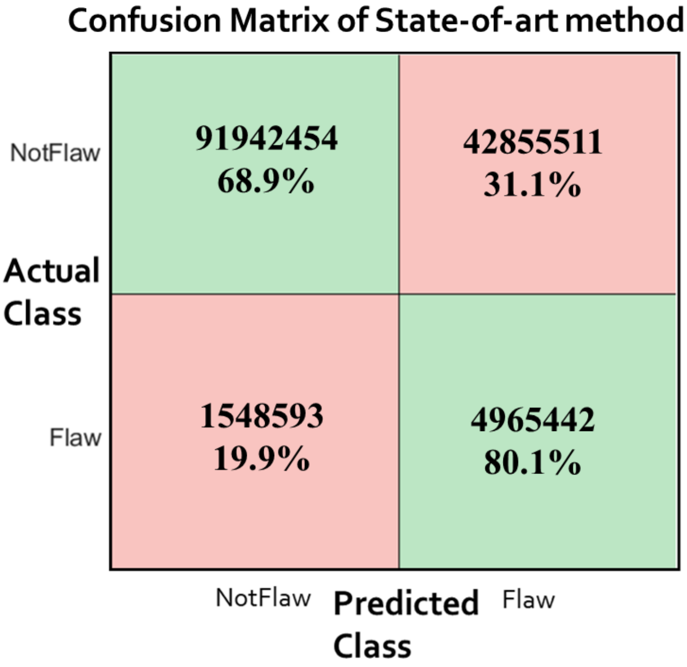
Figure 5-81 FCM Gaussian based results displaying individual steps on same image

5.7.4 State-of-the-art algorithm’s results applied on the same dataset

The algorithm being compared [144] is a state of the art super-pixel-based fast fuzzy c-means clustering algorithm for colour image segmentation. The performance results of state-of-the-art method on all data can be seen through a table showing all metrics as shown in Table 5.18 and can also be highlighted with a confusion matrix in figure 5.81. Individual visual results of the algorithm can be seen in figure 5.82.

Table 5-18 State-of-the-art performance indicators results using same Dataset

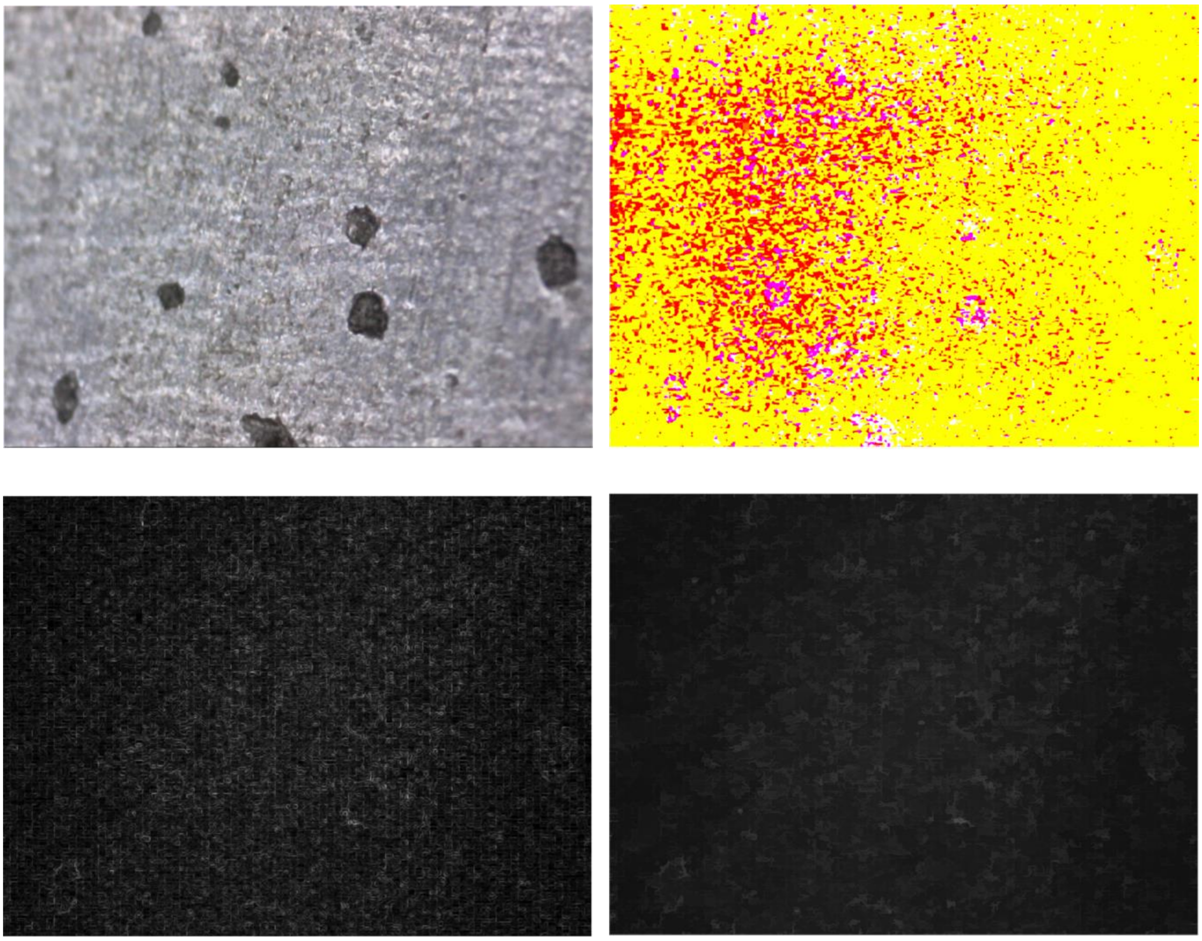
Metrics name	State-of-art
True Positives ↑	4965442
True Negatives ↑	91942454
False Positives ↑	42855511
False Negatives ↑	1548593
Recall/TPR ↑	0.801132191
Specificity/TNR ↑	0.688545564
Precision↑	0.249605223
Accuracy↑	0.685772588
NPV↑	0.985846752
f1↑	0.287644805
RMSE↓	0.296502208
MCC↑	0.295525069
CV↑	0.203155907
RI↑	0.662093678
VOI↓	1.028561891
GCE↓	0.051816341
BE↓	121.0335617

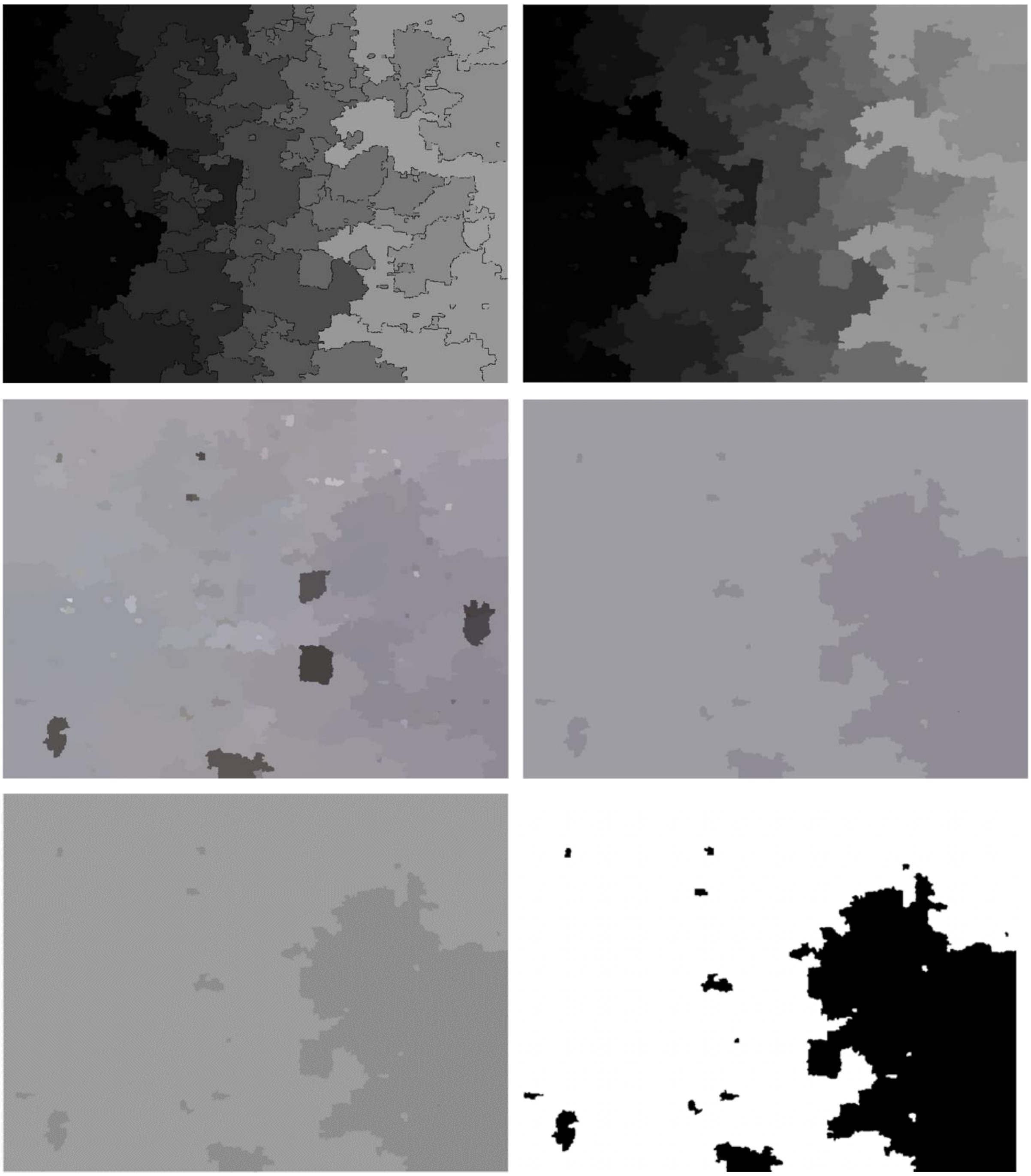


Confusion Matrix of State-of-art method

Actual Class \ Predicted Class	NotFlaw	Flaw
NotFlaw	91942454 68.9%	42855511 31.1%
Flaw	1548593 19.9%	4965442 80.1%

Figure 5-82 State-of-the-art outcome of full dataset displayed by confusion matrix





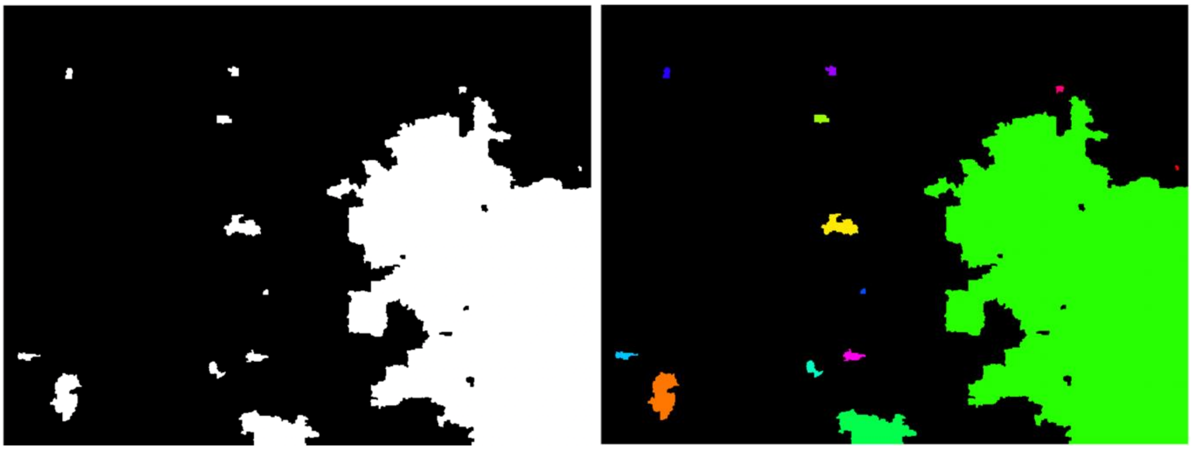


Figure 5-83 State-of-the-art algorithm results displaying individual steps on same image

5.7.5 Conclusion

To compare the three different algorithms applied for this research plus the state-of the art fourth algorithm, multiple metrics are used. Five of the specific segmentation indicators were also applied to evaluate the performance in depth. All the values, shown in table 5.20, are the average performance of each algorithm based on the performances of all individual images. But before that, performance of an individual, Image B919-2016 is shown in table 5.19. It is the same image used to show all the steps earlier in this section.

Table 5-19 Performance evaluation results of all four algorithms on same image

B919-2016	Metrics	Better	Watershed	Morphological	Gaussian	SOTA
1	TP	higher	29812	31068	34609	28956
2	TN	higher	1190002	1190062	1162505	915125
3	FP	lower	2446	2386	29943	277323
4	FN	lower	6540	5284	1743	7396
5	TPR	higher	0.820	0.855	0.952	0.797
6	TNR	higher	0.998	0.998	0.975	0.767
7	PPV	higher	0.924	0.929	0.536	0.095
8	Acc	higher	0.993	0.994	0.974	0.768
9	NPV	higher	0.995	0.996	0.999	0.992
10	FPR	lower	0.002	0.002	0.025	0.233
11	FDR	lower	0.076	0.071	0.464	0.905
12	FNR	lower	0.180	0.145	0.048	0.203
13	f1	higher	0.869	0.890	0.686	0.169

14	RMSE	lower	0.003	0.002	0.023	0.220
15	MCC	higher	0.867	0.888	0.704	0.221
16	CV	higher	0.768	0.802	0.522	0.092
17	RI	higher	0.986	0.988	0.951	0.645
18	VOI	lower	0.109	0.097	0.287	1.014
19	GCE	lower	0.010	0.009	0.011	0.046
20	BE	lower	62.283	27.424	44.364	91.958
21	Time(sec)	lower	12.793	28.836	29.354	16.018

Table 5-20 Performance evaluation results of all four algorithms on same dataset

Metrics#	Metrics name	Watershed	Morphological	Gaussian	State-of-art
1	TP↑	31831	33150	43067	43178
2	TN↑	1155829	1135011	1094056	799500
3	FP↑	38801	37145	78100	372657
4	FN↑	2339	23494	13577	13466
5	Recall↑	0.932	0.635	0.803	0.801
6	TNR↑	0.968	0.968	0.935	0.689
7	Precision↑	0.451	0.496	0.432	0.250
8	Accuracy↑	0.967	0.951	0.925	0.686
9	NPV↑	0.998	0.979	0.988	0.986
10	FPR↓	0.032	0.032	0.065	0.311
11	FDR↓	0.549	0.504	0.568	0.750
12	FNR↓	0.068	0.365	0.197	0.199
13	f1↑	0.607	0.490	0.483	0.288
14	RMSE↓	0.030	0.033	0.061	0.297
15	MCC↑	0.635	0.519	0.509	0.296
16	CV↑	0.436	0.350	0.362	0.203
17	RI↑	0.936	0.924	0.875	0.662
18	VOI↓	0.395	0.410	0.585	1.029
19	GCE↓	0.019	0.030	0.029	0.052

20	BE↓	68.572	121.326	104.826	121.034
21	Time(sec)↓	20.71	32.84	22.28	16.00

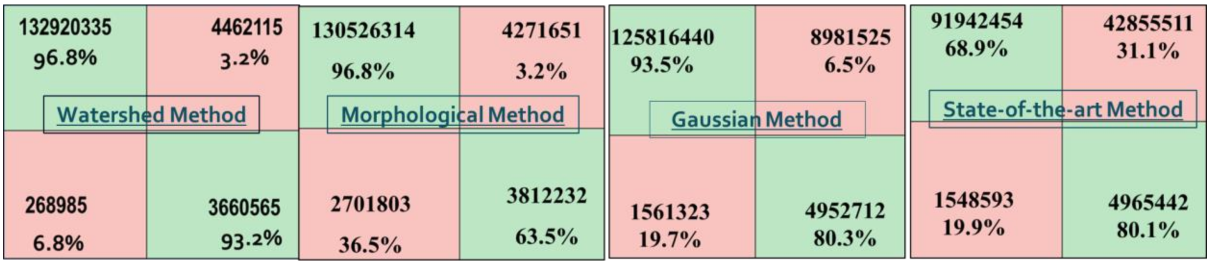
Watershed based algorithm showing best of the performances based on 8 of the metrics, which is highest number of best performing metrics. If just the 5 major metrics specifically used for image segmentation evaluations are considered, then also it can be seen that it gives similar results as above. Out of five metrics applied; Watershed produces best results for two of them and one same as Gaussian. Morphological shows best results for two of the indicators as well and matches watershed while Gaussian shows best for one of them and matching one. But state-of-the-art is still behind in producing as good results for this data set application.

5.8 Chapter Summary

Three versions of image segmentation have been implemented in this chapter, which are watershed-based, Gaussian-based FCM clustering and morphological-based filtering, to investigate and make comparisons between them. Implementation of all algorithms, except the initial trials, follow three major steps which include; pre-processing, segmentation and feature extraction. The outcomes consist of the resultant segmented image along with two excel files. One has the flaw measurements such as flaw length and area and the other has the performance measures such as accuracy, recall and precision etc.

The confusion matrix of all four methods are shown side by side in Table 5.21. The overall performance, using same dataset, of all four methods is illustrated in figure 5.83 using a bar chart. It is the visual representation of the tabular results of table 5.20. It can be seen by the bar chart that both the watershed and morphological algorithm give better results. They show best results based on almost all indicators such as recall, accuracy, TNR, NPV, f1, R1, MCC, CV, VOI, GCE and RMSE.

Table 5-21 Confusion matrix of all four methods side-by-side performed on all data



Watershed Method		Morphological Method		Gaussian Method		State-of-the-art Method	
132920335 96.8%	4462115 3.2%	130526314 96.8%	4271651 3.2%	125816440 93.5%	8981525 6.5%	91942454 68.9%	42855511 31.1%
268985 6.8%	3660565 93.2%	2701803 36.5%	3812232 63.5%	1561323 19.7%	4952712 80.3%	1548593 19.9%	4965442 80.1%

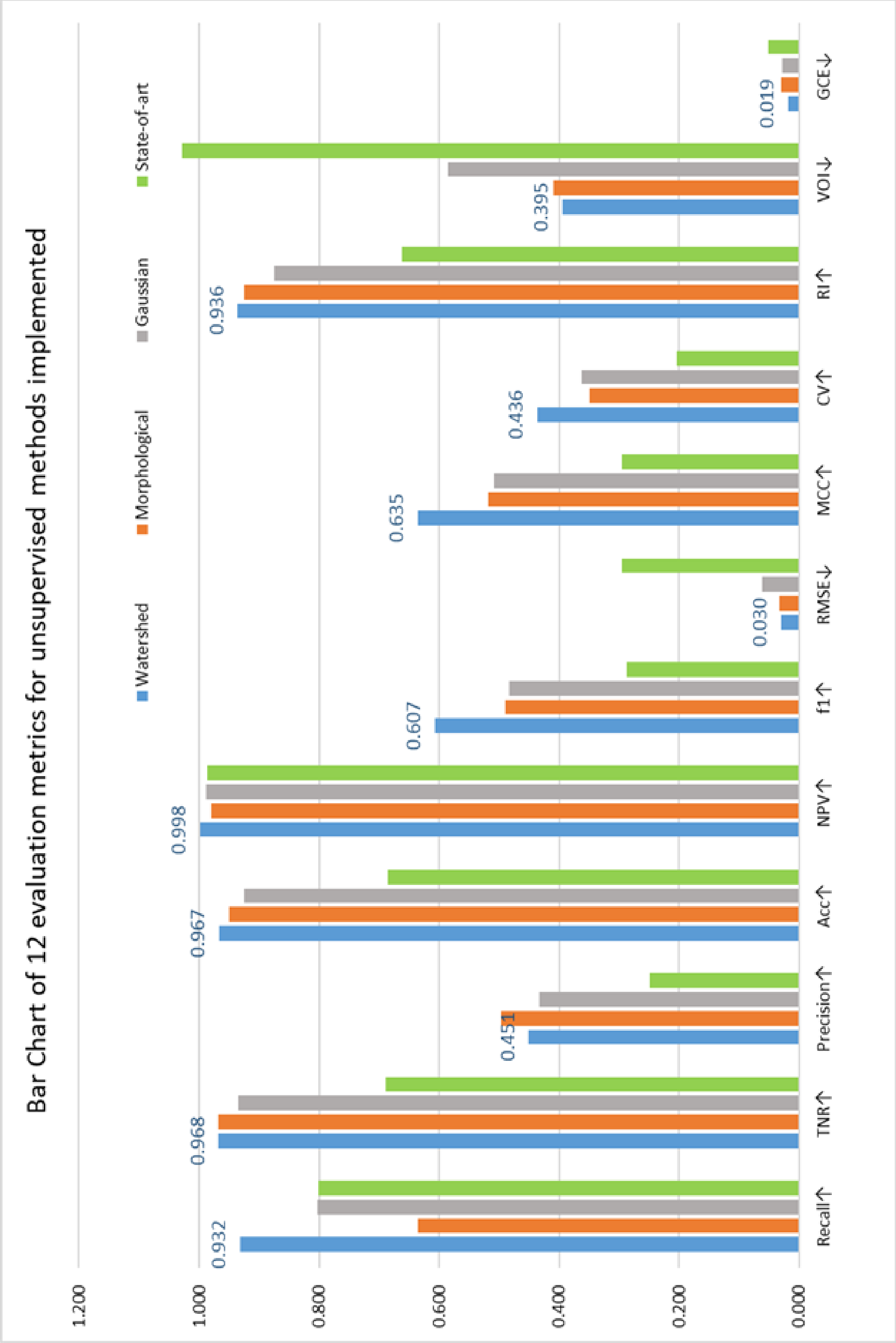


Figure 5-84 Evaluation of all unsupervised methods implemented shown by bar chart

Based on the flaw measurements, Morphological algorithm's measurements are more nearer to the labelled ground truth information. Watershed's results show an edge boundary problem for the flaws detected at the edge, otherwise it is closer to the actual values.

Based on the performance metrics illustrated in figure 5.83, it can be seen that the watershed gives better results in comparison to the rest of the algorithms implemented. Watershed-based algorithm is a simple yet effective method, which shows high performances such as 95.2% accuracy, 55% precision, f1 score 56%, high probabilistic rand Index (PRI) 91.7%, CV is 42.8% and VOI as low as 41.08%, Global consistency error (GCE) as low as 2.6%. The other segmentation method, Morphological-based algorithm shows performance comparable to above such as 95.1% accuracy, 49.6% precision, f1 score 49%, high probabilistic rand Index (PRI) 92.4%, CV is 35% and VOI as low as 41%, Global consistency error (GCE) as low as 3%. Both of them have very similar performance results.

Comparing to the state of the art method used by [144], it can be seen that the results produced in this research, work more accurately for this application data set. Image segmentation has been able to work on both kinds of defects, pits and cracks. For classification purposes, machine learning and deep learning has been implemented, which is discussed in the next chapters in detail.

Chapter 6
Supervised Machine Learning implementation, comparison & evaluation

This chapter provides an overview of the design steps that were necessary to set up supervised classification and begin training for traditional machine learning. This includes discussion of classifier algorithms, spatial and textural feature extractions such as local binary patterns, gradients and other parameter that were applied for the implementation of machine learning models. It then discusses experimental results based on the metrics for comparing and analysing different classifier’s performance

6.1 INTRODUCTION

This chapter discusses supervised learning methods based on traditional machine learning. It lays the foundation to the third main contribution to this research combined with the next chapter on deep learning methods. After implementing unsupervised methods in the previous chapter, next main task is to explore the possibility of implementing an intelligent system on the same data. Especially for the project objective of classifying pits and cracks. Machine Learning (ML) are typically used in problems that maybe couched in terms of classification or forecasting. It teaches machines tasks that are easy for humans to perform but hard for them to formalise how it was performed such as the project problem of classification. This chapter consists of the following:

- Starts with discussing some of the key factors to be considered for applying machine learning (section 6.2)
- Then it shows machine learning initial trials based on the basic RGB feature extraction (section 6.3)
- Discusses three different settings for training the traditional machine learning models. For each of the settings, it explains the feature extracted such as local binary pattern and gradients, then the results of the implementation (section 6.4- 6.6)
- In the end, summary of all the machine learning method's results is concluded with visual illustration (section 6.7)

As researched in chapter 2, machine learning allows machines to learn from experience by finding natural patterns as opposed to being explicitly programmed [145]. These natural trends in the data produces insight and helps to make better decisions and predictions. So, as the number of samples available for learning increases, the algorithms adapt to improve their performance. It is ideal for situations when there is no existing equation or formula, for a complex scenario which deals with massive amount of data and loads of variables.

There are two kinds of machine learning techniques that are used, one is unsupervised learning and the other is supervised learning. Unsupervised is considered useful when the data needs to be explored but doesn't have a specific goal so no output data is available. The advantage of using this technique is that new things can be learnt when they are not known in advance. They can also be used as a pre-processing step for supervised learning. For example, apply clustering to get some features and then use them as an added feature input for the supervised learning model. Most unsupervised learning techniques are a form of cluster analysis where, based on some similarity or shared characteristic, data is partitioned into groups. Some of the commonly used algorithms that are based on clustering include k-means, hierarchical clustering, Gaussian mixture models, hidden Markov models, self-organizing maps, fuzzy c-means clustering, and subtractive clustering.

On the other hand, supervised learning takes a known set of inputs and their outputs (known purpose) to train a model that can make predictions. It is to develop a predictive model based on evidence in the presence of uncertainty to generate reasonable predictions for the response to new data. The foremost difference between them being that the supervised learning has a known purpose and requires output data. All supervised learning techniques are a form of classification or regression. Regression predicts continuous responses while classification is used to predict discrete responses like classifying data into categories. As in the case of the project, the purpose is known hence supervised classification has been considered for further evaluation and implementations. Some of the commonly used supervised classification algorithms include support vector machine (SVM), boosted and bagged decision trees, k-nearest neighbour, Naive Bayes, discriminant analysis, logistic regression, and neural networks.

There is no best method, because different case situations demand different algorithms to be considered. Hence, finding the right algorithm is a core part of the implementation which is partly based on trial and error. It's a trade-off between certain factors like speed of training, memory usage and predictive accuracy on new data. After choosing the model, the next step is to extract different useful features and then evaluate the performance measures of the combination of such implementations.

6.2 IMPLEMENTATION CONSIDERATIONS FOR MACHINE LEARNING

In general there are two steps in building a classifier: first is the training and second is the testing (or classification). These steps can be further broken down into sub-steps.

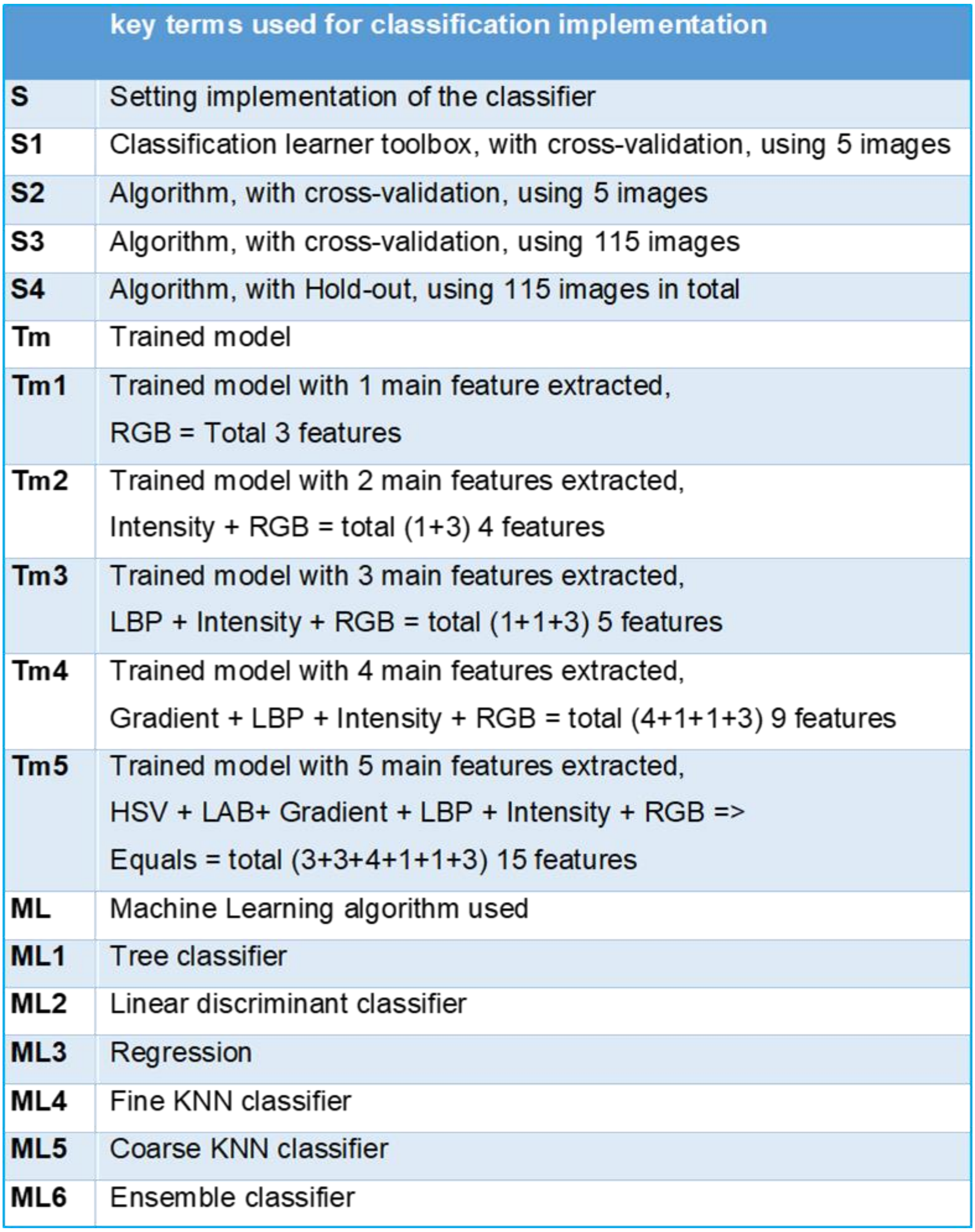
For Training: 1. Pre-processing: Process the data so it is in a suitable form, 2. Feature extraction: Reduce the amount of data by extracting relevant information, usually results in a vector of scalar values. 3. Model Estimation: From the finite set of feature vectors, need to estimate a model (usually statistical) for each class of the training data.

For Testing: 1. Pre-processing: 2. Feature extraction: (both steps are same as above) 3. Classification: Compare feature vectors to the various models and find the closest match. One can match the feature vectors obtained in training set.

A typical machine learning involves the following steps to be performed for classification:

- A matrix of observations and features relevant to the problem at hand.
- Preparing the dataset for processing.
- Splitting the dataset into training and testing sets.
- Selecting and training the classifier using the training set.
- Test the prediction accuracy of the trained classifier using the testing set.
- Evaluate the accuracy of the classifier by using will defined metrics.

Table 6-1 Refers to all the key terms that have been used for classification implementation



	key terms used for classification implementation
S	Setting implementation of the classifier
S1	Classification learner toolbox, with cross-validation, using 5 images
S2	Algorithm, with cross-validation, using 5 images
S3	Algorithm, with cross-validation, using 115 images
S4	Algorithm, with Hold-out, using 115 images in total
Tm	Trained model
Tm1	Trained model with 1 main feature extracted, RGB = Total 3 features
Tm2	Trained model with 2 main features extracted, Intensity + RGB = total (1+3) 4 features
Tm3	Trained model with 3 main features extracted, LBP + Intensity + RGB = total (1+1+3) 5 features
Tm4	Trained model with 4 main features extracted, Gradient + LBP + Intensity + RGB = total (4+1+1+3) 9 features
Tm5	Trained model with 5 main features extracted, HSV + LAB+ Gradient + LBP + Intensity + RGB => Equals = total (3+3+4+1+1+3) 15 features
ML	Machine Learning algorithm used
ML1	Tree classifier
ML2	Linear discriminant classifier
ML3	Regression
ML4	Fine KNN classifier
ML5	Coarse KNN classifier
ML6	Ensemble classifier

Table 6.1 refers to all the short key terms used in this chapter for ease of labelling different experimental setups. The performance of each image, with each pixel accounted for is stored. The table 6.2 shows individual performance results of five test images by applying important metrics. Then the average of all measures for all images are taken, for comparison with the other classifiers. So the average values of set s1tm1ml1 with s1tm1ml2 are considered, to compare two machine learning classifier performance. Similarly, set s1tm1ml1 with s1tm2ml1 values are compared to show the performance difference, with an additional feature extracted for training with same classifiers. In the next sections, only the average values will be discussed unless mentioned otherwise.

Table 6-2 Individual performance results of five test images by applying important metrics

Filename	Better	A186 - 2017	A190 - 2017	A788 - 2017	A825 - 2017	B919 - 2016
TP	higher	2997	5797	10214	19915	1391
TN	higher	1129374	1058494	1074802	938913	1166646
FP	lower	64129	85219	64740	102140	25802
FN	lower	32300	79290	79044	167832	34961
TPR/Recall/Sensitivity	higher	0.085	0.068	0.114	0.106	0.038
TNR/Specificity	higher	0.946	0.925	0.943	0.902	0.978
PPV/Precision	higher	0.045	0.064	0.136	0.163	0.051
Acc/Accuracy	higher	0.922	0.866	0.883	0.780	0.951
NPV	higher	0.972	0.93	0.931	0.848	0.971
FPR/Fallout/(1-TNR)	lower	0.054	0.075	0.057	0.098	0.022
FDR	lower	0.955	0.936	0.864	0.837	0.949
FNR/Miss-rate	lower	0.915	0.932	0.886	0.894	0.962
f1	higher	0.059	0.066	0.124	0.129	0.044
RMSE	lower	0.026	0.005	0.012	0.053	0.007
MCC	higher	0.023	0.006	0.062	0.010	0.019
CV	higher	0.03	0.034	0.066	0.069	0.022
RI	higher	0.852	0.761	0.790	0.640	0.905
VOI	lower	1.136	1.665	1.454	2.515	0.624
GCE	lower	0.054	0.126	0.118	0.188	0.044
BE	lower	94.809	94.652	55.954	50.924	89.916
Time(sec)	lower	0.892	0.958	1.070	1.071	1.575

6.2.1 Reason to compare Classifiers

It is difficult to get to the right model, as each model has its own strengths and weaknesses in a given scenario. There is no straight forward way of determining which model should be used without grossly overgeneralising the considerations. Choosing a data classification model involves to have a solid understanding of what needs to be accomplished keeping the perspective case study in mind. For example, questions like how much data is present, is it continuous, how much storage is required etc. Once all this is known, then the strengths of various models can be looked into. There are a few generic rules of thumb that can help choose the best classification model, but these might just be helpful for initial research. Mostly trial and error is required to attain the right kind of balance among complexity, performance, and accuracy, especially when dealing with large amount of data as a small variance in either performance or accuracy can have a large impact. At the same time, it is important to avoid overfitting a model. The reason being that if the model is super tightly fit with the training data, then when the algorithm faces new set of data, it is not able to cope and produces large errors.

6.2.2 Methods implemented to avoid overfitting

The best way to avoid overfitting is to have a large diverse training set capturing as many learning scenarios as possible. Regularisation method can be tried on the data to correct it. It helps the model from depending too heavily on individual data inputs and becoming too rigid. The lambda value in the objective function determines the strength of overfitting. It can take some time to find the best value of lambda. If it is zero then it means it is not correcting itself for overfitting at all but if the lambda value is too large then it will more likely be a training set fit.

Cross-validation method is generally used to prevent over-fitting because it does not use all the data to train a model. It is a model assessment technique, used to evaluate a machine learning algorithm's performance, when making predictions on new datasets it has not been trained on. This is done by partitioning a dataset and using a subset to train the algorithm and the remaining data for testing. Common techniques of cross validation include k-fold, holdout, and re-substitution.

6.2.3 Choose between Binary vs Multi-class problem

Another point to consider is whether the specific problem has a solution in terms of binary or multi-class. Multi-class classification is generally more challenging than binary as it needs a more complicated model. There are some algorithms which perform more efficiently for binary classification problems like logistic regression. Hence, it is important to know whether it can be solved by simple binary classification to avoid using complex models and computations.

6.2.4 Classifier Models implemented

A useful tool provided by MATLAB is the statistics and Machine Learning Toolbox, which includes classification learner app. It makes it easy to compare between the different models especially for the initial research stage. These classifiers can primarily be divided into seven categories; Naive Bayes Classification, Discriminant Analysis, Ensembles, Decision Trees, Nearest Neighbours, Support Vector Machines (SVM) and Neural Networks. The following sections describe some of the most common classifier models.

6.2.5 Features extracted for machine learning classifiers

Each column in the input training set that is fed into a model, represents a feature extracted from an image. The features that were extracted for the implementations are discussed in their respective experimental settings. The new added feature is explained in the section where it is added for the first time.

6.2.6 Datasets

As discussed earlier, supervised algorithms require response data i.e. manually-annotated ground truth to be fed into the classifier models. This is the challenging part as it requires massive amounts of correctly labelled data. For this research, flaw images were collected as discussed in chapter 4. Total of around 3000 images were collected but many of them had to be discarded due to the bad quality of the images, which includes blurriness and bad lighting. As mentioned before, the data collection process was gradually improved during the course which has helped to gather better quality images later on.

The pit set ‘p1’ consists of 115 images that have pit flaw/s. This p1 data set has been used throughout the classification for comparison purposes, implemented IP, ML and DL techniques. Since this is pixel-wise labelled hence the size of observations as input are, 115 times the image size. The resolution size of the images collected is 1280 times 960 which gives 1,228,800 pixels per image. So in total this means 141,312,000 many training & testing data into the classifier. This is a lot of data that requires a lot of computer processing and storage hence it had a lot of computational and hardware challenges to start with.

For the machine learning purposes, pixel-wise classification has been performed. The data is divided into Training and Test sets if Holdout is used, otherwise images are cross-validated. Algorithms like Tree and k-nearest neighbours have been applied to evaluate but some took a lot of time and could not be used. Later on, comparisons are made to evaluate their performance by using confusion matrix and other such measures.

6.3 INITIAL TRIALS FOR MACHINE LEARNING

These initial experiments were performed by using Classification learner provided by MATLAB’s Statistics and Machine Learning Toolbox. It is able to apply common machine learning classifiers. Tools like Image labeller have been introduced last year which helps to label the data in an easier way. It also has an 'Image Data Store' function that can be used to import data from image collections that are too large to fit in memory.

6.3.1 Features extracted - RGB Colour space values

There are various features based around texture, intensity or colour descriptors. Images have different colour spaces such as RGB, LAB, HSV and different texture variations can be extracted like LBP and gradients. Initially, RGB colours of the original image have been used as the starting three features (Tm1) for training the model.

Colour space is an arrangement of a three-dimensional coordinate system and a subspace of this system, where a single point represents each colour. One of the widely used colour space models is RGB which stands for its red, green and blue colour channels. For humans an image is formed, when these three primary colours combine [90].

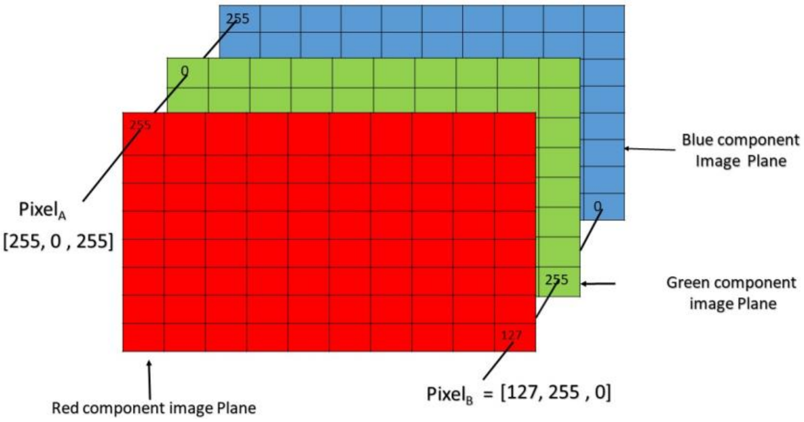


Figure 6-1 Illustration of RGB colour space showing three component images

It can be viewed as three separate images, including a red scale image, a green scale image and a blue scale image, piled up together. In MATLAB, it can be denoted by an array MxNx3 of colour pixel, where M and N stand for the number of columns and rows or it could also be referred as the axis positions. Each colour pixel is associated with three values which correspond to red, blue and green colour component of RGB image at a specified spatial location. The colour of any pixel can be determined by combining the red, green, and blue intensities stored in each colour plane at the pixel's location. Pixel of an RGB image formed from corresponding pixel of the 3 component images [146] can be seen in figure 6.1. Hence, the colour of a pixel can be easily calculated. Pixel$_A$ has (255, 0, 255) value and Pixel$_B$ has (127, 255, 0) value, which were determined by the combination of intensities stored in the red colour plane, green colour plane and blue colour plane respectively, for both the images.

6.3.2 Implementation of trials

For initial evaluation, the simple model consists of the following parameters; number of neighbours are = 100, the distance metric = Euclidean, distance weight= equal, with total number of observations for a single image = 1228800. Since it is based on the RGB colour space values (Tm1) it has 3 features which are Red, Green and Blue meaning 3 columns added for training data.

It was started by training, all 23 algorithms with a single image so that classifiers with higher accuracy and lower run time, can be selected for further experiments. Based on the image segmentation outcomes, it is expected that the KNN might show higher performance.

6.3.3 Results

Training was performed on 23 classifiers by using a single image, such that there are 1228800 number of observations to be trained with 5-fold cross validation applied. From the outcome list produced shown in figure 6.3, it can be seen that many algorithm produced similar results to the KNN-course algorithm with accuracy ranging between 98.1% -98.3% except a few exceptions. Such as SVM models which did not do well with an accuracy of 73% - 90%, allso KNN-Fine algorithm had an accuracy of 96.4% and Ensemble boosted Tress gave an accuracy of 97.1%.

However, taking in account another criteria to choose a model, a few were slower than the others in terms of time taken for training and/or testing such as SVM and KNN-Coarse. Tree-based models were relatively faster in comparison to the others.

Table 6-3 Initial Trials using Classification Learner showing accuracy performance

Classifiers	Method	Accuracy	Features
1.1	Tree: Complex Tree	98.3%	3/3
1.2	Tree: Medium Tree	98.3%	3/3
1.3	Tree: Simple Tree	98.2%	3/3
1.4	Linear Discriminant	98.2%	3/3
1.5	Quadratic Discriminant	98.2%	3/3
1.6	Logistic Regression	98.1%	3/3
1.7	SVM: Linear SVM	90.7%	3/3
1.8	SVM: Quadratic SVM	86.6%	3/3
1.9	SVM: Cubic SVM	73.0%	3/3

1.10	SVM: Fine Gaussian SVM	98.3%	3/3
1.11	SVM: Medium Gaussian SVM	98.3%	3/3
1.12	SVM: Coarse Gaussian SVM	98.3%	3/3
1.13	KNN: Fine KNN	96.4%	3/3
1.14	KNN: Medium KNN	98.2%	3/3
1.15	KNN: Coarse KNN	98.3%	3/3
1.16	KNN: Cosine KNN	98.1%	3/3
1.17	KNN: Cubic KNN	98.2%	3/3
1.18	KNN: Weightd KNN	98.1%	3/3
1.19	Ensemble: Boosted Trees	98.3%	3/3
1.20	Ensemble: Bagged Trees	98.2%	3/3
1.21	Ensemble: Subspace Discriminant	98.2%	3/3
1.22	Ensemble: Subspace KNN	97.6%	3/3
1.23	Ensemble: RUSBoosted Trees	97.1%	3/3

6.4 Learner app models based on 5-images- Setting 1

In the trials, a single image was used to train a model which is incorrect so for this model 5 images were used to train the model classifiers using 5-fold cross validation. With 5-images, there are 1228800*5 pixels = 6144000 inputs into the classifier.

6.4.1 Features Extracted - Intensity and Local Binary Pattern

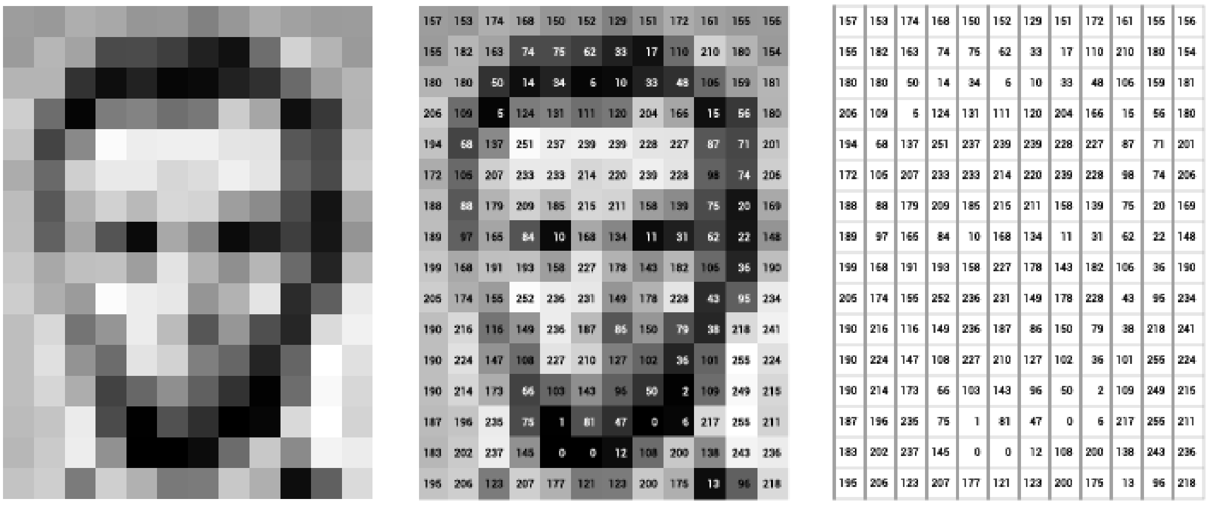
Additional to the RGB values, two other features were also used for modelling such as Intensity vector and an important Local Binary Pattern feature.

6.4.1.1 Intensity vector

In feature extraction, it becomes much simpler and reduces computational requirements if the image is compressed to a 2-D matrix, as sometimes handling the third dimension can be complex and redundant that could increase the amount of training data required to achieve good performance [147].

This can be performed by either converting image to grayscale value or binaries it. Applying conversion to grayscale value is richer than to just plainly binarising pixels into 0s and 1s. Possibly, a simple conversion to greyscale is intensity which is the mean of the RGB channels:

$$\mathcal{G}_{Intensity} \leftarrow \frac{1}{3}(R+G+B)$$



157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

Figure 6-2 Visual display of grayscale intensities [148]

Luminance is designed to match human brightness perception by using a weighted combination of the RGB channels:

$$\mathcal{G}_{Luminance} \leftarrow 0.3R + 0.59G + 0.11B$$

Luminance does not try to match the logarithmic nature of human brightness perception, but this is achieved to an extent with subsequent gamma correction. It is implemented by MATLAB's 'rgb2gray' function, which is frequently used in computer vision [149]

$$\mathcal{G}_{Luma} \leftarrow 0.2126R' + 0.7152G' + 0.0722B'.$$

Luma [147] is also sometimes used which is similarly a gamma corrected form of intensity

6.4.1.2 Local binary Pattern (LBP)

It has emerged as one of the popular texture features [48] with new variants continually being proposed. It is a very simple and efficient operator that is used to extract features to apply classification in computer vision. Its biggest strength lies in its overall computational simplicity tolerance against illumination variations and ease of implementation.

