DEVELOPMENT OF ALGORITHMS TO FACILITATE AUTOMATIC SURFACE DAMAGE DETECTION AND LOCALISATION FROM SCANNED DATA

By

EBUBE EZI EZI

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

School of Computing and Engineering
Centre for Precision Technologies
University of Huddersfield

October 2021

Copyright Statement

Acknowledgement

Firstly, and most importantly, I want to thank God for ending a journey that started a few years ago. It has been the most difficult and challenging part of my life, and I want to say thank you, LORD. There were times when I felt really depressed, but I had to encourage myself through your word.

My sincere and in-depth appreciation goes to my supervisor, Prof Andrew Longstaff, for guiding and supporting me through this journey. Thank you, Andrew, for understanding how challenging this has been and the impact of personal issues on my production level; I had no reason to meet with my tutor because I could discuss these issues with you. Thank you for helping me understand the whole process and the high-level criticisms that were made in a very tactful way. It kept me on my toes when I was "in the valley of shit" for the second half of this journey. I want to use this medium to earnestly appreciate Dr Simon Fletcher for his support and technical advice whenever I needed one. My immense appreciation goes to Andrew Bell, Dr Wencheng Pan (Winston), John Alamina for their support at moments when things get overwhelming. I want to show my gratitude for your time and efforts; sometimes, John keeps awake late for my sake and showed determination when I was discouraged. Winston was very passionate about supporting me. And to my colleagues, other researchers (both graduated and present), and staffs at the Centre for Precision Technologies (CPT), thank you for the few minute conversations, suggestions and help offered throughout this journey. It will be very ungracious for me not to thank the University of Huddersfield (including all departments I encountered) for making this journey possible through the vice chancellor's scholarship and providing the platform for actualising this feat. I want to say thank you greatly.

My unequivocal appreciation goes to my beautiful wife and our baby on the way, my parents, siblings, and the family members that I spoke to about this journey. Thank you, Dad, for your encouragements, thank you, mum, for your continuous prayers, and to my siblings who kept asking how I was faring with my studies. I want to thank my wife for understanding the demanding nature of this journey; she always knows when half of me is thinking about my PhD, when I am under pressure and the psychological effects after a long unfruitful day in school.

Abstract

Surface reconstruction, or reverse engineering of a surface, is the practice of combining data collection and analysis techniques for retrieving measurement data from an object with the aim of generating a digital representation of the object, where the original is either missing or incomplete. Recovering the digital representation of a scanned physical shape will, in most cases, be in the form of a three-dimensional (3D) point cloud. However, the process of capturing the data will inevitably lead to a point cloud contaminated by the uncertainty of measurement and imperfections in the measured object. To reverse engineer an object, especially a used one, consideration must be made for any manufacturing defects (tolerances or imperfections) and in-service damage. Given the significant challenges in designing a surface reconstruction algorithm, this project aims to develop a framework for reverse engineering/ performing surface reconstruction for damaged objects and develop a method for identifying and localising damage on the surface of a model with an accuracy of a few tens of micrometres. This accuracy level is significant due to the resolution of the generated data.

The workflow for reverse engineering considered in this thesis is capturing point cloud data (PCD) using a laser scanner mounted on an Articulated Arm Coordinate Measuring Machine (AACMM). Preprocessing of the raw points is performed to mitigate noise, sparse data, and contamination from the measuring system and identify properties that influence the data file structure, such as interoperability. Before performing detection, it is necessary to identify geometric features from the PCD, since an artefact's "nominal" design geometries are unknown. Hence, a linear fitting method is applied to the input point cloud data, and the system is optimised to obtain a good level of robustness of fitting the line of best fit for estimating "nominals". A combination of edge detection, profile monitoring, model sectioning, and machine learning are used to identify potential damage locations on a model's surface.

The processes are ideally designed to be reproducible, so with reduced human influence. Slices produced from model sectioning of the PCD are unwrapped to create planar (X-Y) data at the micrometre-level, suitable for using machine learning (ML). The ML is trained and tested to classify the spatial-sequences of the data as either "good" or" damaged". When applied to a potentially damaged surface, the sequence classification can then identify the location of damage along the height axis of a model. Future work would generalise this into three dimensions.

The performance of the proposed system is evaluated through mathematical simulation of noisy point cloud data on objects with simulated defects and by physical validation using measurement data. The results indicate that the method can detect and localise areas of damage. This is a stepping-stone to automated repair of damaged surfaces by adaptively reproducing a copy of the original part, considering deviation of the "as manufactured" from the design intent. The thesis contribution to knowledge is a novel method of damage detection and localisation at the micrometre level as a step towards implementing a surface reconstruction framework, which has also been developed in this study.

Table of Contents

List of Figures

List of Tables

Nomenclature

2D	Two-Dimensional
3D	Three-Dimensional
AACMM	Articulated Arm Coordinate Measuring Machine
AFR	Automatic feature recognition
AI	Artificial Intelligence
CAD	Computer-Aided Design
CAE	Computer-Aided Engineering
CAM	Computer-Aided Manufacturing
CAPP	Computer-Aided Process Planning
CCD	Charge-Coupled Device
CMM	Computer Measuring Machine
CNN	Convolutional Neural Network
CSG	Constructive Solid Geometry
CT	Computerized Tomography
CWT	Continuous Wavelet Transform
FPGA	Field-Programmable Gate Array
FPP	Fringe Projection Profilometry
GUI	Graphical User Interface
ISO	International Organization for Standardization
LoG	Laplacian of Gaussian
LPP	Linear Programming Problem
LSI	Linear Scattered Interpolation
LSTM	Long-Short Term Memory
MAT	Medial Axis Transform
ML	Machine Learning
MRI	Magnetic Resonance Imaging
NC	Numerical Control
NDT	Non-Destructive Testing
NEU	Northeastern University, China

NMS	Non-Maximum Suppression
NN	Neural Network
OEM	Original Equipment Manufacturer
OOFF	Object-Oriented Feature Finder
PCD	Point Cloud Data
Q-Q	Quantile-Quantile
RANSAC	Random Sample Consensus
RE	Reverse Engineering
RNN	Recurrent Neural NetworkSLS Structured-light Scanning
SPAT	Single-Point Articulation Test
SPC	Statistical Process Control
SR	Surface Reconstruction
STL	Stereolithography
SWR	Steel Wire Rope
UKAS	United Kingdom Accreditation Service

Definition of Terms

Surface Damage	In this thesis, the use of "surface damage" encompasses both surface defects and surface imperfections. This term is preferred throughout, since defects are normally considered as quality control issues during manufacturing, whereas in this work they relate both to manufacturing and in-service defects and imperfections.
Surface Defect	A defect is defined as not meeting the requirement for designed usage or reasonable expectation (ISO 8402:1995). For this thesis, surface defects are considered as out-of-tolerance deviation in the surface form of actual geometric features from the nominal surface form of those features.
Surface Imperfection	An imperfection is defined as an irregularity that appears on the real surface which might be unintentionally or can be accidentally caused during manufacture, storage, or usage (ISO 8785:1999). It does not necessarily specify if a surface is fit-for-function. As applied in this

thesis, imperfections need to be considered in terms of increased complexity in reverse engineering of measurement data to nominal, but do not require quantification in terms of roughness and waviness.