The main idea of this method is demonstrated in figure 6.3 which can be used to characterise a local texture [84]. Consider a 3 × 3 neighbourhood around each pixel, then compare every pixel to each of its 8 neighbours following in a clockwise, assign value '1' wherever the central pixel value is greater than the neighbour's value otherwise assign '0' value, then read the eight binary numbers associated with the eight neighbours sequentially in the clockwise direction to form a binary number and the finally assign this binary number or its decimal equivalent to the central pixel. The pixels around the central pixel (a) are given value 1, if they are brighter than the central pixel and value 0 otherwise (b) Shows these values (c) If we read these values sequentially in the clockwise direction [84] as shown in figure 6.3.

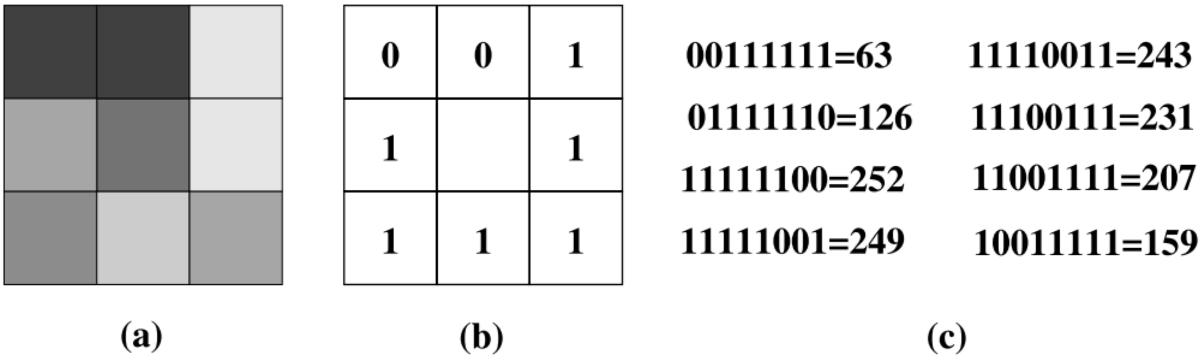


Figure 6-3 Local Binary Pattern (LBP) calculation of central pixel [84]

This seems similar to assign weights to the eight neighbouring pixels according to their relative position with respect to the central pixel. These weights are from 1 to 27, with the first assigned to the neighbour which contributes the most significant digit, the second assigned to the neighbour which contributes the second most significant digit, and so on. This representation makes it sensitive to rotation as some neighbours are given more importance than the others. This means that it is sensitive as to which neighbour is considered to be the first. Also it might not be suitable for macro-structures as it captures only the very local structure of the texture. This makes it very suitable for this research as microscopic images are being dealt with.

6.4.2 Implementation of Setting1

In the trials, a single image was used to train a model which is incorrect so for this model 5 images were used to train the model classifiers using 5-fold cross validation. With 5-images, there are 1228800*5 pixels = 6144000 inputs into the classifier. Since it is based on the RGB colour space values (Tm1) it has 3 features which are Red, Green and Blue, intensity vector (Tm2) and Local Binary Pattern (Tm3), meaning it has 5 columns for training data.

Just 5 images were used as it was restricted due to MATLAB learner. Classification learner crashed if too much data overloaded, didn't work for more than seven images, so it was switched from learner to manual script for the next setting. The five images, of the same size, were used for training are shown in figure 6.4, to compare the algorithms with each other on the basis of these five images.

Instead of choosing 23 classifiers this time, 6 models have been used such as Tree-Fine (Fast), Linear discriminant (different), Regression (different), KNN-fine (to test again for comparison with coarse), KNN-Coarse (as it produced best result in the initial trials), and Ensemble Tress.

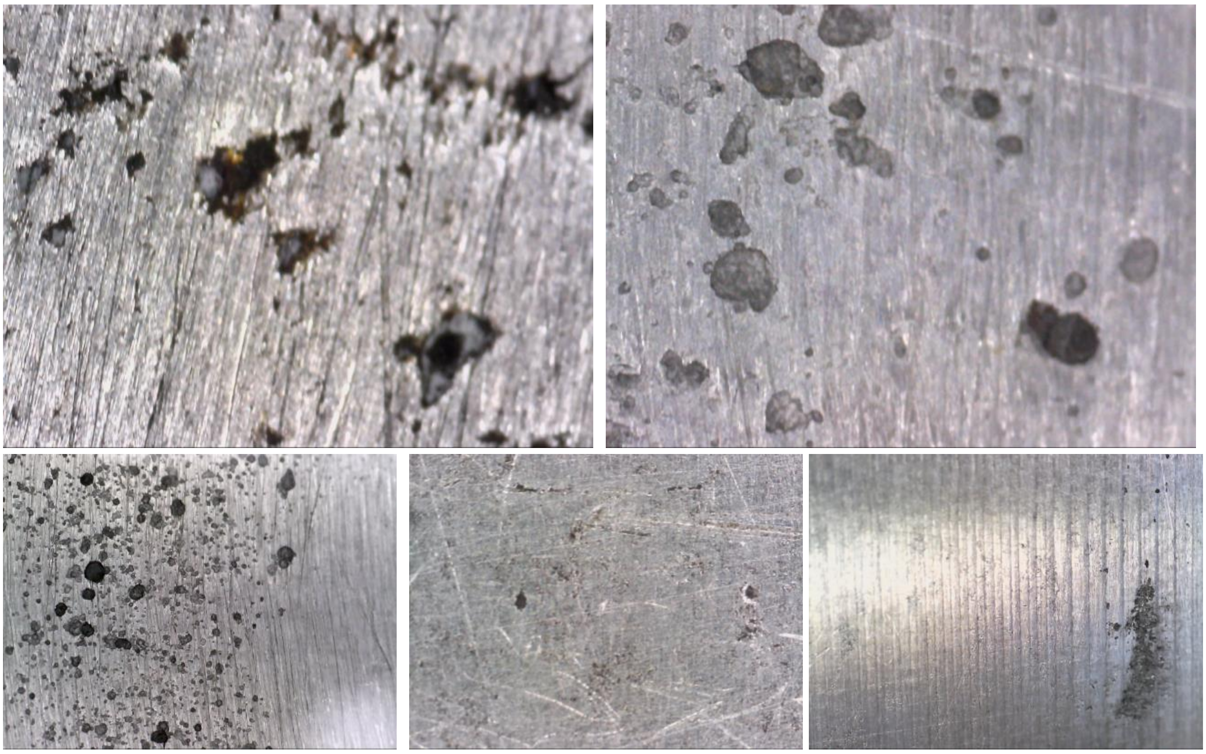


Figure 6-4 Five Test images used for machine learning with setting 1

Hence it was important to select such images which could cover majority of the different variations in the images like lighting, intensity, blurriness, number of flaws, density of the flaws, magnifications etc.

Each model produced was saved and then was run on the rest of the 110 images. These were then compared with the ground truth and performance measurements were made on the basis of all the general metrics discussed earlier. This setting was restrictive and limited as this was through the classification learner tool. It was needed to run all the data and apply cross validation for the model which the tool did not allow. The RGB colours of the original image have been used as the starting three features (Tm1) for training the model, same as in the initial trials. The training accuracy for the five models are presented in table 6.4.

Table 6-4 Training performance results of setting1 by using Classification Learner

Classifiers	Method	Accuracy	Features
1	Tree: Fine Tree	97.2%	3/3
2	Linear Discriminant; Linear Discriminant	96.6%	3/3

3	Logistic Regression: Logistic Regression	97.1%	3/3
4	KNN: Fine KNN	96.7%	3/3
5	KNN: Coarse KNN	97.0%	3/3
6	Ensemble: Boosted Trees	97.2%	3/3

6.4.3 Results for setting 1

sltm1: These results are a list of multiple metrics to judge the performance of the model as shown in table 6.5. It uses the classification learner (s1), with RGB features (tm1), with 5-fold cross validation on 5 training images for 5 classifiers (mls).

Table 6-5 Machine Learning Results for setting 1 with RGB feature

S1-tm1	Better	ML1-tree	ML2-discriminant	ML3-regression	ML4-fineknn	ML5-courseKnn	ML6-ensemble
TP	higher	1211	3528	4133	1964	5497	1489
TN	higher	1156346	1118891	1108133	1141101	940047	1152357
FP	lower	15021	52476	63234	30267	231320	19010
FN	lower	56222	53905	53299	55468	51936	55944
TPR	higher	0.012	0.047	0.057	0.026	0.192	0.016
TNR	higher	0.987	0.954	0.945	0.974	0.807	0.983
PPV	higher	0.048	0.048	0.048	0.047	0.047	0.049
Acc	higher	0.942	0.913	0.905	0.930	0.769	0.939
NPV	higher	0.953	0.953	0.953	0.953	0.947	0.953
FPR	lower	0.013	0.046	0.055	0.026	0.193	0.017
FDR	lower	0.952	0.952	0.952	0.953	0.953	0.951
FNR	lower	0.988	0.953	0.943	0.974	0.808	0.984
f1	higher	0.015	0.038	0.043	0.027	0.029	0.019
RMSE	lower	0.037	0.033	0.033	0.033	0.199	0.035
MCC	higher	0.000	0.001	0.001	0.000	0.000	0.000
CV	higher	0.008	0.020	0.022	0.014	0.015	0.010
RI	higher	0.894	0.846	0.832	0.873	0.884	0.889
VOI	lower	0.564	1.051	1.191	0.847	0.707	0.628
GCE	lower	0.022	0.055	0.064	0.039	0.033	0.027
BE	lower	155	144	144	137	170	154
Time(sec)	lower	1.062	0.999	1.214	35.702	44.263	11.572

s1tm2: In this set, intensity feature was added which is not that significant. It showed a very slight improvements in a few measures as shown in table 6.6. Comparison across the classifiers show that more data and features need to be added to make better decision on choosing a classifier.

s1tm3: In this set, added to the previous two features extracted, another feature was added called the Local Binary Pattern (LBP). It showed a very slight improvements in a few measures as shown in table 6.7. Comparison across the classifiers show that more data and features need to be added to make better decision on choosing a classifier.

Table 6-6 Machine Learning Results for setting 1 with RGB & Intensity feature

S1-tm2	Better	ML1-tree	ML2-discriminant	ML3-regression	ML4-fineknn	ML5-courseKnn	ML6-ensemble
TP	higher	1042	3455	4026	1926	1981	1282
TN	higher	1159197	1120279	1110149	1142245	1142586	1155739
FP	lower	12959	51877	62007	29911	29570	16417
FN	lower	55602	53189	52618	54718	54663	55362
TPR	higher	0.011	0.047	0.056	0.026	0.026	0.014
TNR	higher	0.989	0.955	0.946	0.974	0.974	0.986
PPV	higher	0.048	0.048	0.047	0.046	0.047	0.048
Acc	higher	0.944	0.914	0.907	0.931	0.931	0.942
NPV	higher	0.954	0.954	0.954	0.954	0.954	0.954
FPR	lower	0.011	0.045	0.054	0.026	0.026	0.014
FDR	lower	0.952	0.952	0.953	0.954	0.953	0.952
FNR	lower	0.989	0.953	0.944	0.974	0.974	0.986
f1	higher	0.013	0.038	0.042	0.027	0.026	0.016
RMSE	lower	0.037	0.033	0.032	0.032	0.033	0.036
MCC	higher	0.001	0.001	0.002	0.000	0.000	0.000
CV	higher	0.007	0.020	0.022	0.014	0.013	0.008
RI	higher	0.898	0.847	0.834	0.874	0.875	0.893
VOI	lower	0.528	1.040	1.172	0.837	0.809	0.576
GCE	lower	0.020	0.054	0.063	0.039	0.037	0.024
BE	lower	167	144	145	137	144	161
Time(sec)	lower	14.266	2.119	0.778	45.560	64.105	10.949

Table 6-7 Machine Learning Results for setting 1 with RGB, Intensity & LBP feature

S1-tm3	Better	ML1-tree	ML2-discriminant	ML3-regression	ML4-fineknn	ML5-courseKnn	ML6-ensemble
TP	higher	1541	3493	4007	4196	1766	1847
TN	higher	1151586	1120077	1110510	1099262	1148490	1147182
FP	lower	20570	52079	61647	72894	23666	24975
FN	lower	55103	53151	52637	52448	54878	54797
TPR	higher	0.018	0.047	0.056	0.064	0.021	0.022
TNR	higher	0.982	0.955	0.947	0.937	0.979	0.978
PPV	higher	0.048	0.048	0.047	0.046	0.048	0.048
Acc	higher	0.938	0.914	0.907	0.898	0.936	0.935
NPV	higher	0.954	0.954	0.954	0.954	0.954	0.954
FPR	lower	0.018	0.045	0.053	0.063	0.021	0.022
FDR	lower	0.952	0.952	0.953	0.954	0.952	0.952
FNR	lower	0.982	0.953	0.944	0.936	0.979	0.978
f1	higher	0.019	0.038	0.042	0.044	0.023	0.023
RMSE	lower	0.035	0.033	0.032	0.037	0.033	0.033
MCC	higher	0.000	0.001	0.001	0.001	0.001	0.001
CV	higher	0.010	0.020	0.022	0.023	0.012	0.012
RI	higher	0.888	0.847	0.835	0.818	0.884	0.882
VOI	lower	0.638	1.044	1.173	1.535	0.700	0.712
GCE	lower	0.028	0.055	0.063	0.066	0.032	0.033
BE	lower	158.832	143.897	144.120	136.381	150.679	154.431
Time(sec)	lower	3.882	1.285	2.120	16.770	61.803	12.763

6.5 CLASSIFICATION MODELS IMPLEMENTED ON FEW IMAGES - SETTING 2

This setting depicts that the classification learner has not been used, that is own algorithm has been implemented. Only the first setting uses classification toolbox, the rest are self-implemented algorithm settings. Similar to setting1, this setting also uses 5-fold cross validation with the same 5 images that were used in setting1, to be able to compare performance of self-code algorithm with learner.

6.5.1 Features Extracted - Gradient-based

Digital images, with the first derivative, has the ability to enhance the shift of grayscale values. Hence these_derivative values can be regarded as the corresponding output of boundaries. Thresholds can be set to extract the boundaries. First order derivative based techniques depend on computing the gradient several directions and combining the result of each gradient. The value of the gradient magnitude and orientation is estimated using Horizontal and Vertical convolution masks

Gradient is a vector whose components measure how rapid pixel value are changing with distance in the x and y direction [64]. It is a two-dimensional equivalent of the first derivative, by which edge points can be judged. It can be is defined as a metric for every individual pixel, containing the pixel colour changes in both x-axis and y-axis. This definition is aligned with the gradient of a continuous multi-variable function, which is a vector of partial derivatives of all the variables. Suppose f(x, y) records the colour of the pixel at location (x, y), the gradient vector of the pixel (x, y) is defined as follows:

$$\nabla f(x,y) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \\ \frac{\partial f}{\partial y} \end{bmatrix} = \begin{bmatrix} f(x+1,y) - f(x-1,y) \\ f(x,y+1) - f(x,y-1) \end{bmatrix}$$

There are two important attributes of an image gradient, the magnitude which is the L2-norm of the vector, and the direction which is the arctangent of the ratio between the partial derivatives on two directions.

$$g = \sqrt{g_x^2 + g_y^2}.$$

$$\theta = \arctan\left(g_y / g_x\right)$$

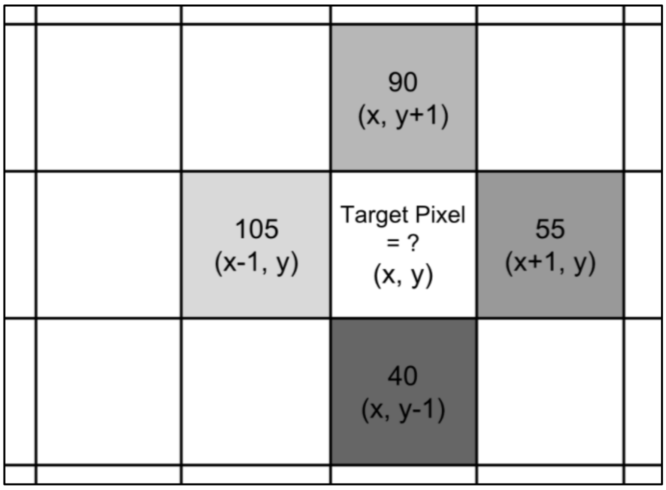


Figure 6-5 Computation of the gradient vector of a target pixel at location (x, y)

To compute the gradient vector of a target pixel at location (x, y), the colours of its four neighbours (or eight surrounding pixels depending on the kernel) need to be known as shown in figure 6.5. The gradient vector of the example is:

$$\nabla f = \begin{bmatrix} f(x+1,y) - f(x-1,y) \\ f(x,y+1) - f(x,y-1) \end{bmatrix} = \begin{bmatrix} 55 - 105 \\ 90 - 40 \end{bmatrix} = \begin{bmatrix} -50 \\ 50 \end{bmatrix}$$

Thus, the magnitude is

$$\sqrt{50^2 + (-50)^2} = 70.71$$

And the direction is,

$$\arctan\left({}^{-50}/_{50}\right) = -45^o$$

It is too slow if the gradient computation process is repeated for every pixel iteratively. Instead, it can be well translated into applying a convolution operator on the entire image matrix, labelled as using one of the specially designed convolutional kernels. Let’s start with the x-direction of the example in figure 6.5 using the kernel [−1,0,1] sliding over the x-axis; ‘∗’ is the convolution operator so

$$G_x = [-1 \quad 0 \quad 1] \times [105 \quad 255 \quad 55] = -105 + 0 + 55 = -50$$

Similarly, on the y-direction, we adopt the kernel $[1 \quad 0 \quad -1]^T$

$$G_y = [1 \quad 0 \quad -1]^T \times [90 \quad 255 \quad 40] = 90 + 0 - 40 = 50$$

There are other common first derivative operators which have similar principles and performance such as Robert, Sobel, Prewitt and Kirsch operators [150]. Different kernels are created for different goals, such as edge detection, blurring, sharpening and many more. For the Prewitt operator, rather than only relying on four directly adjacent neighbours, the Prewitt operator utilises eight surrounding pixels for smoother results. For example,

$$G_x = [-1 \quad -1 \quad -1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1] \times A$$

$$G_y = [10 \quad -1 \quad 10 \;\; -1 \quad 10 \quad -1] \times A$$

While the Sobel operator get assigned higher weights to emphasize the impact of directly adjacent pixels more. As an example,

$$G_x = [-1 \quad -2 \quad -1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 2 \quad 1] \times A$$

$$G_y = [10 \quad -1 \quad 20 \;\; -2 \quad 10 \quad -1] \times A$$

6.5.2 Implementation of Setting 2

This setting depicts that the classification learner has not been used, that is own algorithm has been implemented. Only the first setting uses classification toolbox, the rest are self-implemented algorithm settings. Similar to setting1, this setting also uses 5-fold cross validation with the same 5 images that were used in setting1, to be able to compare performance of self-code algorithm with learner.

Based on the RGB colour space values (Tm1) it has 3 features which are Red, Green and Blue, intensity vector (Tm2) and Local Binary Pattern (Tm3) plus the Gradient magnitude, Gradient Direction, Gradient Gx and Gradient Gy values forming Tm4, consisting of total 5+4, 9 feature columns for training data.

6.5.3 Results for setting 2

s2tm3: There is not much difference in the number of TN values but there is a big increase in the TP values. Along with this, as it can be seen that the FP also decreases. Both these numbers effect on all the rest of the formulas of performance. Hence, regardless of the classifier chosen, between S1 and S2, it is clear from table 6.8 that the self-coded algorithm S2, produces better results than S1.

Table 6-8 Machine Learning Results for setting 2 with RGB, Intensity & LBP feature

S1 vs S2	Better	S1ML1	S1ML6	S2ML1	S2ML6	Method
TP	higher	1541	1847	15803	17443	S2ML6
TN	higher	1151586	1147182	1163790	1162359	S2ML1
FP	lower	20570	24975	8366	9797	S2ML1
FN	lower	55103	54797	40841	39201	S2ML6
TPR	higher	0.018	0.022	0.303	0.332	S2ML6
TNR	higher	0.982	0.978	0.993	0.992	S2ML1
PPV	higher	0.048	0.048	0.647	0.628	S2ML1
Accuracy	higher	0.938	0.935	0.960	0.960	S2ML1/6
NPV	higher	0.954	0.954	0.966	0.967	S2ML6
FPR	lower	0.018	0.022	0.007	0.008	S2ML1
FDR	lower	0.952	0.952	0.353	0.372	S2ML6
FNR	lower	0.982	0.978	0.697	0.668	S2ML1
f1	higher	0.019	0.023	0.322	0.349	S2ML6
RMSE	lower	0.035	0.033	0.034	0.032	same
MCC	higher	0.000	0.001	0.366	0.387	S2ML6
CV	higher	0.010	0.012	0.216	0.234	S2ML6
RI	higher	0.888	0.882	0.925	0.926	S2ML6
VOI	lower	0.638	0.712	0.392	0.411	S2ML1
GCE	lower	0.028	0.033	0.018	0.021	S2ML1
BE	lower	158.832	154.431	70.359	70.112	S2ML6
Time(sec)	lower	3.882	12.763	63.200	110.043	S1ML1

s2tm4: In general, comparison between S1 and S2, shows a massive increase in the performance measures of the classifiers in S2 except the time taken as shown in table 6.9. This setting was used only to make a direct comparison between classification learner and self-code algorithm by using similar classifier and similar dataset. The next task is to increase the training dataset, as setting S2 doesn't put a restriction on the number of observations. So setting 3 uses same 5-fold cross validation on 115 images instead of just 5 images.

Using set s2tm4ml6, a few visual results of the predicted image are shown in figure 6.6 and figure 6.7, along with their respected original image and ground truth. It also shows results, when a post-processing step is applied. The test samples used are Image A186 and A188. The figures show the original, its ground truth, predicted image by Ensemble Classification (S2tm4) implemented on test image, and then post-processing applied after classification.

Table 6-9 Comparison between s2tm3 vs s2tm4 where tm4 has Gradient feature added

Tm3 vs Tm4	Better	Tm3ML1	Tm3ML6	Tm4ML1	Tm4ML6
TP	higher	15803	17443	17659	17618
TN	higher	1163790	1162359	1157001	1156975
FP	lower	8366	9797	15155	15182
FN	lower	40841	39201	38985	39026
TPR	higher	0.303	0.332	0.333	0.333
TNR	higher	0.993	0.992	0.987	0.987
PPV	higher	0.647	0.628	0.541	0.542
Accuracy	higher	0.960	0.960	0.956	0.956
NPV	higher	0.966	0.967	0.967	0.967
FPR	lower	0.007	0.008	0.013	0.013
FDR	lower	0.353	0.372	0.459	0.458
FNR	lower	0.697	0.668	0.667	0.667
f1	higher	0.322	0.349	0.340	0.344
RMSE	lower	0.034	0.032	0.032	0.032
MCC	higher	0.366	0.387	0.365	0.368
CV	higher	0.216	0.234	0.222	0.224
RI	higher	0.925	0.926	0.918	0.918
VOI	lower	0.392	0.411	0.486	0.488
GCE	lower	0.018	0.021	0.026	0.027
BE	lower	70.359	70.112	83.480	82.538
Time(sec)	lower	63.200	110.043	57.913	120.511

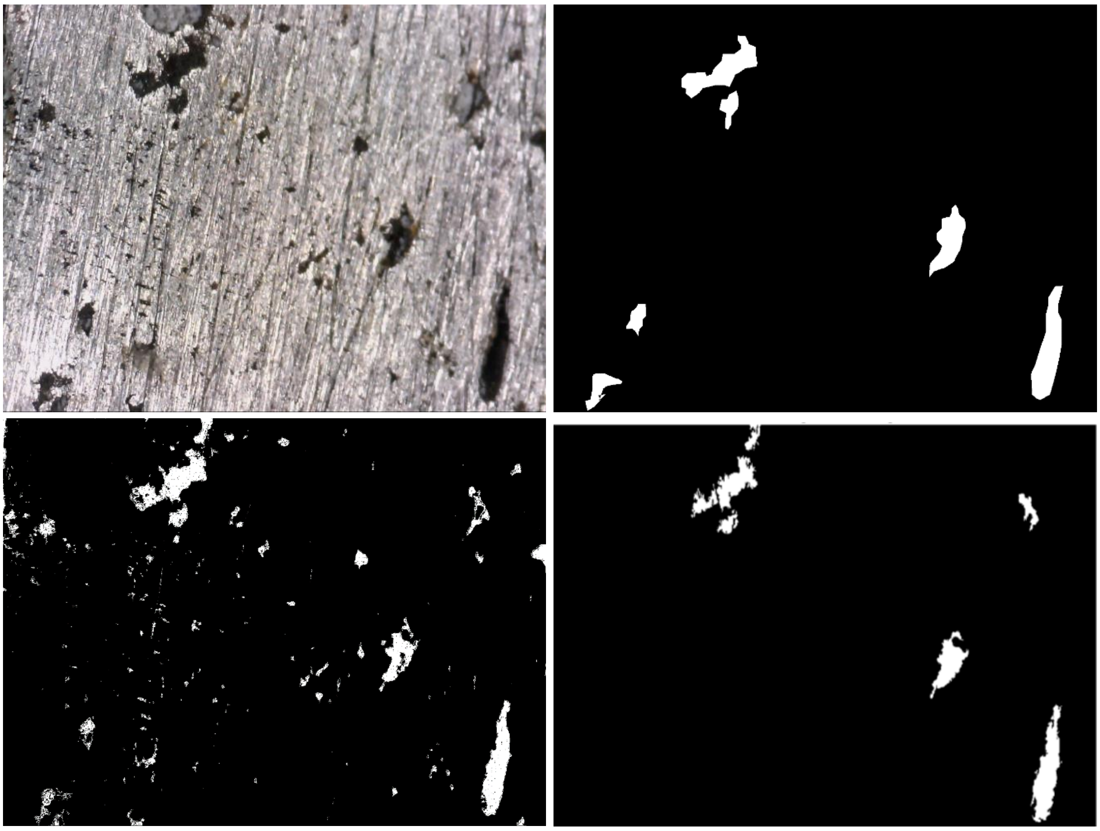


Figure 6-6 Ensemble Classification (S2tm4) implemented on Image A186

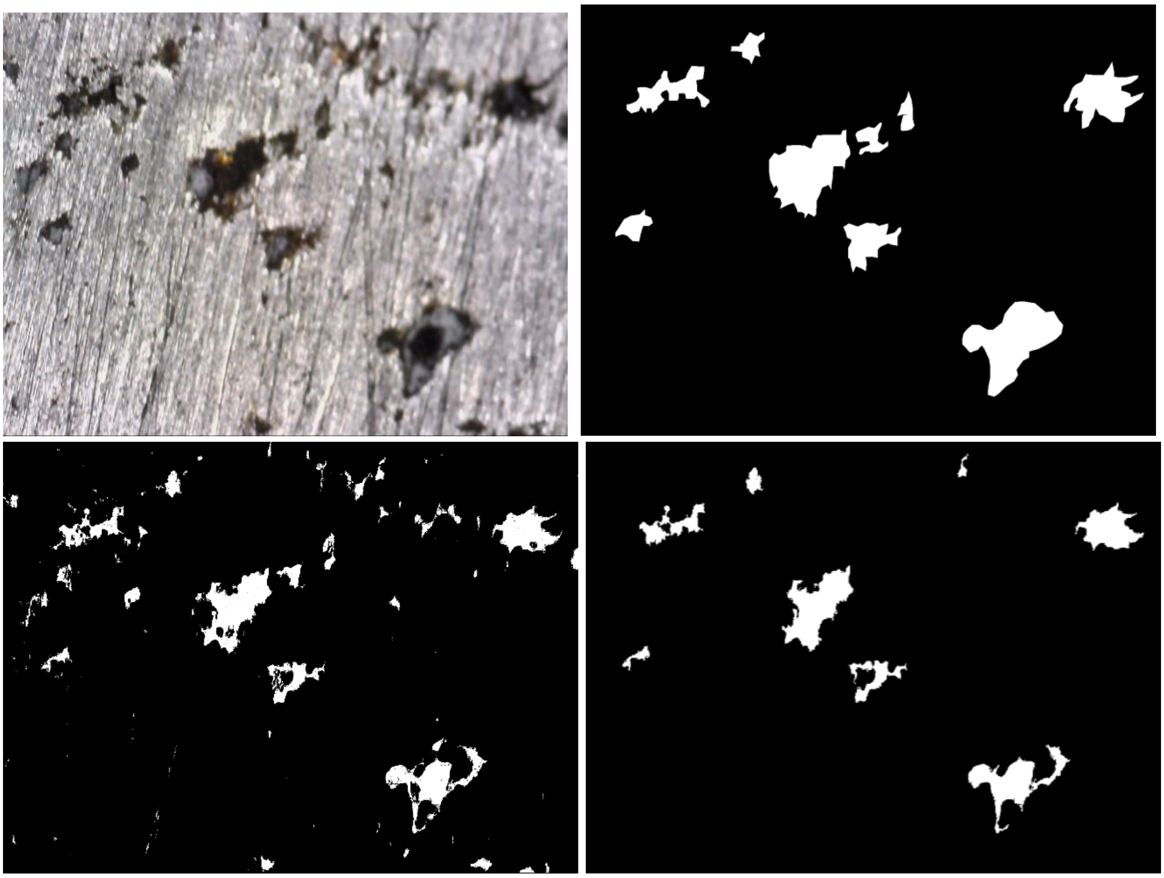


Figure 6-7 Ensemble Classification (S2tm4) implemented on Image A188

6.6 CLASSIFICATION MODELS IMPLEMENTED BASED ON FULL DATA- SETTING 3

This setting of the algorithm applies 5-fold cross validation by using the whole set of 115 images. As it was not restricted by the learner input issues, finally the whole set of images could be used. This is then compared to the previous setting, by evaluating performance of 5-images based trained model with 115-images trained model. This will show the positive effect of increasing data for training.

6.6.1 Features Extracted - L*a*b and HSV colour spaces

6.6.1.1 L*a*b colour space

Colour spaces CIE 1931 (Commission Internationale de l'Elcairage) were introduced to be able to quantitatively express links between distributions of wavelengths in the EM (ElectroMagnetic) visible spectrum along with the physiological perceived colours in human eye sight [151]. The mathematical relationships between these colour spaces serve as fundamental tools for colour management.

Another model that shares similar property to HSV is L∗a∗b∗ where L stands for lightness, 'a' stands for colour component green-red and 'b' for blue-yellow. Research [152] shows results that proves that L*a*b model is able to efficiently distinguish luminance information from colour information, even in presence of saturation, hence offers luminance separation for segmentation tasks. This colour model also offers ‘perceptually linear’ as another important property. It is often used as an interchange format when the task deals with different devices, as it is device independent.

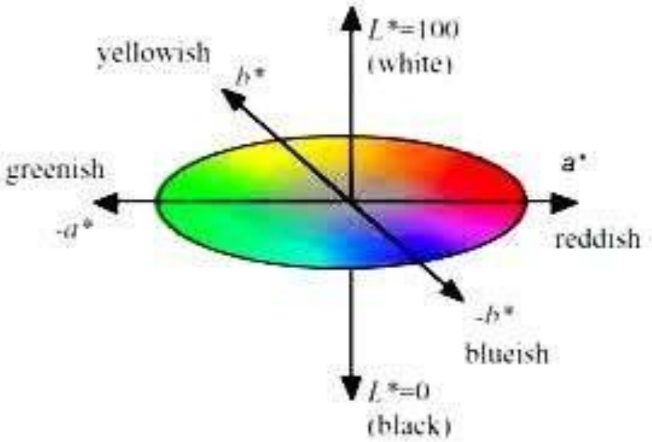


Figure 6-8 Illustration of LAB Colour Space [153]

It is worthwhile using these models, in cases which require luminance invariance. The RGB to L∗a∗b∗ transformation involves two transformations. First RGB colours are transformed into XYZ, and then XYZ to L∗a∗b∗ transformation. The XYZ space was formed on the mathematical limit of human vision as far as colour is concerned. These X, Y and Z are channels extrapolated from the R, G and B channels to prevent the occurrence of negative values. Y represents luminance, Z represents a channel close to blue channel and X represents a mix of cone response curves chosen to be orthogonal to luminance and non-negative. The RGB-to-XYZ transformation is a function of the viewing conditions such as intensity and colour of the lighting.

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 2.7690 & 1.7518 & 1.1300 \\ 1.0000 & 4.5907 & 0.0601 \\ 0.0000 & 0.0565 & 5.5943 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

The XYZ to L∗a∗b∗ transformation is given below,

$$\begin{aligned} L^* &= 116\left(\tfrac{Y}{Y_n}\right)^{\frac{1}{3}} - 16, \\ a^* &= 500\left[\left(\tfrac{X}{X_n}\right)^{\frac{1}{3}} - \left(\tfrac{Y}{Y_n}\right)^{\frac{1}{3}}\right], \\ b^* &= 200\left[\left(\tfrac{Y}{Y_n}\right)^{\frac{1}{3}} - \left(\tfrac{Z}{Z_n}\right)^{\frac{1}{3}}\right], \end{aligned}$$

Where, X_n, Y_n, Z_n are the X, Y, and Z values of the reference, while R = G = B = 1.0. [152].

6.6.1.2 HSV colour space

The RGB values, despite being the more popular colour space, has its downside. One of which is that they get highly affected by illumination [154]. For such cases where illumination value might be a crucial feature, other colour systems such as HSV can be used instead. When humans see colours, they perceive it in a certain way which was taken in account when this model was developed [151]. HSV stands for hue, saturation and value, where the colour hue is described in terms of saturation dealing with shade and value reflecting brightness.

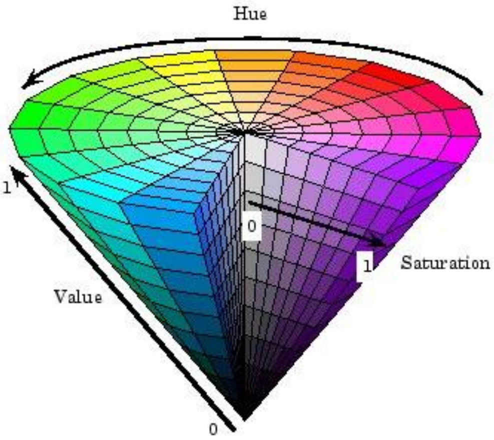


Figure 6-9 Illustration of the HSV Colour Space [155]

The RGB to HSV transformations are fairly standard where R, G, B, H, S,V are the values of the red channel, green channel, blue channel, hue channel, saturation channel and the value channel respectively. [152]

6.6.2 Implementation of Setting 3

This setting of the algorithm applies 5-fold cross validation by using the whole set of 115 images. As it was not restricted by the learner input issues, finally the whole set of images could be used. This is then compared to the previous setting, by evaluating performance of 5-images based trained model with 115-images trained model. This will show the positive effect of increasing data for training.

For ML6 (Ensemble), used bag setting hence applied Random Forest (RF) model in this setting. Based on the RGB colour space values (Tm1) it has 3 features which are Red, Green and Blue, intensity vector (Tm2) and Local Binary Pattern (Tm3), Gradient magnitude, Gradient Direction, Gradient Gx and Gradient Gy values (Tm4), plus added L*a*b and HSV (Tm5) consisting of total 9+6 = 15 feature columns for training data.

6.6.3 Results for setting3

s3tm4: In general, comparison between S2 and S3, shows slight increase in the performance measures of the classifiers in S3 except the massive increase time taken. There is not much difference in the number of TP and TN values plus it can be seen that the FP decreases. Both these numbers improve on all the rest of the formulas of performance. Hence, regardless of the classifier chosen, between S2 and S3, it is clear from table 6.10 that the S3 based on 115 cross-validated images, produces better results than S2, which is based on just 5 training images.

Table 6-10 Comparison of ML models between setting 2 and setting 3

S2 vs S3	Better	S2ML1	S2ML6	S3ML1	S3ML6	Better
TP	higher	17659	17618	16933	17829	S3ML6
TN	higher	1157001	1156975	1163467	1162919	S3ML1/6
FP	lower	15155	15182	8689	9237	S3ML1/6
FN	lower	38985	39026	39710	38815	S3ML6
TPR	higher	0.333	0.333	0.325	0.325	S2
TNR	higher	0.987	0.987	0.993	0.992	S3ML1/6
PPV	higher	0.541	0.542	0.645	0.676	S3ML6
Acc	higher	0.956	0.956	0.961	0.961	S3ML1/6
NPV	higher	0.967	0.967	0.967	0.968	S3ML6
FPR	lower	0.013	0.013	0.007	0.008	S3ML1/6
FDR	lower	0.459	0.458	0.355	0.324	S3ML6
FNR	lower	0.667	0.667	0.675	0.675	S3ML1/6
f1	higher	0.340	0.344	0.340	0.409	S3ML6
RMSE	lower	0.032	0.032	0.033	0.026	S3ML6
MCC	higher	0.365	0.368	0.382	0.435	S3ML6
Time(sec)	lower	57.913	120.511	8082	3536	S2

Table 6.11 shows that the best classifier to choose based on the performance measures is the ML5, coarse k-nearest neighbour model.

Table 6-11 Machine Learning Results for setting 3 of Tree,Knn,Ensemble classifiers

S3tm4mls	Better	ML1-tree	ML5-coarseKnn	ML6-ensemble	Better
TP	higher	16933	18639	17829	ML5
TN	higher	1163467	1164185	1162919	ML1
FP	lower	8689	7972	9237	ML5
FN	lower	39710	38004	38815	ML5
TPR	higher	0.325	0.342	0.325	ML5
TNR	higher	0.993	0.993	0.992	ML1/5
PPV	higher	0.645	0.704	0.676	ML5
Acc	higher	0.961	0.963	0.961	ML5
NPV	higher	0.967	0.968	0.968	ML5/6
FPR	lower	0.007	0.007	0.008	ML1/5
FDR	lower	0.355	0.296	0.324	ML5

FNR	lower	0.675	0.658	0.675	ML5
f1	higher	0.340	0.438	0.409	ML5
RMSE	lower	0.033	0.025	0.026	ML5
MCC	higher	0.382	0.462	0.435	ML5
Time(sec)	lower	8082	32119	3536	ML6

s3tm5: Adding more features increases the performance of the model which can be seen in table 6.12 Tree Model (ML1) was used as an example implementation as it is quick with producing the results in comparison to other classifiers.

Table 6-12 Comparison between s3tm4 vs s3tm5 where L*a*b and HSV features added

Tm4 vs Tm5	Better	Tm4-ML1	Tm5-ML1	Better
TP	higher	16933	17108	Tm5
TN	higher	1163467	1164270	Tm5
FP	lower	8689	7886	Tm5
FN	lower	39710	39536	Tm5
TPR	higher	0.325	0.342	Tm5
TNR	higher	0.993	0.993	Same
PPV	higher	0.645	0.694	Tm5
Acc	higher	0.961	0.961	Same
NPV	higher	0.967	0.967	Same
FPR	lower	0.007	0.007	Same
FDR	lower	0.355	0.306	Tm5
FNR	lower	0.675	0.692	Same
f1	higher	0.340	0.400	Tm5
RMSE	lower	0.033	0.026	Tm5
MCC	higher	0.382	0.430	Tm5

6.7 Chapter Summary

This chapter provided implementation of multiple traditional machine learning algorithms for classification of the microscopic images such as Decision tree, Ensemble Random forest, Ensemble Adaboost and KNN. It then discussed different potential spatial and textural features that are used as an input to the classifiers. Features including local binary pattern (LBP), gradient magnitude & direction, RGB values are extracted from the images to train the model. In the end, variations in the results obtained from these different classifiers with different model settings were discussed.

Classifiers performed differently on the basis of their architecture model and parameters. Overall, as the data size increased and more features were added, the performance of the classifiers showed improvement. S3 with more data images is better than S2. Tm4 is better than Tm1 with added features such as Local Binary Pattern and Gradients in comparison to just the RGB base values of Tm1. It is interesting to see the effect of more task-suitable classifiers, as seen in table 6.13 that ML5 (Coarse KNN) with less features Tm4, gives better output than ML1 with Tm5 features. But the time taken to run the model by KNN is much longer than a simple tree or even ensemble classifiers so it has its computational restrictions, as it also requires more memory to run.

Table 6-13 Illustrating significance of task-suitable classifiers using Tm4ML5 vs Tm5ML1

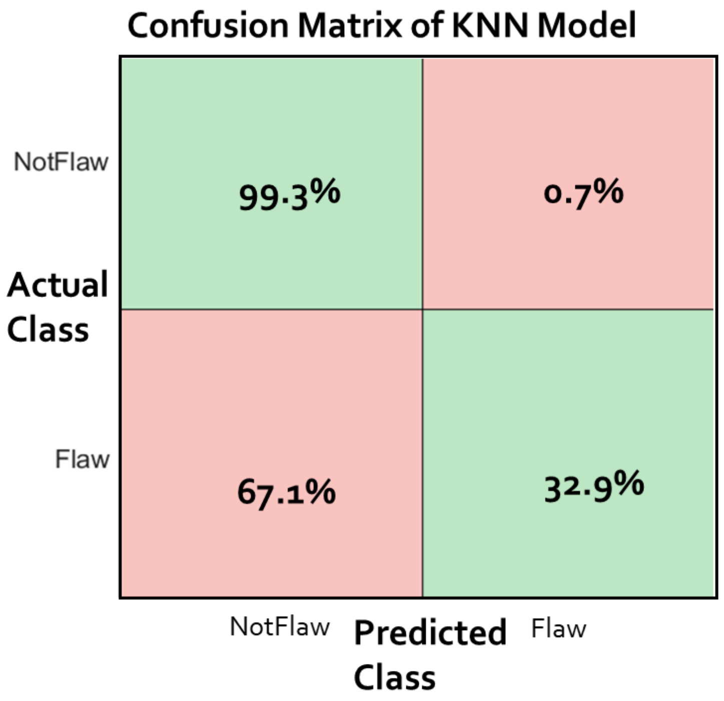
S3	Better	Tm4-ML1-tree	Tm5-ML1	Tm4-ML5-KNN	Tm4-ML6-ensemble	Better
TP	higher	16933	17108	18639	17829	ML5
TN	higher	1163467	1164270	1164185	1162919	ML1
FP	lower	8689	7886	7972	9237	ML5
FN	lower	39710	39536	38004	38815	ML5
TPR	higher	0.325	0.342	0.342	0.325	ML5
TNR	higher	0.993	0.993	0.993	0.992	ML1/5
PPV	higher	0.645	0.694	0.704	0.676	ML5
Acc	higher	0.961	0.961	0.963	0.961	ML5
NPV	higher	0.967	0.967	0.968	0.968	ML5/6
FPR	lower	0.007	0.007	0.007	0.008	ML1/5
FDR	lower	0.355	0.306	0.296	0.324	ML5
FNR	lower	0.675	0.692	0.658	0.675	ML5
f1	higher	0.340	0.4	0.438	0.409	ML5
RMSE	lower	0.033	0.026	0.025	0.026	ML5
MCC	higher	0.382	0.43	0.462	0.435	ML5

Looking purely from the basis of the four major performance metrics, it can be seen from table 6.14, below that with same data and same features extracted, ML5-Coarse KNN classifier gives better results than ML1-Tree and ML6-Ensemble classifier. Taking ML1-tree classifier, for comparison, it can be seen that S3ML1 is better than S1Ml1 with more data populated, plus S3Tm5 is better than S3Tm4 with added extracted features.

Table 6-14 Evaluation and comparison in order to select the best classifier suitable

	Better	S1Tm3-ML1	S3Tm4-ML1	S3Tm5-ML1	S3Tm4-ML5	S3Tm4-ML6	Better
TP	higher	1541	16933	17108	18639	17829	S3Tm4-ML5
TN	higher	1151586	1163467	1164270	1164185	1162919	S3Tm5-ML1
FP	lower	20570	8689	7886	7972	9237	S3Tm5-ML1
FN	lower	55103	39710	39536	38004	38815	S3Tm4-ML5

Hence, it can be concluded that the outcomes are dependent on a combination of three major criteria; added data, added relevant features extracted and more task-suitable classifier models. A confusion matrix of the KNN Course model is shown in figure 6.10 where it can be seen that it is good at classifying the background class at 99.3% but not for the flaw class.



	NotFlaw	Flaw
NotFlaw	99.3%	0.7%
Flaw	67.1%	32.9%

Figure 6-10 Machine learning results of KNN model using Confusion matrix

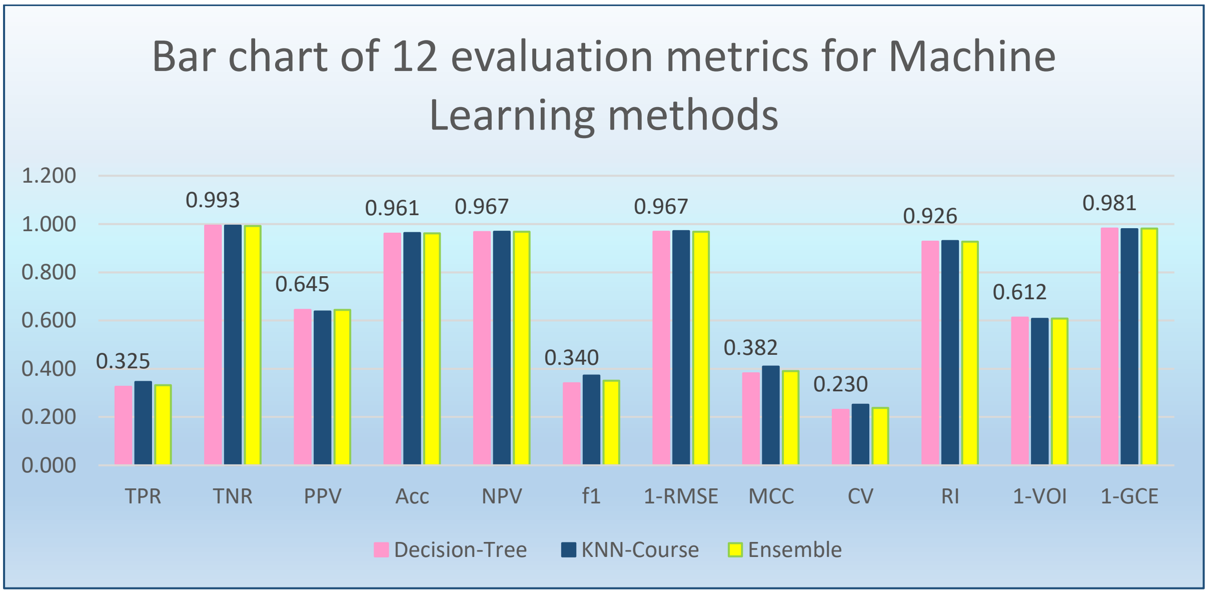


Figure 6-11 Evaluation of implemented supervised machine learning methods by bar chart

Many models were applied including KNN, Tree and ensemble (Random forest) with cross-validation and extracted features such as LBP and gradients. The bar chart in figure 6.11 shows performance of these three classifiers based on 12 evaluation metrics methods such as recall, TNR, PPV, Accuracy, NPV, f1, RMSE, CV, RI, VOI and GCE. After conducting the evaluation based on the performance indicators, it can be seen from the figure that the true positive rate is very low as not many impactful features were extracted. They gave similar results with KNN-course slightly better. The bar chart highlights the fact that even though the accuracy is quite high around 96% but the recall of the ML methods is not adequate. The reason might be that more suitable features need to be extracted. So, the next step was to dig deeper, so started researching deep learning. This is where applying deep learning will be useful as it has the ability to automatically extract suitable features particularly dealing with complex and varied images, as researched in the literature review, chapter 2.

The biggest challenge with traditional ML models is the feature extraction process. It will be useful if right features can automatically be extracted. Capability of learning to focus on the right features by themselves, requiring little guidance makes DL an extremely powerful tool for modern machine learning. DL methods are gaining on traditional ML approaches especially for heavily complicated use cases like image recognition. This is discussed in the next chapter in detail including implementation and evaluation.

Chapter 7
Supervised Deep learning methods implementation, comparison & evaluation

This chapter provides an overview of the design steps that were necessary to set up the network and begin training for deep learning methods. It discusses three main supervised deep learning models implemented for this research such as Image-wise classification model, Pixel-wise UNet model and a novel model which is a combination of UNet_VGG16. For each of them it discusses their detailed architecture, implementation, hyper parameters such as learning rate, batch size, then displays visual results and conclusions

7.1 INTRODUCTION

This chapter discusses supervised learning methods based on deep learning. It is the third main contribution to this research combined with the previous chapter. After implementing and evaluating machine learning models, next main task is to implement intelligent techniques such as CNNs on the same data. The trained model will be able to classify the flaw type (image-wise) and highlight the classified structural flaw location (pixel-wise) as shown in figure 7.1, in the test images. This chapter consists of the following:

- Reasons as to why deep learning is considered for the project task (section 7.2)
- Explains some key implementation considerations for deep learning (section 7.3)
- Experiments of initial trials for deep learning is discussed in detail (section 7.4)
- Next, it discusses the three main deep learning models implemented for this research which are Image-wise classification model AlexNet, Pixel-wise UNet model and a novel model which is a combination of UNet_VGG16. For each of them, it discusses their detailed architecture design, implementation, hyper parameters such as learning rate, batch size, then displays visual results and conclusions (section 7.5 – section 7.7)
- In the end, evaluation of all the supervised learning methods is concluded with visual illustration (section 7.8)

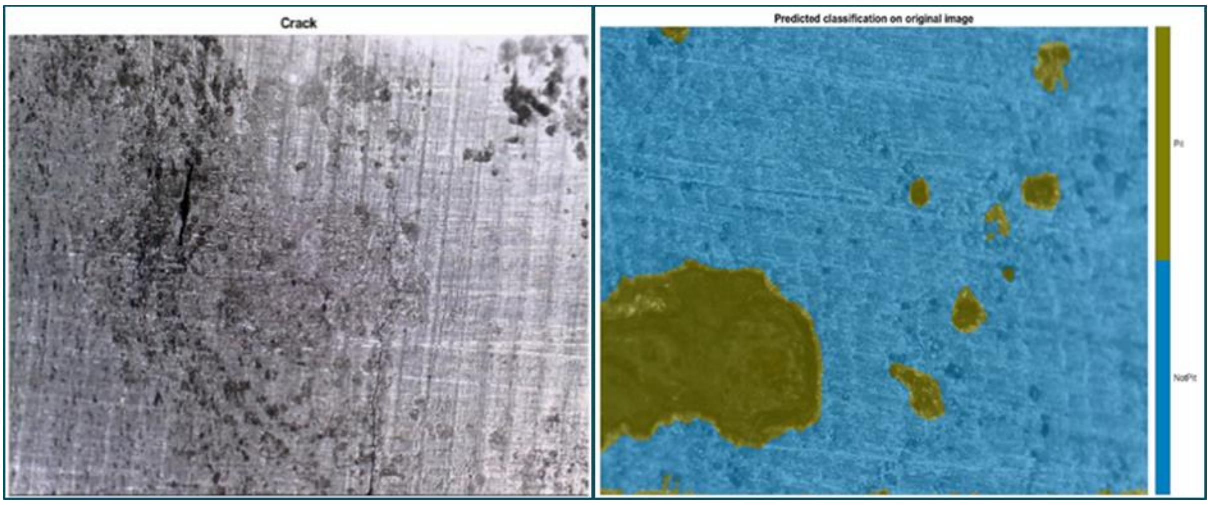


Figure 7-1 Predictions of image-wise & pixel-wise models for classification and localisation

7.2 REASONS AS TO WHY DEEP LEARNING IS CONSIDERED FOR PROJECT TASK

At present, deep learning is the state-of-the-art approach to machine learning and the significant aspect of deep learning algorithms lies in neural networks. They are now being used for complex datasets where traditional techniques have been unable to provide satisfactory or reliable classifications. The major reasons as to why deep learning model has been used for this project research are as follows:

- Extract suitable features: Capability of learning to focus on the right features by themselves, requiring little guidance, makes deep learning an extremely powerful tool for modern machine learning. This is one of the main reasons that deep learning (DL) methods are gaining popularity over traditional machine learning methods. As in the case of the research, it has been extremely challenging to determine in advance which features should be used for the project problem. All successive steps can become futile with a poor choice of features as illustrated by experiments in previous chapter.

- Easier to implement for uncertain data: They are able to improve their performance by considering previous examples, and do not require any programming specific to the task they are learning about. They are particularly useful in domains where knowledge and decision processes are poorly understood, or the data is subject to uncertainty.

- Detect complex trends: They have a proven ability by which they can derive meaning from complicated or imprecise data, they can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. They can be thought of as an "expert" in the category of information it has been given to analyse. Then they can be used to provide projections given new situations of interest.

- Self-adjusting ability: They have an ability to learn how to do tasks based on the data given for training or initial experience known as adaptive learning. They can modify their behaviour in the response to their environment. This factor, more than any other, is responsible for the interest they have received. When they are shown a set of inputs, which have specific desired output, they self-adjust to produce consistent responses.

In short, deep learning models have been implemented for this research as the project dataset is too complex, deals with uncertain factors such as lighting, morphology etc and it would have been highly challenging to script a classification programme that could have taken all the complicated factors such as varied features, shapes, sizes and surface material aspects into account for each of the classes. Hence, for distinguishing between the flaw classes, it is much easier and efficient to implement image-wise classification. Multiple deep learning algorithms with various permutation combinations have been implemented which are discussed in length in the following sections.

7.3 Implementation considerations for deep learning

The design steps start with the identification of available state-of-the-art network architectures, to provide a basis for the subsequent evaluation and then selection of the most suitable network architectures. Furthermore, it is necessary to identify and select the frameworks that are to be used for the implementation from the large number of available frameworks that can be used. Finally, it also provides an overview of the most important hyper-parameters to be evaluated.

To start with implementation, the dataset is split into three parts, the training dataset, validation dataset, and test dataset. Each of the three datasets consists of a number of images and their corresponding ground truth, i.e. the expected pixel-wise labels. For intelligent supervised classification, a predictive model is produced by using training dataset. This model is then evaluated by using the validation dataset images as input, and then comparing the resulting label predictions with the expected label ground truth. The hyper-parameters of the model are then adapted based on the results of the validation through the evaluation metrics. The model that yields good performance on the validation dataset will be selected. The predictive model produced will take previously unused test images, as input and produce predicted pixel-wise labels as output to generate a final evaluation of the model. For image-wise classification, it will predict the flaw class such as pit or a crack.

7.3.1 Datasets

Using deep networks to build applications has always been complex. Firstly because they require detailed and colossal data for training and testing. Secondly because valid datasets are required for network evaluation. Though there are datasets available online but the problem is that different networks are built for different applications. Even when they are available, they might not be suitably designed with the required classes of labels. Also sometimes a specific industry problem needs to be solved so fresh data is required like in the case of this project.

For this research, flaw images were collected as discussed in chapter 4. Total of around 3000 images were collected but many of the images are not considerable due to the bad quality of the images which includes blurriness and bad lighting. As mentioned before, the data collection process was gradually improved during the course which has helped to gather better quality images later on.

Supervised deep learning algorithms rely on manually-annotated ground truth to be fed to the CNN for learning. Pixel-wise labelled dataset of 115 images consisting of only pits, and 20 images that have only cracks was created, to apply the performance measure. This data was divided into four damage categories, based on two magnifications; Cracks only, Pits only, Pit-2-crack and background/ no-damage.

The structure of the data sets was divided for two purposes; one for image-wise classification and the other for pixel-wise classification. The importance of datasets structuring for image-wise classification is explained when discussing the results later on in this chapter.

The pixel-wise data sets were difficult to be created as they require pixel-wise labelling which is extremely time costing. The pixel-wise labelling is the answer to the question of ‘where’ is the flaw located in the image, in terms of classification. While segmentation is able to divide an image into several regions based on grouping similar pixels into one region but it is not able to achieve labelling of pixels into the categorised classes. To perform pixel-wise semantic grouping based on certain label classes, semantic segmentation is used. This creates semantically meaningful outcomes which can then be used for many applications. As an example, for a pit flaw, semantic pixel wise segmentation propagates through each pixel and assigns it a label. This means that each and every pixel in an image belongs to a label.

The initial focus was on corrosion pits, hence the pit set ‘p1’ consists of 115 images that have pit flaw/s. This p1 data set has been used throughout the pixel-wise classification for comparison purposes, implemented IP, ML and now DL techniques. Since this is pixel-wise labelled hence the size of observations as input are, 115 times the image size. The resolution size of the images collected is 1280 times 960 which gives 1,228,800 pixels per image. So in total this means 141,312,000 many training & testing data into the networks. This is a lot of data that requires a lot of computational resources and storage hence faced many computational and hardware challenges. Another data set that was used is a crack set ‘c1’ that consists of 20 images that have a crack flaw/s.

7.3.2 Network Architectures

There is a large number of different neural network architectures for a variety of tasks. The number of nodes in the input and output layers can generally be determined by the dimensionality of the problem.

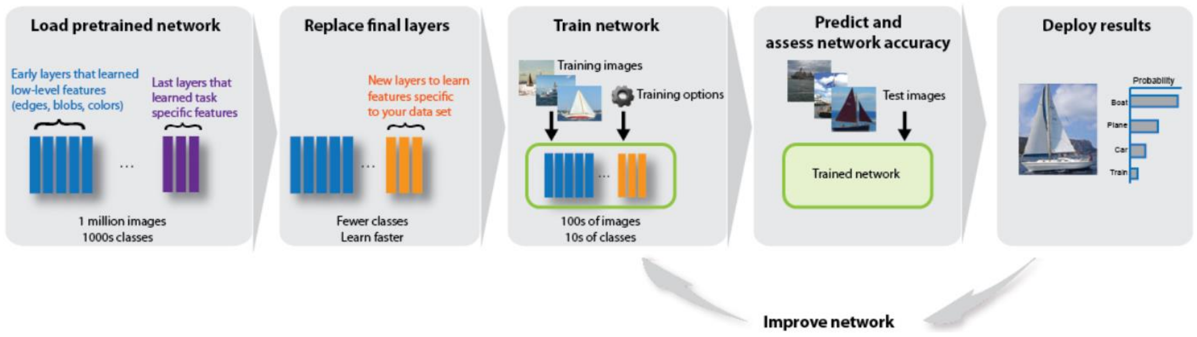


Figure 7-2 Work flow showing how to perform transfer learning [156]

Previously trained networks can be used for classification, feature extraction and/or transfer learning tasks as shown in figure 7.2. They learn new patterns in the new data based on the knowledge transferred from a pre-trained net. Majority of the pre-trained networks are trained by using ImageNet database [115]. These networks have been trained on more than a million images and can classify images into 1000 or more object categories. Using a pre-trained network with transfer learning is typically much faster and easier than training a network from scratch. The ones implemented for deep learning are VGGNet and ResNet in initial trials, AlexNet for Image-wise Classification and later on UNet for Pixel-wise segmentation. Finally in the end, combination of Unet with VGG16 is implemented for the task of Pixel-wise classification.

7.3.3 Frameworks

Development in Machine learning field requires to deal and tackle with tremendous amounts of data for training, learning and testing the systems. Adding to these, when working with image and video datasets, memory and space requirements increase as compared to normal datasets. Graphical processing unit (GPU) is always required on large ML frameworks. Training CNNs involve lots of matrix multiplications and vector operations that can be parallelized in GPUs. Experiments were performed using either Tensor Flow or MATLAB, and ran on the same Nvidia graphics card. Keras was used at the back-end to tensor flow which was implemented for UNet structure. MATLAB was used for image processing, machine learning and deep learning, which is expected to occupy space as well. This work had to face delimitations of RAM and GPU. A few potential high-end experiments that might have benefited the research were not conducted like DenseNet and some were delayed because of such constraints, like implementing UNet. Frameworks that are used for implementation of deep learning, used in this chapter, are MATLAB, Caffe, TensorFlow,Keras using Nvidia CUDA graphics card.

7.3.4 Fine tuning Hyper parameters

For machine learning, there are some parameters that are set before the start of the learning process and they are not influenced by the learning process itself, these are called the hyper-parameters. For deep learning, the training algorithms are almost always iterative, and as such they require a pre-set starting points, that can be used as the basis for the iterations. If this initial state is set badly, the learning time can be greatly increased. It is even possible that the algorithm may never converge at all [92]. There are several strategies for the initial setting of the parameters. The search for the best hyper-parameters is a large part of building any suitable model [120].

Using a pre-trained network with transfer learning is typically much faster and easier than training a network from scratch. In this research, pre-trained models were used hence they required fine-tuning of the parameters. In most researches they appeared to use the stochastic gradient descent (SGD) or mini-batch gradient descent (MBGD) optimization. Mini-batch size is often referred to as batch size and it is usually limited by the CPU or GPU memory requirements.

There are many CNN hyper-parameters that need to be considered and might be tuned for achieving best performance. The use of the dropout technique as a regularization mechanism to avoid over-fitting has become a standard practice in modern CNN designs and not surprisingly, it is used in most of the current researches. Similarly, ReLu is now the de facto standard activation function of deep learning, except for one or two studies that use the sigmoid or tanh activation function in their hidden layers.

7.4 Initial Trials of Deep Learning

In general, apart from the first task, which is research oriented, the rest of the tasks will involve both implementation and evaluation for the initial trial phase of the research. The implementation part will contain the building of the machine learning pipeline, with the goal of allowing interchangeability of networks so a large number of architectures and hyper-parameters can be evaluated with minimal effort, as well as the implementation of the selected network architectures themselves. While the evaluation part will deal with the training runs themselves.

7.4.1 Datasets used for initial trials

For the initial trial phase results, the size of the data is roughly around 125 images, but the data used was not good enough quality as it was taken in the initial phase of data collection. Plus, the labelling method used was also not that sensitive. The selected methods for data collection and labelling, discussed in earlier chapters, had not been identified by that time. Hence the results show quite low performance in general, compared to the remaining later stage algorithms applied. Just by upgrading the dataset made a substantial difference in the performance results, which shows the importance of right data being used for learning.

7.4.2 First Task

For the first task, in order to select the models to be implemented for deep learning, their performance at image recognition and image segmentation challenges is evaluated. Comparing two architectures is not always achievable as they might be trained on different datasets, or with different hyper-parameters, for example differing splits into training and validation datasets. The ones that have no detailed explanation of their architecture and implementation have not been included. For instance, LeNet on the Pascal VOC and ILSVRC datasets shows no previous training so this is excluded from the comparative research. It is the first CNN with a relatively simple architecture that cannot cope with large images.

The two challenge datasets used are Pascal VOC and ILSVRC. There are other challenges available such as the classification of human actions in an image, but the focus here is on the segmentation challenge. The Pascal Visual Object Classes (VOC) challenge provides a large, standardised dataset of images as well as their ground truth for pixel-wise segmentation. The main metric for the Pascal VOC challenge is the Average Precision (AP). The Image-Net Large Scale Visual Recognition Challenge (ILSVRC) is another popular dataset [115] available for image segmentation challenges. It consists of over 14 million labelled images, with bounding boxes for each object to be identified.

The available data for the six selected network architectures shows that the VGGNet can perform the object segmentation with high performance. The GoogLeNet performs worse than the VGGNet architecture on the same dataset [112]. This has been verified by [57] that FCN-VGG16 give better results in comparison to FCN-AlexNet and FCN-googleNet on the basis of mean IU as can be seen in table 7.1.The ResNet in its basic configuration also performs slightly worse than the VGGNet.

While the identified data has some large gaps for several of the architectures, the analysis of available performance evaluations shows that the VGGNet can still be considered state-of-the-art for image segmentation, as demonstrated by its high performance on the Pascal VOC dataset, while the ResNet performs seemingly better for classification tasks on the ILSVRC dataset compared to the VGGNet but in its FCN variant slightly worse on the classification tasks. The DenseNet is closely behind both of these in segmentation performance.

Table 7-1 Performance comparison on common deep learning models

Model	*Pascal VOC*	*ILSVRC*		*References*
	AP (%)	mean IU	top-5 error (%)	
LeNet	-	-	-	
AlexNet	-	39.8	16.4	Long et al. (2015), Krizhevsky et al. (2012)
GoogLeNet	?[2]	42.5	6.7	Long et al. (2015), Krizhevsky et al. (2012)
VGGNet	82.1	56.0	7.3	Long et al. (2015)
ResNet	80.7	-	3.57	Krizhevsky et al. (2012)
DenseNet	78.3	-	8.3	Krapac and Šegvic (2017)

[2] No exact numbers found, but results found at https://github.com/DeepSegment/FCN-GoogLeNet indicate worse performance than VGG)

In conclusion, for the pixel-wise classification, the selected architectures for this part of the work are the VGGNet and ResNet, as they show the best performance on the Pascal VOC image segmentation challenge and as reference implementations for both of them are publicly available. A typical CNN architecture, either inspired by LeNet or VGGNet, with some variations has been used in many researches [97]. The strategy adopted here was to find architectures that have been mentioned through challenge results via a research paper along with an implementation reference. DenseNet shows promise and is a candidate for continuing evaluation in further works. UNet was considered quite late in the research hence it is not included in the initial trials but will be discussed in the experimental testing and training stage in detail. It shows potential, specific to the research study because of its unique shape and design.

Understanding the selected Architectures

VGG16: The VGG network architecture was introduced by Simonyan and Zisserman in their 2014 paper [113], where the authors explored the effect that, depth of a convolutional network, has on its image recognition accuracy. This network is characterized by its simplicity by using only 3×3 convolutional layers stacked on top of each other in increasing depth up to 19 weight layers. This showed a significant improvement on the previous state-of-the-art. The team won the first and the second places in the localization and classification tracks respectively at the ImageNet Challenge 2014 submission. There is a number of different network configurations presented in [113] ranging from 11 to 19 weight layers. But due to its depth and number of fully-connected nodes, they weigh quite large and also it is slow to train them.

VGG is used in many deep learning image classification problems; however, sometimes smaller network architectures are needed such as SqueezeNet, GoogLeNet etc. This VGGNet was adapted by [57], specifically the VGG-16 with 16 weight layers, as fully convolutional. Of the three evaluated architectures, the FCN-VGG16 performs thebest.

ResNet: The deep residual network, or Resnet in short, is a very deep neural network architecture, also known as network-in-network architecture. It was developed [117] with the deepest evaluated network having a depth of 152 layers, about 8 times deeper than the VGGNet architecture. It demonstrates that extremely deep networks can be trained using standard SGD and a reasonable initialization function, by using residual modules. ResNet is a form of exotic architecture that banks on micro-architecture modules unlike the traditional sequential network architectures such as AlexNet and VGG. The term micro-architecture refers to the set of “building blocks” used to construct the network. A collection of micro-architecture building blocks, including the standard CONV, POOL, etc. layers, leads to the macro-architecture itself.

7.4.3 Second task

For the second task, different configurations of the selected architectures, VGGNet and ResNet are run, with identical hyper-parameters to allow a comparison between the runs.

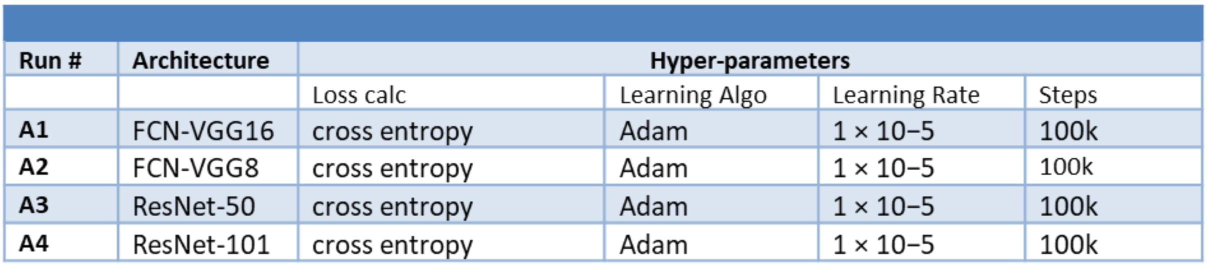
With the two main network architectures under evaluation selected earlier as the FCN-VGGNet and the ResNet, there is a number of different forms of these two architectures to be evaluated. The FCN-VGGNet is evaluated in the FCN-VGG16 version and the FCN-VGG8 version. These two differ only in the architecture of their final, fully-connected layer. While the FCN-VGG16 combines the predictions of the final convolutional and the fourth pooling layer to form upsampled predictions with a stride of 16, the FCN-VGG8 does so with stride 8 upsampled predictions and combining not only the fourth pooling and last convolutional but also the third pooling layer together. According to [57], the FCN- VGG8 should thus provide increased precision compared to the other architecture. Hence both of these models have been applied in order to verify this claim of improved performance of the latter architecture. The ResNet is available in a large number of different versions. For this study, Resnet-50 with a total number of 50 residual layers and Resnet-101 with a total number of 101 residual layers were selected. This shows higher performance of the Resnet-101 when compared to the Resnet-50 but again this has been verified for the available dataset by running both architectures with identical hyper-parameter configurations.

Table 7-2 Deep Learning dataset split for the initial trials

Stage	Percentage	Total	Positive examples	Negative examples
Training	80 %	100	59	41
Validation	15 %	18	11	7
Testing	5 %	8	5	3

This dataset is from the initial image collection. The dataset is split into a training, validation and test dataset as shown in Table 7.2.

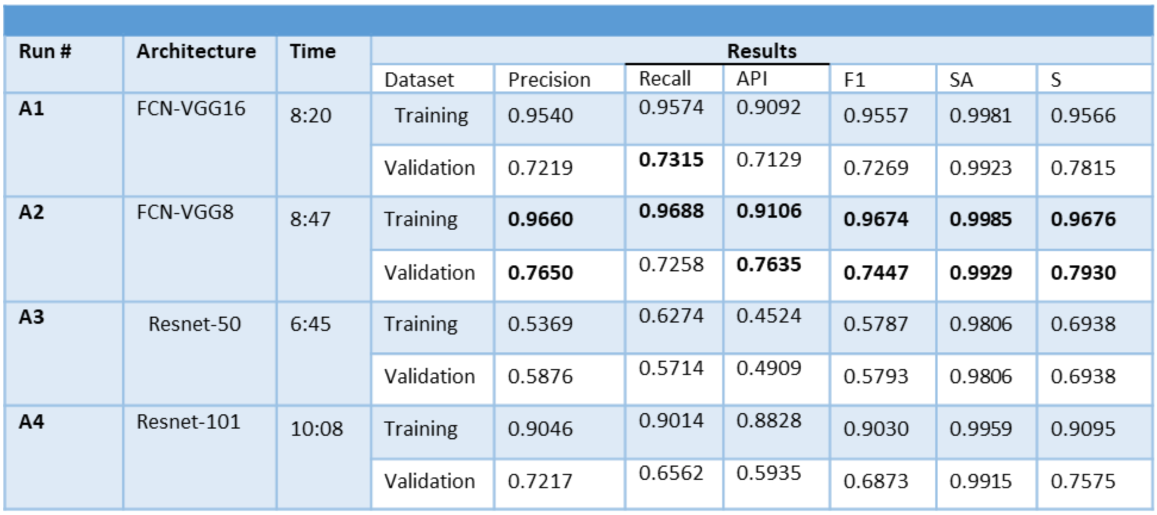
Table 7-3 Listing of runs performed in the network architecture evaluation stage



Run #	Architecture	Hyper-parameters			
		Loss calc	Learning Algo	Learning Rate	Steps
A1	FCN-VGG16	cross entropy	Adam	1 × 10−5	100k
A2	FCN-VGG8	cross entropy	Adam	1 × 10−5	100k
A3	ResNet-50	cross entropy	Adam	1 × 10−5	100k
A4	ResNet-101	cross entropy	Adam	1 × 10−5	100k

The hyper-parameters are set to a comparable default that is unchanged for all four of the architecture evaluation runs as shown in table 7.3. The loss calculation is done by cross entropy for all runs, where cross entropy is defined as the arithmetic mean of the cross entropy of all images, and the cross entropy of a single image is calculated by combining the pixel-wise classification loss in the image. The learning algorithm used is Adam, with a learning rate of 10 × 10−5. There are a total of 100000 iterations performed for each run. Due to the constant batch size of 1 image, this means that it will take 100 iterations to go through the entire training dataset once, resulting in a total of 1000 epochs for training.

Table 7-4 Deep Learning initial trial results of performance measure



Run #	Architecture	Time	Results						
			Dataset	Precision	Recall	API	F1	SA	S
A1	FCN-VGG16	8:20	Training	0.9540	0.9574	0.9092	0.9557	0.9981	0.9566
			Validation	0.7219	**0.7315**	0.7129	0.7269	0.9923	0.7815
A2	FCN-VGG8	8:47	Training	**0.9660**	**0.9688**	**0.9106**	**0.9674**	**0.9985**	**0.9676**
			Validation	**0.7650**	0.7258	**0.7635**	**0.7447**	**0.9929**	**0.7930**
A3	Resnet-50	6:45	Training	0.5369	0.6274	0.4524	0.5787	0.9806	0.6938
			Validation	0.5876	0.5714	0.4909	0.5793	0.9806	0.6938
A4	Resnet-101	10:08	Training	0.9046	0.9014	0.8828	0.9030	0.9959	0.9095
			Validation	0.7217	0.6562	0.5935	0.6873	0.9915	0.7575

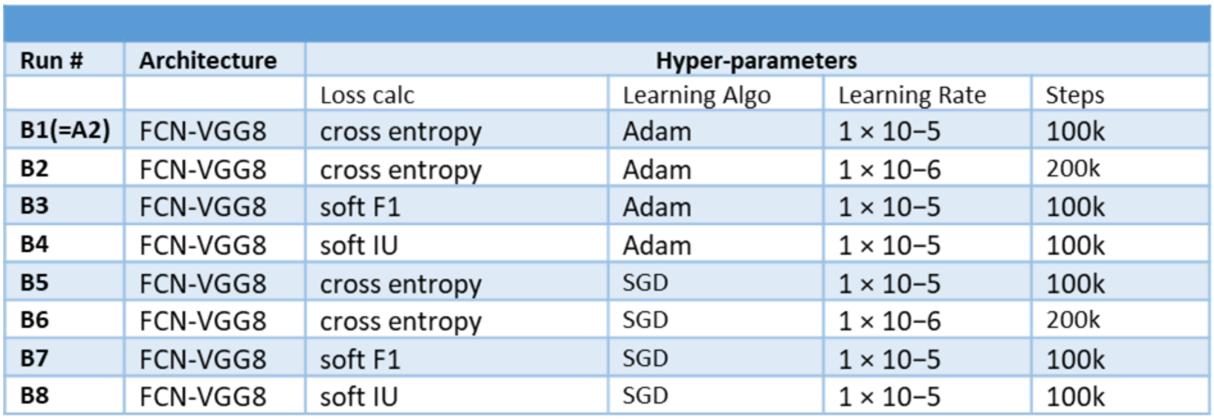
The results in table 7.4 show how important the correct selection of evaluation criteria is in order to measure the performance of the models: consistently for all runs, the Segmentation Accuracy (SA) is very high (> 99 %) compared to all other metrics. This is because the Segmentation Accuracy measures if each pixel has been correctly classified, and in the dataset used there is a large discrepancy between the number of pixels that should correctly be classified as negatives (24226334/24576001 = 0.9858% of all pixels in the validation dataset) and those that should be classified as positives. The other metrics only take into account segmentation performance for the positive class, and accordingly show values that look much worse.

The weight loss for the Resnet trainings takes longer to approach zero than the weight loss for the VGGNets. As the batch size is set to 1 for all runs, the weight loss is unlikely to reach exactly zero as there is a different image trained each step and as such even when a model would already have achieved its minimum, there is always a small weight loss due to the difference in each image. The continuous plots of the F1 and Average Precision score show that after already a comparatively small amount of training steps (20000 for the VGGNets, 30000 for the ResNets) there is little improvement in either score when calculated for the validation dataset. There are high fluctuations between the performances of each evaluated training step. This may be caused by the small batch size causing weight rebalancing in each step. The Resnet-50 takes the shortest time to train, with 6:45 hours, run on an Nvidia Geforce 1080Ti GPU. The FCN-VGG8 takes a bit longer than the FCN-VGG16 with 8:47 and 8:20 hours respectively, while the Resnet-101 takes the longest with 10:08 hours to train.

7.4.4 Third task

For the third task, the better architecture is selected and the hyper-parameters are optimized for this architecture, with identical network architecture so as to allow a comparison between the runs. With the network selected as the FCN-VGG8, the next step is the hyper-parameter configuration of the network so as to maximise the segmentation performance, as measured by the main metrics described in chapter 4.10. There are a total of eight runs performed during this phase as shown in table 7.5. Four runs with the Adam learning algorithm and four runs with the stochastic gradient descent algorithm. Runs with the Adam learning algorithm use 1 × 10−5 as the default learning rate. The learning rate is adapted to 1 × 10−6 for run B2, and the total number of steps increased to 200,000. SGD runs use a default learning rate of 0.01, with run B6 adapting this to train with a learning rate of 0.001 and 200,000 steps. Furthermore, each learning algorithm is trained once with the loss calculation done using cross entropy, once with a soft F1 score, and once with a soft IU score.

Table 7-5 Listing of trial runs performed in the hyper-parameter evaluation stage

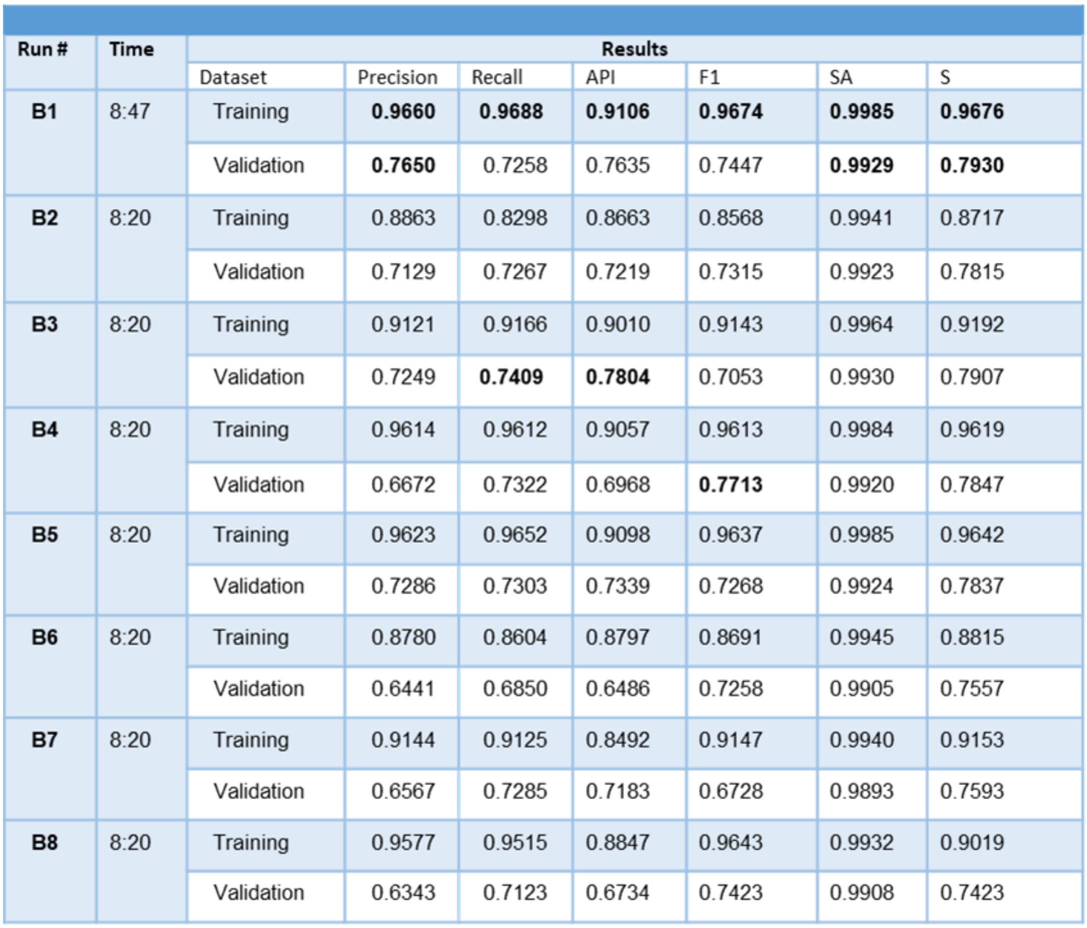


Run #	Architecture	Hyper-parameters			
		Loss calc	Learning Algo	Learning Rate	Steps
B1(=A2)	FCN-VGG8	cross entropy	Adam	1 × 10−5	100k
B2	FCN-VGG8	cross entropy	Adam	1 × 10−6	200k
B3	FCN-VGG8	soft F1	Adam	1 × 10−5	100k
B4	FCN-VGG8	soft IU	Adam	1 × 10−5	100k
B5	FCN-VGG8	cross entropy	SGD	1 × 10−5	100k
B6	FCN-VGG8	cross entropy	SGD	1 × 10−6	200k
B7	FCN-VGG8	soft F1	SGD	1 × 10−5	100k
B8	FCN-VGG8	soft IU	SGD	1 × 10−5	100k

The cross entropy loss function increases if the predicted probability diverges more from the actual label. Another loss function used for training during this phase is the soft F1 which applies the F1 score for the loss function. The third metric that is evaluated for the loss calculation is the soft Intersection over Union (soft IU), which calculates the mean IU and again subtracts it from 1, thereby calculating a loss between 0 and 1.

The results in table 7.6 show that the loss calculation has a minor but noticeable impact on the performance of the final model. The models using the cross entropy loss function consistently outperform the other models when measured on the precision, SA and S scores. However, the best recall and best average precision score is achieved in run B3 which applies cross entropy loss. The effects are similar for the runs applying SGD learning algorithm.

Table 7-6 DL trial model results performed in the hyper-parameter evaluation stage



Run #	**Time**	**Results**						
		Dataset	Precision	Recall	API	F1	SA	S
B1	8:47	Training	**0.9660**	**0.9688**	**0.9106**	**0.9674**	**0.9985**	**0.9676**
		Validation	**0.7650**	0.7258	0.7635	0.7447	**0.9929**	**0.7930**
B2	8:20	Training	0.8863	0.8298	0.8663	0.8568	0.9941	0.8717
		Validation	0.7129	0.7267	0.7219	0.7315	0.9923	0.7815
B3	8:20	Training	0.9121	0.9166	0.9010	0.9143	0.9964	0.9192
		Validation	0.7249	**0.7409**	**0.7804**	0.7053	0.9930	0.7907
B4	8:20	Training	0.9614	0.9612	0.9057	0.9613	0.9984	0.9619
		Validation	0.6672	0.7322	0.6968	**0.7713**	0.9920	0.7847
B5	8:20	Training	0.9623	0.9652	0.9098	0.9637	0.9985	0.9642
		Validation	0.7286	0.7303	0.7339	0.7268	0.9924	0.7837
B6	8:20	Training	0.8780	0.8604	0.8797	0.8691	0.9945	0.8815
		Validation	0.6441	0.6850	0.6486	0.7258	0.9905	0.7557
B7	8:20	Training	0.9144	0.9125	0.8492	0.9147	0.9940	0.9153
		Validation	0.6567	0.7285	0.7183	0.6728	0.9893	0.7593
B8	8:20	Training	0.9577	0.9515	0.8847	0.9643	0.9932	0.9019
		Validation	0.6343	0.7123	0.6734	0.7423	0.9908	0.7423

Run B1 scores the highest on all metrics when measured on the training dataset, but it beats the other runs only in precision, SA and S score when measured on the validation dataset. This shows that there is not necessarily a correlation between the performance on the training and on the validation dataset.

Both runs with the learning rate decreased and the step number doubled perform worse than the runs with the same hyper-parameters but higher learning rate and lower step number, showing that 1 × 10−6 for Adam and 0.001 with SGD is too low of a learning rate to use. The weight loss plots of runs B2 and B5 show that the weight loss does not go down significantly enough even after all 200,000 steps. At the same time, the time required for training is doubled. Run B3 performs the best when measured on the average precision and F1 score, and is thus selected as the best performing model.

7.4.5 Fourth task

Finally in the fourth task, the best-performing architecture and hyper-parameter configuration is identified on the basis of the initial trials and data used. Run B3, using the FCN-VGG8 architecture with soft F1 loss calculation, the Adam learning algorithm with a learning rate of

1×10^{-5} and 100,000 steps of training achieved the best performance measured both on the Average Precision and the F1 score. The achieved Average Precision score of 0.7804 is lower than the 0.821 score that [57] achieved when training on the Pascal VOC dataset, however this may be in larger part due to the much smaller dataset size (126 images compared to almost 9,993) and could be improved by training on a larger dataset. Evaluating the run B3 on the test dataset, which consists of 8 images that have so far been completely disregarded, gives an Average Precision performance of 0.7756, very similar to the score achieved on the validation dataset, showing the ability of the trained model to generalize to previously unknown data.

7.4.6 Conclusion

The experiments started with the design phase, where available state of the art network architectures were identified. This provided a basis for first task of the research which was to perform subsequent evaluation in order to select comparatively better models. Later on, for the second task, the most suitable network architectures selected, were to be compared and evaluated in further depth by varying the hyper-parameters. Alongside choosing network architectures, frameworks were also identified and selected from a large number of available frameworks that could be used. Experiments were performed using either Tensor Flow or MATLAB. They ran on the same Nvidia graphics card, allowing a direct comparison of the performance of all runs to each other.

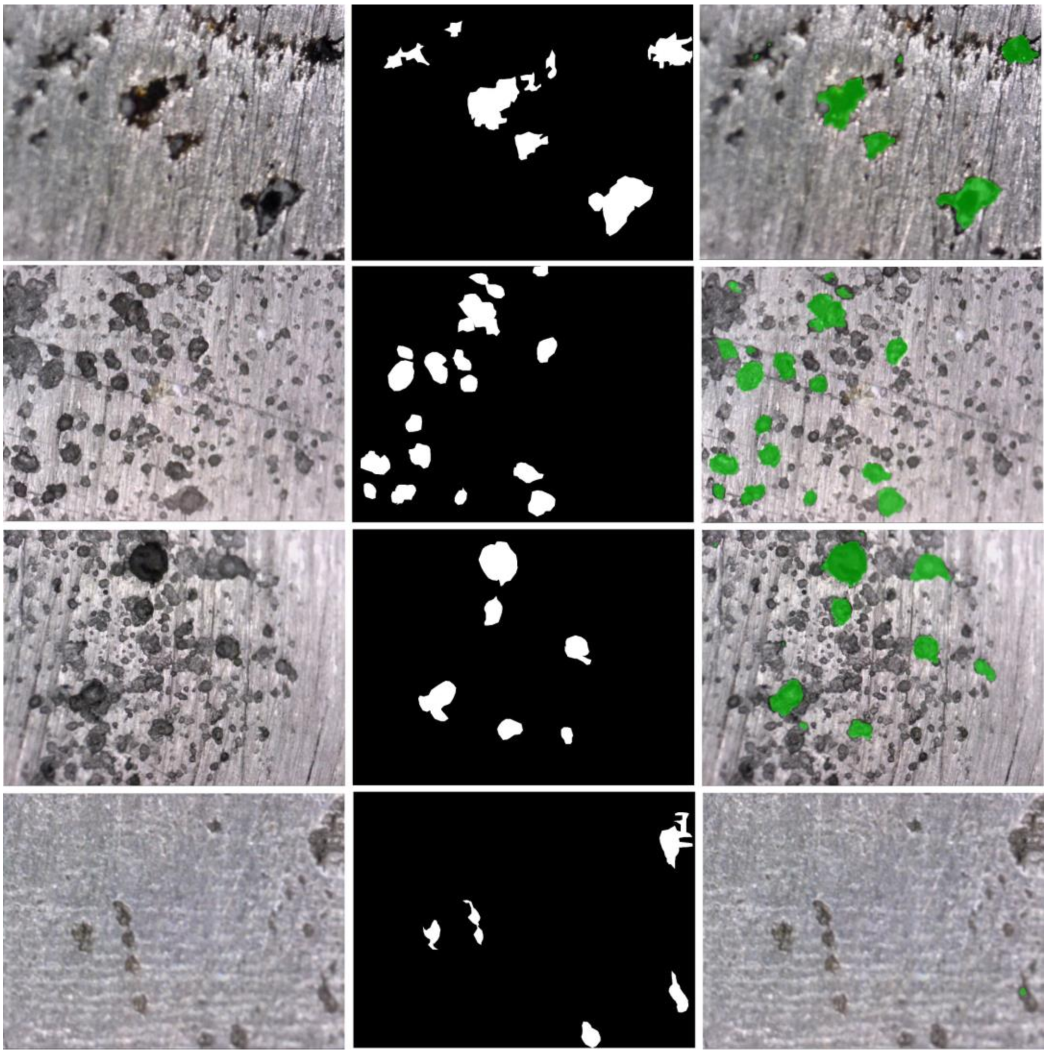


Figure 7-3 Deep Learning pixel-wise classification results in the initial trials

Although these initial trials did not produce good results because of the dataset used, but it laid a good foundation. Deep learning outputs of the run B3 trained model from the initial trials is shown in figure 7.3. In the figure, the left side shows the original image, the middle shows the ground truth and the right side is the output from the trained model. Basically, all these experiments in the initial trials were performed so that a base line setup of the system could be implemented and better understanding of deep learning models could be reached, which was achieved.

7.5 Image-wise Classification using pre-trained AlexNet

It is one of the pioneer Deep Neural Net that was developed in [112]. It won the ImageNet challenge in 2012 by a large margin as one of the first CNNs to be used for image classification problems. It was initially trained to recognize 1000 different objects. It achieved a top5 error rate of 15.3 %, over 10 % better than the runner-up that did not apply CNNs [115]. From then onwards, more or less following nets are based on its architecture [157].

7.5.1 Architecture

It contains five convolutional layers from C1 to C5 plus three fully connected layers from FC6 to FC8 with the final one being a softmax output layer as shown in figure 7.4. Additionally, it has three layers applying max pooling as shown in table 7.7. It uses ReLU activation function instead of Sigmoid or Tanh functions which makes it speed up by more than 5 times with same accuracy. To reduce overfitting, dropout is applied in first two fully-connected layers with a probability of 0.5 for each neuron. At test time, output of each neuron is multiplied by 0.5 [112].

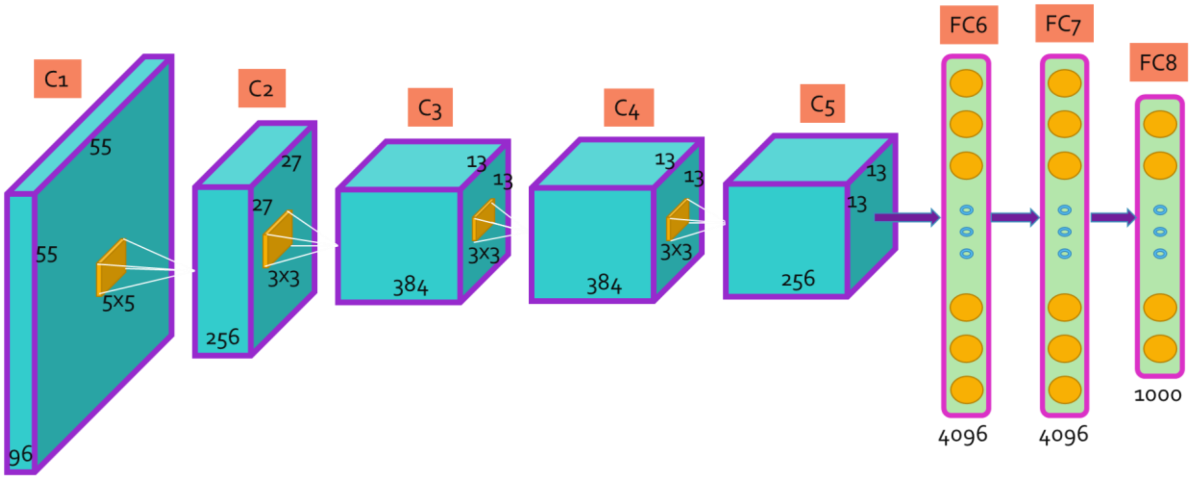


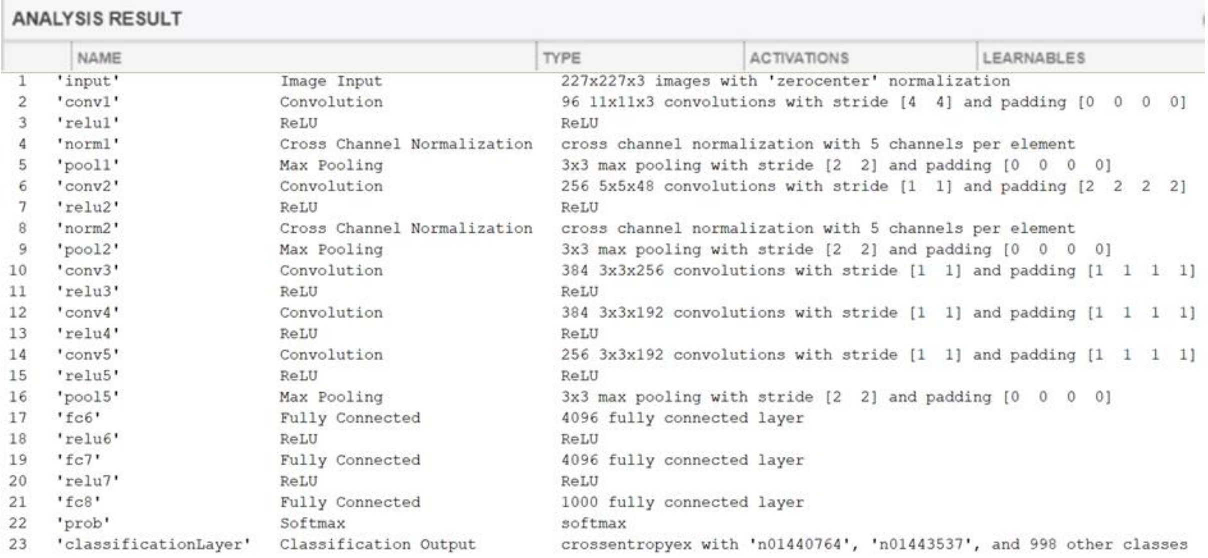
Figure 7-4 Architecture of the implemented pretrained AlexNet Deep Learning Model

It is designed as a fully convolutional network in [57], however it is not able to accomplish the same image segmentation performance as its more complex and innovative rival, VGG architecture.

7.5.2 Implementation

Pre-trained Alex has been implemented for the data-driven application to distinguish between cracks and pits. The splitting of the labels is done by proportions and then each class data is split 0.7% for training data and 0.3% for test data.