Reference Surface — This is a geometrical surface on which the dimensions of a surface imperfection are assessed. They pass through the highest point of the real surface with exception of the imperfections.

Chapter 1 Introduction

1.1 General Overview

The concept of surface reconstruction (SR) is part of a process known as reverse engineering (RE). “Reverse engineering” of an object has existed in practice for centuries. Whenever someone saw an item they wanted but could not obtain the original designs, they would try to work out how to make it themselves. Sometimes RE is undertaken for industrial espionage, but in many cases, it is because an item has become obsolete and can no longer be procured. At a very basic level, the term RE could be applied to “borrowing” the design concept. The more recognised forms of RE are taking measurements of an object. Traditionally, this has been individual, manual, discrete measurements of key features.

The concept of Reverse Engineering (RE) is a systematic approach that uses the technique of examining an existing engineering product or artefact with unknown dimensions to extract detailed design information and specification such that it can be accurately reproduced. In contrast to the traditional engineering process, this technique helps learn or understand how products are made and how they function by digitising the shape information of a real object using the technology of three-dimensional (3D) scanning. This technique also helps in duplicating or remodelling a product for enhanced performance. The application of RE has historically been on hardware components, but relatively recent attention has also been given to RE in forensic aiding crime scene and accident scene investigation. This engineering application cuts across several fields like manufacturing, animation, military and medical, to mention a few.

The concept of RE is happening all around us every day. Companies are always trying to access information about a competitor’s product to understand the functionality and the rationale behind their design to enhance their product to surpass a competitor’s product. In medical applications, drugs can be reverse engineered to obtain elements used in developing such a drug to either improve or investigate why and how it functions the way it does. There is more to RE than just copying another person’s works, and perhaps it is unethical in some countries with this motive. RE is employed when interfacing between two or more systems, and the main requirement for this is a product’s interoperability. This is one reason why communication can be established between a mathematical application and CAD software,

why a platform like Microsoft office can communicate with other applications, and why data captured from any experimental platform can be analysed, visualised, and altered on a different application completely independent of the capturing system.

RE is applied to situations of obsolescence where a product has become obsolete with no available design drawing and is no longer manufactured by the Original Equipment Manufacturer (OEM). The only way to incorporate it into modern technology is to reverse engineer it. Another application of RE which can be quite challenging is medical image-based modelling for constructing 3D virtual models of the human body anatomical structures constructed from anatomical data scans such as computerized tomography (CT) and Magnetic resonance imaging (MRI) [1], and laser scanning [2]. One of the most recent RE applications is in forensics, aiding in crime and accident scene investigation [3, 4]. The recent technological advancement on how data for RE is captured has aided in more application areas. It has made RE popular and a viable technique in developing a 3D virtual model of an existing physical product using 3D CAD, CAM, and CAE applications and is known as digitising.

The conventional engineering process follows the path shown in Figure 1.1. Considering the product's functionality, the idea is conceptualised and visualised using design tools. These design tools produce CAD models as a representation of the concept, and it is at this stage that the functionality of a product can be assessed and modified. Once the design process is satisfactory, the product is manufactured, and validation is performed to correlate the initial idea. In most cases, the validation is aided by the process of RE, where the digitisation of the manufactured part is validated against the CAD model of the design stage.

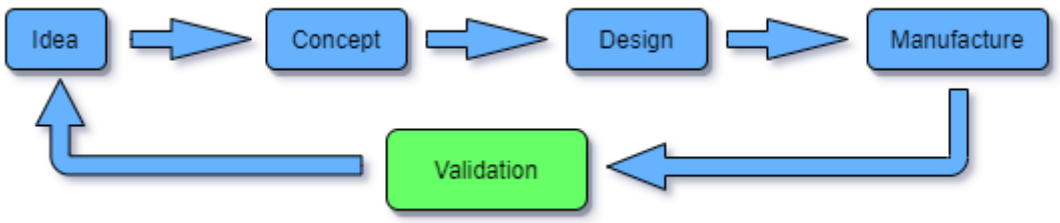


Figure 1.1: The conventional engineering concept which is a forward process, and the manufactured part should validate the idea

The reverse engineering theory is, in essence, quite different, as shown in Figure 1.2. A manufactured part is interrogated through a chosen measurement tool to obtain the relevant

dimensions and geometry. However, without a design drawing, it is the judgement of the person undertaking the RE to decide which are the “key” features. From this data, a design can be created to produce a similar product. Ideally, they require sufficient indicators and engineering knowledge to be able to generate a design with the correct functional requirements. However, without access to the original “design intent”, there is no guarantee that the correct attributes have been captured, or that conversion from an as-manufactured product, through data capture and analysis will not have introduced errors in the new design.

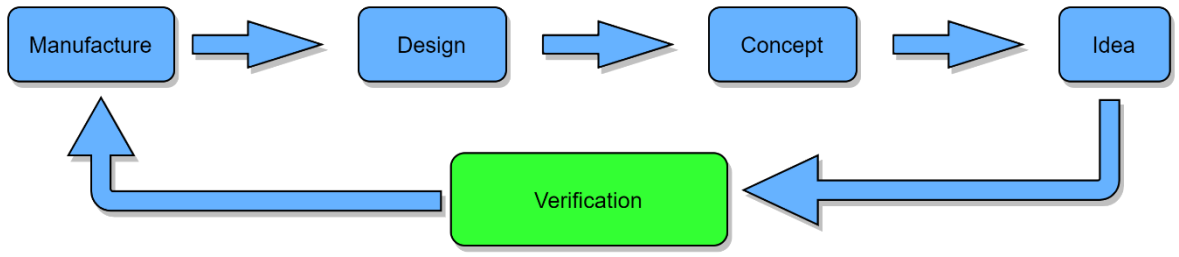


Figure 1.2: The reverse engineering theory where the idea obtained should be verified against the manufactured part

Although RE has been in existence for a long time, surface reconstruction from a point cloud of measurement points has attracted growing interest over the last two decades due to advances and the greater availability of scanning devices. These devices can generate hundreds of thousands of point samples reflected from a geometric object’s surface. With the massive amount of data that is currently obtainable, surface reconstruction has gained more research attention, investigating the mechanism and challenges in using scattered sampled points for performing surface reconstruction. Also, concentrating on the compensation existing for optical systems between data accuracy, processing speed and scan depth. SR is used in applications such as computer visualisation of physical parts, development of medical imaging, reverse engineering for product remodelling, computational science, object detection, and virtual, augmented, and mixed reality.

Surface reconstruction is used where the original design data of a machined part or physical object is missing, does not exist or has become inaccessible by either the initial designer or no backup was made. This concept involves a retrieval process of capturing measurement data or geometric information of an existing object using a process of data acquisition that

are generally classified as contact (tactile) and non-contact (optical) techniques [5, 6]. Point cloud data is obtained through these acquisition techniques as a 3D representation of the object before a computer-aided design (CAD) model is developed. Computer analysis, geometric feature extraction, measurement information, and part functionality are deduced from the acquired data. The method of retrieving an object's physical image is vital as several variables can influence the data integrity, including measurement error, lost data, and any flaws with the soft computing scanning tool for data capturing [7, 8]. In some manufacturing processes, the reconstruction of a physical model to produce a prototype is often faster than creating a new one. Reconstruction reduces time and cost by shortening the design cycle with improved accuracy [9] and provides an enhanced view of the system's capabilities [10, 11]. Once a CAD model has been generated, it can be processed using computer-aided manufacturing (CAM) module to produce numerical controlled (NC) codes that can be run on CNC manufacturing machines [12].

Detecting the amount of damage on a part for which a CAD drawing exists is relatively straightforward since the cloud point data can be aligned to the nominal drawing, and deviations can be measured. However, in RE, there is no nominal drawing, so the problem is much harder. In this situation, it is good practice to estimate the nominal points or boundary of the object using the captured data. Gao et al. [13] used a method known as cross-section curve on a model (straight blade) and extending the curve to a nominal height along the Z-axis aided in bending or stretching the curve. For a curved model, they used cross-section curves at different heights. The same authors used a similar approach in [14] but used the surface model's parametric and geometric continuity. Wilson et al. [15] used a semi-automated geometric reconstruction and a laser direct deposition process on defective voids in turbine airfoils. Using a Boolean discrepancy between the original faulty and final reconstructed models, they created a geometric parameterisation representation of the restored volume. A multi-probe measurement system was developed by Zexiao et al. [16] using a CMM, a structured-light sensor, a trigger probe and a rotary table. For this system, the profile of the part was scanned from different views using the structured-light sensor, and for measuring the edge and vital features, the trigger probe was used. They were able to obtain the coordinate points of a complicated part. A similar method was that of Li et al. [6, 17, 18], where a multi-sensor system was created. They described an effective approach of

compensating data from a laser scanner with a tactile probe for accurate RE, and the integration of a contact-optical coordinate measuring system was proposed. A five-step method for performing automatic RE was developed by [19]; they include mesh simplification, quadrilateral mesh generation, curve net construction, connectivity data preparation and several surface fittings bearing tangential continuity across boundaries.

The significant challenge for most proposed methods is producing a watertight surface when captured points are polygonised. The scanning method can produce missing or incomplete data, and the surface points of the cavities or concave regions can be easily missed. The projector and sensors can neither project nor see the light when the concave area is less than the angle created by the triangulation of the captured points. Invisible sections of the scanned part tend to appear as holes, and hole-filling is necessary for filling these holes and creating the correct surface continuity required for accurate reconstruction. Much research on RE concentrated on various aspects relating to the process of extracting information from a known object. They include scanning methods (looking at the advantages and frailty of scanning systems), image processing and computer vision, multi-probing methods, integrating the process with rapid prototyping, scan path planning and development of machining path, surface fitting and data point preprocessing [5].

1.2 Aims, Objectives, Scope

1.2.1 Aims

This study aims to develop a reverse engineering framework for reconstructing a damaged surface such that a copy of the original part can be produced.

This study also aims to develop an algorithm to automatically process measurement data scanned from a potentially damaged object such that surface damage can be detected and localised.

1.2.2 Objectives

- To develop a framework that uses the concept of reverse engineering in performing a reconstruction of a damaged surface.
- To identify alternative methods in achieving the stages presented in the reverse engineering framework.
- To develop an algorithm suitable for preprocessing unorganised measurement data.

- To develop an algorithm that produces micrometre-level data for representing the properties of a model's surface.
- To develop a machine learning algorithm that can learn from the micrometre-level data and identify locations of damage on a model.

1.3 Motivation

The 3D modelling of measurement data encounters uncertainties such as the integrity of the data in terms of capturing parameters, data transfer, measurement systems, and associated errors. These uncertainties are important factors for consideration when modelling an object to obtain its digital representation for analysis and understanding its functionality inferred from its geometric features. With these challenges, there is the need to develop an "all-in-one" system capable of measuring a potentially damaged object by detecting damaged areas, automatically reconstructing a model, and transfer to a format usable by a manufacturing machine. This can be achieved by developing a structured framework with a critical investigation of its procedures in performing experiments and preprocessing measurement data. Inferring design rationale of objects to understand functionality, design intent and representation of features to reduce the time taken to reconstruct a potentially damaged surface. The reconstructed model should be a correct representation of the measured object. This project aims to develop a structured process for achieving the overall ambition of developing an "all-in-one" system for reconstructing damaged surfaces, concentrating on damage detection. The investigation of detecting and localising damage is an important element of the SR/RE framework because identifying the damage location and its properties can aid in distinguishing potential damage and an intended design.

1.4 Contributions

Contributions were made with respect to damage detection and localisation at the microns level as a step to implementing the proposed surface reconstruction process. This process was based on a developed reverse engineering framework for performing reverse engineering/surface reconstruction on a damaged engineering component by identifying the necessary steps from data capturing, identifying alternative methods in each step, detecting

damaged locations, and verification of reconstructed models, concentrating on detecting damaged locations.

A model visualisation method of inspecting the profile of a model using rotation matrix was developed. Profiling the boundary of a model was analysed using a new approach different from the trivial method of using fitting techniques. This was achieved using a rotational matrix to produce a boundary examination of a model's surface as it circumvolves about an axis.

Finally, a method of sectioning a model to produce slices of micrometre resolution to extract dimensional information was developed. Unwrapping the raw data of each slice and automatically classifying them into specified classes. The unwrapped raw data produce numeric sequence data at the microns level after automatically classifying them into specified classes, and the data is used for training an LSTM neural network. The LSTM data can be used to perform prediction by detecting and localising damage on the Z-axis of a model.

1.5 Thesis Structure

This section presents an overview of the thesis structure as follows:

Chapter 2: Literature review, the concept of reverse engineering and surface reconstruction

Chapter 2 presents the theoretical background and an in-depth review of literature on reverse engineering algorithms and its application in the current work. It further presents a review of existing literature on surface reconstruction and some commercial applications of the SR algorithms presented. Building from the literature on surface reconstruction, the literature review was narrowed down to a critical appraisal of damage detection algorithms using machine learning (ML). This chapter also discussed the processes involved in dealing with measurement data and associated uncertainties during data acquisition concentrating on capturing methods and instruments. In this current work, the Articulated Arm CMM was used as the main data capturing instrument while dimension verification and analysis such as circularity, diameter, height, and width properties of a cylindrical, spherical and cuboid surfaces were performed using a CMM as indicated in Appendix A Circularity Test for Both the Arm Calibration Sphere and a Cylindrical Part used in Chapter 5 section 5.2.