Table 7-7 Architecture of implemented Image-wise Classification with full details



```
ANALYSIS RESULT
     NAME                             TYPE                          ACTIVATIONS              LEARNABLES
 1   'input'               Image Input                   227x227x3 images with 'zerocenter' normalization
 2   'conv1'               Convolution                   96 11x11x3 convolutions with stride [4  4] and padding [0  0  0  0]
 3   'relu1'               ReLU                          ReLU
 4   'norm1'               Cross Channel Normalization   cross channel normalization with 5 channels per element
 5   'pool1'               Max Pooling                   3x3 max pooling with stride [2  2] and padding [0  0  0  0]
 6   'conv2'               Convolution                   256 5x5x48 convolutions with stride [1  1] and padding [2  2  2  2]
 7   'relu2'               ReLU                          ReLU
 8   'norm2'               Cross Channel Normalization   cross channel normalization with 5 channels per element
 9   'pool2'               Max Pooling                   3x3 max pooling with stride [2  2] and padding [0  0  0  0]
10   'conv3'               Convolution                   384 3x3x256 convolutions with stride [1  1] and padding [1  1  1  1]
11   'relu3'               ReLU                          ReLU
12   'conv4'               Convolution                   384 3x3x192 convolutions with stride [1  1] and padding [1  1  1  1]
13   'relu4'               ReLU                          ReLU
14   'conv5'               Convolution                   256 3x3x192 convolutions with stride [1  1] and padding [1  1  1  1]
15   'relu5'               ReLU                          ReLU
16   'pool5'               Max Pooling                   3x3 max pooling with stride [2  2] and padding [0  0  0  0]
17   'fc6'                 Fully Connected               4096 fully connected layer
18   'relu6'               ReLU                          ReLU
19   'fc7'                 Fully Connected               4096 fully connected layer
20   'relu7'               ReLU                          ReLU
21   'fc8'                 Fully Connected               1000 fully connected layer
22   'prob'                Softmax                       softmax
23   'classificationLayer' Classification Output         crossentropyex with 'n01440764', 'n01443537', and 998 other classes
```

Network weights for individual convolutional layers can be visualised as seen in figure 7.5, 7.6 & 7.7, after scaling and re-sizing the weights. Figure 7.5 shows montage of Image-wise network for iw-db3-crack, weights of the first convolution layer or 3rd layer of the model whereas figure 7.6 and 7.7 shows fc7 and fc8.

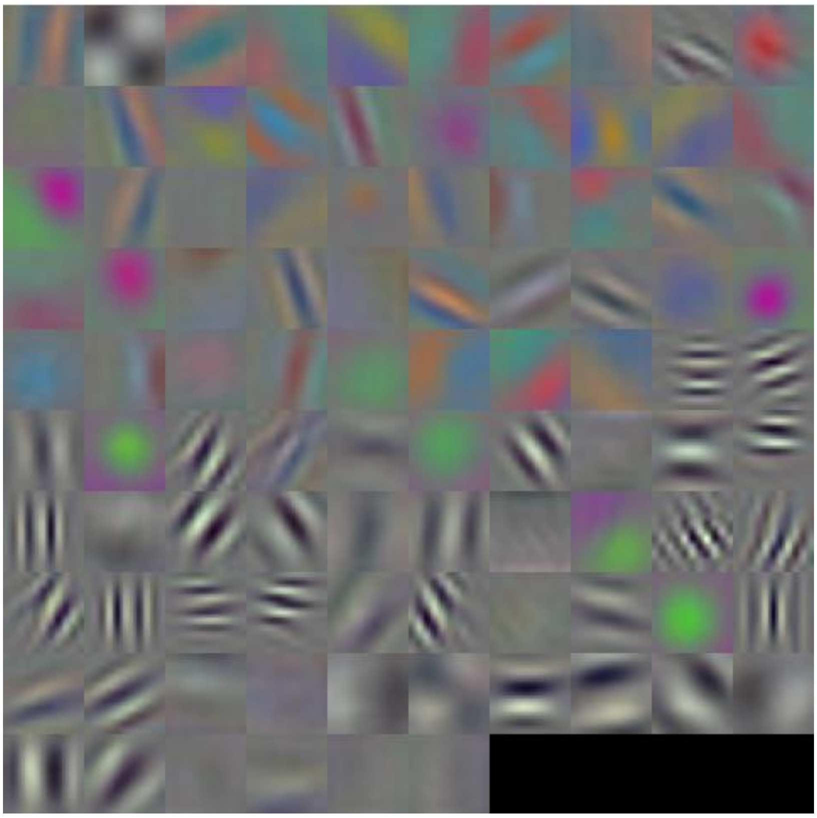


Figure 7-5 Montage of Image-wise network weights of the first convolution layer

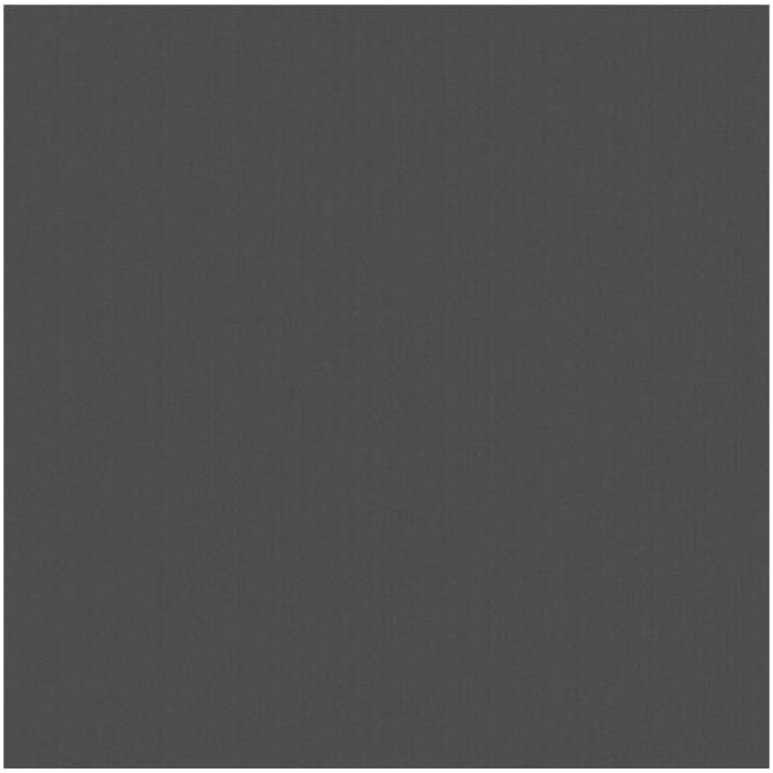


Figure 7-6 Montage of Image-wise network weights of the fc7 convolution layer

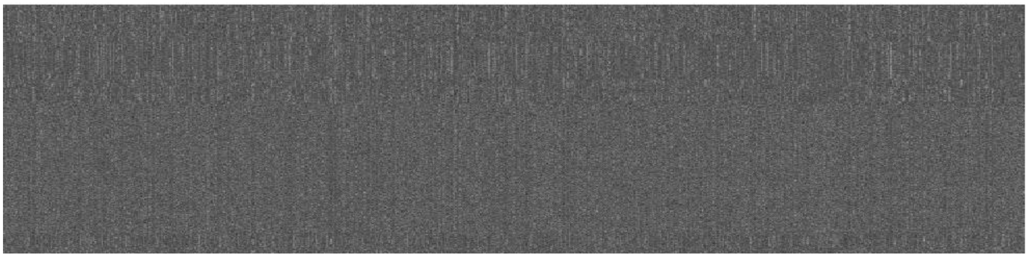


Figure 7-7 Montage of network weights of fc8 which is last convolution layer

7.5.3 Datasets for Image-wise classification

The image wise data sets have been created to use them to classify between a pit and a crack flaw as shown in table 7.8. This is basically to answer the ‘what’ question of classification. For the initial stages different variations were tried and tested which are named as iw_db1, iw_db2 and so on, where, iw=image_wise and db=database. Later on, for image-wise classification for deep learning two models were designed separately; Pit or not-pit model, and crack or not-crack model.

Table 7-8 Different datasets used for Image-wise classification using AlexNet

DB #	DB consists of: /Classification type	No of images	Classes
iw_db1	{pit, crack and both} where Pit = Pit only, crack = crack only and both = images with pits and/or cracks + pit2crack (p2c) images.	272	3
Iw_db2	crack, crackNot, where crack = crack images and crackNot = all the other images including background, corrosion, Pit2crack and pits.	-	2
Iw_db3	Crack Classifier Model with categories = {crackOnly, crackNot}, where crack = crackOnly images and crackNot = all the other images including background, corrosion and pits but not Pit2crack images.	-	2
Iw_db4	pit, NotPit, where pit = PitOnly + pit with slight corrosion images and NotPit = includes background, crackOnly and general corrosion crack, crackNot where crack = crackOnly + pit2crack images and crackNot = includes background, very evident corrosion and pit only	193 162	2 2

Different database structure and classes produced different outcomes from the classifier. The complexity of the image categories based around useful life expectancy of a component, is displayed in figure 7.8. It is a challenging task to categorise them into different classes. To make multi-class data easier to understand, it is visualised in an image.

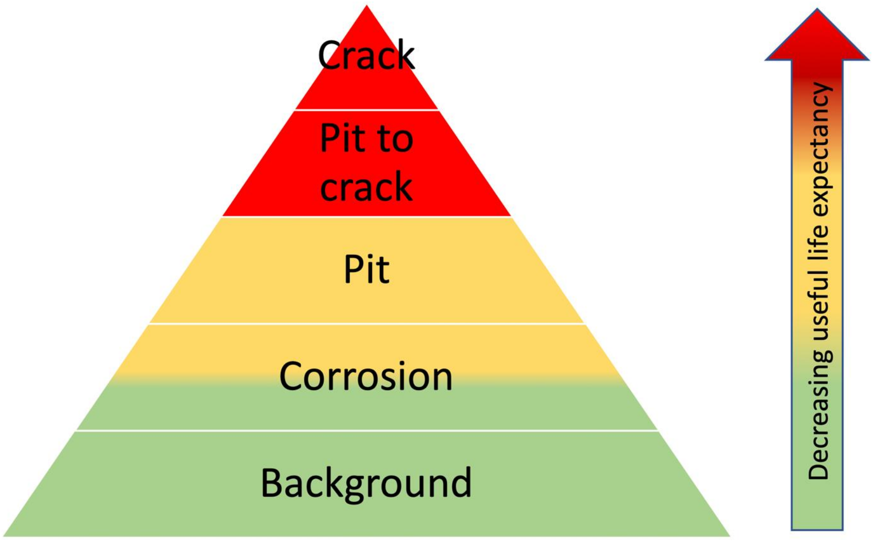
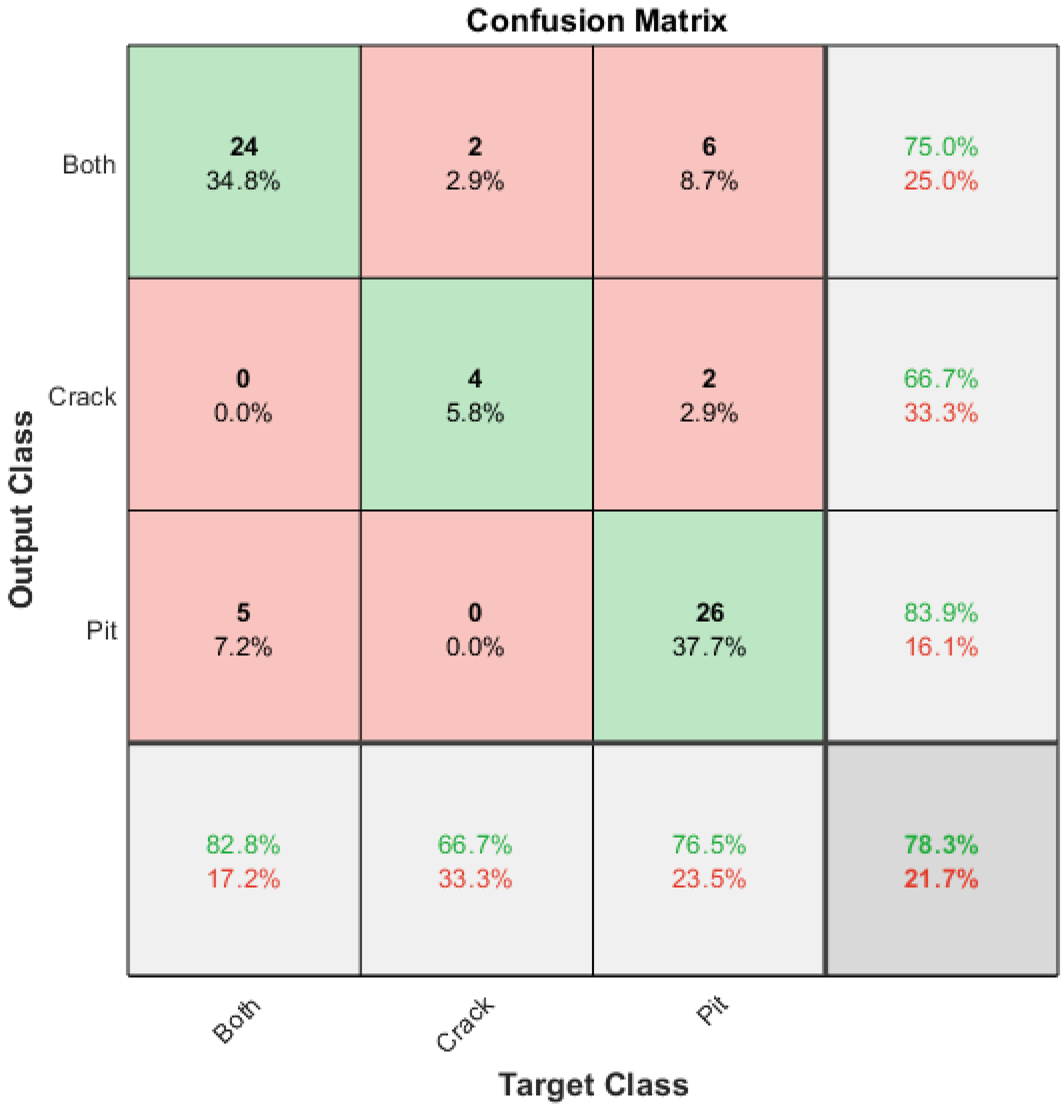


Figure 7-8 Showing project complexities of multi-class data

7.5.4 Results of Image-wise classification by Confusion Matrix

First set: In the first trial, categories = {pit, crack and both} where Pit = Pit only, crack = crack only and both = images with pits and/or cracks + pit2crack (p2c) images. Total images used are 272 in total, out of which background = 8, both = 96, crack = 21, pit2crack = 4, and pit = 155. The splitting is set to 0.7% (191 images) for training data and 0.3% (81 images) for test data. The results to such labelled data was not efficient giving just 78% accuracy as shown in figure 7.9 by using confusion matrix. This was caused by the confused image-wise labelling itself. The 'both' category labelling consisted of cracks, pits and also pit2cracks so it didn't know which category to place it in. This was incorrect labelling. This is why appropriate labelling is crucial for deep learning systems.



Confusion Matrix

	Both	Crack	Pit	
Both	24 34.8%	2 2.9%	6 8.7%	75.0% 25.0%
Crack	0 0.0%	4 5.8%	2 2.9%	66.7% 33.3%
Pit	5 7.2%	0 0.0%	26 37.7%	83.9% 16.1%
	82.8% 17.2%	66.7% 33.3%	76.5% 23.5%	78.3% 21.7%

Output Class
Target Class

Figure 7-9 Deep Learning Image-wise classification results on iw_db1 by Confusion Matrix

<u>Second set</u> of the database attempted for binary classification for a Crack Classifier Model with categories = {crack, crackNot}, where crack = crack images and crackNot = all the other images including background, corrosion, Pit2crack and pits. This immediately improved the accuracy to 91.7% as shown in figure 7.10.

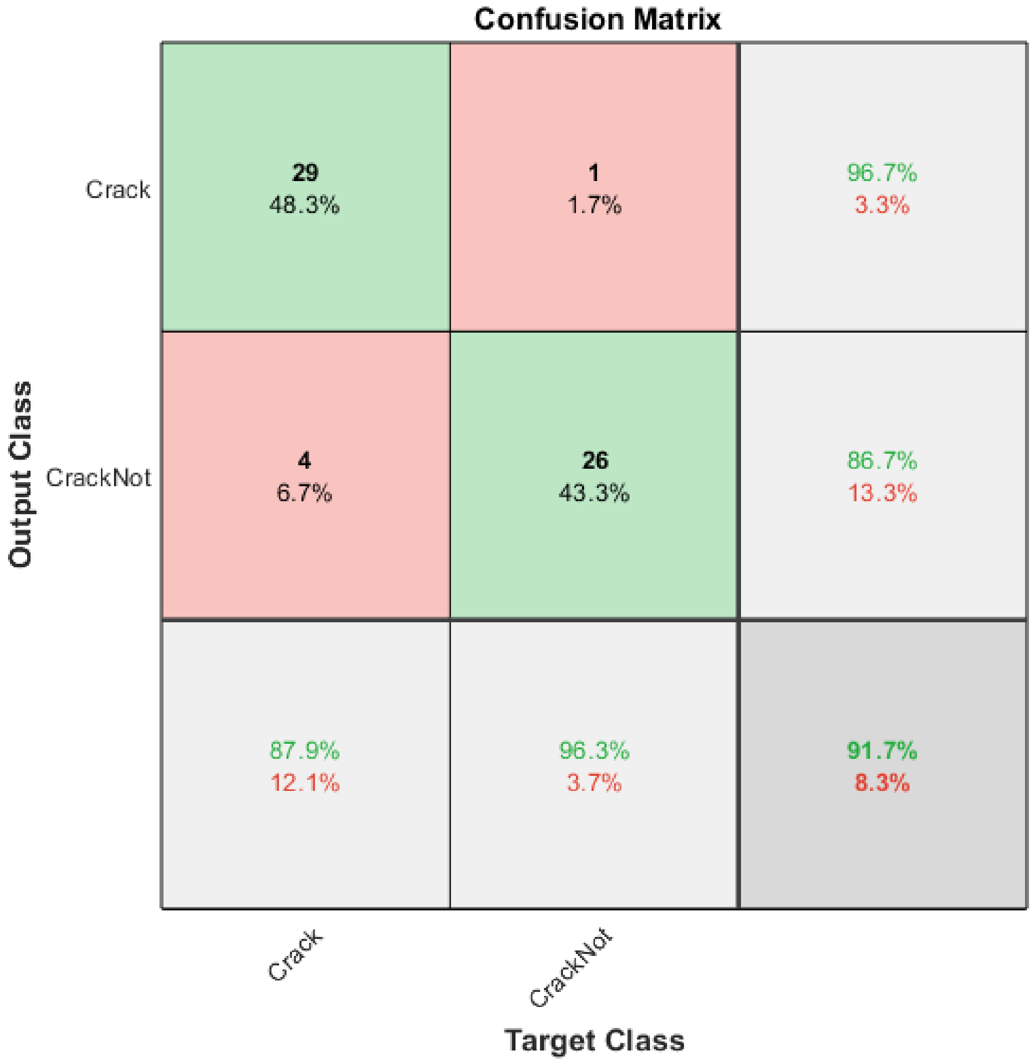


Figure 7-10 Deep Learning Image-wise classification results on iw_db2 by Confusion Matrix

<u>Third set</u> of the database (iw_db3) also based for binary classification for a Crack Classifier Model with categories = {crackOnly, crackNot}, where crack = crackOnly images and crackNot = all the other images including background, corrosion and pits but not Pit2crack images. This bounced the accuracy to 100% as shown in figure 7.11. But this is not an inclusive structure as it does not include an important category p2c, where a pit is initiating to become a crack hence it has both a pit and a crack within itself. This was the most difficult category to deal with in the research for data labelling task.

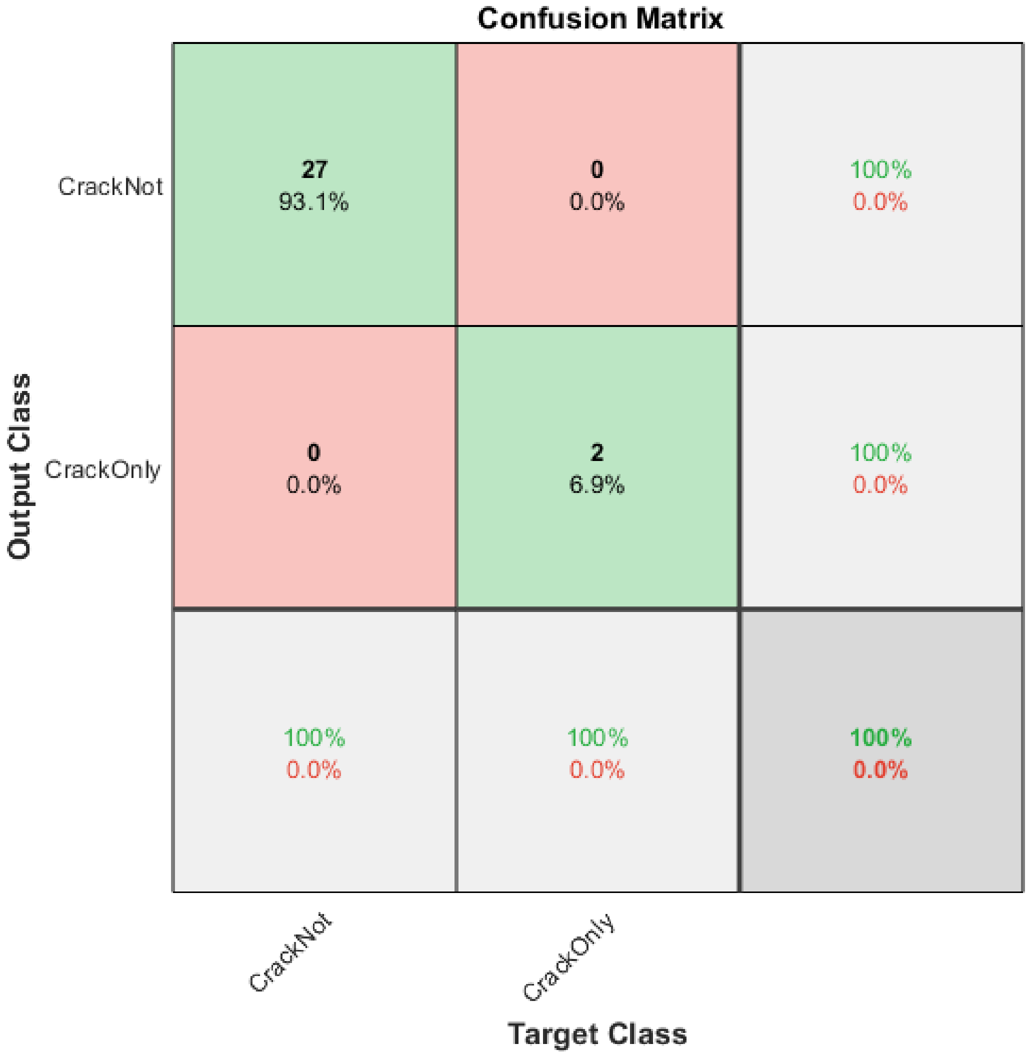


Figure 7-11 Deep Learning Image-wise classification results on iw_db3 by Confusion Matrix

Proposed system consists of two binary classification models; a Crack Classifier Model and a Pit Classifier Model separately based on database (iw_db4)

Pit model consists of categories = {pit, NotPit }, here pit = PitOnly + pit with slight corrosion images and NotPit = includes background, crackOnly and general corrosion images. It contains total 193 images with 135 for training. This accuracy for the first pit model classifier is 91.4% as shown in figure 7.12.

Crack model consists of categories = {crack, crackNot}, but here crack = crackOnly + pit2crack images and crackNot = includes background, very evident corrosion and pit only images. It contains total 162 images with 113 for training. This improved the accuracy to 98% which is still higher in comparison to the second model, as shown in figure 7.13.

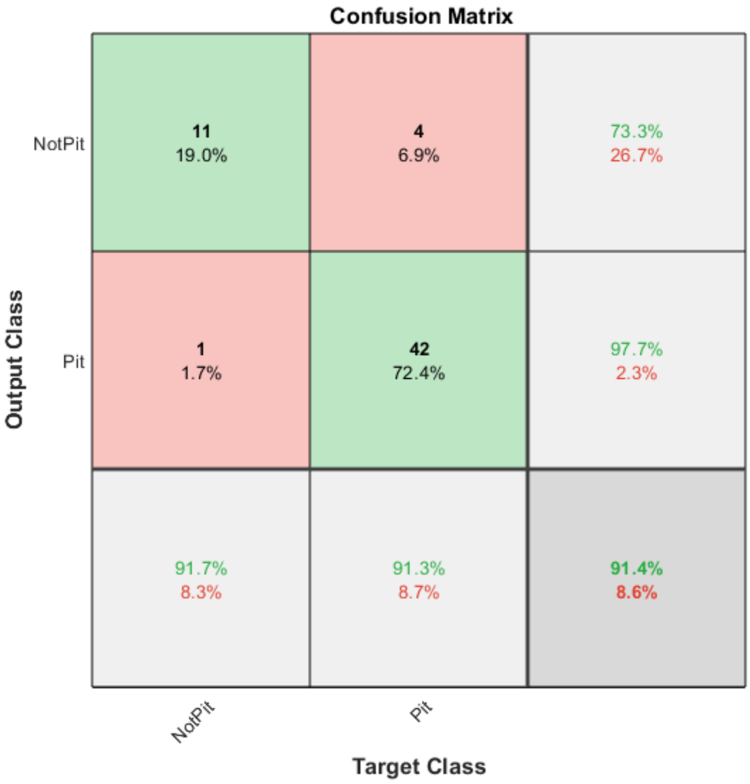


Figure 7-12 Deep Learning Image-wise classification results for Pit model

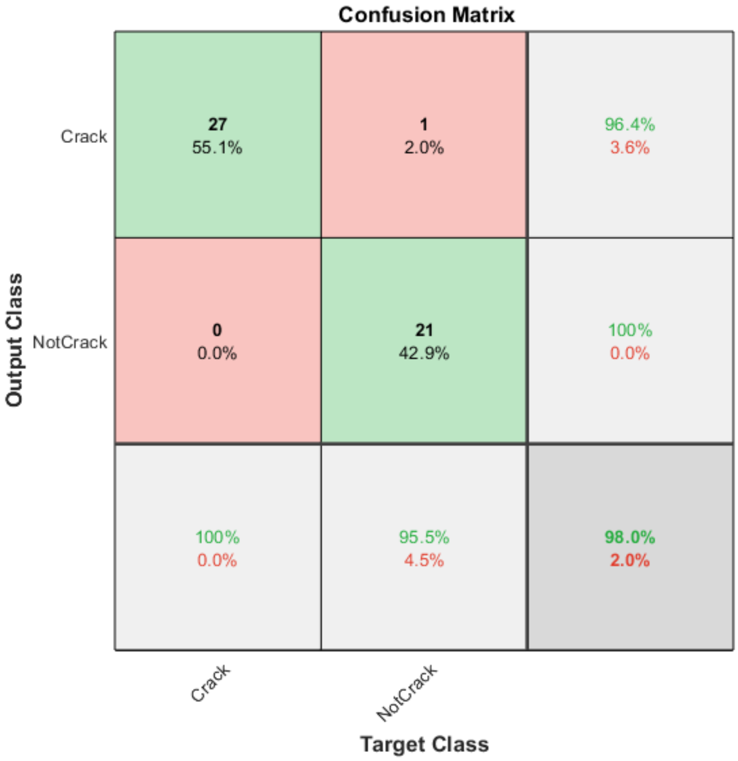


Figure 7-13 Deep Learning Image-wise classification results for Crack model

7.5.5 Results of image-wise classification on test data

Few examples of the resultant outcomes from the system can be seen in figure 7.14. The predicted classified classes, from being classified as a crack or not a crack, can be seen on the top of the test image.

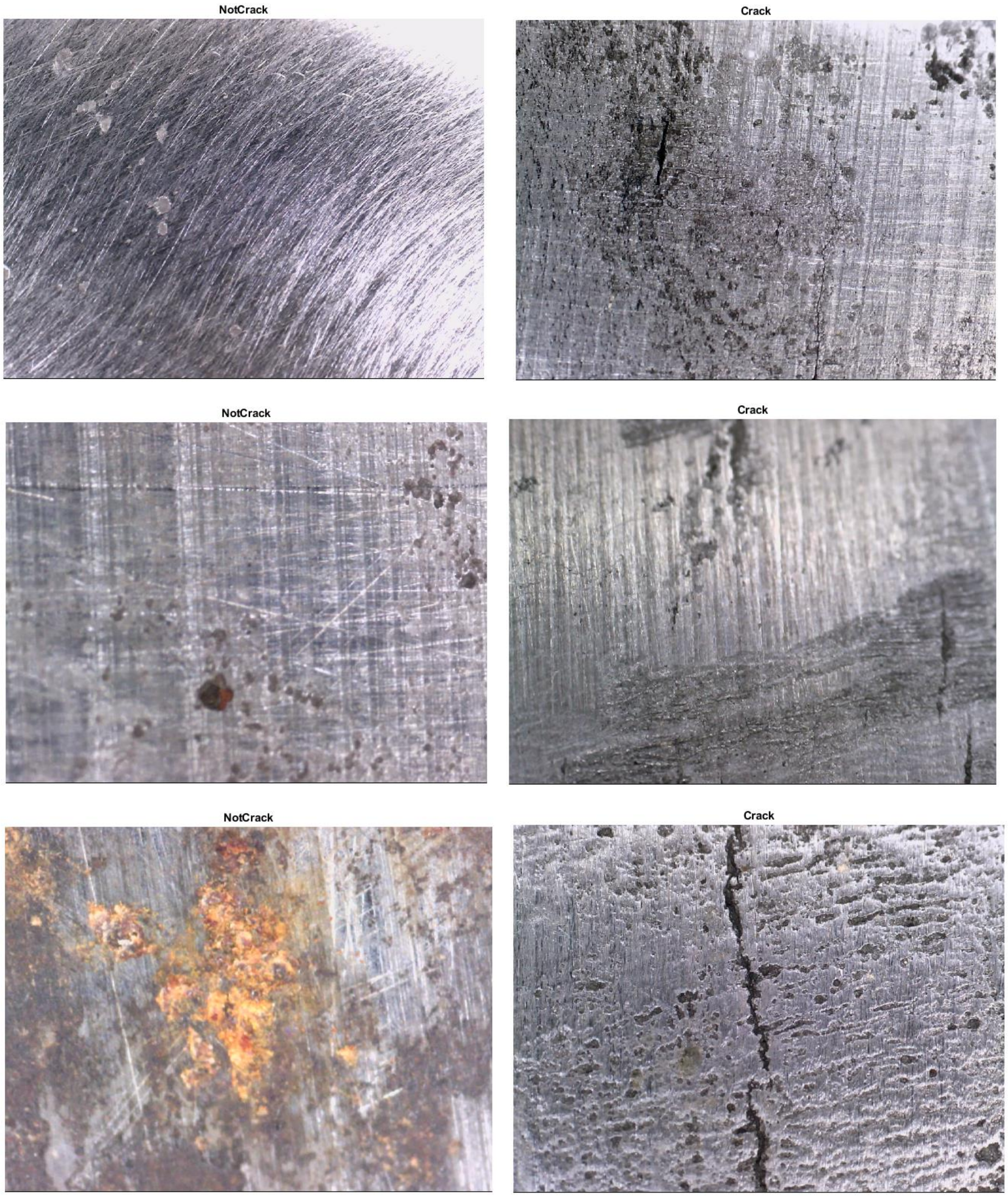


Figure 7-14 Deep Learning Image-wise classification results by the proposed models

7.5.6 Conclusion

The image-wise classification model, pre-trained with AlexNet, is the first model to be implemented after the initial trails. This classifier consists of two binary sub-models, a Crack Classifier Model and a Pit Classifier Model. With the combination of both models, the system is able to classify between a crack and a pit. The crack model shows whether it is a crack or not, while the pit model classifies the images into pit or not. The resultant outcome displays the class of the image to which it belongs. If the image has a pit, then the Pit model is able to pick it with 91.4% accuracy, and if the image has a crack, it is picked by the Crack model with a high accuracy of 98% as shown in figure 7.15 using confusion matrix and visual outcomes.

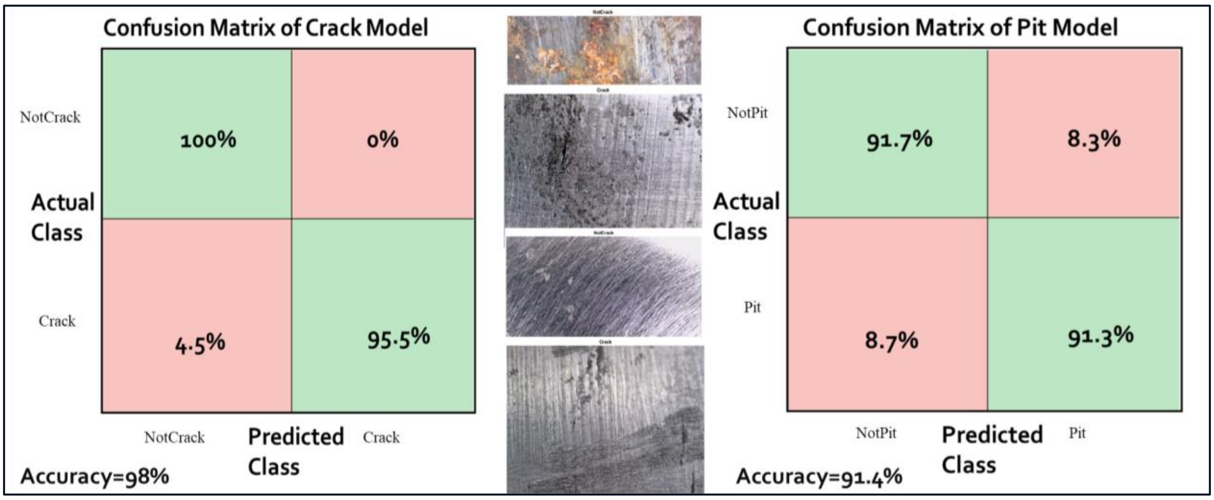


Figure 7-15 Test data result of Image-wise classification of both models

7.6 PIXEL-WISE UNET MODEL

UNet is convolutional network architecture for fast and precise segmentation of images. It performed better than sliding-window convolutional network method on the ISBI challenge for segmentation which shows its potential [158]. For image segmentation, the datasets consist of at most thousands of images in most cases because it requires manual labelling of the ground truth which is a very costly procedure. UNet has a capacity of learning from relatively small training sets, which is why it is selected to be experimented in this research. Not much work has been done with this architecture apart for a few medical applications.

7.6.1 Architecture

The key architecture consists of a contracting path to capture context and of a symmetrically expanding path that enables precise localization [159] as shown in figure 7.16.

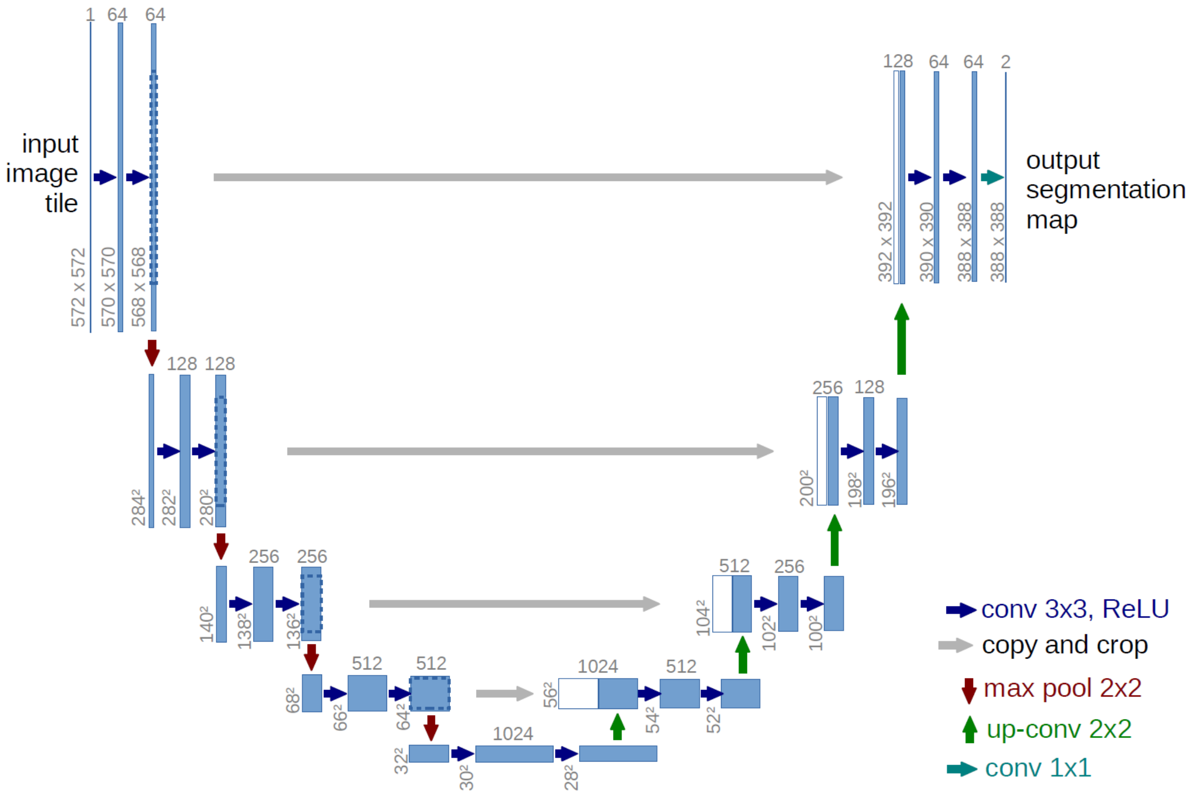


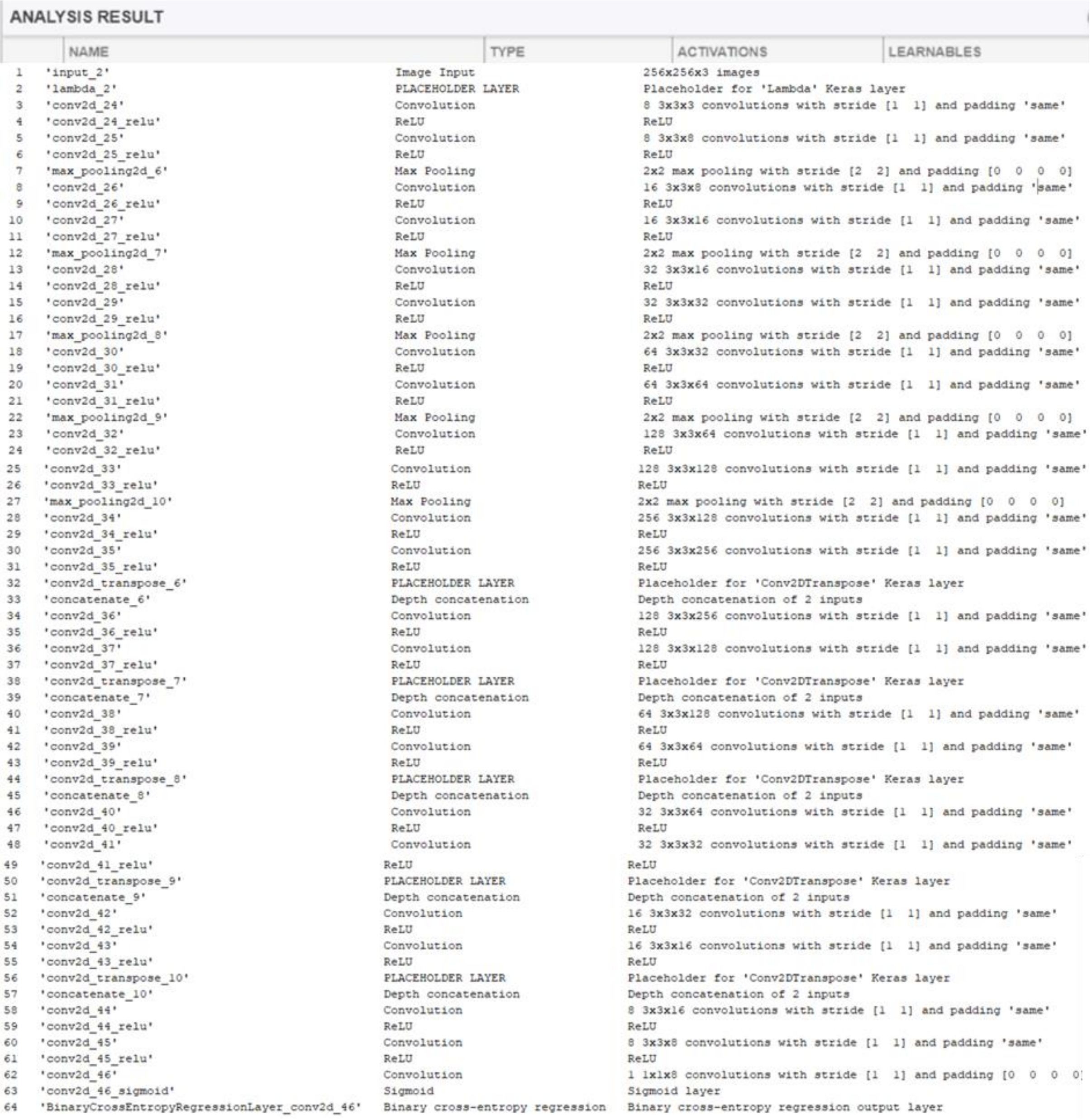
Figure 7-16 Architecture example of a U-NET Structure [158]

The adjustment made in unet is that there are a large number of feature channels in the up-sampling part. This allows the network to propagate context information to higher resolution layers. The expansive path is more or less symmetric to the contracting part, and yields a u-shaped architecture (figure 7.14). This UNet architecture example is for 32x32 pixels in the lowest resolution. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

7.6.2 Implementation

The main design is that the contracting path is a typical convolutional network that consists of repeated application of convolutions, each followed by a rectified linear unit (ReLU) and a max pooling operation. During the contraction, the spatial information is reduced while feature information is increased. The expansive pathway combines the feature and spatial information through a sequence of up-convolutions and concatenations with high-resolution features from the contracting path. This structure is shown in a more elaborate manner in table 7.9 with an input image size of 128x128x3 while in the original paper the size was 572x572x3. For this example, the main components will remain the same with changes in the sizes at various locations.

Table 7-9 Detailed implemented UNET Architecture summarised



ANALYSIS RESULT

	NAME	TYPE	ACTIVATIONS	LEARNABLES
1	'input_2'	Image Input	256x256x3 images	
2	'lambda_2'	PLACEHOLDER LAYER	Placeholder for 'Lambda' Keras layer	
3	'conv2d_24'	Convolution	8 3x3x3 convolutions with stride [1 1] and padding 'same'	
4	'conv2d_24_relu'	ReLU	ReLU	
5	'conv2d_25'	Convolution	8 3x3x8 convolutions with stride [1 1] and padding 'same'	
6	'conv2d_25_relu'	ReLU	ReLU	
7	'max_pooling2d_6'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0 0 0]	
8	'conv2d_26'	Convolution	16 3x3x8 convolutions with stride [1 1] and padding 'same'	
9	'conv2d_26_relu'	ReLU	ReLU	
10	'conv2d_27'	Convolution	16 3x3x16 convolutions with stride [1 1] and padding 'same'	
11	'conv2d_27_relu'	ReLU	ReLU	
12	'max_pooling2d_7'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0 0 0]	
13	'conv2d_28'	Convolution	32 3x3x16 convolutions with stride [1 1] and padding 'same'	
14	'conv2d_28_relu'	ReLU	ReLU	
15	'conv2d_29'	Convolution	32 3x3x32 convolutions with stride [1 1] and padding 'same'	
16	'conv2d_29_relu'	ReLU	ReLU	
17	'max_pooling2d_8'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0 0 0]	
18	'conv2d_30'	Convolution	64 3x3x32 convolutions with stride [1 1] and padding 'same'	
19	'conv2d_30_relu'	ReLU	ReLU	
20	'conv2d_31'	Convolution	64 3x3x64 convolutions with stride [1 1] and padding 'same'	
21	'conv2d_31_relu'	ReLU	ReLU	
22	'max_pooling2d_9'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0 0 0]	
23	'conv2d_32'	Convolution	128 3x3x64 convolutions with stride [1 1] and padding 'same'	
24	'conv2d_32_relu'	ReLU	ReLU	
25	'conv2d_33'	Convolution	128 3x3x128 convolutions with stride [1 1] and padding 'same'	
26	'conv2d_33_relu'	ReLU	ReLU	
27	'max_pooling2d_10'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0 0 0]	
28	'conv2d_34'	Convolution	256 3x3x128 convolutions with stride [1 1] and padding 'same'	
29	'conv2d_34_relu'	ReLU	ReLU	
30	'conv2d_35'	Convolution	256 3x3x256 convolutions with stride [1 1] and padding 'same'	
31	'conv2d_35_relu'	ReLU	ReLU	
32	'conv2d_transpose_6'	PLACEHOLDER LAYER	Placeholder for 'Conv2DTranspose' Keras layer	
33	'concatenate_6'	Depth concatenation	Depth concatenation of 2 inputs	
34	'conv2d_36'	Convolution	128 3x3x256 convolutions with stride [1 1] and padding 'same'	
35	'conv2d_36_relu'	ReLU	ReLU	
36	'conv2d_37'	Convolution	128 3x3x128 convolutions with stride [1 1] and padding 'same'	
37	'conv2d_37_relu'	ReLU	ReLU	
38	'conv2d_transpose_7'	PLACEHOLDER LAYER	Placeholder for 'Conv2DTranspose' Keras layer	
39	'concatenate_7'	Depth concatenation	Depth concatenation of 2 inputs	
40	'conv2d_38'	Convolution	64 3x3x128 convolutions with stride [1 1] and padding 'same'	
41	'conv2d_38_relu'	ReLU	ReLU	
42	'conv2d_39'	Convolution	64 3x3x64 convolutions with stride [1 1] and padding 'same'	
43	'conv2d_39_relu'	ReLU	ReLU	
44	'conv2d_transpose_8'	PLACEHOLDER LAYER	Placeholder for 'Conv2DTranspose' Keras layer	
45	'concatenate_8'	Depth concatenation	Depth concatenation of 2 inputs	
46	'conv2d_40'	Convolution	32 3x3x64 convolutions with stride [1 1] and padding 'same'	
47	'conv2d_40_relu'	ReLU	ReLU	
48	'conv2d_41'	Convolution	32 3x3x32 convolutions with stride [1 1] and padding 'same'	
49	'conv2d_41_relu'	ReLU	ReLU	
50	'conv2d_transpose_9'	PLACEHOLDER LAYER	Placeholder for 'Conv2DTranspose' Keras layer	
51	'concatenate_9'	Depth concatenation	Depth concatenation of 2 inputs	
52	'conv2d_42'	Convolution	16 3x3x32 convolutions with stride [1 1] and padding 'same'	
53	'conv2d_42_relu'	ReLU	ReLU	
54	'conv2d_43'	Convolution	16 3x3x16 convolutions with stride [1 1] and padding 'same'	
55	'conv2d_43_relu'	ReLU	ReLU	
56	'conv2d_transpose_10'	PLACEHOLDER LAYER	Placeholder for 'Conv2DTranspose' Keras layer	
57	'concatenate_10'	Depth concatenation	Depth concatenation of 2 inputs	
58	'conv2d_44'	Convolution	8 3x3x16 convolutions with stride [1 1] and padding 'same'	
59	'conv2d_44_relu'	ReLU	ReLU	
60	'conv2d_45'	Convolution	8 3x3x8 convolutions with stride [1 1] and padding 'same'	
61	'conv2d_45_relu'	ReLU	ReLU	
62	'conv2d_46'	Convolution	1 1x1x8 convolutions with stride [1 1] and padding [0 0 0 0	
63	'conv2d_46_sigmoid'	Sigmoid	Sigmoid layer	
64	'BinaryCrossEntropyRegressionLayer_conv2d_46'	Binary cross-entropy regression	Binary cross-entropy regression output layer	

It has been implemented by using Keras with Tensorflow (backend) has been used to develop the model. Python 3.5 in Jupyter Notebook was used for test script in Anaconda Environment. The same Pixel-wise 115 Pit images dataset has been used as input. Figure 7.17 and 7.18 show montages of the model weights of its first and last convolutional layer.