Furthermore, this chapter also discussed the analysis of surface types, structures, characteristics, and application to SR. An overview to computational geometry, defect detection and handling noisy data. Using these ideas as a basis and lowering the amount of human engagement, machine learning techniques were utilised to locate and measure the damage that had been observed. This study utilised neural network systems such as the convolutional neural network (CNN) and the sequence-to-sequence system of long-short term memory (LSTM).

Chapter 3: The methodology for performing reconstruction using reverse engineering

Chapter 3 presents a detailed description of the methodology for this project and the proposed reverse engineering/surface reconstruction framework used to achieve the project aims and objectives. This includes data capturing, preprocessing, dimensioning, feature identification, damage detection and dimensioning, performing reconstruction, and validation of the newly developed model. The methodology was developed in a 'mind mapping' application with a breakdown and description of the various tasks involved in achieving the aims of this project. Most of the work was focused on identifying features within a model and detecting possible damage that might exist on the surfaces of the identified features. However, the data capturing and preprocessing were based on existing methods. The preprocessing of some of the data was challenging, leading to the development of new preprocessing methods for the extraction of valid information from the data. This was mainly experienced during the CAD and mathematical software communication. The feature identification process investigated various ways for identifying features in both three dimensional and two-dimensional applications. A new damage detection method was developed, exploring its capability in identifying damage in terms of accuracy, repeatability and reduced human influence.

Chapter 4: Edge detection algorithms for image processing and system optimisation

Chapter 4 discusses the concept of edge detection for detecting potential damage on the surface of a model. Several edge detection algorithms were investigated. Following experiments performed using various edge detection algorithms from the literature [20], the Canny edge algorithm was found to be a more suitable detection algorithm for use in the current work because of its low sensitivity to noise and produces single response point, which

was verified by performing experiments with various algorithms. An essential step in Canny edge detection algorithm processing is the thresholding for determining what edge points are real edges and non-edges. A method known as hysteresis thresholding was investigated in dealing with such noise in an image. System optimisation in the form of simplex optimisation was also investigated to understand how data distribution behaves in the presence of noise. This simplex optimisation process was used as an analysis step in extracting vital information used as input for the machine learning application in the detection of damage.

Chapter 5: Automatic feature recognition

This chapter provides a review of the concept used for recognising geometric features known as Automatic Feature Recognition (AFR). It classifies several approaches for AFR into logic rules and those related to artificial neural networks with brief descriptions. Further investigation into the process of AFR taking into account both techniques, one with the concept of interpolation and the other with sample consensus considering the distribution of points within a specified region of interest.

Chapter 6: Feature profile extraction for damage detection

Chapter 6 investigated the concept of profile monitoring using a rotational matrix for extracting the boundary contours of a part. Although the rotational matrix is not a new concept, its application for damage detection has not been investigated The detection process was by visualisation, which involved a reasonable level of human interaction with prior knowledge of the model, resulting in using machine learning algorithms to reduce human mitigation. The machine learning approach brought about processing methods such as wavelet transform and model sectioning (slicing) to investigate points in the global reference coordinate system within the data compared to images—furthermore, validation of the proposed methods using measurement data from AACMM.

Chapter 7: Validation of the proposed method using measurement data

This chapter describes the validation of the proposed method by applying the steps in chapter six using measurement data with a description of validation steps and discussion of results. There was little control of some parameters with measurement data, such as slice resolution,

but parameters like the height and radius could not be altered, and it is expected that the extracted information should compare well to a standard CMM measurement.

Chapter 8: Conclusions and future work

Finally, this chapter will present a summary of all the concepts discussed in this thesis, a general conclusion of the work and the proposed methods as a contribution to knowledge. It also stated the current limitations of the work and suggested further research areas to improve the method.

Chapter 2 Literature Review

2.1 Surface Reconstruction

There has been an accelerated rate of industrial interest in modern technologies dealing with how 3D point clouds are captured, processed, and represented, leading to the advancement of 3D visualisation and analysis. This has led to numerous research work on several ways of reconstructing a surface from 3D points. However, the reconstruction process is accompanied by challenges capable of affecting the desired result. In reconstruction using point cloud data, there are other concepts related to reconstructing actual elements from the point cloud of static objects and scenes obtained using 3D scanners. These include urban reconstruction [21], reconstruction of statutory objects, which could be structural or sculptural and interpolatory reconstruction. However, as explained in Chapter 1, this thesis is concerned with the surface reconstruction of machined parts from scanned data. The choice of machined parts is dependent on the gap of micrometre-level RE from literature where concentration is on surface texture and structure, with particular interest in the B-spline nature of a model. The developed RE framework is more industrial inclined, focusing on engineering components than application on statues, structural, and urban reconstruction due to its high tolerance application.

For reconstruction approaches where point cloud is used as measurement data, their properties are essential factors in comprehending the nature and behaviour of reconstruction methods. These properties range from the sampling density of points, the noise and outliers that have the ability of affecting the integrity of data, and misalignment and missing data can make reconstruction challenging to execute. The basic forms of input point cloud are the surface normal which is further categorised into oriented and unoriented normal, scanner information and RGB imagery. Surface normal are uniquely defined at every point and it indicates the perpendicular direction of each point to the tangent space. Oriented surface normal is consistently pointing inwardly or outwardly of the surface, while unoriented normal possess no direction [22]. Hence, obtaining correct connectivity among captured points is a big step in overcoming the difficulty faced by the reconstruction process as laser scanners produce unorganised data set.

For the representation of surfaces, surfaces can be classified into two categories of explicit and implicit surfaces. The exact positioning of a surface is represented by explicit surfaces. Explicit surfaces can handle huge data set and produces reconstructed surface that are piecewise linear as well as noisy and unorganised data, but obvious deformations and change in geometric shapes are difficult to deal with. Examples of explicit surfaces are parametric surfaces under which B-spline and NURBS are classified, and triangulated surfaces under which Voronoi diagram and Delaunay triangulation are classified. Implicit surfaces are also known as volumetric representation of a model. This is mostly in 2.5D and a slight variation to the traditional approach of working with 3D volumetric shape where several primitives are fitted together in representing the shape of a complex 3D model. Examples of implicit surfaces include least squares, Poisson surface reconstruction, partial differential equation (PDE) and level set method.

Surface reconstruction process requires soft computing methods to deal with optimisation problems and improve the modelling of reconstruction systems. These soft computing methods include neural network (NN), simulated annealing, particle swarm optimisation (PSO), and differential evolution [23]. There has been an increased application of NN in modelling reconstruction algorithms because it is able to represent the network structure of the data points and at the same time deal with unorganised density data. Neural network also produces better results due to it iterative training steps.

From literature [23-25], surface reconstruction algorithms can be classified into triangular mesh and parametric surface algorithms, which are further divided into other sub-surface algorithms. For B-Spline and NURBS triangular mesh algorithms, smooth enough surfaces can be produced to meet machining requirements [26]; however, most algorithms take a long time even if they produce dense data with less tolerance [11, 27]. Parameterisation (surface approximation) of the measured points is implemented first before applying several fitting strategies, thereby producing surfaces based totally on minimisation conditions [24]. The triangular mesh algorithms can transform measured points into a mesh surface but not without problems influenced by noise when performing reverse engineering. These surfaces are broadly used due to their widespread industrial standards for representing surfaces [19]. This is slightly different for the parametric surface, representing reconstructed models more efficiently and precisely [24].

2.2 Surface Reconstruction Algorithms

Most reconstruction algorithms generate a piecewise linear approximation of the curves and surfaces being sampled. Talking about approximation, we mean that the output directly represents the geometry and topology of the shape. The approximation of the sample point is only possible when the points are dense enough to represent the features of the object being sampled. Many algorithms for reconstruction use the data structures known as Voronoi diagrams and their dual known as Delaunay triangulations. A Voronoi diagram is a versatile geometric structure produced when a surface S is partitioned into regions, while Delaunay triangulation of a set of points P is the planar subdivision whose polygon surfaces are triangles and vertices are points contained in P [28].

Amenta and Bern [29] examined the reconstruction problem of producing triangular mesh whose vertices equal the number of points representing a surface. They used a combinatorial algorithm of both the Delaunay triangulation and Voronoi diagram. The Voronoi vertices were used in extracting triangles in the Delaunay triangulation using only a subset of the vertices. This produced a piecewise linear surface pointwise that converges to a smooth two-dimensional manifold after applying postprocessing steps of filtration by normal and trimming. If the sample is dense enough, this combinatorial approach can be used on non-uniform sampled points. Amenta et al. [30] improved on the algorithm of [29] by proving it is homeomorphic to a model's surface if a set of triangles T spanning all sample points satisfies three conditions of smoothness, compact, and sufficiently dense.

Eck and Hoppe [31] developed a method for reconstructing a tensor B-spline surface from a series of scanned 3D data points. They aimed to reconstruct an arbitrary topological surface from an unorganised set of points. The surface was defined as a network of B-spline patches before tackling the problem of reconstructing both the network of patches and the parameterisation of points found in those patches. When unorganised points were used as input data, an original surface approximation of dense triangular mesh was constructed before applying a re-parametrizing procedure to achieve a simple triangular base complex and quadrilateral domain complex. They used harmonic maps in reducing the effect of distortion in the corresponding reparameterisation, hence, optimising the system. Surface spline was designed in such a way that the device would maintain tangential surface continuity between patches by translating them into a format suitable to B-spline surfaces.

This new format is a network of quadrilateral regions that can be accomplished by creating a constrained surface continuity over the boundary. Ying and Hong [32] developed a model known as a triangular B-spline to address this problem. The developed model created a single triangular B-spline with variation continuity applicable on smooth regions and positional continuity on sharp features. An algorithm that can automatically incorporate these points into the network of triangular B-spline was proposed. This proposed algorithm eliminated the need for trimming and patching so that parametric domains having arbitrary topology can be handled quickly. Although this method is an improvement of the method by Eck and Hoppe [31], it is not supported by most modelling systems as the structure and topology of the simplified mesh may influence the surface reconstruction [19].

The process of achieving a smooth continuity between detached surfaces can be difficult for some systems. Ma & Zhao [33] proposed a technique using Catmull-Clark surfaces in place of a B-spline to simultaneously fit several surfaces from a point cloud that are smoothly linked and have arbitrary topology. The Catmull-Clark techniques used a linear least-square fitting approach to achieve a network of smoothly linked bi-cubic B-spline surfaces. Catmull-Clark surfaces with curvature continuity for surfaces with exceptional corner indicates a tangential continuity was achieved as it is difficult to achieve curvature continuity using this technique.

A two-approach system for remodelling triangular mesh into quadrilateral mesh was presented by Tsai, et al. [19] by firstly splitting each triangle into three quadrangles by linking its centre to the mid-point of its three edges, and secondly by merging two triangles into a quadrangle. In dealing with B-spline surface reconstruction from a large dataset having triangular meshes, a five-step integrated procedure was developed, and this includes: mesh simplification, quadrilateral mesh generation, curve net construction, connectivity data preparation, and constrained surface fitting. The implementation of this procedure was accompanied by challenges where the curve net generation and fitting of several surfaces having constrained boundary continuity resulted in situations of the meshes being inversed, overlayed, and distortion of boundaries. During the mesh simplification process, the model's boundary must be maintained to keep its original structure, as the possibility of achieving boundary continuity between patches will be affected. This led to developing a condition checklist to monitor the procedure to avoid losing the model’s boundary. Lau, et al. [34] proposed an algorithm for automatically generating adaptive quadrilateral mesh. The

combination of two triangles at a time will turn a triangular mesh into quadrilaterals using a carefully regulated operation. Two approaches for generating quadrilateral meshes was stated by Lau, et al. [34], the direct and the indirect schemes. The most traditional method is the direct scheme that uses mapping techniques to create physical domain components by mapping normal or quadrilateral meshes almost consistent to the physical domain in a parametric coordinate plane. Quadrilateral elements are created in the indirect scheme by merging and splitting the elements in a triangular mesh into quadrilateral meshes. The adaptive method of generating quadrilateral mesh is an indirect scheme with a structural merging technique. This technique can be divided into three stages: preprocessing a triangular mesh, merging triangles, and enhancing mesh quality. By splitting each triangle into three quadrangles using a sub-divisional technique, Kobbelt [35] and Borouchaki and Frey [36] were able to convert triangular meshes into quadrangles.

Hoppe, et al. [37] proposed an algorithm for surface reconstruction whose only input is unorganised points. They used the method of estimating tangent planes and tracing contours. This is similar to the concept presented by Boissonnat [38]. To obtain the estimation of the tangent plane, the principal component analysis and computing of an oriented tangent plane for every data point is used. An algorithm from dense samples was presented by Cheng, et al. [39], which was formed in a variant of the standard octree. They built a prototype using the algorithm to obtain locally compatible subsamples without any specialised structure of data. Dumitriu, et al. [40] used these subsamples in developing an algorithm needing fewer geometries but requires the distance between the sampled points. The extraction method for these samples was fast and efficient, and Cocone was used to raise its efficiency to about 68%. Amenta, et al. [30] used Cocone to simplify the Crust algorithm proposed by Amenta & Bern [18] by eliminating one Delaunay pass and a conventional trimming stage. Similar steps as the Crust case were employed where specific triangles were chosen using Cocone at each stage before extracting a manifold. Dey, Funke and Ramos [41] have demonstrated, using an approximate nearest neighbour searching data structure, that the Cocone algorithm can be altered for operation in $O(nlogn)$ time. The Crust algorithm, according to Amenta, et al. [30], is used for the reconstruction of surfaces that have both geometric and topological guarantees when Voronoi is used as input data. The Crust, an extended form of the Delaunay-

based system, replaces the Delaunay triangulation of points using a weighted Voronoi diagram known as the power diagram.