Figure 7-17 Montage of Pixel-wise UNet network weights of the first convolution layer

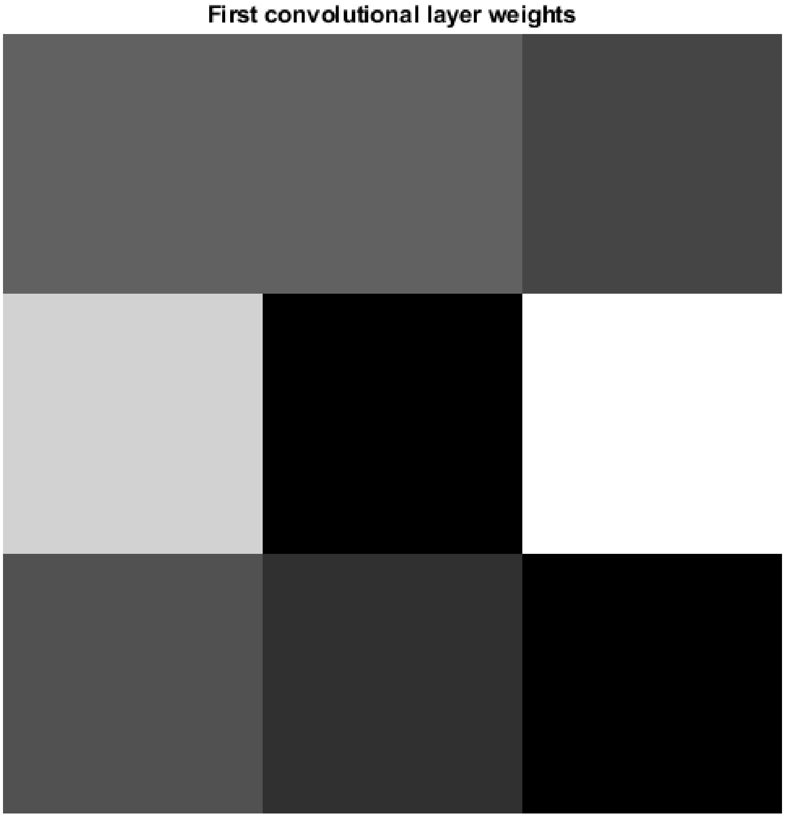


Figure 7-18 Montage of Pixel-wise UNet network weights of the last convolution layer

7.6.3 Results of Pixel-wise UNet

The section shows performance results of the pixel-wise classification using UNet model. First it shows a table as shown below in Table 7.10, then visual output of few test images as shown in figure 7.19, 7.20 and 7.21. These images illustrate the results from Deep Learning Pixel-wise Unet model showing both original image and the resultant Predicted image by the model. The images selected for test sample consist of varied sample images including crack image, pit image and a less pitted image to show diverse result outcomes.

Table 7-10 Deep Learning pixel-wise UNet performance results on same Dataset

Metrics#	Metrics names	UNet
1	TP↑	43690
2	TN↑	1085872
3	FP↑	93726
4	FN↑	5512
5	Recall↑	0.914
6	TNR↑	0.921
7	Precision↑	0.336
8	Accuracy↑	0.919
9	NPV↑	0.995
10	FPR↓	0.079
11	FDR↓	0.664
12	FNR↓	0.086
13	f1↑	0.445
14	RMSE↓	0.072
15	MCC↑	0.491
16	CV↑	0.319
17	RI↑	0.862
18	VOI↓	0.674
19	GCE↓	0.031
20	BE↓	96.135
21	Time(sec)↓	0.71

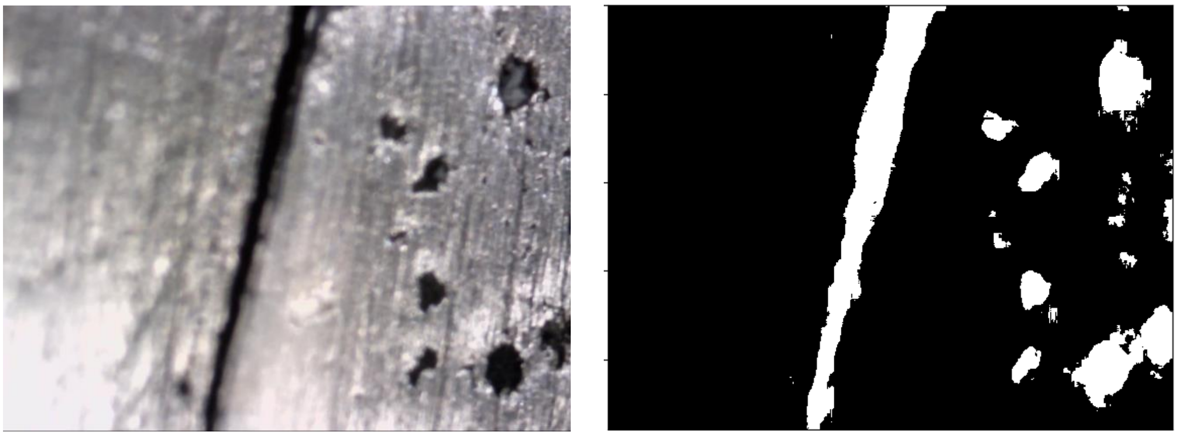


Figure 7-19 Deep Learning UNet model showing results for a crack sample image

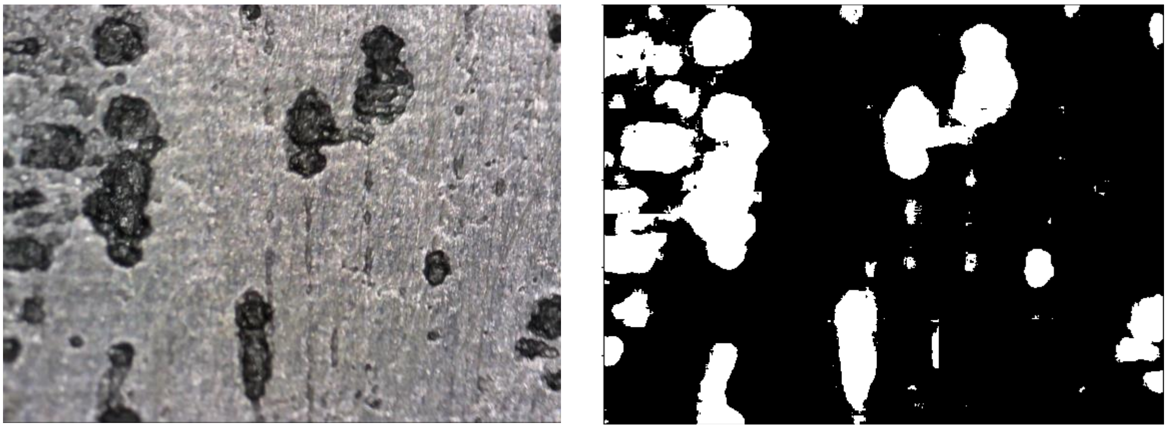


Figure 7-20 Deep Learning UNet model showing results for pits sample image

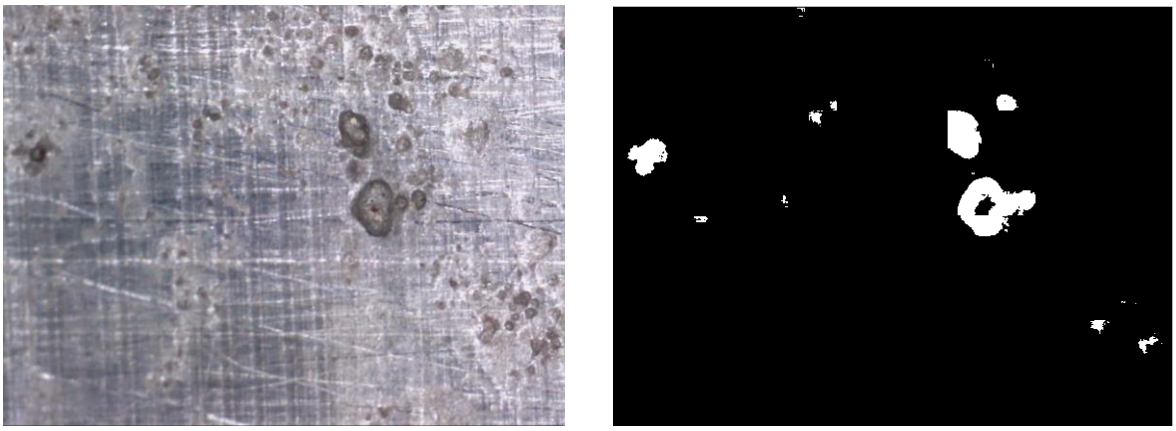
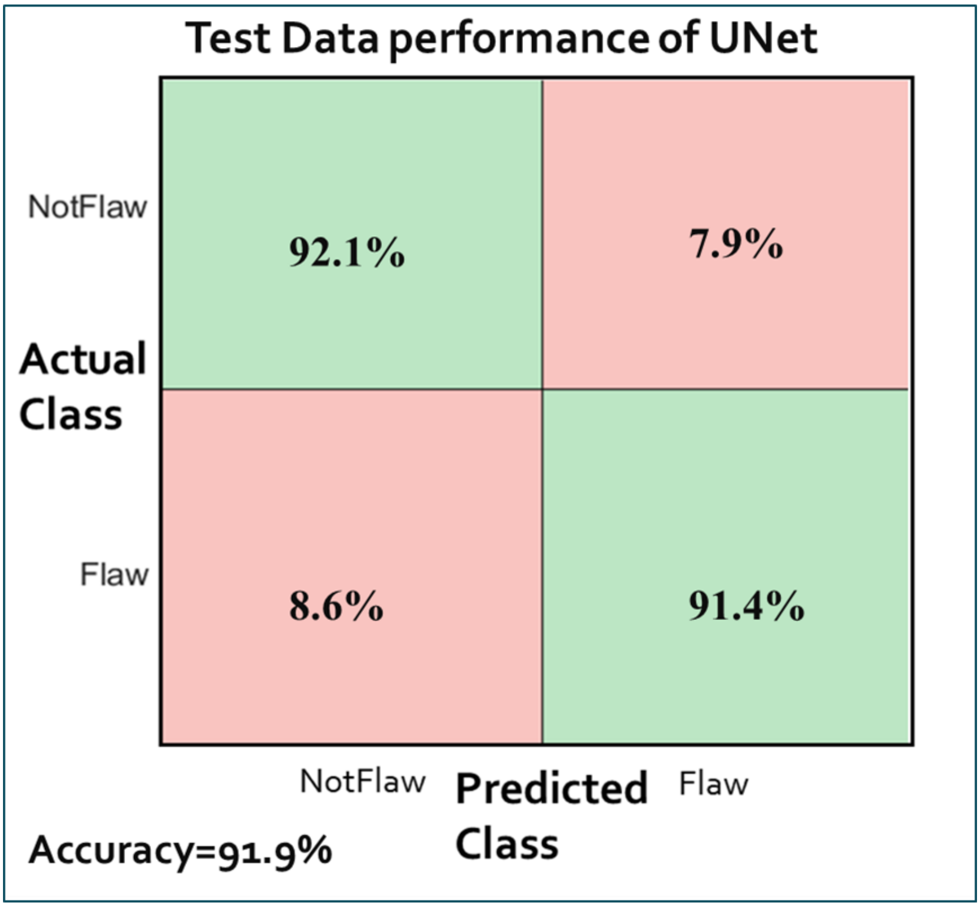


Figure 7-21 Deep Learning UNet model showing results for less pitted sample image

7.6.4 Conclusion

Pixel-wise UNet classification model is the second one implemented, which was selected because of its interesting architectural design. The model is able to produce, same size predicted image as the size of input image entered as shown in figure 7.22. It is able to detect and measure the flaws in the image with an accuracy of 89.4%. It is able to outline the shape of the defect with a good effect to localise the object detected.



Test Data performance of UNet

Actual Class	Predicted Class: NotFlaw	Predicted Class: Flaw
NotFlaw	92.1%	7.9%
Flaw	8.6%	91.4%

Accuracy=91.9%

Figure 7-22 Test result of UNet Pixel-wise classification model by confusion matrix

7.7 Pixel-wise UNet with VGG16 improved model

This method is the proposed method for pixel-wise classification. It is a combination of two architectures, so this is a Unet that is fine-tuned with VGG-16 encoders, which are pre-trained encoders on image-net. It consists of encoder with several layers of convolution and pooling for down-sampling and the second half includes decoder that uses up-sampling and convolution layers. As it is a pre-trained model, the training starts with initial weights of the pre-trained model, which was designed for object classification. Though pre-trained model is trained on a different task than the task at hand but provides a very useful starting point because the features learned while training on the old task are useful for the new task.

7.7.1 Architecture of UNet_VGG16 Pixel-wise model

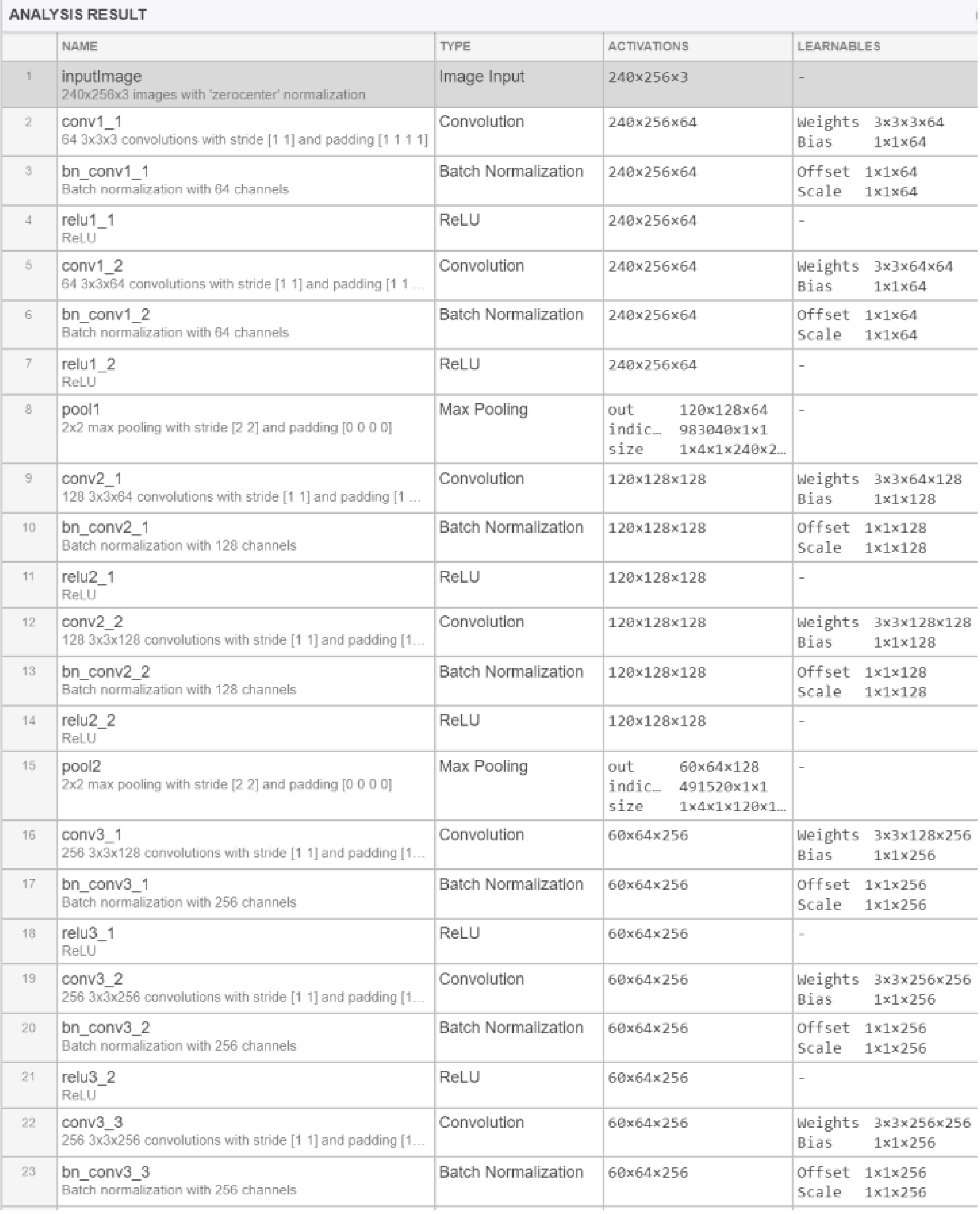
This architecture has an encoder network and a corresponding decoder network as seen in table 7.11. It has a total of 91 layers overall that includes a final pixel-wise classification layer right in the end. Based on the VGG16's first 13 convolutional layers, this architecture also contains 13 convolutional layers. Each encoder performs convolution with a filter bank to produce a set of feature maps. Each of the convolution layer is followed by a ReLU activation function, an element-wise rectified linear non-linearity max (0, x). These are then batch normalised. By removing the fully connected layers, the number of parameters decreases remarkably and this also helps in preserving higher resolution feature maps at the deepest encoder output.

Following that, max-pooling with a 2 × 2 window and stride 2 (non-overlapping window) is performed and the resulting output is sub-sampled by a factor of 2. While several layers of max-pooling and sub-sampling can achieve more robust classification. The increasingly lossy (boundary detail) image representation is not beneficial for segmentation where boundary delineation is vital. Therefore, it is necessary to capture and store boundary information in the encoder feature maps before sub-sampling is performed. If memory during inference is not constrained, then all the encoder feature maps (after subsampling) can be stored. But this is usually not the case in practical applications. It involves storing only the max-pooling indices, i.e, the locations of the maximum feature value in each pooling window is memorised for each encoder feature map. In principle, this can be done using 2 bits for each 2 × 2 pooling window and is thus much more efficient to store as compared to memorising feature map(s) in float precision. This lower memory storage results in a slight loss of accuracy but is still suitable for practical applications.

Each encoder layer has a corresponding decoder layer and hence as expected the decoder network also has 13 layers. The final decoder output is fed to a multi-class soft-max classifier to produce class probabilities for each pixel independently. The appropriate decoder in the decoder network up-samples its input feature map(s) using the memorised max-pooling indices, from the corresponding encoder feature map(s). This step produces sparse feature map(s). These feature maps are then convolved with a trainable decoder filter bank to produce dense feature maps. A batch normalization step is then applied to each of these maps. Note that the decoder corresponding to the first encoder (closest to the input image) produces a multi-channel feature map, although its encoder input has 3 channels (RGB). This is unlike the other decoders in the network which produce feature maps with the same number of size and channels as their encoder inputs.

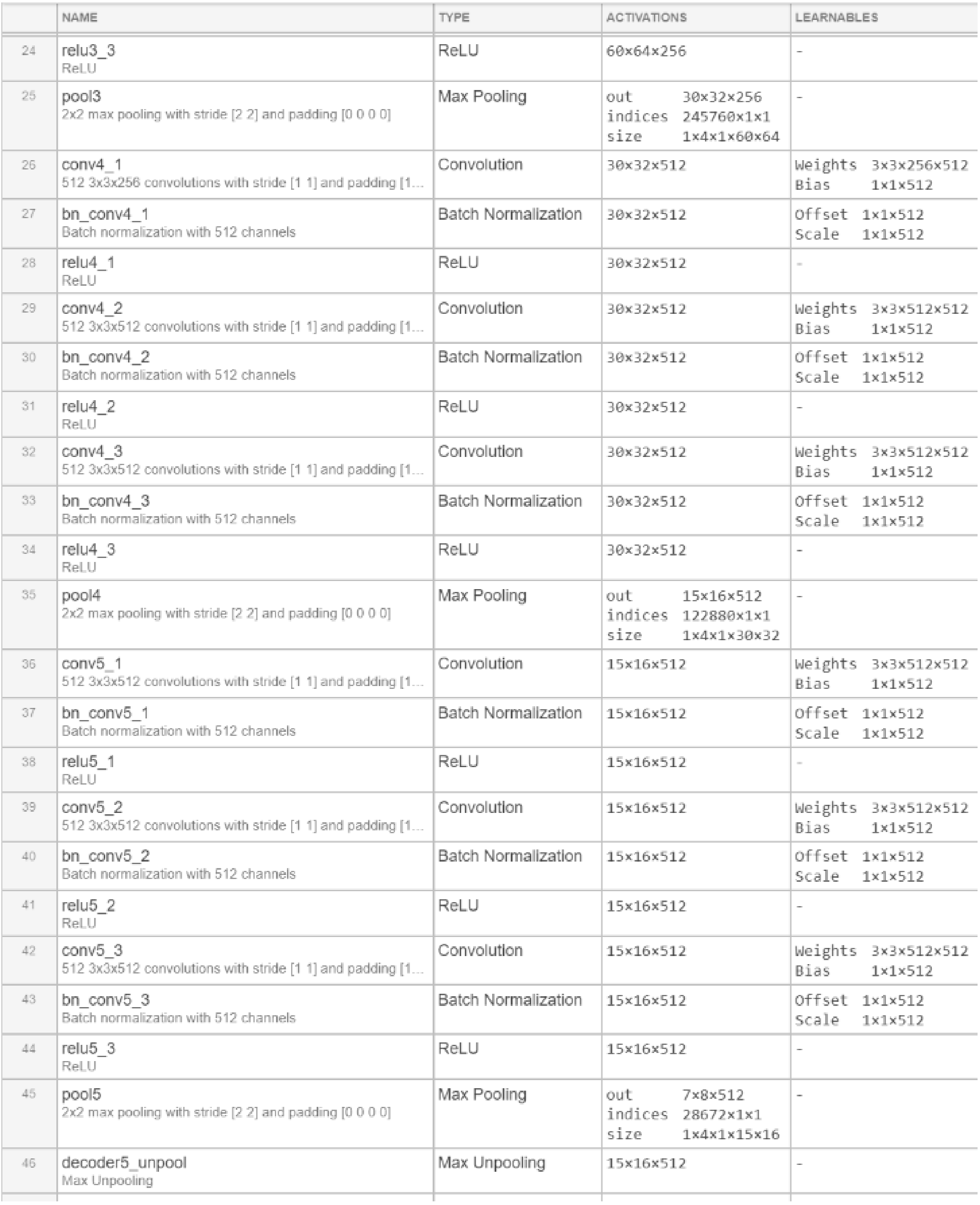
The high dimensional feature representation at the output of the final decoder is fed to a trainable soft-max classifier. This soft-max classifies each pixel independently. The output of the soft-max classifier is a K channel image of probabilities where K is the number of classes. The predicted segmentation corresponds to the class with maximum probability at each pixel.

Table 7-11 Architecture of the implemented UNet-Vgg16 method with full detail

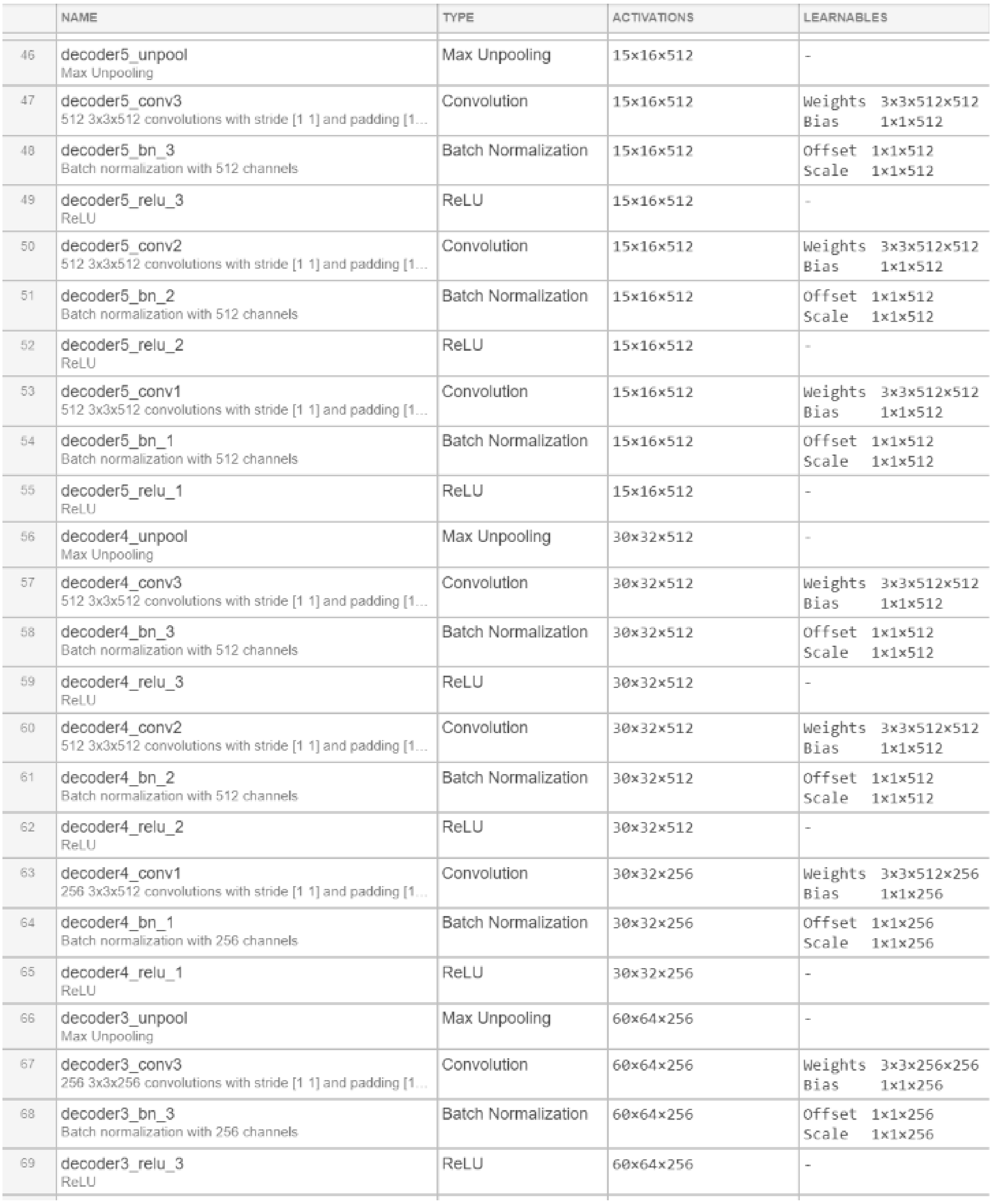


ANALYSIS RESULT

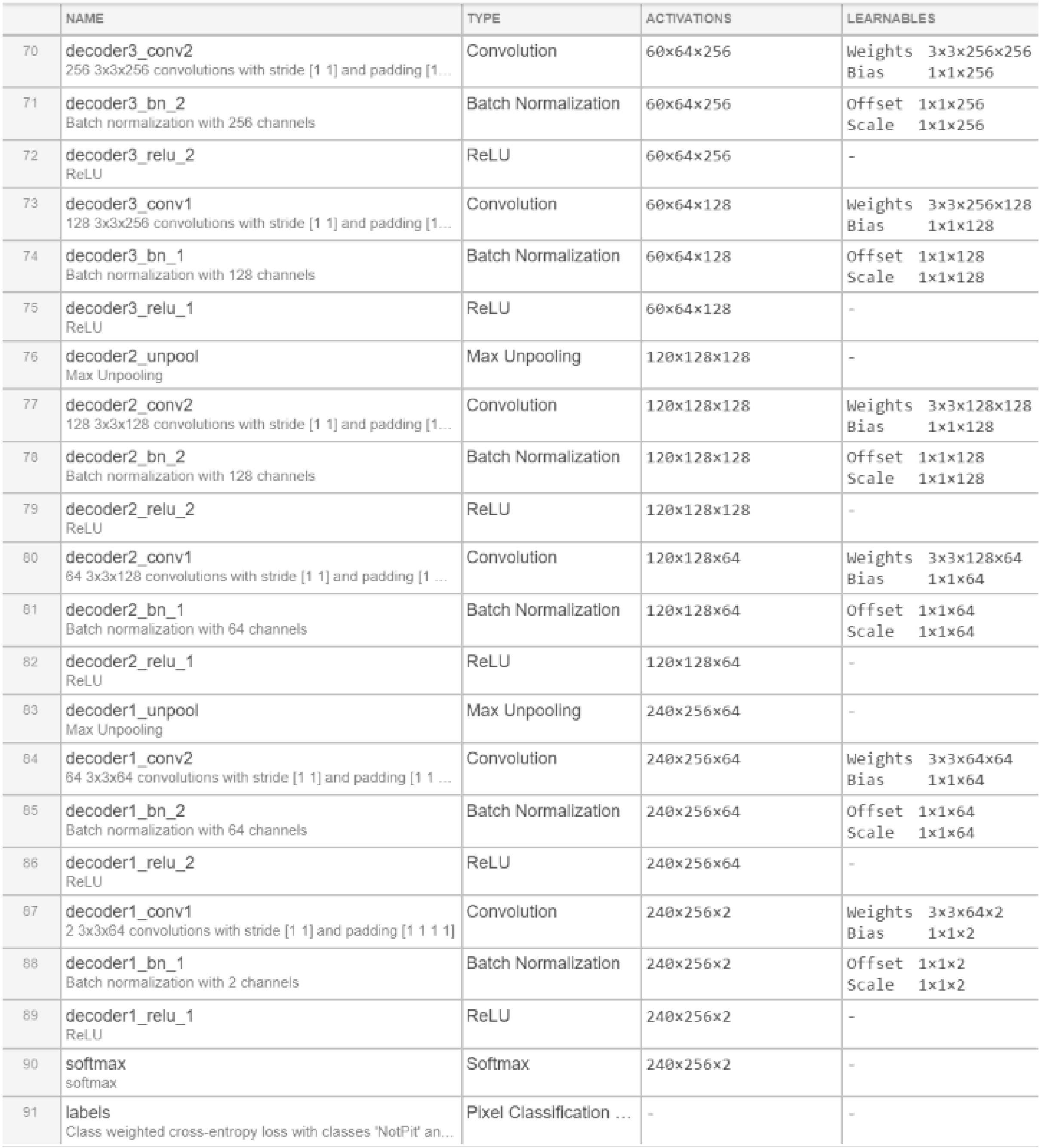
	NAME	TYPE	ACTIVATIONS	LEARNABLES
1	inputImage 240x256x3 images with 'zerocenter' normalization	Image Input	240×256×3	-
2	conv1_1 64 3x3x3 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	240×256×64	Weights 3×3×3×64 Bias 1×1×64
3	bn_conv1_1 Batch normalization with 64 channels	Batch Normalization	240×256×64	Offset 1×1×64 Scale 1×1×64
4	relu1_1 ReLU	ReLU	240×256×64	-
5	conv1_2 64 3x3x64 convolutions with stride [1 1] and padding [1 1 …	Convolution	240×256×64	Weights 3×3×64×64 Bias 1×1×64
6	bn_conv1_2 Batch normalization with 64 channels	Batch Normalization	240×256×64	Offset 1×1×64 Scale 1×1×64
7	relu1_2 ReLU	ReLU	240×256×64	-
8	pool1 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	out 120×128×64 indic… 983040×1×1 size 1×4×1×240×2…	-
9	conv2_1 128 3x3x64 convolutions with stride [1 1] and padding [1 …	Convolution	120×128×128	Weights 3×3×64×128 Bias 1×1×128
10	bn_conv2_1 Batch normalization with 128 channels	Batch Normalization	120×128×128	Offset 1×1×128 Scale 1×1×128
11	relu2_1 ReLU	ReLU	120×128×128	-
12	conv2_2 128 3x3x128 convolutions with stride [1 1] and padding [1…	Convolution	120×128×128	Weights 3×3×128×128 Bias 1×1×128
13	bn_conv2_2 Batch normalization with 128 channels	Batch Normalization	120×128×128	Offset 1×1×128 Scale 1×1×128
14	relu2_2 ReLU	ReLU	120×128×128	-
15	pool2 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	out 60×64×128 indic… 491520×1×1 size 1×4×1×120×1…	-
16	conv3_1 256 3x3x128 convolutions with stride [1 1] and padding [1…	Convolution	60×64×256	Weights 3×3×128×256 Bias 1×1×256
17	bn_conv3_1 Batch normalization with 256 channels	Batch Normalization	60×64×256	Offset 1×1×256 Scale 1×1×256
18	relu3_1 ReLU	ReLU	60×64×256	-
19	conv3_2 256 3x3x256 convolutions with stride [1 1] and padding [1…	Convolution	60×64×256	Weights 3×3×256×256 Bias 1×1×256
20	bn_conv3_2 Batch normalization with 256 channels	Batch Normalization	60×64×256	Offset 1×1×256 Scale 1×1×256
21	relu3_2 ReLU	ReLU	60×64×256	-
22	conv3_3 256 3x3x256 convolutions with stride [1 1] and padding [1…	Convolution	60×64×256	Weights 3×3×256×256 Bias 1×1×256
23	bn_conv3_3 Batch normalization with 256 channels	Batch Normalization	60×64×256	Offset 1×1×256 Scale 1×1×256



	NAME	TYPE	ACTIVATIONS	LEARNABLES
24	relu3_3 ReLU	ReLU	60×64×256	-
25	pool3 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	out 30×32×256 indices 245760×1×1 size 1×4×1×60×64	-
26	conv4_1 512 3x3x256 convolutions with stride [1 1] and padding [1…	Convolution	30×32×512	Weights 3×3×256×512 Bias 1×1×512
27	bn_conv4_1 Batch normalization with 512 channels	Batch Normalization	30×32×512	Offset 1×1×512 Scale 1×1×512
28	relu4_1 ReLU	ReLU	30×32×512	-
29	conv4_2 512 3x3x512 convolutions with stride [1 1] and padding [1…	Convolution	30×32×512	Weights 3×3×512×512 Bias 1×1×512
30	bn_conv4_2 Batch normalization with 512 channels	Batch Normalization	30×32×512	Offset 1×1×512 Scale 1×1×512
31	relu4_2 ReLU	ReLU	30×32×512	-
32	conv4_3 512 3x3x512 convolutions with stride [1 1] and padding [1…	Convolution	30×32×512	Weights 3×3×512×512 Bias 1×1×512
33	bn_conv4_3 Batch normalization with 512 channels	Batch Normalization	30×32×512	Offset 1×1×512 Scale 1×1×512
34	relu4_3 ReLU	ReLU	30×32×512	-
35	pool4 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	out 15×16×512 indices 122880×1×1 size 1×4×1×30×32	-
36	conv5_1 512 3x3x512 convolutions with stride [1 1] and padding [1…	Convolution	15×16×512	Weights 3×3×512×512 Bias 1×1×512
37	bn_conv5_1 Batch normalization with 512 channels	Batch Normalization	15×16×512	Offset 1×1×512 Scale 1×1×512
38	relu5_1 ReLU	ReLU	15×16×512	-
39	conv5_2 512 3x3x512 convolutions with stride [1 1] and padding [1…	Convolution	15×16×512	Weights 3×3×512×512 Bias 1×1×512
40	bn_conv5_2 Batch normalization with 512 channels	Batch Normalization	15×16×512	Offset 1×1×512 Scale 1×1×512
41	relu5_2 ReLU	ReLU	15×16×512	-
42	conv5_3 512 3x3x512 convolutions with stride [1 1] and padding [1…	Convolution	15×16×512	Weights 3×3×512×512 Bias 1×1×512
43	bn_conv5_3 Batch normalization with 512 channels	Batch Normalization	15×16×512	Offset 1×1×512 Scale 1×1×512
44	relu5_3 ReLU	ReLU	15×16×512	-
45	pool5 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	out 7×8×512 indices 28672×1×1 size 1×4×1×15×16	-
46	decoder5_unpool Max Unpooling	Max Unpooling	15×16×512	-



	NAME	TYPE	ACTIVATIONS	LEARNABLES
46	decoder5_unpool Max Unpooling	Max Unpooling	15×16×512	-
47	decoder5_conv3 512 3x3x512 convolutions with stride [1 1] and padding [1…	Convolution	15×16×512	Weights 3×3×512×512 Bias 1×1×512
48	decoder5_bn_3 Batch normalization with 512 channels	Batch Normalization	15×16×512	Offset 1×1×512 Scale 1×1×512
49	decoder5_relu_3 ReLU	ReLU	15×16×512	-
50	decoder5_conv2 512 3x3x512 convolutions with stride [1 1] and padding [1…	Convolution	15×16×512	Weights 3×3×512×512 Bias 1×1×512
51	decoder5_bn_2 Batch normalization with 512 channels	Batch Normalization	15×16×512	Offset 1×1×512 Scale 1×1×512
52	decoder5_relu_2 ReLU	ReLU	15×16×512	-
53	decoder5_conv1 512 3x3x512 convolutions with stride [1 1] and padding [1…	Convolution	15×16×512	Weights 3×3×512×512 Bias 1×1×512
54	decoder5_bn_1 Batch normalization with 512 channels	Batch Normalization	15×16×512	Offset 1×1×512 Scale 1×1×512
55	decoder5_relu_1 ReLU	ReLU	15×16×512	-
56	decoder4_unpool Max Unpooling	Max Unpooling	30×32×512	-
57	decoder4_conv3 512 3x3x512 convolutions with stride [1 1] and padding [1…	Convolution	30×32×512	Weights 3×3×512×512 Bias 1×1×512
58	decoder4_bn_3 Batch normalization with 512 channels	Batch Normalization	30×32×512	Offset 1×1×512 Scale 1×1×512
59	decoder4_relu_3 ReLU	ReLU	30×32×512	-
60	decoder4_conv2 512 3x3x512 convolutions with stride [1 1] and padding [1…	Convolution	30×32×512	Weights 3×3×512×512 Bias 1×1×512
61	decoder4_bn_2 Batch normalization with 512 channels	Batch Normalization	30×32×512	Offset 1×1×512 Scale 1×1×512
62	decoder4_relu_2 ReLU	ReLU	30×32×512	-
63	decoder4_conv1 256 3x3x512 convolutions with stride [1 1] and padding [1…	Convolution	30×32×256	Weights 3×3×512×256 Bias 1×1×256
64	decoder4_bn_1 Batch normalization with 256 channels	Batch Normalization	30×32×256	Offset 1×1×256 Scale 1×1×256
65	decoder4_relu_1 ReLU	ReLU	30×32×256	-
66	decoder3_unpool Max Unpooling	Max Unpooling	60×64×256	-
67	decoder3_conv3 256 3x3x256 convolutions with stride [1 1] and padding [1…	Convolution	60×64×256	Weights 3×3×256×256 Bias 1×1×256
68	decoder3_bn_3 Batch normalization with 256 channels	Batch Normalization	60×64×256	Offset 1×1×256 Scale 1×1×256
69	decoder3_relu_3 ReLU	ReLU	60×64×256	-



	NAME	TYPE	ACTIVATIONS	LEARNABLES
70	decoder3_conv2 256 3x3x256 convolutions with stride [1 1] and padding [1…	Convolution	60×64×256	Weights 3×3×256×256 Bias 1×1×256
71	decoder3_bn_2 Batch normalization with 256 channels	Batch Normalization	60×64×256	Offset 1×1×256 Scale 1×1×256
72	decoder3_relu_2 ReLU	ReLU	60×64×256	-
73	decoder3_conv1 128 3x3x256 convolutions with stride [1 1] and padding [1…	Convolution	60×64×128	Weights 3×3×256×128 Bias 1×1×128
74	decoder3_bn_1 Batch normalization with 128 channels	Batch Normalization	60×64×128	Offset 1×1×128 Scale 1×1×128
75	decoder3_relu_1 ReLU	ReLU	60×64×128	-
76	decoder2_unpool Max Unpooling	Max Unpooling	120×128×128	-
77	decoder2_conv2 128 3x3x128 convolutions with stride [1 1] and padding [1…	Convolution	120×128×128	Weights 3×3×128×128 Bias 1×1×128
78	decoder2_bn_2 Batch normalization with 128 channels	Batch Normalization	120×128×128	Offset 1×1×128 Scale 1×1×128
79	decoder2_relu_2 ReLU	ReLU	120×128×128	-
80	decoder2_conv1 64 3x3x128 convolutions with stride [1 1] and padding [1 …	Convolution	120×128×64	Weights 3×3×128×64 Bias 1×1×64
81	decoder2_bn_1 Batch normalization with 64 channels	Batch Normalization	120×128×64	Offset 1×1×64 Scale 1×1×64
82	decoder2_relu_1 ReLU	ReLU	120×128×64	-
83	decoder1_unpool Max Unpooling	Max Unpooling	240×256×64	-
84	decoder1_conv2 64 3x3x64 convolutions with stride [1 1] and padding [1 1 …	Convolution	240×256×64	Weights 3×3×64×64 Bias 1×1×64
85	decoder1_bn_2 Batch normalization with 64 channels	Batch Normalization	240×256×64	Offset 1×1×64 Scale 1×1×64
86	decoder1_relu_2 ReLU	ReLU	240×256×64	-
87	decoder1_conv1 2 3x3x64 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	240×256×2	Weights 3×3×64×2 Bias 1×1×2
88	decoder1_bn_1 Batch normalization with 2 channels	Batch Normalization	240×256×2	Offset 1×1×2 Scale 1×1×2
89	decoder1_relu_1 ReLU	ReLU	240×256×2	-
90	softmax softmax	Softmax	240×256×2	-
91	labels Class weighted cross-entropy loss with classes 'NotPit' an…	Pixel Classification …	-	-

7.7.2 Implementation

As discussed earlier, this architecture is a combination of two architectures, so this is a Unet that is fine-tuned with VGG-16 encoders, which are pre-trained encoders on image-net. It is similar to Unet in the sense that it has an encoder and a corresponding decoder network but it doesn't use feature maps.

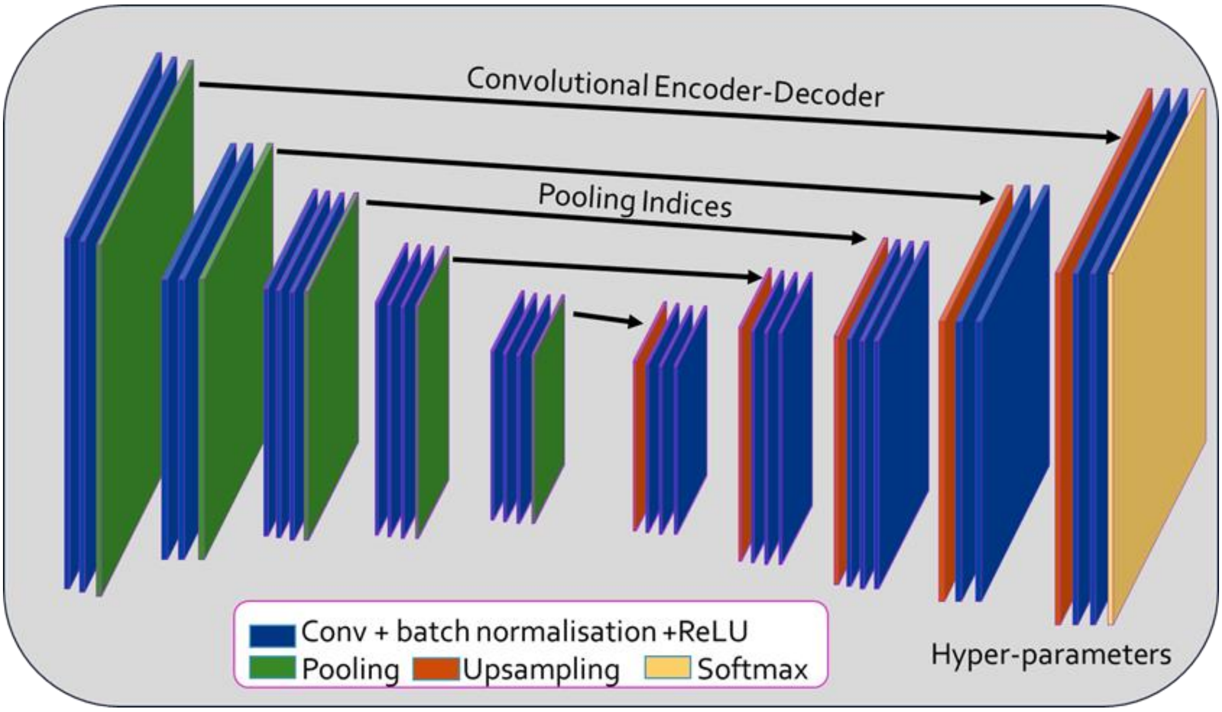

Convolutional Encoder-Decoder
Pooling Indices
Conv + batch normalisation +ReLU
Pooling
Upsampling
Softmax
Hyper-parameters

Figure 7-23 Overview of the implemented UNet+VGG16 architecture

VGG16 contains thirteen convolutional layers, as shown in figure 7.23, each followed by a ReLU activation function, and five max polling operations, each reducing feature map by2. All convolutional layers have 3×3kernels which is the filter size. The first convolutional layer produces 64 channels (filters) and then, as the network deepens, the number of channels doubles after each max pooling operation until it reaches 512. On the following layers, the number of channels does not change. To construct an encoder, the fully connected layers are removed and they are replaced with a single convolutional layer of 512 channels.