From usage, it has been observed that the Crust and Cocone algorithms create holes on the reconstructed surface due to a noisy dataset, non-smoothness, or inappropriate sampling, providing an incomplete representation of the surface. In offering a solution to this, Dey & Goswami [42] developed a simple algorithm to fill up all holes to create a watertight surface. In using this algorithm, no additional data points are required but used the first data and generated a three-dimensional surface that interpolates the input sampling points. The decimation algorithm presented by Schroeder, et al. [43] is similar to that of [42] with the goal of minimising the total number of triangles needed to remodel and secure the topology and geometry of an object using triangle meshes. It creates a hole in the mesh when a vertex satisfies the requisite decimation criteria and is removed. The filling of the holes forms a local triangulation in the mesh, and the elimination of the vertexes is an ongoing process until certain termination conditions are met.

A power crust was presented by Amenta, et al. [44] that approximates the Medial Axis Transform (MAT) of objects and then used an inverse transform from the MAT to produce surface representation. This algorithm gives a theoretical guarantee not relying on the quality of the input sample. This is similar to the algorithm by Boissonnat [38], where the method was presented to represent three-dimensional shapes with a set of points on their frontier using Delaunay triangulation. Gibes & Oudot [45] implemented an algorithm in which witness complexes supplement the Delaunay triangulation. The witnesses used inter-sample distances and can apply to any metric space. This algorithm demonstrated its good relationship with Delaunay triangulation in 2D and its sparse relationship in 3D. Despite witness complex being a weaker version of Delaunay triangulation, it has played a significant role in the context of topological data analysis described by De Silva and Carlsson [46]. It could replace it as the triangulation completely infers that a point cloud should always represent a single class of shapes. A simplification algorithm was suggested by Cohen, et al. [47] that would offset a mesh along its surface normal. This simplification can proceed to a mesh intersection. Garland & Heckbert [48] and Hoppe [49] used borders as the basis for simplification and suggested different rules for simplifying meshes.

For their industrial standards [19], most algorithms were developed based on tensor B-spline and NURBS surfaces. These algorithms had a primary objective of transforming triangulated points into a quadrilateral mesh with defined boundaries. The stability of most patches was retained by analysing the continuity of boundary conditions as patches are merged. Preprocessing [50], curve net construction, surface filling [51], post-blending and trimming are some of the processes involved in performing surface reconstruction [19]. They require interactive and iterative procedures for implementation, and they were a manual approach. This manual procedure can be avoided using macros by learning to avert the repetitive procedures. Macros record the reconstruction processes that can then be used on a different model, and identical procedures are performed on the new model. The use of macros will be an abducted method for this thesis when working with CAD software, and some algorithms proposed by literature are implementable in Geomagic Studio 12 CAD software. Very few approaches used raw measurement data for evaluation, but this thesis used both measurement and simulated data for evaluation of the processes used in this thesis. Simulated data are flexible to work with where parameters can be altered to investigate the performance of an algorithm. Once the algorithm performed satisfactorily, measurement data were used for cross-validation. Surface Imperfection and Defects on Machined Components

Surface imperfections are damage on the reference surface of a geometric surface, and it is important not to mistake surface imperfections for surface parameters such as surface roughness and waviness. According to ISO 8785:1999 [52], it is not recommended to state surface imperfections as a surface defects, and the definition of the term can be found in ISO 8402:1995 [53]. The reference surface, according to [52], “passes through the highest peak of the real surface excluding the imperfections, and is equidistant from the mean surface determined by the least-squares method”. Owing to imperfection on the actual surface, the specified surface is not inherently unacceptable, meaning that the surface can be fit for purpose and not easily discarded for such imperfections. Imperfection assessment and acceptability will rely on the surface application or function and will be defined accordingly, for example, length, depth, height, width, and number per unit area.

According to ISO 8402:1995, defects are defined as a nonfulfillment of purpose, usage requirement or acceptable quality, and the aspect of safety. Imperfections could be

unintentional due to uncertainties in manufacturing, but they could meet usage requirements and not necessarily be discarded. For objects such as machinery, propeller, aircraft, turbine blades and car parts, consist of components having freeform and machined surfaces that encounter several forms of forces. Due to continuous usage and product life cycle analysis, it is observed that these surfaces can get damaged or deformed from the original geometry. In situations where this falls below the acceptable tolerance level for a specific purpose, there is a risk to the system's operation in terms of optimum performance and hence, the need for detection and reconstruction.

For freeform surfaces such as a turbine blade, damage can be categorised as a damage that changes the whole dimension of the part and minor damage located on the surface that does not affect the functionality and are within tolerance. Machined surfaces take specific geometric shapes, and this could be intertwined, but their features are identifiable.

2.2.1 Specific Types of Surface Imperfections

There are four categories of surface imperfections as presented by ISO 8785:1999 in Table 2.1. These include recession, raising, combined, and area appearance. The recession surface imperfections can be described as an inwardly directed surface imperfection from the reference surface, and this is the sort of imperfections found on most manufactured surfaces. Raising surface imperfections are outwardly directed protrusion of the surface involved. The class of surface imperfection for this project is a recession type. The other classification is the combined surface imperfection which is a combination of partially inwardly and partially outwardly. The last classification is the area appearance that occur closest to the surface layer and usually do not have sharp contours with no practicable measurable depth or height. Figure 2.1 illustrates some of the different types of surface imperfection and a pictorial representation can be found in ISO 8785:1999.

Table 2.1: List of specific surface imperfections with their direction of appearance relative to the reference surface

Surface Imperfections			
Recession - inwardly directed surface imperfection	Raising – outwardly directed surface imperfection	Combined – partially inwardly and partially outwardly directed surface imperfection	Area appearance – scattered imperfections in the outermost surface layer
Groove	Wart	Crater	Skidding
Scratch	Blister	Lap	Erosion
Crack	Buckle (convex)	Scoring	Corrosion
Pore	Scale	Chip rest	Pitting
Blowhole	Inclusion		Crazing
Buckle (concave)	Burr		Spot patch
Dent	Flash		Discoloration
Wane	Deposits		Streak
Shrink crevice			Cleavage flaking
Shrinkage hole			

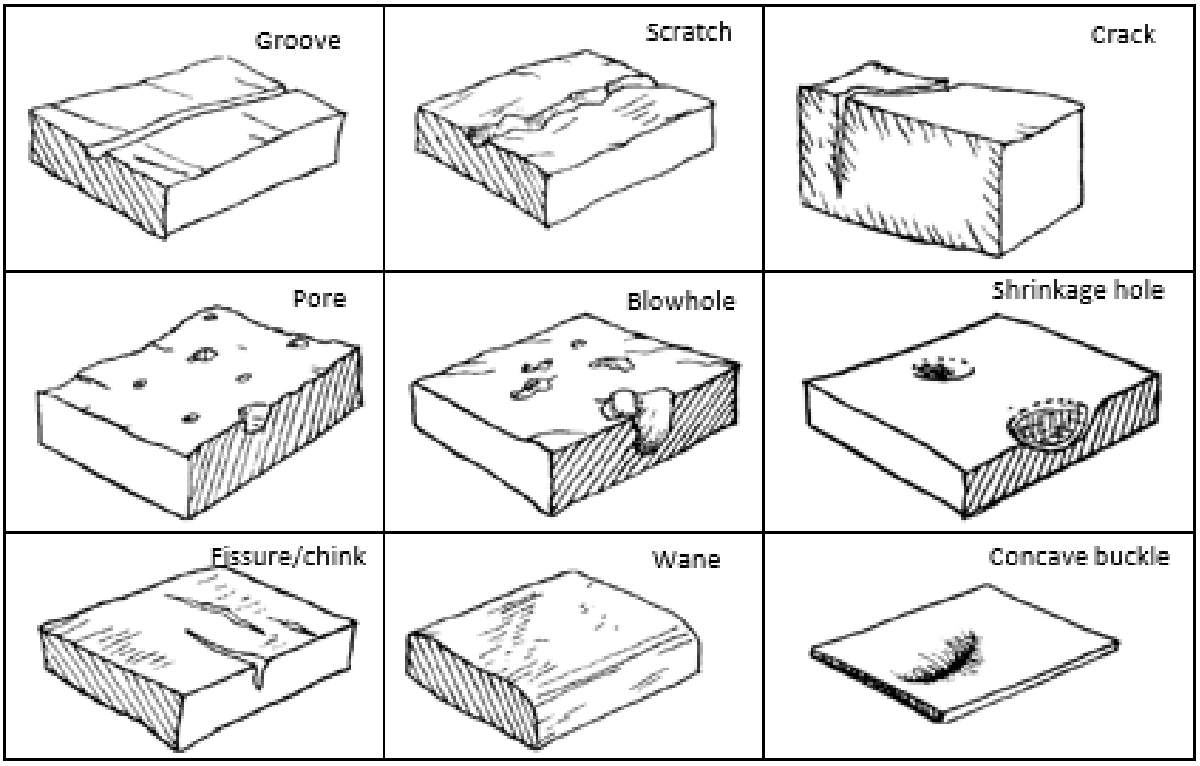


Figure 2.1: Different types of surface imperfection. Reproduced from [52]

2.2.2 Defect Identification

Defects are experienced in various sectors that deal with surfaces, and over time, there have been several applications for identifying defects. These surface imperfections affect appearance as well as reducing the properties of a material, such as corrosion and abrasive resistance and fatigue strength. Some imperfections can be identified by visual inspection of the part [54, 55], and for relatively fine and undetectable imperfections, optical methods are employed [56]. The captured data is processed using image processing techniques [54], pattern recognition [56], and artificial intelligence (AI) [57]. A typical example of the challenges of identifying defects is the Northeastern University (NEU), China [58] surface imperfection database of six typical surface imperfections of hot-rolled steel classified as intra-class and inter-class, as shown in Figure 2.2. In this application, two challenges encountered are the intra-class (same kind) defects producing significant discrepancies in appearance when the inter-class (different classes) defects produce identical aspects. The effect of light and the rate of material change is also another factor [59]. An improved steel surface inspection was developed by Song and Yan [59] using an adjacent evaluation window

to improve noise interference using the NEU database. They used annotations in specifying the class and location of defects in each image, and this is illustrated in Figure 2.3.