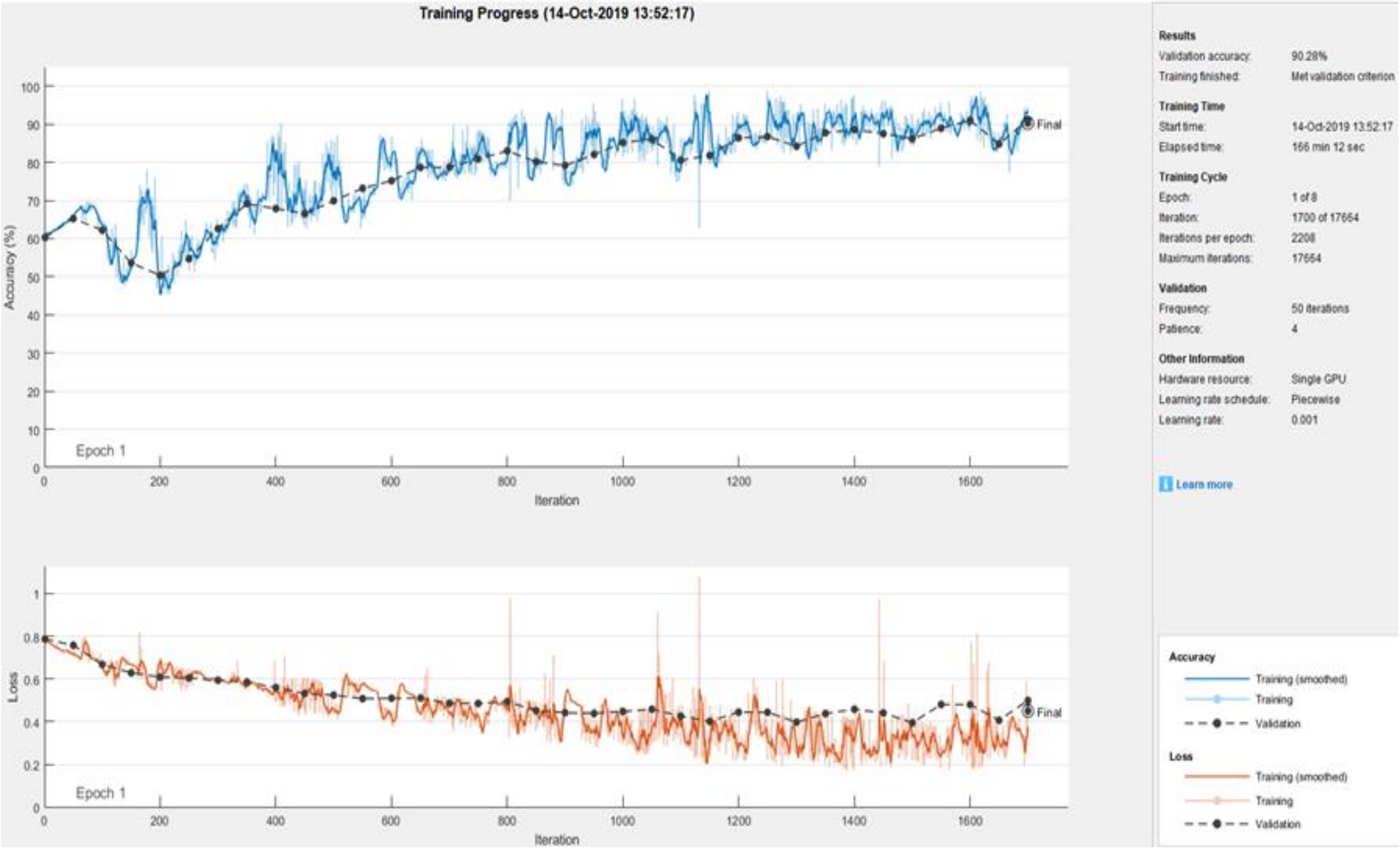
To construct the decoder, transposed convolutions layers are used that doubles the size of a feature map while reducing the number of channels by half. And the output of a transposed convolution is then concatenated with an output of the corresponding part of the decoder. The resultant feature map is treated by convolution operation to keep the number of channels the same as in a symmetric encoder term. This up-sampling procedure is repeated 5 times to pair up with 5 max-poolings. Technically fully connected layers can take an input of any size, but because we have 5 max-pooling layers, each down-sampling an image two times, only images with a side divisible by 64 can be used as an input to the current network implementation.

7.7.3 First few versions tuning the hyper parameters

The initial parameters were setup and then they were fine-tuned till they started working, as it gave errors in the first few trials because of the memory restrictions. The Optimizer which was used, was the Stochastic Gradient Descent Algorithm with a learning rate of 1e-3 with a momentum value of 0.9. Stochastic Gradient Descent Algorithm is a bit slow as compared to Adam's Optimizer Algorithm but gave more accurate results than other optimizer algorithms. The Loss function used was Categorical Cross-entropy, which gave the probabilities of each corresponding class. .

If the GPU does not have sufficient memory, either images need to be re-sized into to smaller sizes or reduce the training batch size. Both were done here. Started off with mini-batch size (mbs) of 64 and the image as its original size of from 960x1280. This gave an error of merge sort that said failed to get memory buffer. With same settings, decreased the batch size to 32 and it still gave the same error. The batch sized was again reduced from 32 to 16 but to no effect as it gave, out of memory error as well. After this, other changes were made like epoch from 30 to 8. But still it didn't work. Lowered batch size even further down to 8, but still there gave the same memory error. Then at batch size 8, learning rate was changed to 0.01, and also the image size was decreased to 240x256 by using patch extraction and finally it worked. Even though it worked but the learning rate was too high for the learning process so changed back the learning rate to 0.001 and further decreased the batch size to 4, which also worked.

So the images were divided into the batch-size of 4. The total number of epochs on which transfer learning VGG16 model ran was 8, thus performing 2208 iterations per epoch with maximum 17,664 iterations. A validation patience of 4 was used which gave accuracy of 90.28% in its first iteration at 1700 epochs as it met the validation criteria. So the learning stopped with it as can be seen in figure 7.24. So for the next fine-tuning, the validation patience was kept as infinity expecting a positive effect on the result as it will be able to learn more. Once all the settings are selected, data is ready to be trained by the model.



Results

Validation accuracy: 90.28%

Training finished: Met validation criterion

Training Time

Start time: 14-Oct-2019 13:52:17

Elapsed time: 166 min 12 sec

Training Cycle

Epoch: 1 of 8

Iteration: 1700 of 17664

Iterations per epoch: 2208

Maximum iterations: 17664

Validation

Frequency: 50 iterations

Patience: 4

Other Information

Hardware resource: Single GPU

Learning rate schedule: Piecewise

Learning rate: 0.001

Figure 7-24 Dataset being trained showing progress of UNet+VGG16 initial model

7.7.4 Final version

Dataset: The total dataset is split into 3 sets, 60% of images are taken as training data, 20% of the images as validation data and the rest of the 20% are taken for testing.

Class Balancing: Fully convolutional training can balance classes by weighting or sampling the loss.

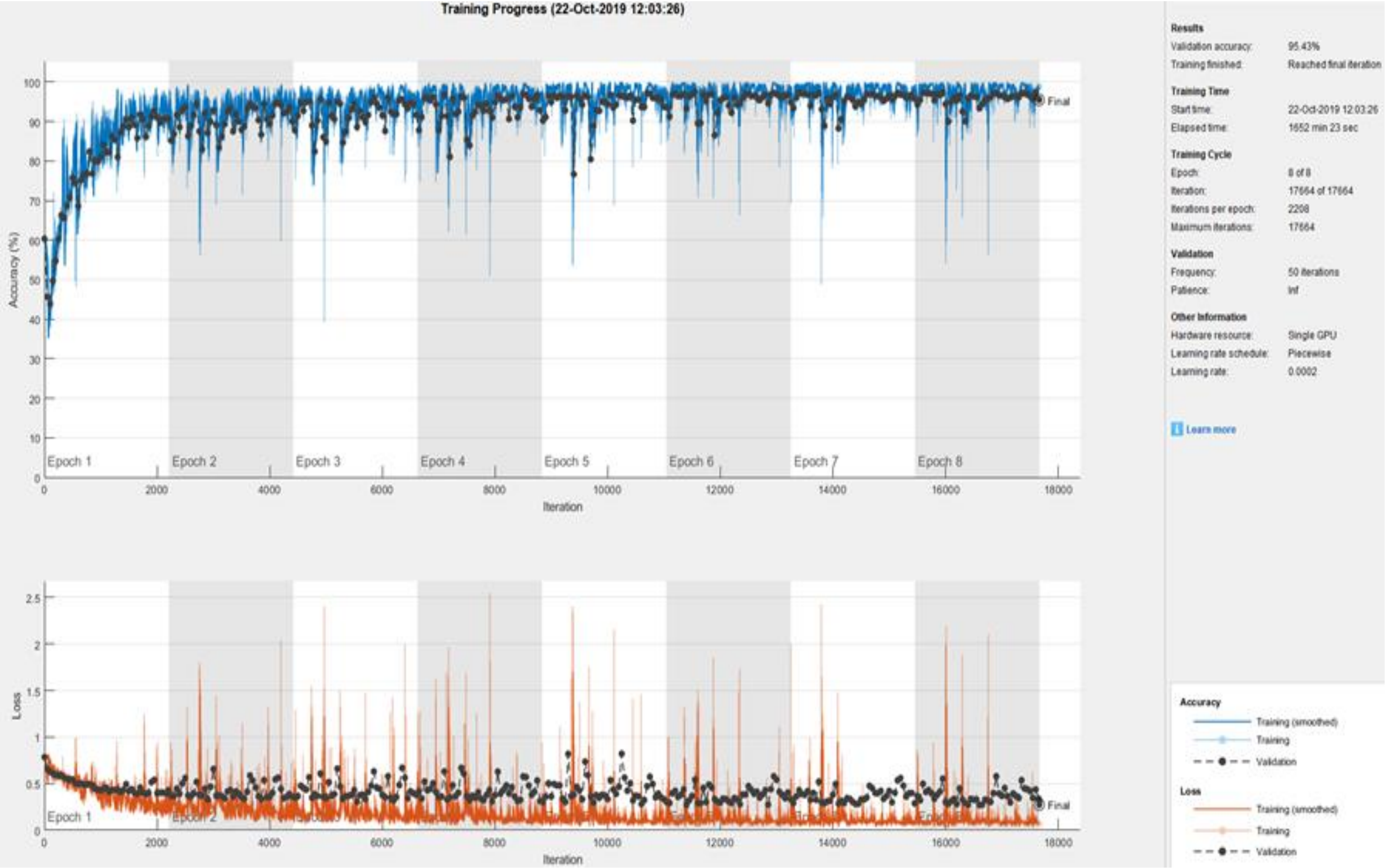
Optimisation: A sgdm optimiser has been used with a value of 0.9 momentum. The learn rate schedule is piece-wise with a learn rate drop factor as 0.2. The learn rate is set to be 1e-3 with a GPU execution environment and loss as Cross Entropy. The mini-batch size set is 4 because of hardware restrictions.

Fine-tuning: Training from scratch is not feasible considering the time required to learn the base classification nets. Fine-tuning takes three days on a single GPU for the coarse FCN-32s version, and about one day each to upgrade to the FCN-16s and FCN-8s versions.

Patch sampling: Performed patch extraction on images to convert from 960x1280 to 240x256. Hence increasing number of observations. But this is not done by randomly sampling patches over a full dataset which may cause in higher variance batches that may accelerate convergence [57]. Instead, it is more like image training effectively batches each image into a regular grid of large overlapping patches.

Iterations: Number of Iterations/Epochs is taken to be 2208. And step_per_epoch which is equal to test images divided by batch size taken, is used correspondingly. Same is applicable for validation set also.

7.7.5 Results



Results

Validation accuracy: 95.43%

Training finished: Reached final iteration

Training Time

Start time: 22-Oct-2019 12:03:26

Elapsed time: 1652 min 23 sec

Training Cycle

Epoch: 8 of 8

Iteration: 17664 of 17664

Iterations per epoch: 2208

Maximum iterations: 17664

Validation

Frequency: 50 iterations

Patience: Inf

Other Information

Hardware resource: Single GPU

Learning rate schedule: Piecewise

Learning rate: 0.0002

Figure 7-25 Dataset being trained showing progress of UNet+VGG16 final model

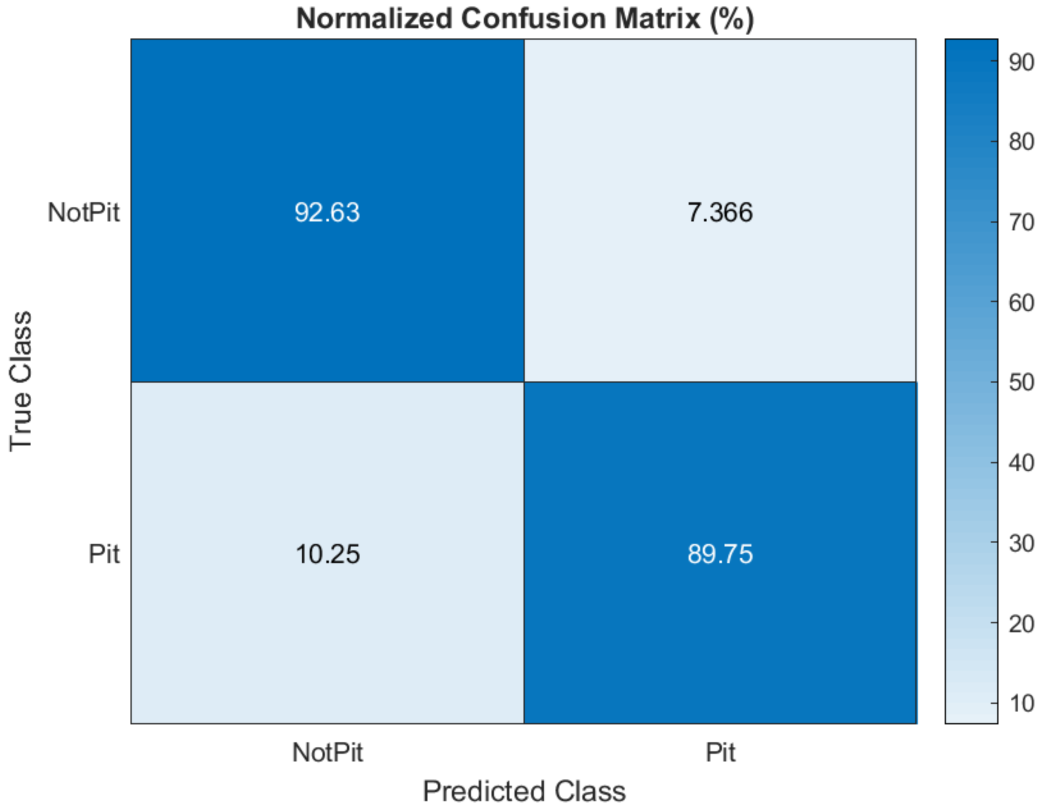


Figure 7-26 Deep Learning UNet=Vgg16 performance using confusion matrix

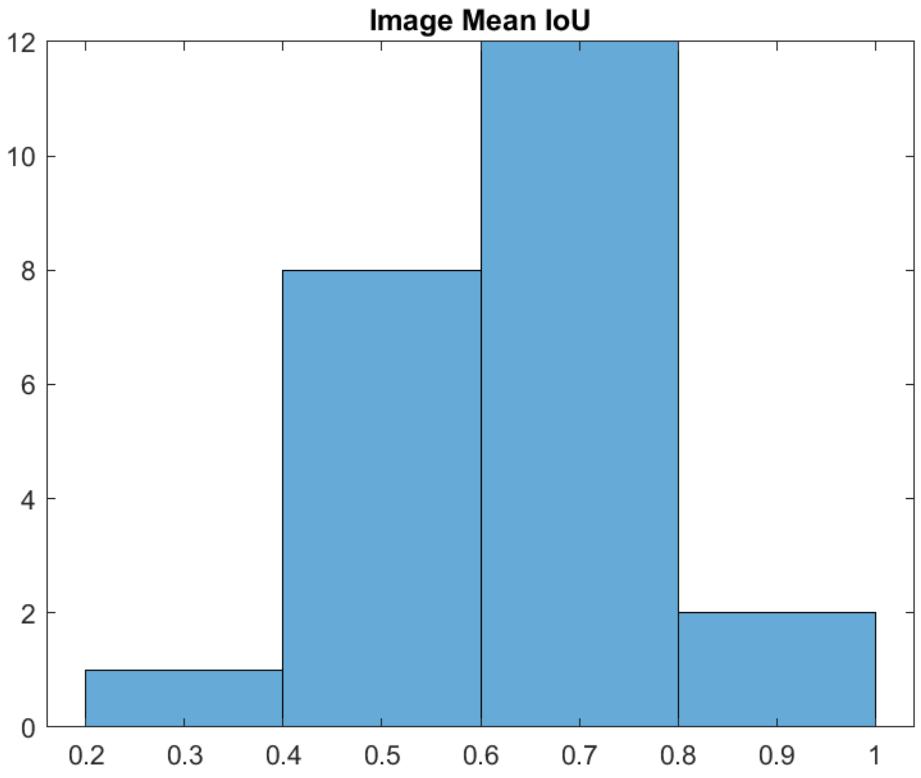
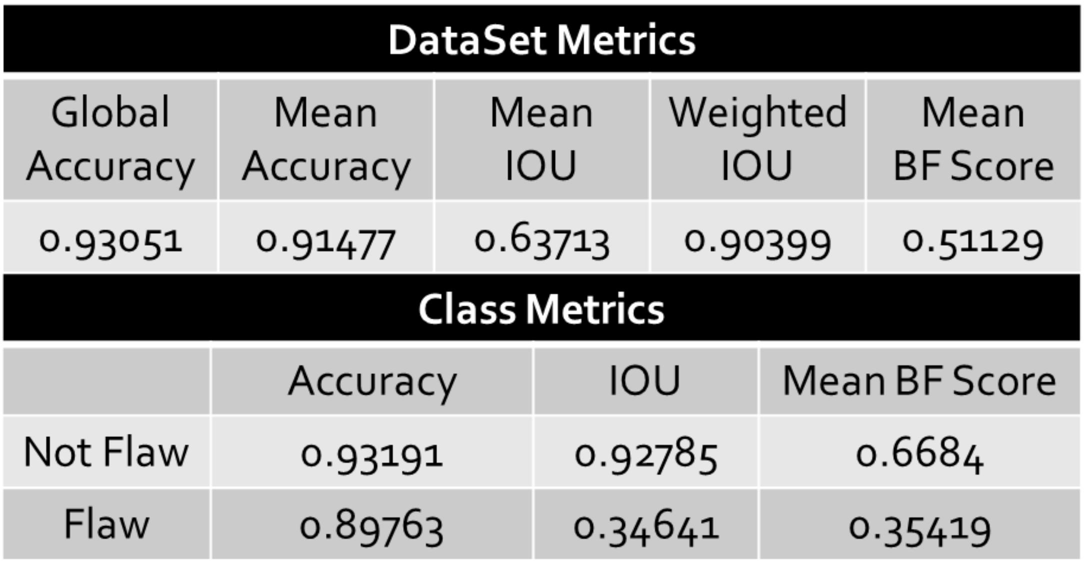


Figure 7-27 Deep Learning UNet-Vgg16 performance displaying mean IoU

Table 7-12 Deep learning UNet-Vgg16 performance on test dataset



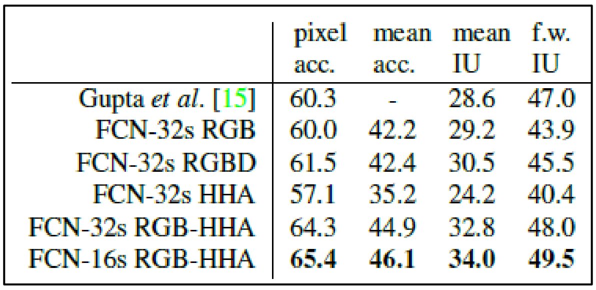
DataSet Metrics				
Global Accuracy	Mean Accuracy	Mean IOU	Weighted IOU	Mean BF Score
0.93051	0.91477	0.63713	0.90399	0.51129

Class Metrics			
	Accuracy	IOU	Mean BF Score
Not Flaw	0.93191	0.92785	0.6684
Flaw	0.89763	0.34641	0.35419

The model's performance, with a validation accuracy of 95% can be seen in figure 7.25. The Testing result outcome of this model, can be seen in table 7.12 and figure 7.26 to 7.27, It shows Global accuracy of 93% with mean of Pit and NotPit classes being 91%. Mean Iou is 63.71% with a weighted IoU being 90.4%. The mean BF Score shows that is 51.1%.

The interesting outcome of the developed system is that the test performance results show higher mean accuracy and weighted IoU as shown in table 7.12, to the state of the art techniques [57], as can be seen in Table 7.13 but dataset used is different, hence not much can be deducted from this comparison.

Table 7-13 Comparison of UNet_vgg16 method with state-of-art indicators [57]



	pixel acc.	mean acc.	mean IU	f.w. IU
Gupta *et al.* [15]	60.3	-	28.6	47.0
FCN-32s RGB	60.0	42.2	29.2	43.9
FCN-32s RGBD	61.5	42.4	30.5	45.5
FCN-32s HHA	57.1	35.2	24.2	40.4
FCN-32s RGB-HHA	64.3	44.9	32.8	48.0
FCN-16s RGB-HHA	**65.4**	**46.1**	**34.0**	**49.5**

Following are some of the examples from the test data. One example is shown in a larger size while a few other examples will be minimised for display. The green coloured depicts false negatives (FN), while magenta represents false positive (FP). It is more dangerous if the green is more, which will mean that it was not able to detect a flaw though it was actually there. Hence, the criteria to look out for is, that the FNs (green) are as minimum as possible in comparison to the FPs (magenta). Results are shown in multiple different ways, on same image from figure 7.28-7.37. They are easy to interpret due to colour-coded visual outcome.

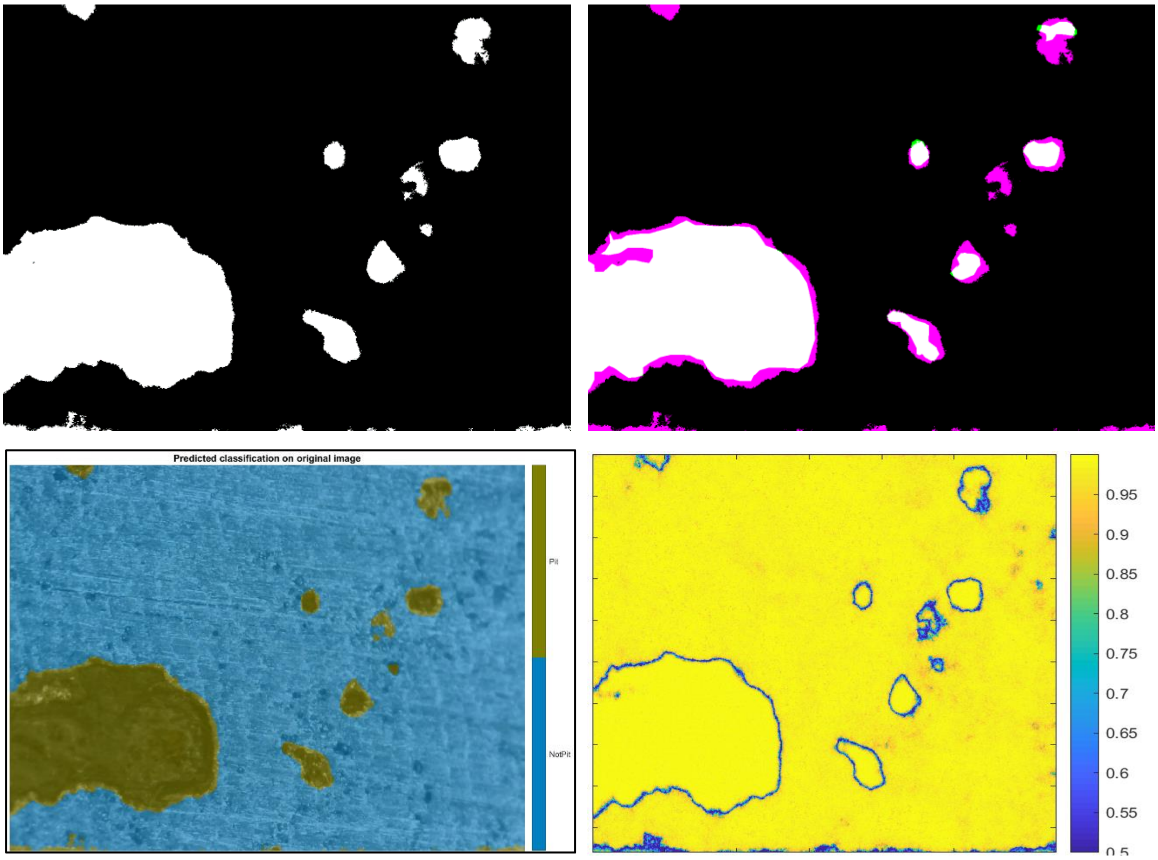


Figure 7-28 Deep Learning UNet-Vgg16 performance on test image

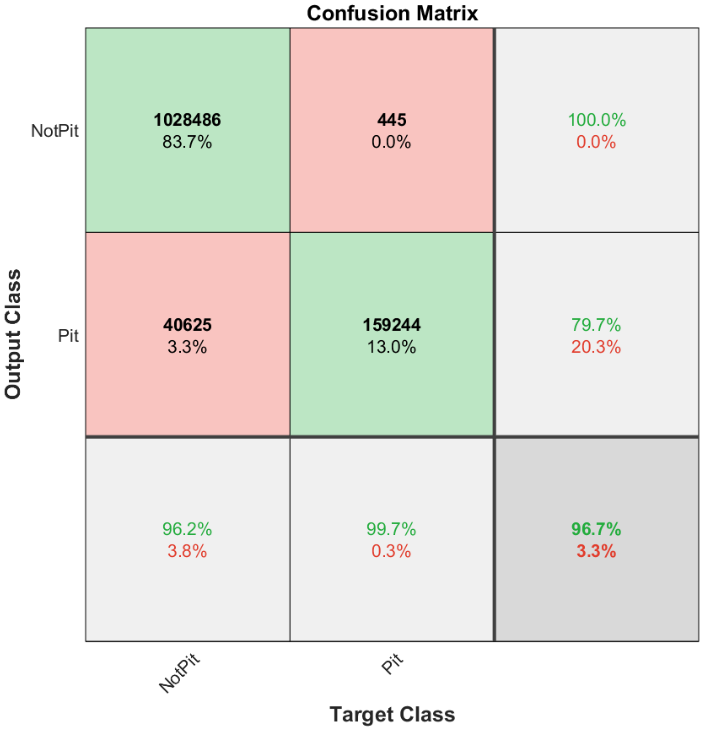


Figure 7-29 Confusion matrix displaying individual image performance

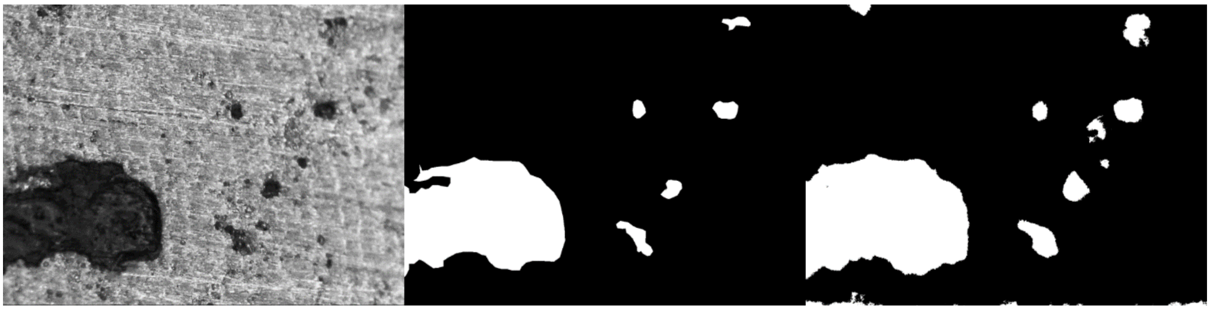


Figure 7-30 Deep Learning pixel-wise UNet-Vgg16 performance on Image A925-17

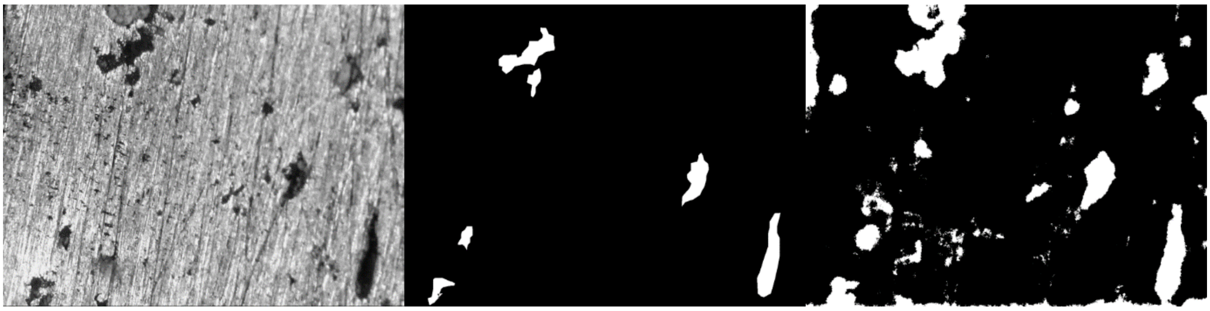


Figure 7-31 Deep Learning pixel-wise UNet performance on Image A186-17

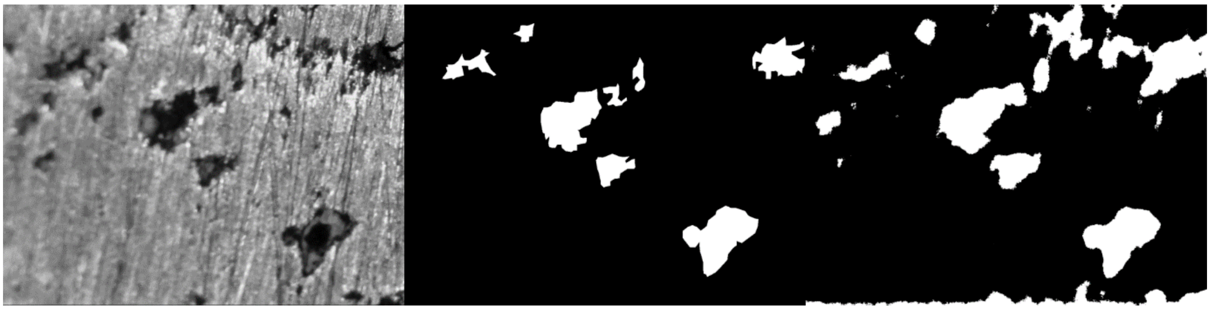


Figure 7-32 Deep Learning pixel-wise UNet performance on Image A190-17

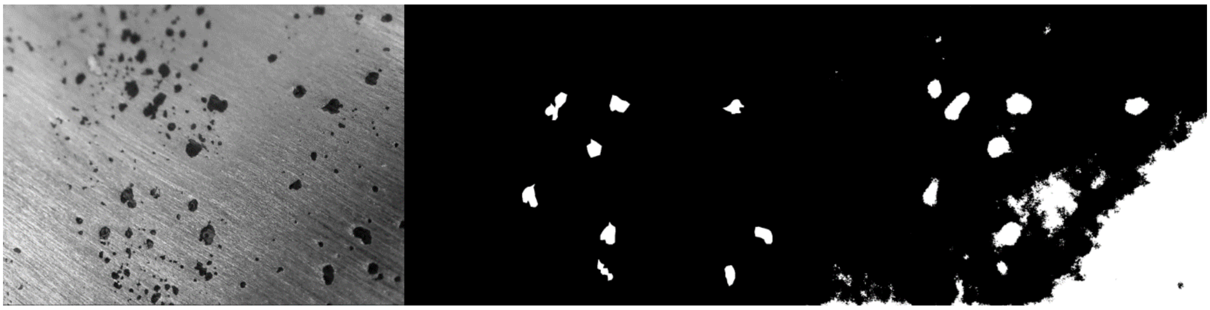


Figure 7-33 Deep Learning pixel-wise UNet performance on Image A652-17

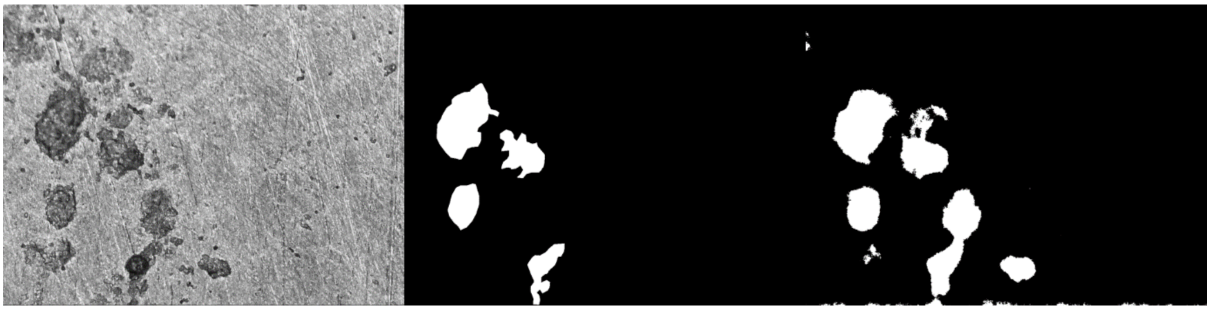


Figure 7-34 Deep Learning pixel-wise UNet performance on Image A762-16

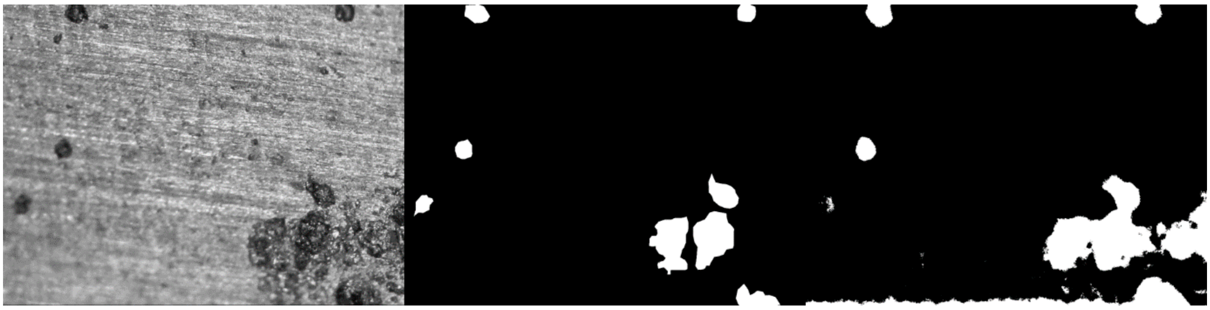


Figure 7-35 Deep Learning pixel-wise UNet performance on Image A917-17

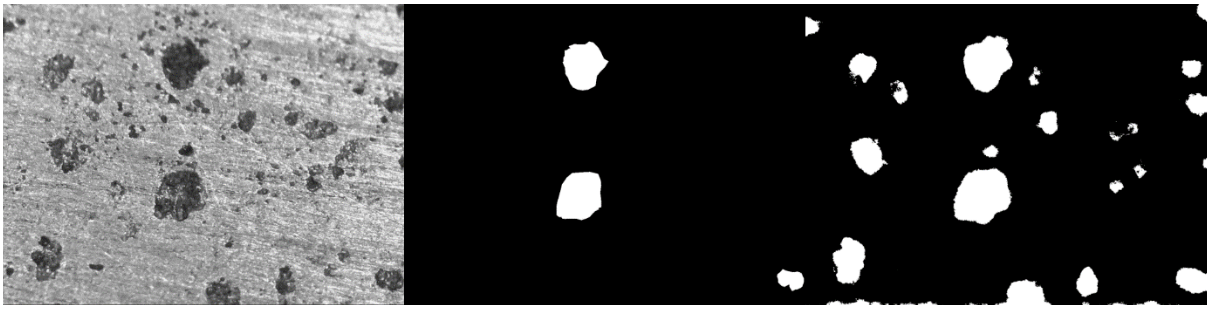


Figure 7-36 Deep Learning pixel-wise UNet performance on Image A994-16

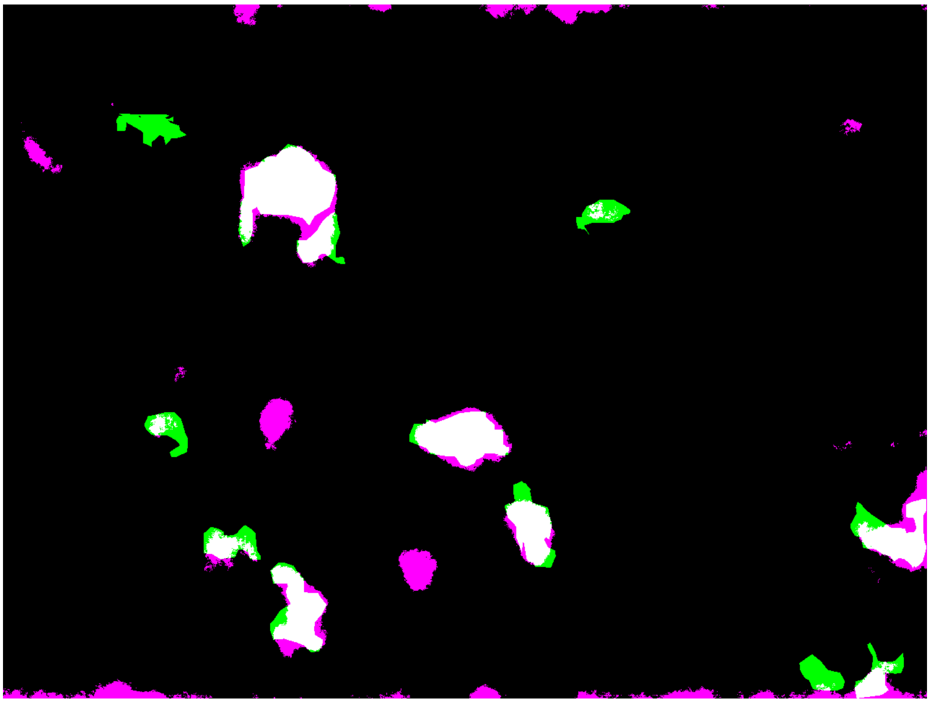


Figure 7-37 Deep Learning UNet performance on Image A502 by Jaccard mean IoU

The result outcomes from the Unet_vgg16 model includes Jaccard mean IoU which can be seen in figure 7.37 for test Image A502. An edge line can be seen on the top and bottom of the image, these are the false positives which can be solved by applying CRF for future work. As the FP doesn't affect the detection that much, as long as it is able to identify and detect flaws when they are there keeping the false negatives low. Another solution is to delete the border line in the pre-processing step.

A comparison between simple UNet and improved Unet can be shown in figure 7.38 by using image A502.which reflects the performance of both models. The figure shows the test image on the top left, with its ground truth on top right, then UNet predicted image in the bottom right and right beside it UNet+Vgg16 predicted image to show comparison.

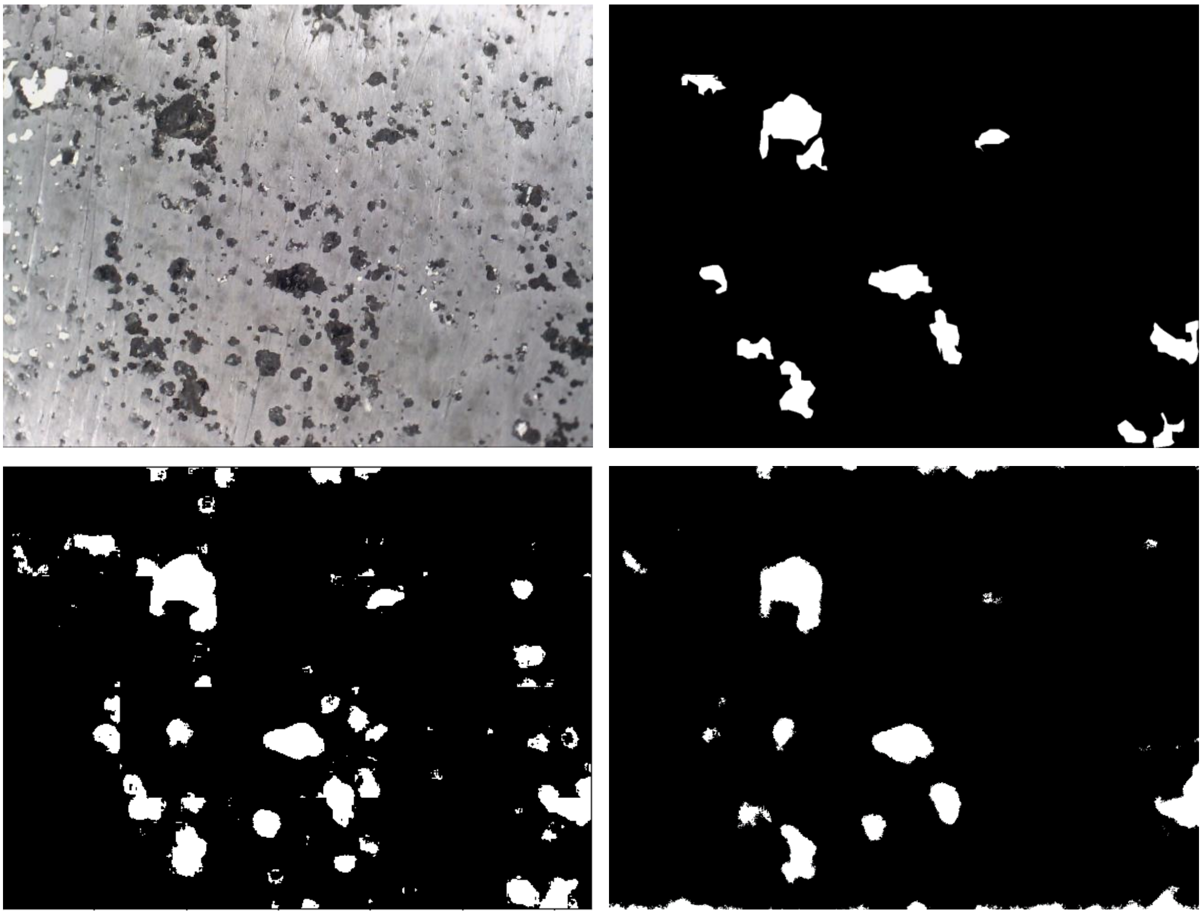
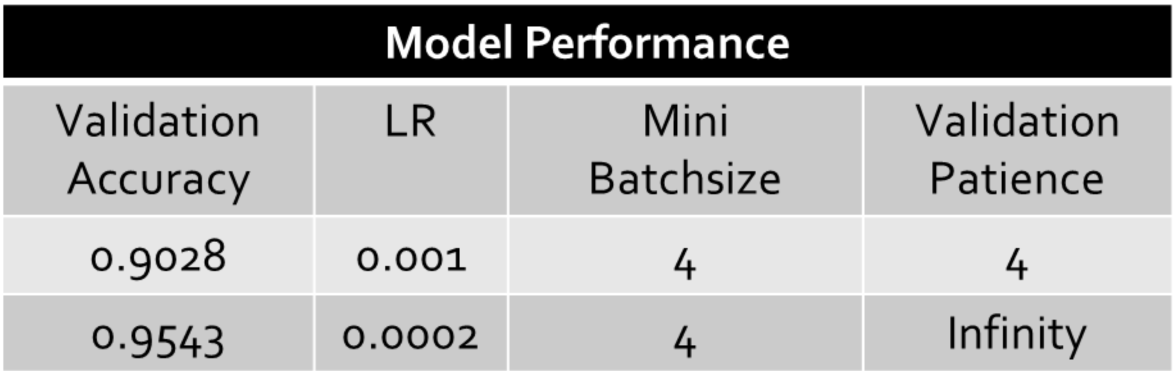


Figure 7-38 Deep Learning on Image A502 showing predictions from UNet & UNet+VGG16

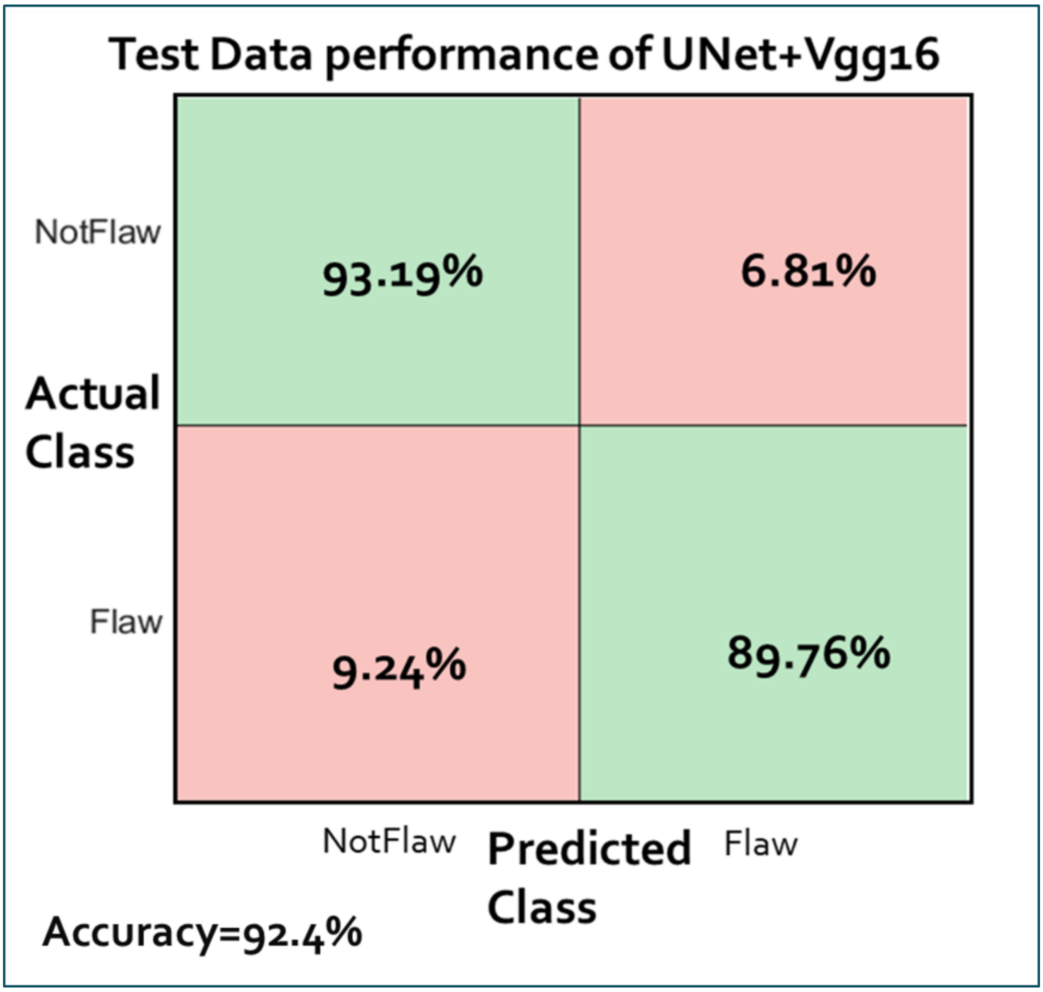
7.7.6 Conclusion

With a combination of two state-of the art models, VGG16 and UNet, the third model implemented, shows performance with global accuracy of 93%. It is a pixel-wise classification model that is able to detect and measure the defects. With a validation accuracy of 95% on validation training dataset and testing mean accuracy of 91%. The mean accuracy is an average of both the classes, Pits and Not-Pits. Mean Iou is 63.71% with a weighted IoU being 90.4%. The mean BF Score shows that is 51.1%. An interesting outcome of the developed system is that the test performance results show higher mean accuracy and weighted IoU, to the state of the art techniques [57], with a mean accuracy of 46.1% and mean IoU of 34%. Table 7.14 and figure 7.39 show the validation and testing performance of the model.

Table 7-14 Validation performance of initial and final version of UNet+Vgg16 model



Model Performance			
Validation Accuracy	LR	Mini Batchsize	Validation Patience
0.9028	0.001	4	4
0.9543	0.0002	4	Infinity



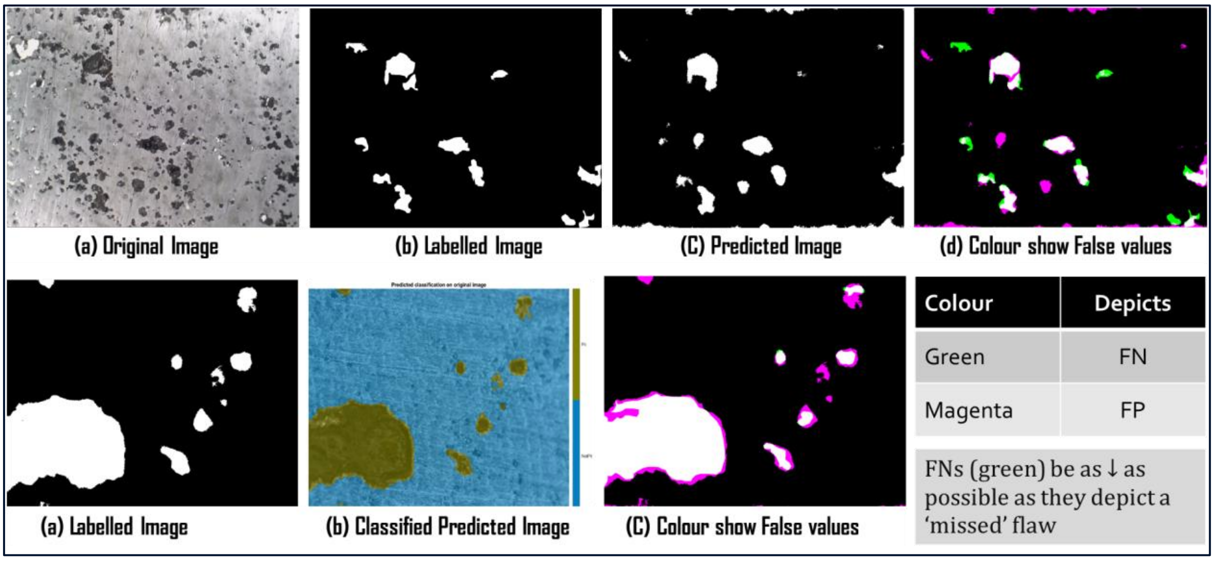
Test Data performance of UNet+Vgg16

Actual Class	NotFlaw	Flaw
NotFlaw	93.19%	6.81%
Flaw	9.24%	89.76%

Figure 7-39 Test data result of Pixel-wise classification of UNet+VGG16 model

7.8 CHAPTER SUMMARY

This chapter shows full execution of three supervised intelligent models implemented in this research. This was done by discussing the pre-trained model's architecture, the implementation parameters and steps, and then displaying the outcomes of the system in tabular as well visual image formats as shown in figure 7.40.


(a) Original Image
(b) Labelled Image
(C) Predicted Image
(d) Colour show False values
(a) Labelled Image
Predicted classification on original image
(b) Classified Predicted Image
(C) Colour show False values

Colour	Depicts
Green	FN
Magenta	FP

FNs (green) be as ↓ as possible as they depict a 'missed' flaw

Figure 7-40 Deep Learning classification performance on test images

There were a few initial trials performed to check working and setup of deep learning by varying different parameters in section 7.4 based on pixel-wise classification. It can be seen visually in figure 7.41, when outcome of initial trials CNN model is compared to the final deep learning models results, based on same test image, that the detection has vastly improved, as the database to the model started to improve. In figure 7.41, top images are the original test sample and ground truth and the bottom three images show results produced by three CNN models based on a) CNN model initial version b) UNet model and c) UNet+Vgg16 model.

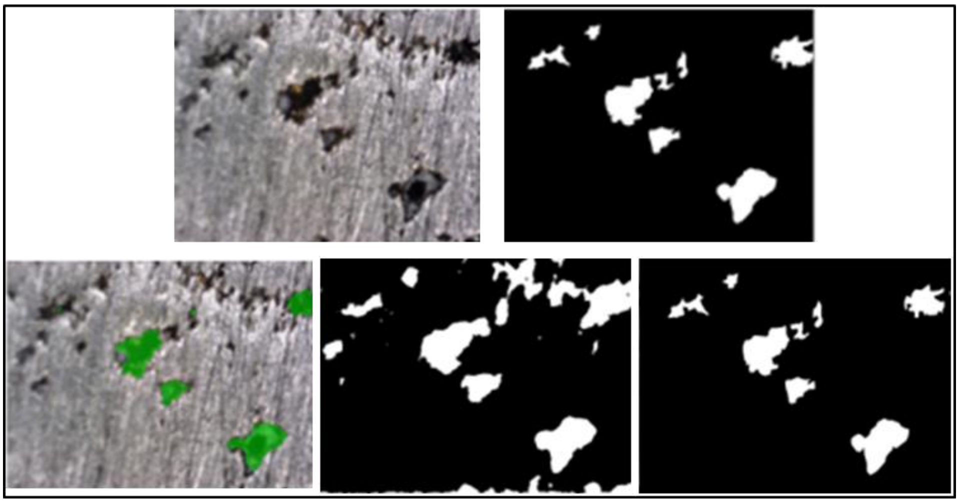


Figure 7-41 Comparison of all pixel-wise CNN models using same test image

The image-wise classification model, pre-trained with AlexNet, is the first model to be implemented after the initial trails. This classifier consists of two binary sub-models, a Crack Classifier Model and a Pit Classifier Model. With the combination of both models, the system is able to classify between a crack and a pit. The crack model shows whether it is a crack or not, while the pit model classifies the images into pit or not. The resultant outcome displays the class of the image to which it belongs as shown in figure 7.42. If the image has a pit, then the Pit model is able to pick it with 91.4% accuracy, and if the image has a crack, it is picked by the Crack model with a high accuracy of 98%. This model for crack detection shows 98% test accuracy while from the literature review, the object-detection research discussed in section 2.4.2, shows 96.5% of accuracy for detection of cracks. Both models have been trained on their own respective databases.

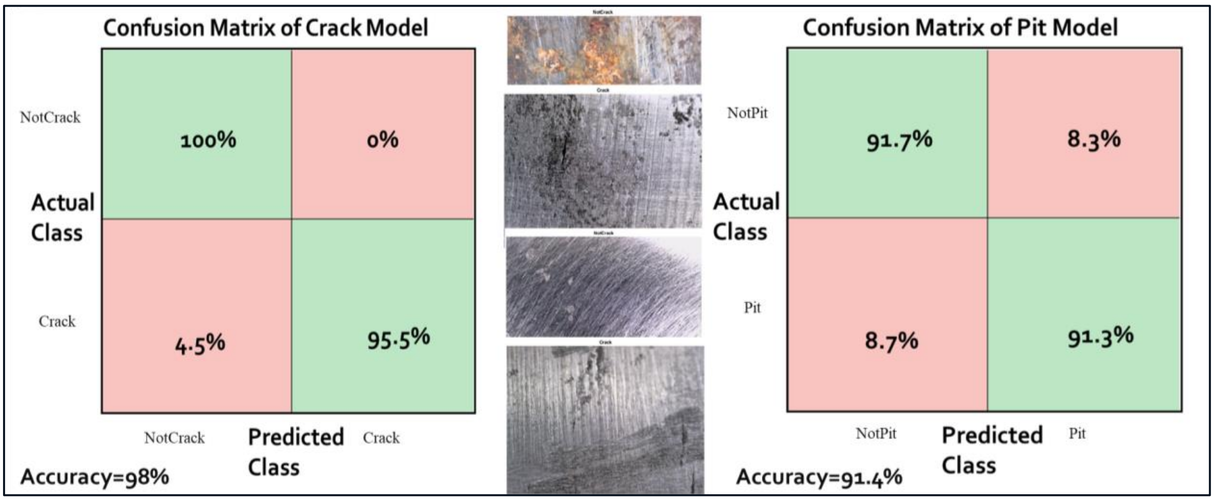
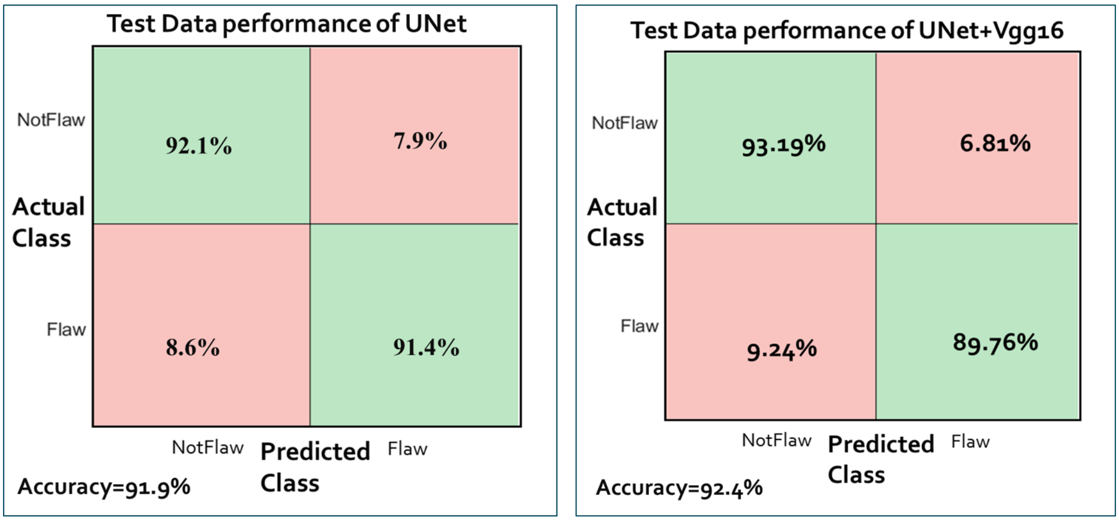


Figure 7-42 Deep Learning image-wise classification test outcome of two models

Pixel-wise UNet classification model is the second one implemented, which was selected because of its interesting architectural design. The model is able to produce, same size predicted image as the size of input image entered. It is able to outline the shape of the defect with a good effect to localise the object detected. It is able to detect and measure the flaws in the image with an accuracy of 91%. as shown in figure 7.43.



Test Data performance of UNet

Actual Class	NotFlaw (Predicted Class)	Flaw (Predicted Class)
NotFlaw	92.1%	7.9%
Flaw	8.6%	91.4%

Accuracy=91.9%

Test Data performance of UNet+Vgg16

Actual Class	NotFlaw (Predicted Class)	Flaw (Predicted Class)
NotFlaw	93.19%	6.81%
Flaw	9.24%	89.76%

Accuracy=92.4%

Figure 7-43 Deep Learning pixel-wise classification test resultant outcomes

With a combination of two state-of the art models, VGG16 and UNet, the third model implemented, shows performance with global accuracy of 93%. It is a pixel-wise classification model that is able to detect and measure the defects. With a validation accuracy of 95% on validation training dataset and testing mean accuracy of 91%. The mean accuracy is an average of both the classes, Pits and Not-Pits. Mean Iou is 63.71% with a weighted IoU being 90.4%. The mean BF Score shows that is 51.1%. An interesting outcome of the developed system is that the test performance results show higher mean accuracy and weighted IoU, to the state of the art techniques [57], with a mean accuracy of 46.1% and mean IoU of 34%.

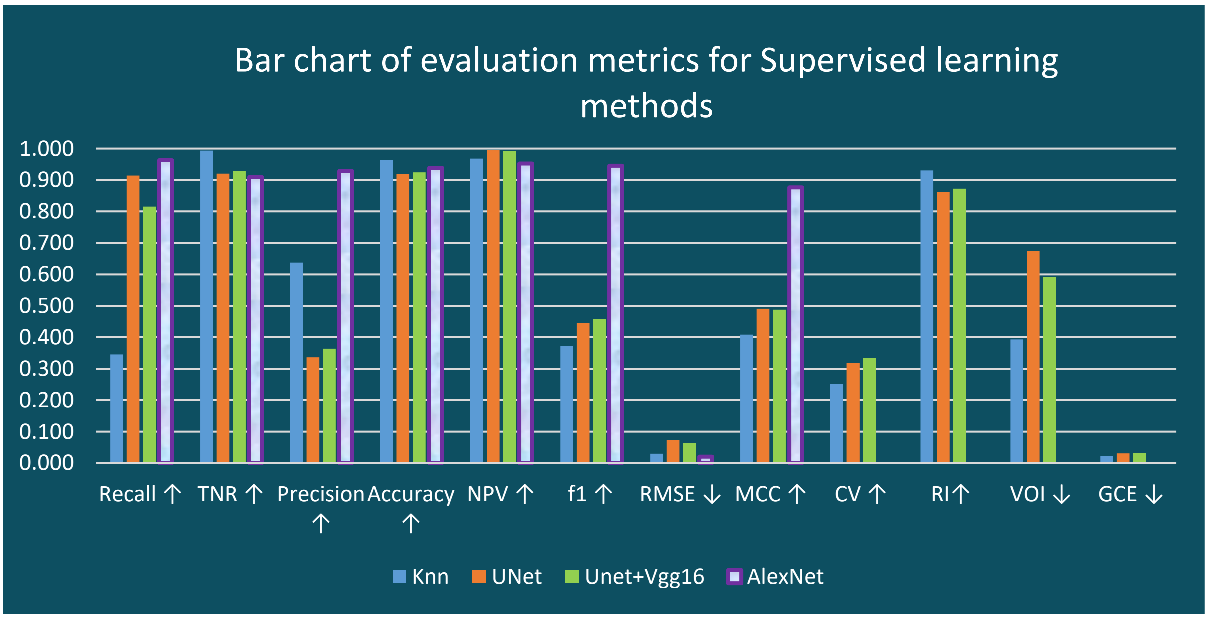


Figure 7-44 Illustrating performance results of supervised learning methods

Performance of all supervised learning methods based on multiple evaluation metrics is illustrated in figure 7.44. It shows comparison between three pixel-wise methods which are KNN classifier, UNet and the combo Unet+Vgg16 models. It also shows an image-wise performance with the end bar in purple outline. As it is image-wise, it doesn't perform on the segmentation metrics evaluation. This graph is an overview of all the results of supervised learning in one plot. As can be seen, KNN performs quite well for the background class but has very low recall value. Unet and combo unet+vgg16 produce quite similar results and are higher than the machine learning results. Image-wise Alexnet performs well based on accuracy, recall as well as precision and also it is easy to label for image-wise classification while pixel-wise takes an hour for each image.

Chapter 8
Defect Detection System (DDS) and its Industrial impact

This chapter presents the Defect Detection system (DDS) along with its industrial impact (RAAI). It includes explaining the automatic structural health inspection using microscope to improve both efficiency and accuracy. The research especially focuses on the inspection for pits and cracks of rail axles and implementation of AI techniques for automatic inspection using both supervised and unsupervised methods.

8.1 INTRODUCTION

This chapter discusses the overall Defect Detection system (DDS) along with its industrial impact (RAAI) which is the fourth main contribution to this research. It is the whole system in general, starting from system setup to data collection, then labelling it (both image-wise and pixel-wise), image analysis implementation including on-site validation and inspection. This chapter explains the design workflows of two systems, then compares them, selects one of them and then demonstrates results from the actual on-site inspection. It is implemented to improve both efficiency and accuracy of the automatic structural health inspection. This research is especially focused on the inspection for pits and cracks of rail axles using microscope, which include tasks like creation of database and implementation of AI techniques for automatic inspection using both supervised and unsupervised methods. Two designs of the Defect Detection System have been proposed and implemented. The preferred method has been implemented in the industry for on-site inspection and assessment. This chapter consists of the following:

- Design 1 of the defect detection system (DDS) and its workflow (section 8.2)
- Design 2 of the defect detection system (DDS) and its workflow (section 8.3)
  This includes comparison of both the designs and their advantages
- Next it discusses the selected design with an on-site demonstration of the application (section 8.4)
- In the end, industrial impact of the application (section 8.5)

As it is based on a real industry problem, this project comes from the actual needs of the industrial work. Based on the project aim and industrial need, an automated inspection system has been designed and implemented that can detect, measure and classify on-surface flaws such as pits and cracks. It reduces the time required to manually count the defects on the images and helps in distinguishing defects. It consists of two sub-systems,

Defect Detection system (DDS) = data collection system + image analysis system

It requires a laptop, portable microscope and an automated scanner. The laptop is attached to the microscope which capture flaws with up to 0.08mm size length sensitivity and saves data in images/video format. The microscope is mounted on the scanner, that auto-rotates the camera circumferential as well as in axial axes, along the structure component being inspected. This system has been refined by some upgrades along the way such as from hand-held camera to automated scanner. Once the data has been collected, it is passed onto the defect detection system.

Major tasks performed for the detection and measurement are; Pre-processing and feature extraction of the interesting properties such as length, area, density etc. Tasks for classification are; label the data, then training and validation of data to produce a model classifier, then test the data. On the basis of all the research and the conclusions, two versions of the Defect Detection System are proposed.

8.2 Defect Detection System (DDS) - Design 1

Design1 of DDS is the preferred method that has been implemented plus also tested in the industrial application. This solution is more practical, ready-to-use and resource-efficient which is an important factor for industry-based solutions. It is based on both supervised and unsupervised learning methods by combining their strengths.

Detects, measures and localises by unsupervised image segmentation so it doesn’t need to perform lengthy pixel-wise labelling; and classifies the flaws by using deep learning so it doesn’t need to hard-code complex computational values. It is simple yet efficient. The overview of the defect detection system (DDS) using design 1, is shown in figure 8.1, by using combined techniques of Unsupervised Image Segmentation and Supervised Image Classification

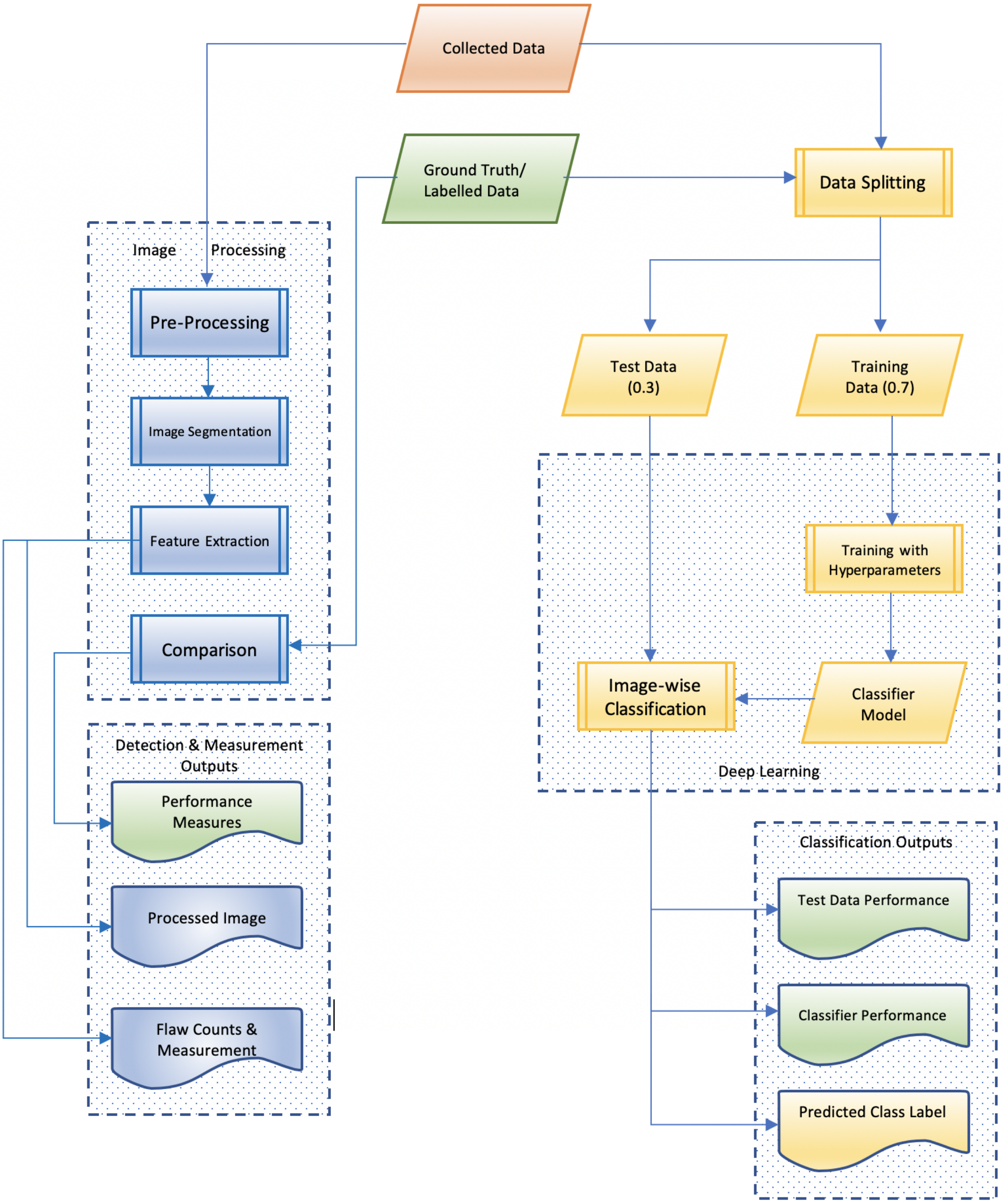


Figure 8-1 Overview of implemented Defect Detection System (DDS) workflow

8.3 Defect Detection System (DDS) - Design 2

This is the second implemented design. It has novelty in terms of the technique used, which is a combination of pixel-wise UNet and VGG16 that performs detection and measurement classification. This is completely deep learning-based design using the latest research methodologies which has produced results that are comparable to the latest state-of the art techniques. The overview of the deep learning-based detection system is shown in figure 8.2.

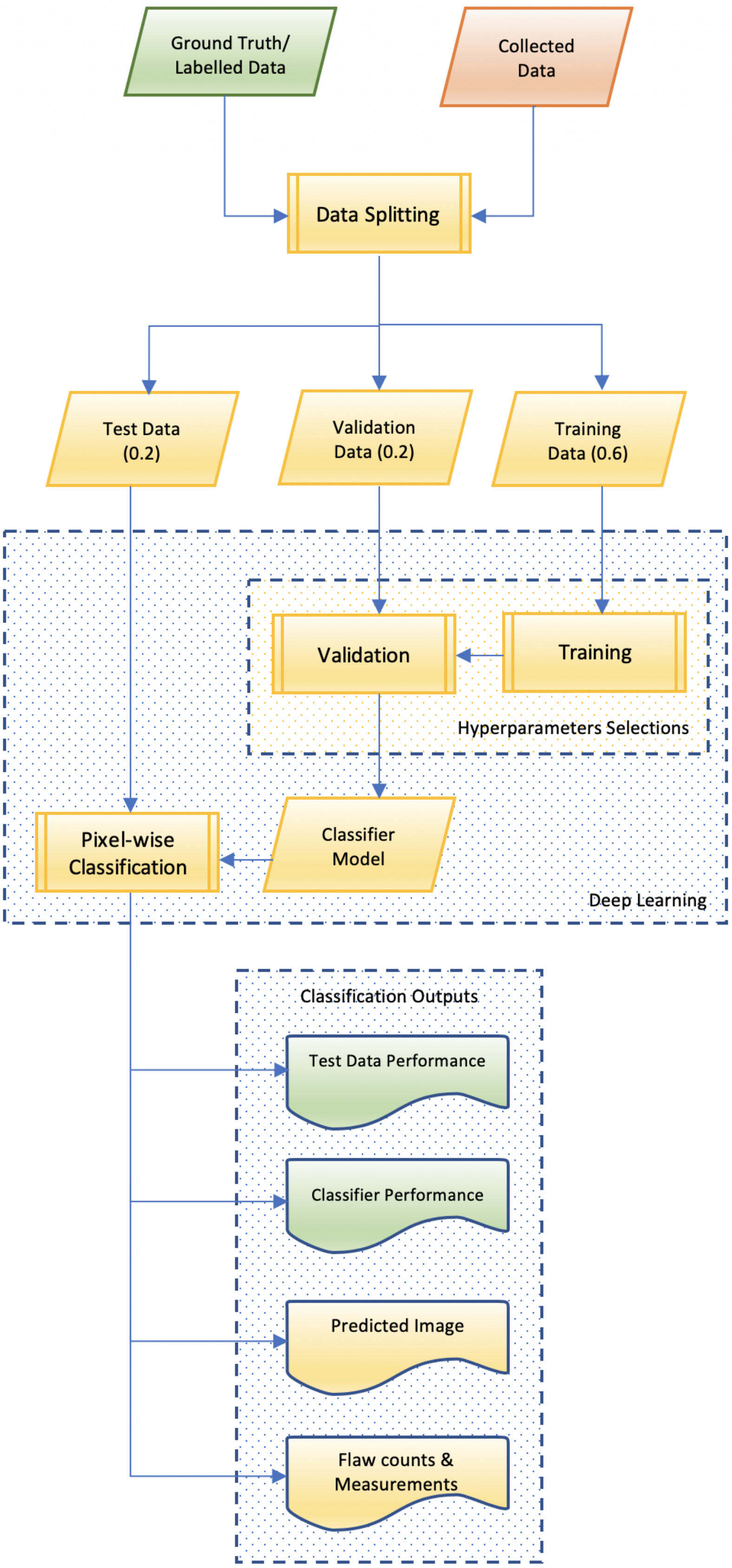


Figure 8-2 Overview of Pixel-wise classification defect detection system (DDS2)

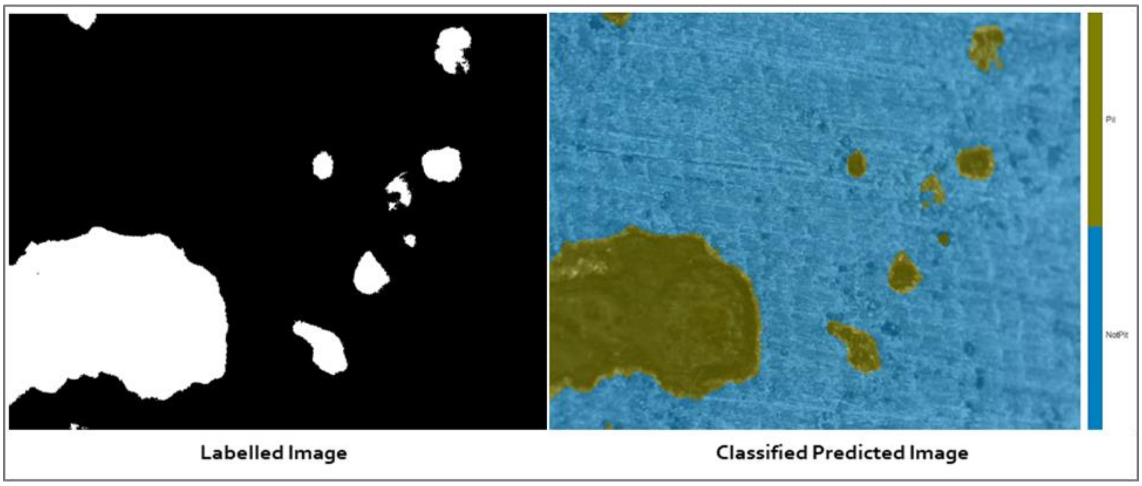


Figure 8-3 Pixel-wise classification on a test image using design-2 of DDS

The above outcomes shown in figure 8.3 results from this design 2 that uses pixelwise deep learning so it classifies each pixel as a ‘Pit’ or ‘Not-Pit’. This gives both classification as well as localisation of the defect. It produces good results but the biggest downside is that it requires pixel-wise labelling which has been extremely challenging and time-consuming. It is more reasonable to use unsupervised image segmentation that produces similar or better results while doesn’t require any labelling, but that just performs localisation, and not classification. So, for classification, use image-wise, as labelling image-wise is easier and takes less time, along with very high accuracy rate. This is exactly what design 1 is based on and hence for practical purposes the first design was implemented for on-site use.

8.4 On-site demonstration of Defect Detection System (DDS)

Based on the performance measures of the supervised and unsupervised results which are shown in figure 8.4. It can be seen that image-wise (pink) supervised classification shows high recall, precision and accuracy rate for classification of pits and cracks and pixel-wise (blue) unsupervised segmentation method also shows high recall and accuracy rate. The best of the pixel-wise method has been selected. Hence combination of both is an ideal solution to our research problem. Downside of supervised is labelled data and downside of unsupervised is hard-core complex computational programming. Both are not required as only the strengths of both are picked, which is classification is best done by deep learning and flaw measurements doesn’t require pixel-wise labelling of data which is extremely time-consuming and cumbersome.

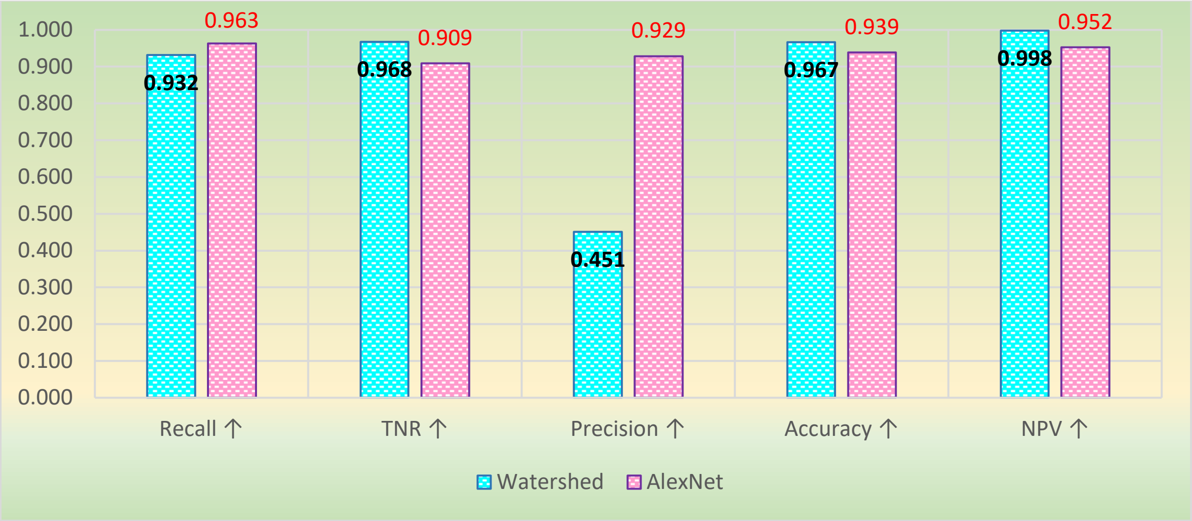


Figure 8-4 Resultant performance of DDS implemented using both methods

Hence, this system detects, measures and localises by unsupervised image segmentation so it doesn't need to perform lengthy pixel-wise labelling; and classifies the flaws by using deep learning so it doesn't need to hard-code complex computational values. It is simple yet efficient. The overview of the combined defect detection system is shown in figure 8.5

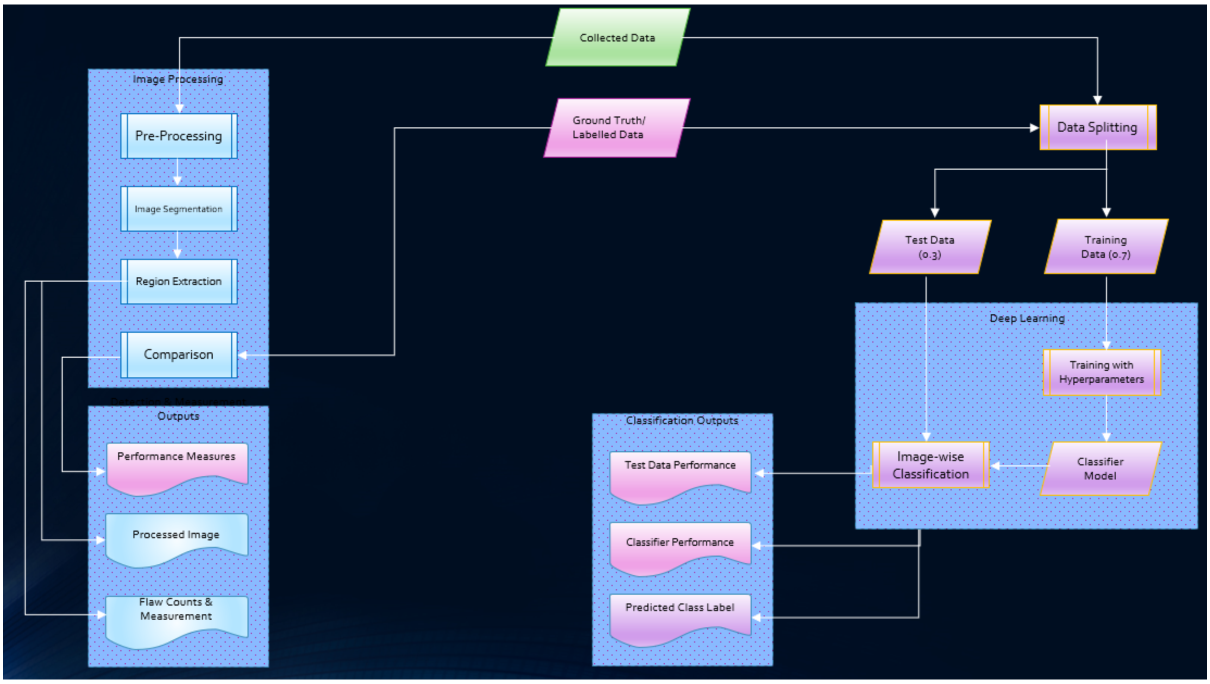
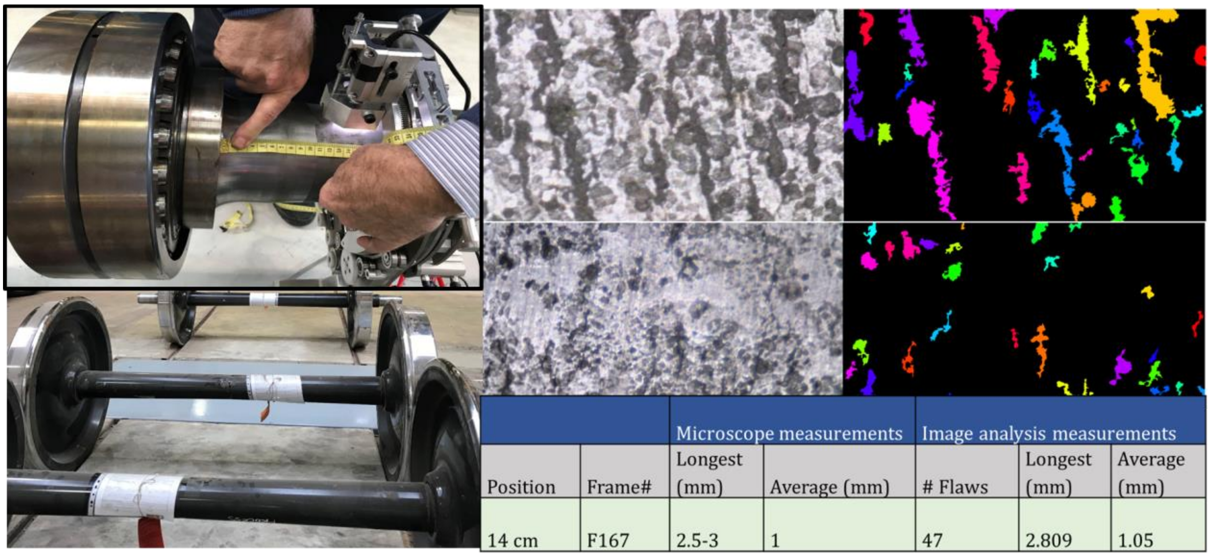


Figure 8-5 Overview of the combined Defect Detection System (DDS)

This implemented DDS design has gone through trials and validations in the industry. This application was tested on-site as can be seen in figure 8.6 by following same data collection operational protocol discussed in Chapter 3, as shown in figure 8.7. This design solution is more practical, ready-to-use and resource-efficient which is an important factor for industry-based solutions. It has been implemented to improve both efficiency and accuracy of the automatic structural health inspection.



		Microscope measurements		Image analysis measurements		
Position	Frame#	Longest (mm)	Average (mm)	# Flaws	Longest (mm)	Average (mm)
14 cm	F167	2.5-3	1	47	2.809	1.05

Figure 8-6 On-site inspection results of the Defect Detection System (DDS)

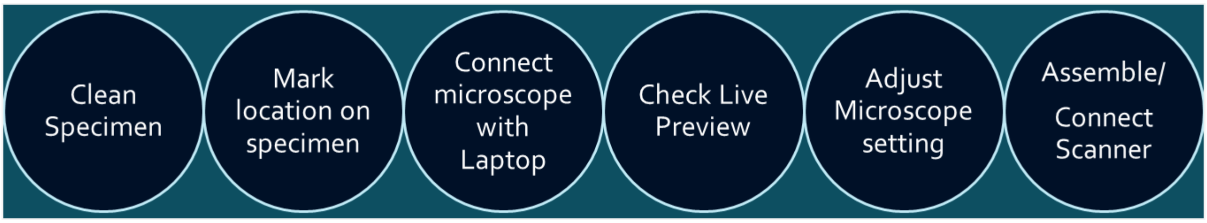

Clean Specimen
Mark location on specimen
Connect microscope with Laptop
Check Live Preview
Adjust Microscope setting
Assemble/ Connect Scanner

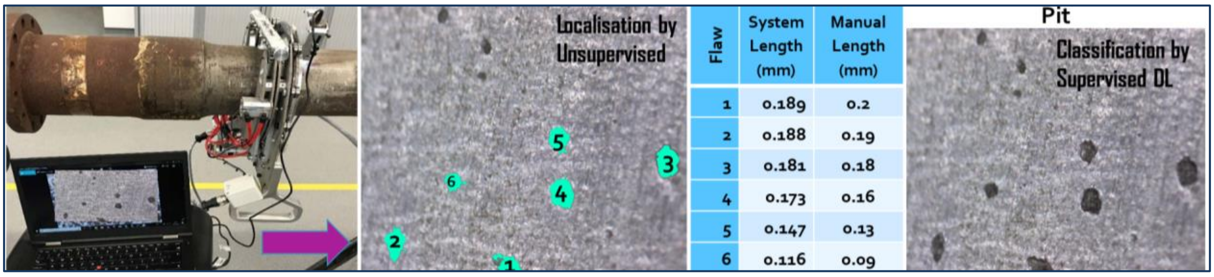
Figure 8-7 Data collection's operational procedure for DDS implementation

This system was demonstrated by going to multiple sites. Figure 8.6 indicates the flaw positions on the axle using a measuring tape. The position of the flaw was marked which produced the results also shown in figure 8.6 in a tabular form. These results were measured manually which validated the results from the defect detection system (DDS) as shown in the table within figure 8.6. These results are then passed into a model that is used to predict the structure's lifetime. The results required from the DDS are the longest crack length and average lengths of the flaws. Then it is also used as a tool to show visual results of flaw assessment. The scanner used has been upgraded to semi-automatic. It rotates axial and circumferentially on the structure to collect data.

8.5 INDUSTRIAL IMPACT

This system has been used in structural reliability assessment on rail axles as well as for pipelines that can potentially reduce the maintenance costs and still extend the useful life of a structure. Moreover, the condition of the structure health can be judged in a more objective way. Hence, from the above demonstration of the application in section 8.4, it can be seen as illustrated in figure 8.8 that this system has been able to successfully:

- Classify between pits and cracks
- Localise each micro-flaw
- Count the total number of flaws in the frame
- Measures the length and area of each flaw (saves it as an excel file)
- The time taken to process an individual image or predict on a test image is just a few seconds



Flaw	System Length (mm)	Manual Length (mm)
1	0.189	0.2
2	0.188	0.19
3	0.181	0.18
4	0.173	0.16
5	0.147	0.13
6	0.116	0.09

Figure 8-8 Successful implementation of on-site Defect detection System

This system is at the high-end of the on-surface detection as it is effective to detect micro-scale flaws yet it is extremely portable and cost effective. This system has attained high classification accuracy as well as high efficiency (comparison from manual counting to automation) so it saves great amount of valuable time of inspection as well as it is able to detect the pits and cracks at a high performance of 98% as shown in the previous chapters.

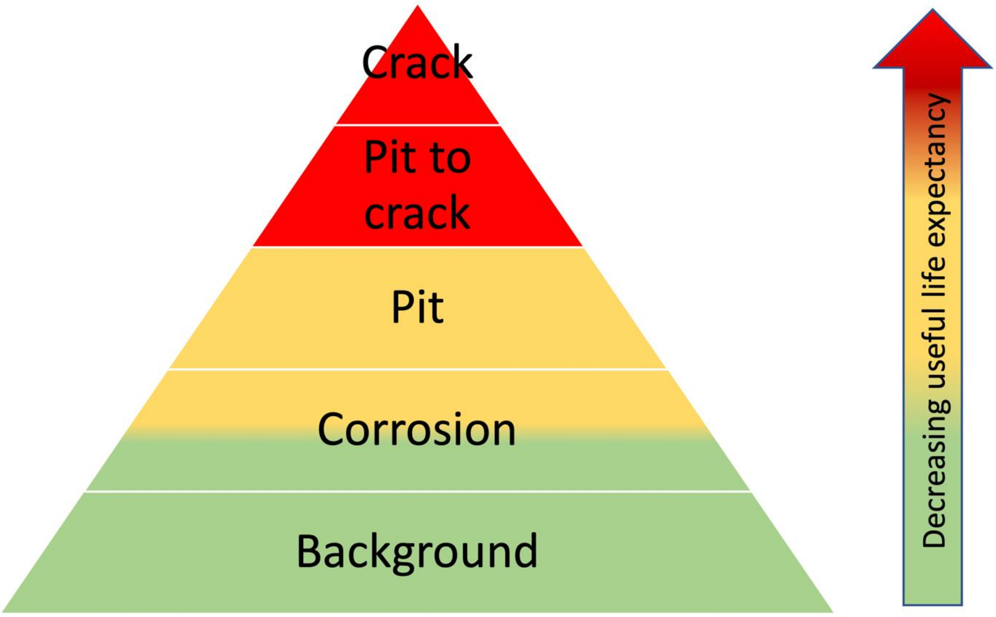


Figure 8-9 Different classes involved in flaw assessment in respect to useful life estimation

The application has been used for two purposes. First purpose is to create a tool that can provide assistance to a corrosion assessment operator, where figure 8.9 shows different flaw classes involved for assessment of corrosion fatigue. This means that the output of the defect detection system (DDS) will display the processed image by highlighting the flaws, save the flaw information in a file. This has been implemented by using unsupervised image segmentation such that it visually displays the location of the flaw by highlighting, along with the flaw count numbered, provides measurements of each of the counted flaw in an excel file. It is also able to distinguish between a crack and a pit flaw by using supervised deep learning such that it classifies the images into label classes such as pit or a crack.

The second purpose of the system is to estimate the remaining life of the axle given the presence of corrosion fatigue. It does this by detecting microscopically small cracks, which appear originating from corrosion pits in the corrosion fatigue process. Hence it is important to be able to capture the initial pit to crack phase by using a microscopic device. Then the life is estimated from the average length of the cracks. This has been implemented by using unsupervised learning methods. The outcome from the DDS system has dimensions of all the cracks and its average. This serves as an input value into the remaining-life software in the RAAI project [160] which has been validated with Polimi data.

Chapter 9
Conclusion and future works

This chapter summarises and concludes on the major points discussed throughout the thesis. This list includes data collection, creation and labelling and then all the experimental results produced by unsupervised image segmentation, supervised learning with extracted features such as local binary pattern, and deep learning performing pixel-wise segmentation as well as image-wise classification. In the end, it discusses possible improvements to enhance performance for future research

9.1 CONCLUSIONS

From this research work, an effective automated system has been developed that can detect and measure on-surface corrosion fatigue including classification of pits and cracks for microscopic visual inspection in non-destructive testing by using AI techniques, as discussed in the previous chapter. Figure 9.1 shows an overview of pixel-wise segmentation performance results of the major methods implemented, along with an image-wise in yellow at the end.

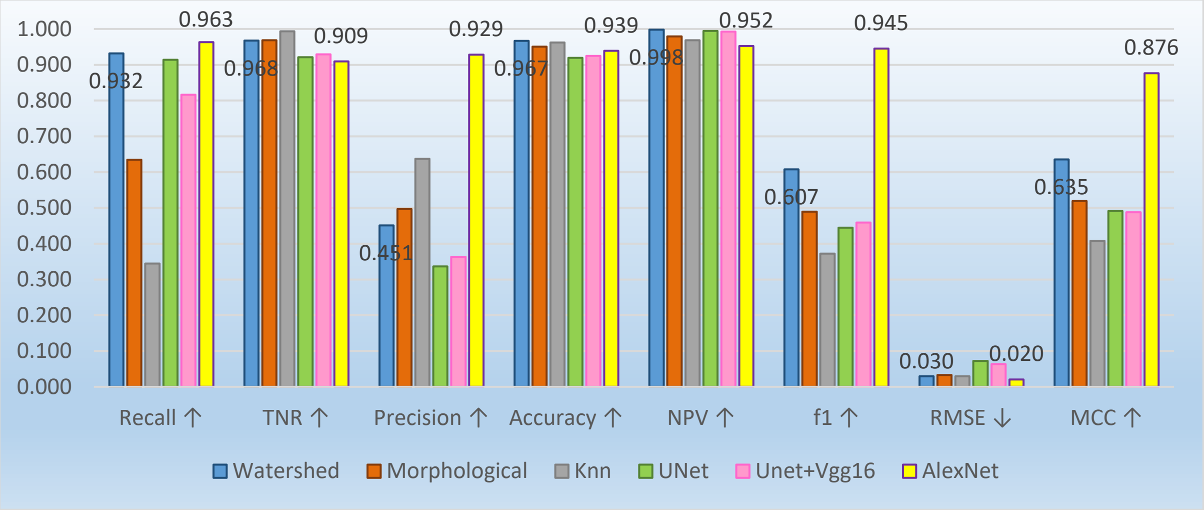


Figure 9-1 Performance overview of all main implemented methods

In order to design and implement this defect detection system (DDS), several experimental and research-oriented tasks were performed throughout the project. These tasks are focussed around four major areas: data collection and labelling (chapter 3 and 4), unsupervised image segmentation (chapter 5), feature extraction and machine learning (chapter 6) and deep learning (chapter 7). Core research tasks, experimental work and findings from these areas have been concluded in the following sections.

9.1.1 Data collection conclusion

From the <u>data collection</u> point of view,

- Investigated NDT techniques and after experiments, it was concluded that the data will be gathered by using microscopic visual inspection NDT technique with/without MPI.
- Designed a data collection system that included initial steps such as correct sample selection, thorough sample preparation, appropriate camera selection and proper hardware setup.
- Images were collected from different data sources to expand the depth of the database being created.

- Some of the noticeable findings from site visits during data collection are: degree of cleaning, influences the appearance as the contrast relies on oxide remaining; Using MPI enhances the contrast of the images especially for the purpose of crack detection; Cracks could start in machining lines which makes them difficult to distinguish.
- It was deduced from discussion with the inspection operators, that the detection unit of 0.2-0.3mm is considered as significant damage to be looked and investigated. Hence, the system device was tested to check its sensitivity, which is able to see a flaw of 0.2mm length with low magnification and 0.08mm at higher magnification. Therefore, similar measurements were used for ground truth labelling, 0.3 mm at low and 0.08 at high mag.
- It showed evidence that using the higher magnification gave more depth (clarity) to the outline of the detected flaw.
- Image quality gets affected by dark lighting conditions and improper data handling.
- The image processing results were validated by the Polimi data showing that the implemented method is able to provide a prompt outcome including highlighting, measuring and counting specific features, using on-site data. Thus, it reduces the skill level requirement of an operator, as the algorithm sets a standard for the desired defects to be counted in quantitative measures. However, this is only possible once the samples have been cleaned.
- Created a pixel-wise labelled database of 115 microscopic pit images and 20 crack images were also labelled. Steps that were taken to create ground truth include setting flaw size criteria, calibrating flaws and then labelling each of them into classes (crack, pit, pit2crack and background)
- In total, sixteen performance measurements were applied and a few added metrics for visual display. With this, the performance of the system could now be quantitatively measured such as Accuracy, Precision and Recall, based on the ground truth as the target image. So later on, when it is said that the system performed better, it means overall performance of all these measures combined.

9.1.2 Detection system conclusion

From the <u>defect detection, measurement and classification</u> point of view,

- Each of the three key areas, further consist of three implementations such as; Watershed, Morphological and FCM gaussian have been developed for unsupervised image segmentation; Decision Tree, KNN-Coarse and Ensemble classifiers for feature-based learning; Image-wise classification, Pixel-wise UNet classification and Pixel-wise UNet_VGG16 classification models for deep learning.

- All of these, except the Image-wise classifier, have been tested with the same dataset so that comparisons can be made throughout the experiments. This assures that the quality of the models can be quantifiably measured for evaluations.

- These methods work better when the images are taken at a high magnification as they have clear edges.

- The assumption made in this study, verified based on the data collection findings, is that flaw indications are of darker colour in comparison to the background. The second assumption is regarding the type of flaw, if it’s a pit then it is likely to have an elliptical shape and if it’s a crack then it has an elongated shape.

9.1.3 Unsupervised Image segmentation

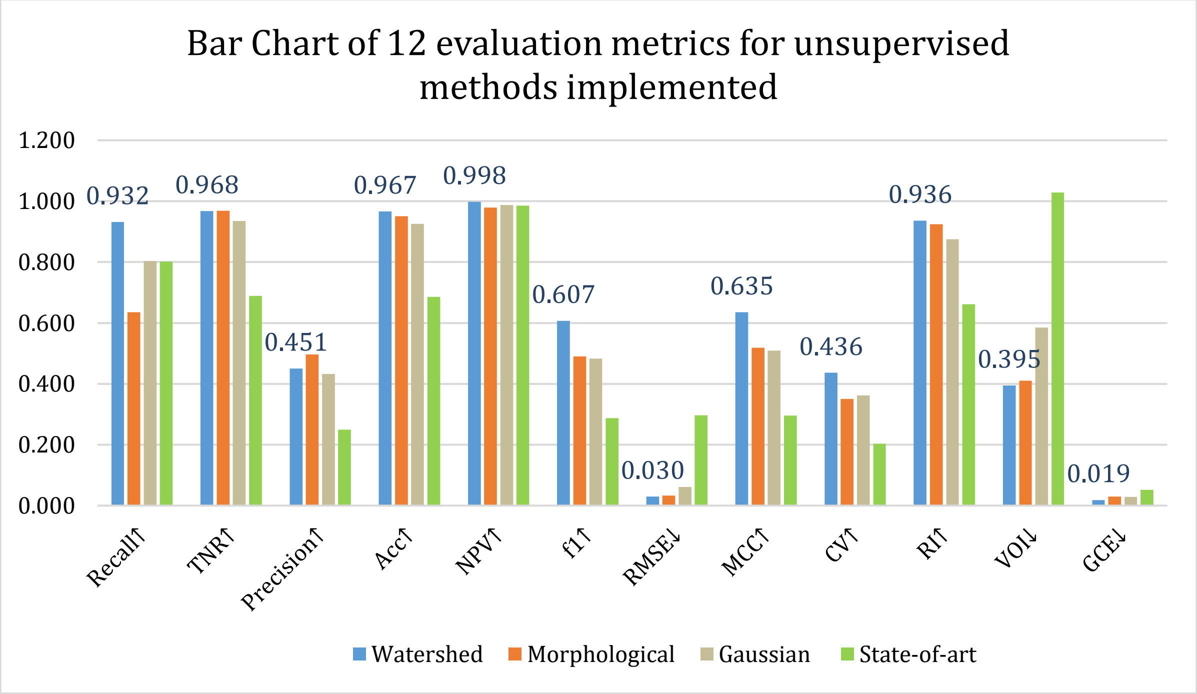
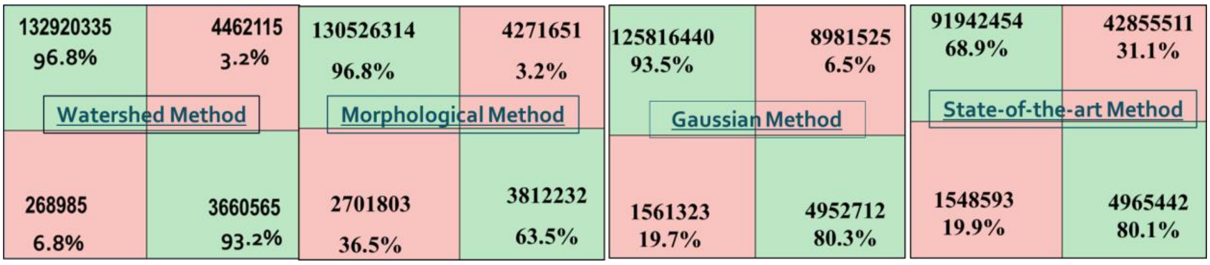


Figure 9-2 Performance overview of implemented unsupervised methods

- Implementation of all algorithms, except the initial trials, follow three major steps which include; pre-processing, segmentation and feature extraction. The outcomes consist of the resultant segmented image along with two excel files. One has the flaw measurements such as flaw length and area and the other produces the performance measures of the system such as Boundary Error, Covering segmentation, Global Consistency Error.

- Watershed-based algorithm provides an effective tool that deals with the object's boundary and finds local changes. It has been able to detect and count the number of flaws, for both pits and cracks, with good shape extraction especially the outline details. This is the easiet yet effective method that fits the requirement smartly.
- Morphological-based algorithm attempts to apply Pit Assessment, based from some of the points in the industry API-579 standard. This algorithm works by extracting shapes from the input image, based around a selected morphological structuring element. It involves operations such as erosion, dilation, reconstruction, opening and closing in a specifically designed arrangement to get better results. The shape information is extremely effective for detecting flaws which do not have a high contrast with the background.
- FCM Gaussian-based algorithm is based built around LoG edge with the watershed algorithm to generate results with less over segmentation to highlight edges. The mask of LoG is set as 5x5 which can be modified for obtaining a better segmentation result.
- Flaw measurement: Morphological algorithm's measurements are closer in value to the labelled ground truth information. Watershed's results show an edge boundary problem for the flaws detected at the edge, otherwise it is closer to the actual values.
- Performance Metrics: Watershed-based algorithm gives best performances based on 8 of the metrics applied, which is highest number of best performing metrics, followed by morphological-based. It has been able to work on both kinds of defects. Watershed shows high performances such as 95.2% accuracy, 55% precision, f1 score 56%, high probabilistic rand Index (PRI) 91.7%, CV is 42.8% and VOI as low as 41.08%, Global consistency error (GCE) as low as 2.6.



Watershed Method

132920335 96.8%	4462115 3.2%
268985 6.8%	3660565 93.2%

Morphological Method

130526314 96.8%	4271651 3.2%
2701803 36.5%	3812232 63.5%

Gaussian Method

125816440 93.5%	8981525 6.5%
1561323 19.7%	4952712 80.3%

State-of-the-art Method

91942454 68.9%	42855511 31.1%
1548593 19.9%	4965442 80.1%

Figure 9-3 Performance of unsupervised learning by using confusion matrix

- This method was compared to the state of the art method used by [144], on this dataset and it can be seen that the results produced in this research work perform better for this application as it shows higher accuracy, specificity, precision values.

9.1.4 Machine learning

- Classifiers performed differently on the basis of their model and parameters. Overall, as the data size increased and more features were added, the performance of the classifiers showed improvement. S3 with more data images, is better than S2. Tm4 with added features such as Local Binary Pattern and Gradients is better than Tm1 with just RGB base values. But it could also be because of the significance of the features extracted.

- Coarse KNN classifier gives better results than Tree and Ensemble classifiers; on the basis of the four major performance metrics; True Positive, True Negatives, False positives and False Negatives, using same data and extracted same features

- But the time taken to run the model by KNN is much longer than a simple tree or even ensemble classifiers and also requires more memory to run, so it has its computational restrictions which need to be considered.

- It is interesting to see the effect of more task-suitable classifiers, such that Coarse KNN even with just nine features, gives higher MCC value in comparison to Decision Tree with fifteen features.

- Taking Tree classifier for comparison, the set with 141312000 observations, performs better than the set with 6144000 number of input observations. This shows that as the number of images increases, the performance of learning also increases.

- When significant features are extracted such as local binary patterns and gradients then the performance of the system increases.

- Hence, it can be concluded by experiment that the outcomes are dependent on a combination of multiple factors such as; more data, more relevant and significant features extracted, and more task-suitable classifier models. Plus the time taken for training and testing the data should also be considered when choosing the best classifier.

- However, the biggest challenge with traditional ML models is the feature extraction process. It will be useful if right features can automatically be extracted. Capability of learning to focus on the right features by themselves, requiring little guidance makes DL an extremely powerful tool for modern machine learning.

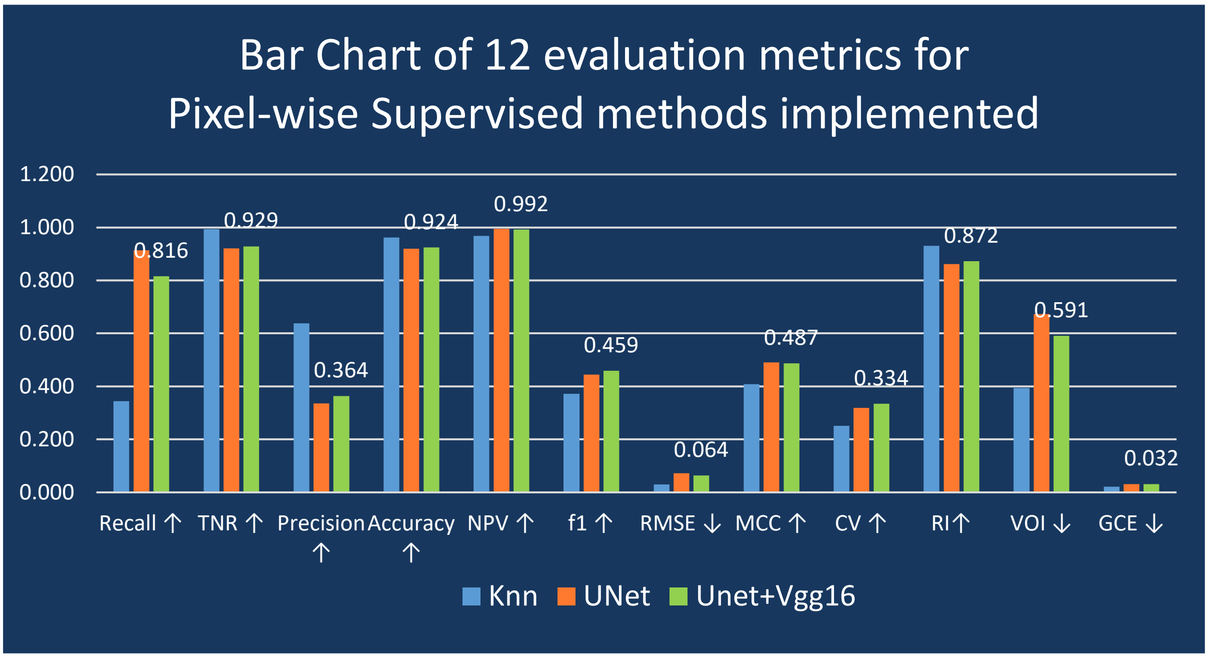


Figure 9-4 Performance overview of implemented pixel-wise supervised models

9.1.5 Deep learning

- The first deep learning model is an Image-wise classification model, pre-trained with AlexNet. This classifier consists of two binary sub-models, a Crack Classifier Model and a Pit Classifier Model. With the combination of both models, the system is able to classify between a crack and a pit. The crack model shows whether it is a crack or not, while the pit model classifies the images into pit or not. The resultant outcome displays the class of the image to which it belongs. If the image has a pit, then the Pit model is able to pick it with 91.4% accuracy, and if the image has a crack, it is picked by the Crack model with a high accuracy of 98%.

- The second deep learning model is a Pixel-wise UNet segmentation model which was implemented because of its interesting U-shaped architectural design. Because of which the model is able to produce, same size predicted image as the size of input image entered. It is able to detect and measure the flaws in the image with an accuracy of 91%. It is able to outline the shape of the defect with a good effect to localise the object detected.

- The third deep learning model is a Pixel-wise segmentation which is a combination of two state-of the art models, UNet with VGG16. It shows performance with global accuracy of 93%. With a validation accuracy of 95% on validation training dataset and testing mean accuracy of 93%. The mean accuracy is an average of both the classes, Pits and Not_Pits. Mean IoU is 63.71% with a weighted IoU being 90.4%. The mean BF Score shows that is 51.1%.

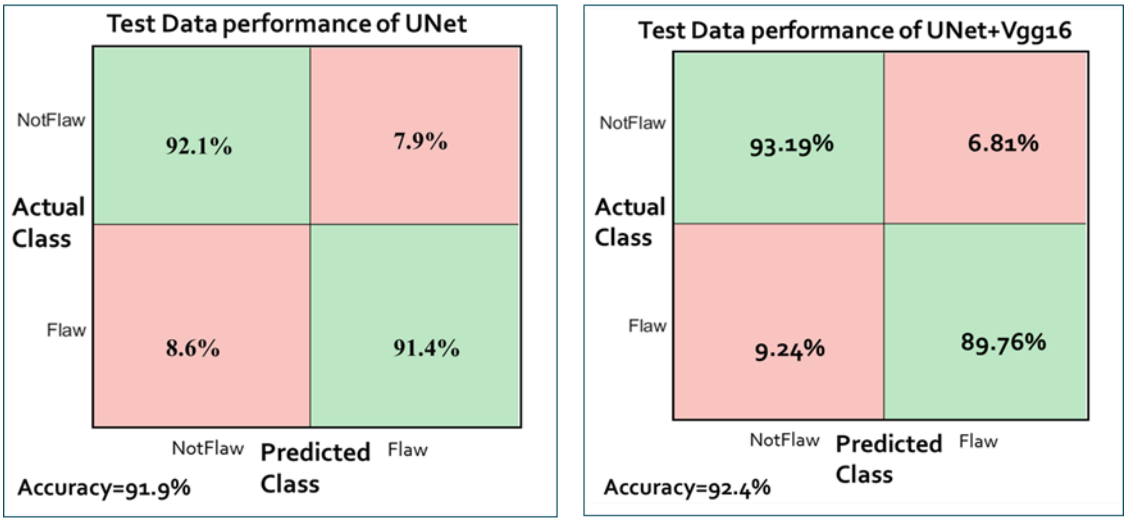


Figure 9-5 Performance overview of implemented deep learning methods

- An interesting outcome of the developed system is that the test performance results show higher mean accuracy and weighted IoU, to the state of the art techniques [57], with a mean accuracy of 46.1% and mean IoU of 34%.

9.1.6 Overall

- If a system needs to be evaluated, it needs to have a labelled data against which the quality performance can be quantified.

- Limitation of Supervised learning methods is that they require massive amount of labelled data while on the other hand, unsupervised learning can produce results without it

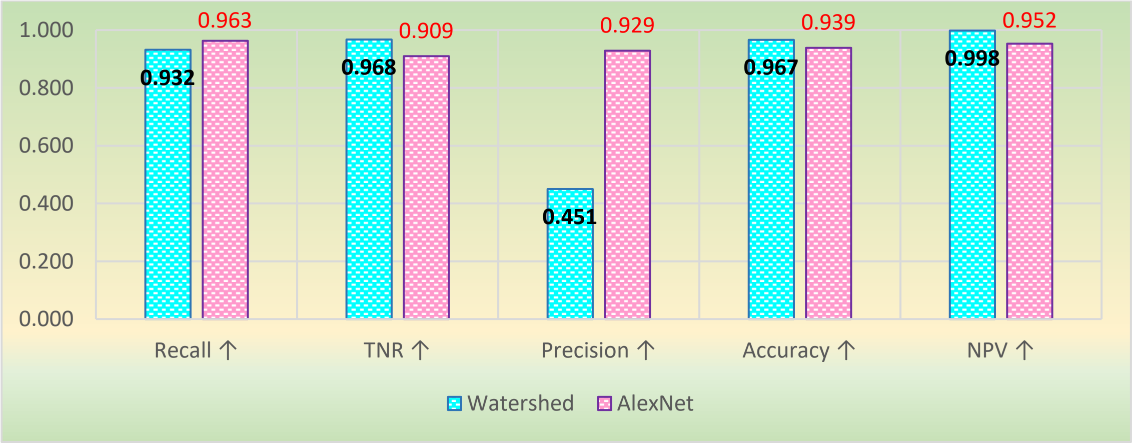


Figure 9-6 Overall performance of the implemented DDS showing both methods

- For supervised methods, for image-wise classification the procedure for labelling the ground truth is simple yet efficient unlike Pixel-wise classification. Imagine the comparison of labelling such as, image-wise : Pixel_wise. = 1: image_size, as it requires each pixel value to be labelled and the number of input values increase with that rate as well which requires high computational equipment.

- Unsupervised methods require prior hard-coded values to extract the features while deep learning extract features by automatic adaptive learning. Hence useful to implement in applications where it is difficult to define a problem such as difference between pits and cracks.

9.2 Future Works

There are a few areas of this project that can be further extended in future work provided the said resources are available. If a larger pixel-wise labelled dataset becomes obtainable, training with this larger dataset may yield significantly better results than have been achieved so far. For example, the Pascal VOC segmentation dataset contains 9,993 images, compared to the only 115 images that were used for training in this work.

If more powerful computing resources are provided, which would allow an increase of the training batch sizes, could yield a more consistent learning rate throughout each step, increasing the overall performance of the model. Furthermore, other architectures may be explored that could yield good results as well, such as the DenseNet. Post-processing methods such as Conditional Random Fields (CRF) could be applied to improve segmentation performance and could thus be an area of interest for future work as well.

Upgrading the data collection device to 3D-format may add more accuracy to the location of the defects to build the 3D Defect detection system. Performance of the two cameras could be analysed with the 3D camera adding the third dimension of depth, which might hugely increase the efficiency of detecting the defects correctly. After the pit depths are known, Industry standard API-579 could be applied for pit assessment including classifying different grading levels of pitting corrosion assessment.

Deep learning is an exciting novel method that can be applied to solve many different engineering and scientific problems especially for classification. With the combination of Image segmentation, which is a powerful tool to detect and measure interesting objects by feature extraction, this research may have a significant impact on automatic detection systems. The chief concern in many structural integrity applications is the capability of the closely related forms of corrosion to lead to accelerated failure of structural components by damage or by acting as an initiation site for cracking, causing severe loss of functionality or even catastrophic failure. Hence there are many areas where this system may be applicable such as monitoring, grading assessment, similar applications of detection and measurements such as ultrasound images, 3D additive manufacturing area.

References

[1] G. K. Choudhary and S. Dey, "Crack Detection in Concrete Surfaces using Image Processing, Fuzzy Logic, and Neural Networks," in *IEEE fifth International Conference on Advanced Computational Intelligence(ICACI)*, 2012, pp. 404–411.

[2] S. Arunachalam and S. Fawaz, "Test method for corrosion pit-to-fatigue crack transition from a corner of hole in 7075-T651 aluminum alloy," *Int. J. Fatigue*, vol. 91, pp. 50–58, 2016.

[3] F. Bonnin-pascual and A. Ortiz, "Detection of Cracks and Corrosion for Automated Vessels Visual Inspection," in *Frontiers in Artificial Intelligence and Applications*, 2010, no. October 2016.

[4] B. M. Schönbauer *et al.*, "Fatigue life estimation of pitted 12% Cr steam turbine blade steel in different environments and at different stress ratios," *Int. J. Fatigue*, vol. 65, pp. 33–43, 2014.

[5] R. Ebara, "Corrosion fatigue crack initiation in 12% chromium stainless steel," *Mater. Sci. Eng. A*, vol. 468–470, no. SPEC. ISS., pp. 109–113, 2007.

[6] Industrial Applications and Chemistry Section, "Liquid Penetrant and Magnetic Particle Testing at Level 2," Vienna, Austria LIQUID, 2000.

[7] N. O. Larrosa, M. D. Chapetti, and R. A. Ainsworth, "PVP2015-45562 ASSESSING FATIGUE ENDURANCE LIMIT OF PITTED SPECIMENS BY MEANS OF," in *Proceedings of the ASME 2015 Pressure Vessels & Piping Conference PVP2015*, 2015, pp. 1–9.

[8] J. Rudlin, S. Beretta, and A. Loconte, "Estimation of Corrosion Fatigue Status on Rail Axles," no. Ecndt, 2014.

[9] Anon, "Fitness-For-Service, API Recommended Practice 579," Second., American Petroleum Institute, 2007.

[10] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, *Digital Image Processing Using MATLAB*, 2nd ed. Gatesmark Publishing, 2009.

[11] R. C. Gonzalez and W. Paul, *Digital Image Processing*, 2nd ed. Addison-Wesley Publishing Company, Inc., 1987.

[12] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, “Deep learning for visual understanding : A review,” *Neurocomputing*, vol. 187, pp. 27–48, 2016.

[13] J. Jiang, P. Trundle, and J. Ren, “Medical image analysis with artificial neural networks,” *Comput. Med. Imaging Graph.*, vol. 34, no. 8, pp. 617–631, 2010.

[14] H. Debar, M. Becker, and D. Siboni, “A neural network component for an intrusion detection system,” in *Proceedings 1992 IEEE Computer Society Symposium on Research in Security and Privacy*, 1992, pp. 240–250.

[15] K. Gurney, *Introduction to neural networks*. London: UCL Press, 1997.

[16] B. Cheng and D. M. Titterington, “Neural networks: a review from a statistical perspective,” *Stat. Sci.*, vol. 9, no. 1, pp. 2–30, 1994.

[17] Y. Bengio, “Learning Deep Architectures for AI,” *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.

[18] Arminox Stainless, *Corrosion Aspects of Galvanic Coupling Between Carbon Steel and Stainless Steel in Concrete*. 1999.

[19] S. Kharkovsky and P. Giri, “Detection of crack in cement-based specimens using microwave imaging with the 3-axis scanning system,” in *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, 2016, vol. 2016-July.

[20] N. Giel, *Corrosion Engineering Guide*. KCI Publishing B.V., 2008.

[21] A. Turnbull, “Corrosion pitting and environmentally assisted small crack growth.,” *Proc. Math. Phys. Eng. Sci.*, vol. 470, no. 2169, p. 20140254, 2014.

[22] X. Huang and J. Xu, “3D analysis for pit evolution and pit-to-crack transition during corrosion fatigue,” *J. Zhejiang Univ. Sci. A*, vol. 14, no. 4, pp. 292–299, 2013.

[23] R. Akid, “Corrosion Fatigue,” in *Shreir’s Corrosion*, 4th ed., R. . Cottis, M. . Graham, R. Lindsay, S. . Lyon, J. . Richardson, J. . Scantlebury, and F. . Stort, Eds. Elsevier Inc, 2010, pp. 928–953.

[24] P. Prasanna *et al.*, “Automated Crack Detection on Concrete Bridges,” *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 591–599, 2016.

[25] K. Tamaki, S. Tsujikawa, and Y. Hisamatsu, “Development of a New Test Method for Chloride Stress Corrosion Cracking of Stainless steels in Dilute NaCl Solutions.,” in *Advances in Localized Corrosion*, 1991, pp. 207–214.

[26] X. Zhang, S. Li, R. Liang, and R. Akid, “Effect of corrosion pits on fatigue life and crack initiation,” in *13th International Conference on Fracture*, 2013, pp. 1–9.

[27] G. S. Chen, K.-C. Wan, M. Gao, R. P. Wei, and T. H. Flournoy, “Transition from pitting to fatigue crack growth—modeling of corrosion fatigue crack nucleation in a 2024-T3 aluminum alloy,” *Mater. Sci. Eng. A*, vol. 219, no. 1–2, pp. 126–132, 1996.

[28] C. M. Holtam, D. P. Baxter, I. A. Ashcroft, and R. C. Thomson, “Effect of crack depth on fatigue crack growth rates for a C – Mn pipeline steel in a sour environment,” *Int. J. Fatigue*, vol. 32, no. 2, pp. 288–296, 2010.

[29] S. Xu and Y. Weng, “A new approach to estimate fractal dimensions of corrosion images,” *Pattern Recognit. Lett.*, vol. 27, no. 16, pp. 1942–1947, 2006.

[30] S. I. Rokhlin, J.-Y. Kim, H. Nagy, and B. Zoofan, “Effect of pitting corrosion on fatigue crack initiation and fatigue life,” *Eng. Fract. Mech.*, vol. 62, no. 4–5, pp. 425–444, 1999.

[31] M. Cerit, K. Genel, and S. Eksi, “Numerical investigation on stress concentration of corrosion pit,” *Eng. Fail. Anal.*, vol. 16, no. 7, pp. 2467–2472, 2009.

[32] ASTM G46-94, “Standard Guide for Examination and Evaluation of Pitting Corrosion,” West Conshohocken, 2013.

[33] F. Moretti, S. Beretta, A. Lo Conte, and D. Straub, “Corrosion-fatigue under rainwater of a q&t steel: Experiments and probabilistic description,” *Procedia Eng.*, vol. 74, pp. 12–17, 2014.

[34] F. Zana and J.-C. Klein, “segmentation of vessel-like patterns using mathematical morphology and curvature evaluation.,” *IEEE Trans. Image Process.*, 2001.

[35] S. Iyer and S. K. Sinha, “A robust approach for automatic detection and segmentation of cracks in underground pipeline images,” *Image Vis. Comput.*, vol. 23, pp. 921–933, 2005.

[36] R. Amhaz and S. Chambon, “A NEW MINIMAL PATH SELECTION ALGORITHM FOR AUTOMATIC CRACK DETECTION ON PAVEMENT IMAGES,” in *International Conference on image processing (ICIP)*, 2014, pp. 788–792.

[37] ASM International, “Inspection Methods - Overview and Comparison,” in *Inspection of Metals—Understanding the Basics*, F. C. Campbell, Ed. 2013.

[38] M. Willcox and D. George, “A review of common nondestructive tests,” *TPJ - Tube Pipe J.*, pp. 1–4, 2006.

[39] H. G. Ramos and A. L. Ribeiro, “Image P os - Processing and Inversion for Eddy Current Crack Detection Problems,” *IEEE*, pp. 203–208, 2016.

[40] M. Willcox and G. Downes, “A Brief Description of NDT Techniques,” no. 771. pp. 1–22, 1981.

[41] S. C. Her and S. T. Lin, “Non-destructive evaluation of depth of surface cracks using ultrasonic frequency analysis,” *Sensors (Switzerland)*, vol. 14, no. 9, pp. 17146–17158, 2014.

[42] P. P. Raghu and B. Yegnanarayana, “Segmentation of Gabor-filtered textures using deterministic relaxation,” *IEEE Trans. Image Process.*, vol. 5, no. 12, pp. 1625–1636, 1996.

[43] Z. Chen and T. C. Hutchinson, “Image-Based Framework for Concrete Surface Crack Monitoring and Quantification,” *Adv. Civ. Eng.*, vol. 2010, pp. 1–18, 2010.

[44] V. Musoko, M. Kolinová, and A. Procházka, “Image Classification Using Competitive Neural Networks.”

[45] J. Nagi, S. K. Ahmed, and F. Nagi, “A MATLAB based Face Recognition System using Image Processing and Neural Networks,” in *4th International Colloqium on Signal Processing and its Applications*, 2008, pp. 83–88.

[46] S. A. Anwar and M. Z. Abdullah, “Micro-crack detection of multicrystalline solar cells featuring an improved anisotropic diffusion filter and image segmentation technique,” *EURASIP J. Image Video Process.*, vol. 2014, no. 15, pp. 1–17, 2014.

[47] H. Xu, Y. Tian, S. Lin, and S. Wang, “Research of image segmentation algorithm applied to concrete bridge cracks,” in *IEEE Third International Conference on Information Science and Technology (ICIST)*, 2013, pp. 1637–1640.

[48] M. Quintana, J. Torres, and J. M. Menendez, “A Simplified Computer Vision System for Road Surface Inspection and Maintenance,” *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 3, pp. 608–619, 2016.

[49] S. K. Sinha and P. W. Fieguth, “Automated detection of cracks in buried concrete pipe images,” *Autom. Constr.*, vol. 15, pp. 58–72, 2006.

[50] C. Koch and I. Brilakis, “Pothole detection in asphalt pavement images,” *Adv. Eng. Informatics*, vol. 25, no. 3, pp. 507–515, 2011.

[51] Q. Zou, Y. Cao, Q. Li, Q. Mao, and S. Wang, “CrackTree : Automatic crack detection from pavement images,” *Pattern Recognit. Lett.*, vol. 33, no. 3, pp. 227–238, 2012.

[52] F. Bonnin-Pascual and A. Ortiz, *Corrosion Detection for Automated Visual Inspection*. 2014.

[53] V. Malekian, R. Amirfattahi, M. Rezaeian, A. Aghaei, and P. Rahimi, “Automatic Detection and Localization of Surface Cracks in Continuously Cast Hot Steel Slabs Using Digital Image Analysis Techniques,” *Int. J. ISSI*, vol. 9, no. 1, pp. 30–40, 2012.

[54] S. Anwar and M. Abdullah, “Micro-crack detection of multicrystalline solar cells featuring an improved anisotropic diffusion filter and image segmentation technique,” *EURASIP J. Image Video Process.*, vol. 2014, no. 1, p. 15, 2014.

[55] Y. Fang, “Simulation , measurement and Image analysis of corrosion initiation and growth rate for Aluminum 2024 and Steel 304,” Virginia Commonwealth University, 2011.

[56] J. Ma.Guadalupe D. Gutierrez-Padilla a, Angela Bielefeldt a, Ovtchinnikovb, Serguei; Pellegrinoa and J. Silversteina, “Simple scanner-based image analysis for corrosion testing: Concrete application,” *J. Mater. Process. Technol.*, vol. 209, no. 1, pp. 51–57, 2009.

[57] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07-12-June, pp. 3431–3440.

[58] F. Buggenthin *et al.*, “Prospective identification of hematopoietic lineage choice by deep learning,” *Nat. Methods*, vol. 14, no. 4, pp. 403–406, 2017.

[59] A. Jansche, A. Choudary, R. Buttner, and D. Sutter, “Machine Learning in the Microscopy Lab,” *Wiley Analytical Science*, 2019. [Online]. Available: https://analyticalscience.wiley.com/do/10.1002/imaging.6770.

[60] R. Buettner, M. Bilo, N. Bay, and T. Zubac, “A Systematic Literature Review of Medical Image Analysis Using Deep Learning,” *2020 IEEE Symp. Ind. Electron. Appl. ISIEA 2020*, no. August, 2020.

[61] R. Buettner and M. Schunter, “Efficient machine learning based detection of heart

disease," *2019 IEEE Int. Conf. E-Health Networking, Appl. Serv. Heal. 2019*, 2019.

[62] H. Baumgartl, J. Tomas, R. Buettner, and M. Merkel, "A deep learning-based model for defect detection in laser-powder bed fusion using in-situ thermographic monitoring," *Prog. Addit. Manuf.*, vol. 5, no. 3, pp. 277–285, 2020.

[63] M. Salman and V. Baporikar, "Image Based Detection and Inspection of Cracks on Bridge Surface Using an Autonomous Robot [Review]," *Int. J. Recent Innov. Trends Comput. Commun.*, vol. 3, no. 2, pp. 23–27, 2015.

[64] J. H. Pujar, P. S. Gurjal, D. S. Shambhavi, and K. S. Kunnur, "Medical Image Segmentation based on Vigorous Smoothing and Edge Detection Ideology," *World Acad. Sci. Eng. Technol. Int.*, vol. 4, no. 8, pp. 1143–1149, 2010.

[65] R. Maini and H. Aggarwal, "A comprehensive review of image enhancement techniques," *J. Comput.*, vol. 2, no. 3, pp. 8–13, 2010.

[66] D. L. Pham, C. Xu, and J. L. Prince, "CURRENT METHODS IN MEDICAL IMAGE SEGMENTATION," in *Annu. Rev. Biomed. Eng.*, 2000.

[67] W. M. Wells, W. E. L. Crimson, R. Kikinis, and F. A. Jolesz, "Adaptive segmentation of mri data," *IEEE Trans. Med. Imaging*, vol. 15, no. 4, pp. 429–442, 1996.

[68] W. K. Pratt, *Processing Digital Image Processing*, Third., vol. 5, no. 11. Los Altos, California: A Wiley-Interscience Publication, 2001.

[69] C. Wang and Z. Ye, "Brightness preserving histogram equalization with maximum entropy: A variational perspective," *IEEE Trans. Consum. Electron.*, vol. 51, no. 4, pp. 1326–1334, 2005.

[70] S. W. Franklin and S. E. Rajan, "Computerized screening of diabetic retinopathy employing blood vessel segmentation in retinal images," *Biocybern. Biomed. Eng.*, vol. 34, no. 2, pp. 117–124, 2014.

[71] A. Drobchenko, J. Vartiainen, and J. Kämäräinen, "Thresholding Based Detection of Fine and Sparse Details," Lappeenranta, Finland, 2005.

[72] S. Rodriguez, Y. Wang, R. Akid, R. Leiva, and W. Yin, "Design of an FPGA-based eddy current instrument for the detection of corrosion pits," *Conf. Rec. - IEEE Instrum. Meas. Technol. Conf.*, vol. 2015-July, pp. 705–710, 2015.

[73] Madhulika *et al.*, "Implementing Edge Detection for Medical Diagnosis of a Bone in

Matlab," in *5th International Conference on Computational Intelligence and Communication Networks Implementing*, 2013.

[74] R. Maini and H. Aggarwal, "Study and comparison of various image edge detection techniques," *Int. J. Image Process.*, vol. 3, no. 1, pp. 1–12, 2009.

[75] Z. Liu, M. Genest, and D. Krys, "Processing thermography images for pitting corrosion quantification on small diameter ductile iron pipe," *NDT E Int.*, vol. 47, pp. 105–115, 2012.

[76] W. Zhang, Z. Zhang, D. Qi, and Y. Liu, "Automatic crack detection and classification method for subway tunnel safety monitoring," *Sensors*, vol. 14, no. 10, pp. 19307–19328, 2014.

[77] A. Landstrom and M. J. Thurley, "Morphology-based crack detection for steel slabs," *IEEE J. Sel. Top. Signal Process.*, vol. 6, no. 7, pp. 866–875, 2012.

[78] H. Wu, R. Zhao, and B. Li, "Crack image processing using probability based threshold," *IEEE*, vol. 2, pp. 3904–3907, 2011.

[79] B. Shan, S. Zheng, and J. Ou, "A Stereovision-based Crack Width Detection Approach for Concrete Surface Assessment," vol. 20, pp. 803–812, 2016.

[80] A. Z. Arifin and A. Asano, "Image segmentation by histogram thresholding using hierarchical cluster analysis," *Pattern Recognit. Lett.*, 2006.

[81] C. Fang, L. Zhe, and Y. Li, "Images Crack Detection Technology based on Improved K-means Algorithm," *J. Multimed.*, vol. 9, no. 6, pp. 822–828, 2014.

[82] Y. Jiang and Z. Zhou, "SOM Ensemble-Based Image Segmentation," *Neural Process. Lett.*, vol. 20, pp. 171–178, 2004.

[83] K. Haris, S. N. Efstratiadis, N. Maglaveras, and A. K. Katsaggelos, "Hybrid image segmentation using watersheds and fast region merging," *IEEE Trans. Image Process.*, vol. 7, no. 12, pp. 1684–1699, 1998.

[84] M. Petrou and P. Garcia, *Image Processing Dealing with Texture*. John Wiley & Sons Ltd, 2006.

[85] R. Guo *et al.*, "Pixel-wise classification method for high resolution remote sensing imagery using deep neural networks," *ISPRS Int. J. Geo-Information*, vol. 7, no. 3, 2018.

[86] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” *Proc. - 2005 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognition, CVPR 2005*, vol. I, pp. 886–893, 2005.

[87] V. Andrearczyk and P. F. Whelan, “Texture segmentation with Fully Convolutional Networks,” 2017.

[88] G. Srinivasan and G. Shobha, “Statistical texture analysis,” in *PROCEEDINGS OF WORLD ACADEMY OF SCIENCE, ENGINEERING AND TECHNOLOGY*, 2008, vol. 36, no. December, pp. 1264–1269.

[89] P. Prasanna, K. Dana, N. Gucunski, and B. Basily, “Computer Vision Based Crack Detection and Analysis,” in *Proc. SPIE 8345, Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems*, 2012.

[90] W. Castro, J. Oblitas, M. De-la-torre, C. Cotrina, H. Avila-george, and K. Baz, “Using machine learning techniques and different color spaces for the classification of Cape gooseberry (Physalis peruviana L .) fruits according to ripeness level ∗,” no. 2006, pp. 1–19, 2019.

[91] G. Guo, J. Peng, K. Yang, L. Xie, and W. Song, “Wheel Tread Defects Inspection Based on SVM,” pp. 251–253, 2017.

[92] I. Goodfellow and A. Bengio, Yoshua; Courville, *Deep Learning*. The MIT Press, 2016.

[93] T. Mitchell, *Machine Learning*. 1997.

[94] R. J. Schalkoff, *Pattern recognition: statistical, structural and neural approaches*. New York: John Wiley and Sons, 1992.

[95] X. Wu *et al.*, *Top 10 algorithms in data mining*, vol. 14, no. 1. 2008.

[96] X. K. V. Wu, *The Top ten algorithms in Data Mining*. CRC Press, 2009.

[97] K. Gopalakrishnan, “Deep learning in data-driven pavement image analysis and automated distress detection: A review,” *Data*, vol. 3, no. 3, 2018.

[98] A. E. P. Villa, W. Duch, and D. Hutchison, “Artificial Neural networks and Machine Learning – ICANN 2012,” 2012, no. September.

[99] C. M. Lynch *et al.*, “International Journal of Medical Informatics Prediction of lung cancer patient survival via supervised machine learning classification techniques,” *Int. J. Med. Inform.*, vol. 108, no. April 2016, pp. 1–8, 2017.

[100] S. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, “Machine learning: A review of classification and combining techniques,” no. May 2014, 2007.

[101] P. Wang and C. C. Seepersad, “A Comparative Evaluation of Supervised Machine Learning Classification Techniques for Engineering Design Applications,” vol. 141, no. December, 2019.

[102] P. Baudron, F. Alonso-sarría, J. L. García-aróstegui, F. Cánovas-garcía, D. Martínez-vicente, and J. Moreno-brotóns, “Identifying the origin of groundwater samples in a multi-layer aquifer system with Random Forest classification,” *J. Hydrol.*, vol. 499, pp. 303–315, 2013.

[103] R. Noori, G. Hoshyaripour, K. Ashrafi, and B. N. Araabi, “Uncertainty analysis of developed ANN and ANFIS models in prediction of carbon monoxide daily concentration,” *Atmos. Environ.*, vol. 44, no. 4, pp. 476–482, 2010.

[104] C. Peng and X. Wen, “Recent Applications of Artificial Neural Networks in Forest Resource Management: An Overview. American Association for Artificial Intelligence (AAAI) Technical Report WS-99-07,” 1999.

[105] R. S. Adhikari, O. Moselhi, and A. Bagchi, “Image-based retrieval of concrete crack properties for bridge inspection,” *Autom. Constr.*, vol. 39, pp. 180–194, 2014.

[106] B. Chandra and R. K. Sharma, “Fast learning in Deep Neural Networks,” *Neurocomputing*, vol. 171, pp. 1205–1215, 2016.

[107] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford: CLARENDON PRESS, 1995.

[108] M. Singh and K. Verma, “Speech recognition using neural networks,” *Int. J. Technol. Eng. Syst.*, vol. 2, no. 1, pp. 108–110, 2011.

[109] M. A. Nematollahi, M. Farid, M. R. Hematiyan, and A. A. Safavi, “Crack detection in beam-like structures using a wavelet-based neural network,” *Proc. IMechE*, vol. 226, pp. 1243–1254, 2011.

[110] J. Schmidhuber, “Deep learning in neural networks : An overview,” *Neural Networks*, vol. 61, pp. 85–117, 2015.

[111] L. Liu, B. Bahramimianrood, H. Zou, and S. Yang, “Welding Defects Detection and Classification by Using Eddy Current Thermography,” pp. 75–80, 2017.

[112] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2012.

[113] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.

[114] Y. Gao and K. M. Mosalam, “Deep Transfer Learning for Image-Based Structural Damage Recognition,” *Comput. Civ. Infrastruct. Eng.*, vol. 33, no. 9, pp. 748–768, 2018.

[115] ImageNet, “ImageNet LSVRC,” 2012. .

[116] M. Lin, Q. Chen, and S. Yan, “Network In Network,” pp. 1–10, 2013.

[117] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 770–778, 2016.

[118] G. Huang, S. Liu, L. Van Der Maaten, and K. Q. Weinberger, “CondenseNet: An Efficient DenseNet Using Learned Group Convolutions,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 2752–2761, 2018.

[119] Raschka, “Gradient Descent,” 2017. .

[120] M. Claesen and B. De Moor, “Hyperparameter Search in Machine Learning,” in *MIC 2015: The XI Metaheuristics International Conference*, 2015, pp. 10–14.

[121] N. Srivastava, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” *J. Mach. Learn. 15 1929-1958*, 2014.

[122] R. S. Weinstein *et al.*, “Overview of telepathology, virtual microscopy, and whole slide imaging: prospects for the future,” *Hum. Pathol.*, vol. 40, no. 8, pp. 1057–1069, 2009.

[123] K. Dorst and N. Cross, “Creativity in the design process: Co-evolution of problem-solution,” *Des. Stud.*, vol. 22, no. 5, pp. 425–437, 2001.

[124] L. P. Shari and M. A. Joannne, *Software engineering*. Pearson Prentice Hall, 2006.

[125] J. Sharma and J. K. Rai, “Enhancement of Mammogram Images,” in *International Conference on Medical Imaging, m-Health and Emerging Communication Systems (MedCom). IEEE.*, 2014, pp. 115–119.

[126] J. J.R, *Introductory digital image processing: A remote sensing perspective*. United States: Prentice Hall, Inc.,Old Tappan, NJ, 1986.

[127] S. A. S. Abedi, "Exploring Discrete Cosine Transform for Multi- resolution Analysis," Georgia State University, 2005.

[128] Q. Z. Q. Zhu and A. Alwan, "An efficient and scalable 2D DCT-based feature coding scheme for\nremote speech recognition," *2001 IEEE Int. Conf. Acoust. Speech, Signal Process. Proc. (Cat. No.01CH37221)*, vol. 1, pp. 113–116, 2001.

[129] M. H. Hassoun, *Fundamentals of artificial neural networks*. 1995.

[130] T. Jia, Yangqing and Shelhamer, Evan and Donahue, Jeff and Karayev, Sergey and Long, Jonathan and Girshick, Ross and Guadarrama, Sergio and Darrell, "Caffe," 2014. .

[131] Tensorflow, "Tensorflow," 2018. [Online]. Available: https://www.tensorflow.org/guide. [Accessed: 20-Sep-2007].

[132] S. Bahrampour, N. Ramakrishnan, L. Schott, and M. Shah, "Comparative Study of Deep Learning Software Frameworks," 2015.

[133] D. Kondermann, "Ground Truth Design Principles An Overview," in *Proceedings of the International Workshop on Video and Image Ground Truth in Computer Vision Applications*, 2013, pp. 1–4.

[134] P. Acharjya and A. Sinha, "a New Approach of Watershed Algorithm Using Distance Transform Applied To Image Segmentation," *Int. J. Innov. Res. Comput. Commun. Eng.*, vol. 1, no. 2, pp. 185–189, 2013.

[135] M. Xess and S. Agnes, "Analysis of Image Segmentation Methods Based on Performance Evaluation Parameters," *Int. J. Comput. Eng. Res.*, vol. 4, no. 3, pp. 68–75, 2014.

[136] K. Appiah, A. Hunter, P. Dickinson, and H. Meng, "Accelerated hardware video object segmentation: From foreground detection to connected components labelling," *Comput. Vis. Image Underst.*, vol. 114, no. 11, pp. 1282–1291, 2010.

[137] A. Arena, C. Delle, and J. Sarout, "A new computational approach to cracks quantification from 2D image analysis : Application to micro-cracks description in rocks," *Comput. Geosci.*, vol. 66, pp. 106–120, 2014.

[138] Mathworks, “Connected Components,” 2019. [Online]. Available: https://uk.mathworks.com/help/images/label-and-measure-objects-in-a-binary-image.html. [Accessed: 20-Jun-2009].

[139] O. Marques, “Feature Extraction and Representation,” in *Practical Image and Video Processing Using MATLAB®*, John Wiley & Sons Inc., 2011, pp. 447–474.

[140] R. M. Pidaparti, B. S. Aghazadeh, A. Whitfield, A. S. Rao, and G. P. Mercier, “Classification of corrosion defects in NiAl bronze through image analysis,” *Corros. Sci.*, vol. 52, no. 11, pp. 3661–3666, 2010.

[141] G. Liang, K. Zheng, S. Wang, C. Tu, and X. Wang, “Existing Weld Seam Recognition Based on Image Processing,” pp. 181–186, 2017.

[142] S. Iyer and S. K. Sinha, “A robust approach for automatic detection and segmentation of cracks in underground pipeline images,” vol. 23, pp. 921–933, 2005.

[143] J. Kittler and J. Illingworth, “Minimum Error Thresholding,” *Pattern Recognit.*, vol. 19, no. 1, pp. 41–47, 1986.

[144] T. Lei, X. Jia, Y. Zhang, S. Liu, H. Meng, and A. K. Nandi, “Superpixel-Based Fast Fuzzy C-Means Clustering for Color Image Segmentation,” *IEEE Trans. Fuzzy Syst.*, vol. 27, no. 9, pp. 1753–1766, 2019.

[145] A. L. Samuel, “Some Studies in Machine Learning,” *IBM J. Res. Dev.*, vol. 3, no. 3, pp. 210–229, 1959.

[146] John, “RGB Image representation,” 2000. [Online]. Available: https://www.geeksforgeeks.org/matlab-rgb-image-representation/. [Accessed: 20-Sep-2010].

[147] C. Kanan and G. W. Cottrell, “Color-to-Grayscale : Does the Method Matter in Image Recognition ?,” *PLoS One*, vol. 7, no. 1, 2012.

[148] Ransac, “Image Filtering,” 2008. [Online]. Available: https://ai.stanford.edu/~syyeung/cvweb/tutorial1.html. [Accessed: 20-Jul-2006].

[149] A. Bosch, A. Zisserman, and X. Mu, “IEEE 2007 - Image Classification using Random Forests and Ferns.pdf,” *2007 IEEE 11th Int. Conf. Comput. Vis.*, pp. 1--8, 2007.

[150] Z. Yichi, W. Lv, and L. Xuedong, “Defects Detection of Cold-roll Steel Surface Based on MATLAB,” in *Third International Conference on Measuring Technology and*

Mechatronics Automation Defects, 2011, pp. 26–29.

[151] S. N. Gowda and C. Yuan, “ColorNet: Investigating the importance of color spaces for image classification,” in *Asian Conference on Computer Vision 2018*, 2019, pp. 1–17.

[152] A. Woodland and F. Labrosse, “On the separation of luminance from colour in images On the separation of luminance from colour in images,” in *Vision, Video and Graphics*, 2005, pp. 29–36.

[153] D. Jyoti Bora, “Importance of Image Enhancement Techniques in Color Image Segmentation: A Comprehensive and Comparative Study,” *Indian J.Sci.Res*, vol. 15, no. 1, pp. 115–131, 2017.

[154] J. Moka, “Image classification with HSV Color Model Processing,” 2017. .

[155] M. Poorani, T. Prathiba, and G. Ravindran, “Integrated Feature Extraction for Image Retrieval,” *Ijcsmc*, vol. 2, no. February, pp. 28–35, 2013.

[156] Mathworks, “Pretrained Deep Neural Networks,” 2019. .

[157] S. Collet, “AlexNet,” 2017. .

[158] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9351, pp. 234–241, 2015.

[159] Utkarsh, “Semantic Segmentation of Aerial images Using Deep Learning,” 2019. .

[160] S. Beretta, F. Sangalli, J. Syeda, D. Panggabean, and J. Rudlin, “RAAI Project: Life-prediction and prognostics for railway axles under corrosion-fatigue damage,” *Procedia Struct. Integr.*, vol. 4, pp. 64–70, 2017.