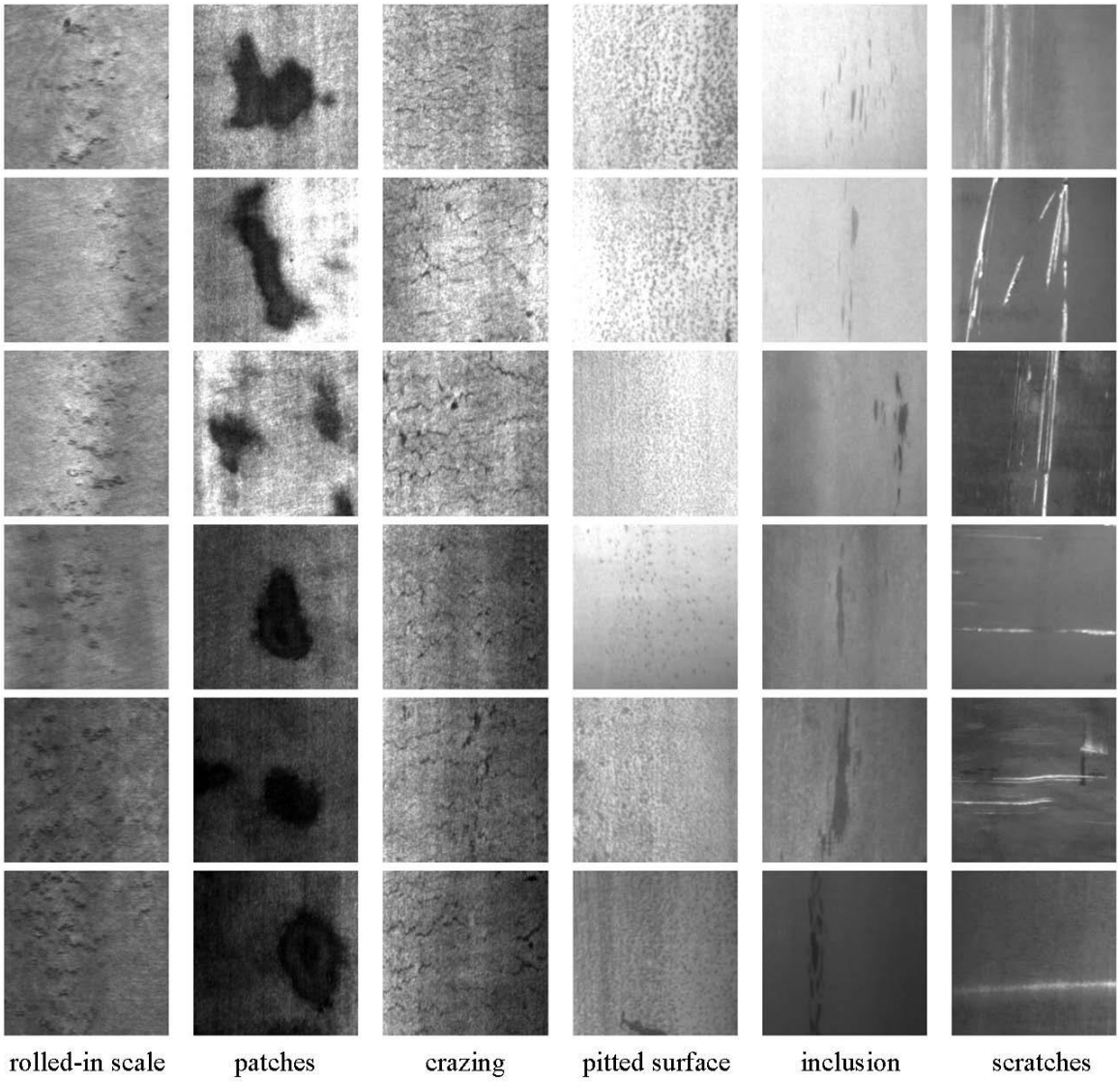


Figure 2.2: The Northeastern University (NEU) China, surface imperfection database, six kinds of typical surface imperfections of the hot-rolled steel strip. Reproduced from [58, 59]

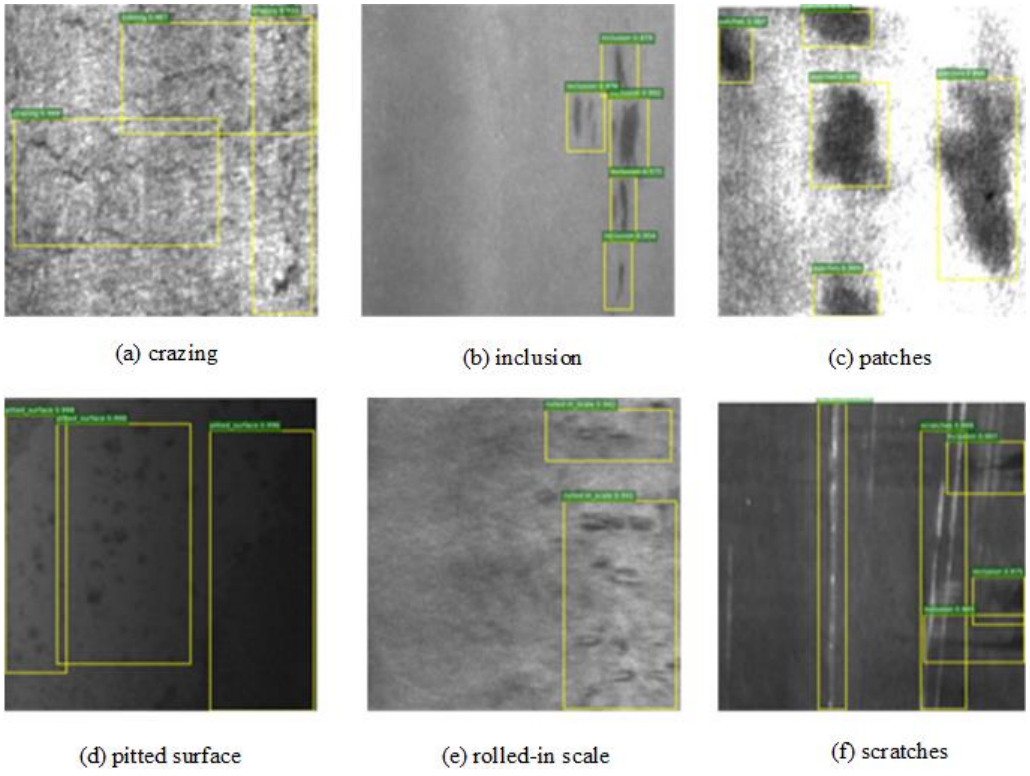


Figure 2.3: Defect annotation and location on the NEU surface imperfection database. Reproduced from [58]

A vision system technology inspection method was developed by Galan et al., [60] to detect defects on the surface of a metal component using a casting process. They first collected the binary representation of the bright and dark surface regions to produce a set of images. Related components of these images are processed in identifying shadows from defects. This is similar to the work by Rosati, et al. [61], where a curved mirror transfers light rays onto a surface that is reflected and captured by a charge-coupled device (CCD) camera. The detected defects appear as shadows having different geometries and dimensions after surface treatment due to the high reflectivity of the surface. A multi-angle illumination system for surface defect detection was proposed by Liu et al., [62] using several lighting sources and cameras for capturing several images. Satorres et al., [63] developed a machine vision image fusion technique by fusing images acquired from different lighting conditions. The same authors used a supervised machine learning classifier to classify images into defective and non-defective, determining true-positive and false-positive classification. The use of multiple lighting sources and cameras for capturing and the process of fusing images can be time-consuming. A one-shot detection method using one camera and a planar lighting source was

developed by Lin et al., [64]. Using texture orientation histogram, statistical information was developed for defect judgment compared to template matching of images. Although this approach improves the multiple light source approach by eliminating the bad direction of reflected light, it cannot be applied to edge collision damage.

A visual inspection system using image processing techniques was developed by Chen et al., [54]. Edge detection was used to segment potential defects from its background to obtain feature parameters used to develop a classification algorithm. They used a morphological approach similar to the work by Alaknanda et al., [65], incorporating an edge detection algorithm for analysing X-ray images. A segmentation of X-ray images for defect detection was proposed by Tang et al., [66] using a bound histogram. The fuzzy exponential entropy between the feature and its background is computed, and this information was used in creating an ideal threshold for segmentation. However, using a bound histogram to separate features from its background in X-ray images requires prior knowledge of the image. Also, X-ray images are challenging to work with due to noise, sometimes degraded by intermittent illumination, and can retain low contrast.

A systematic approach was developed by Huang et al., [67] for detecting and monitoring defects using measurement data of a three-dimensional curved surface acquired by optical means. This approach includes region division of curved surfaces after outliers are removed and boundaries are recognised. The region division was organised by dividing them into sub-regions with millions of measured points, comparable to a plane using wavelet packets for its decomposition, as shown in Figure 2.4. The wavelet packet entropy and normal vector were used to determine if the sub-regions are out-of-limit of their specifications and assess features of each sub-regions features. Three quality parameters were used in the monitoring of these parameters by calculating their values based on the clusters of out-of-limit sub-regions. The proposed approach was validated using a comparison of the profile monitoring using a Quantile-Quantile (Q-Q) plot for assessing the assumption of normality. When measurement data is acquired as distribution points, a highly linear Q-Q plot is obtained with in-control point cloud data, and out-of-control point cloud data will generate a Q-Q plot that deviates from linearity, and this is illustrated in Figure 2.5.

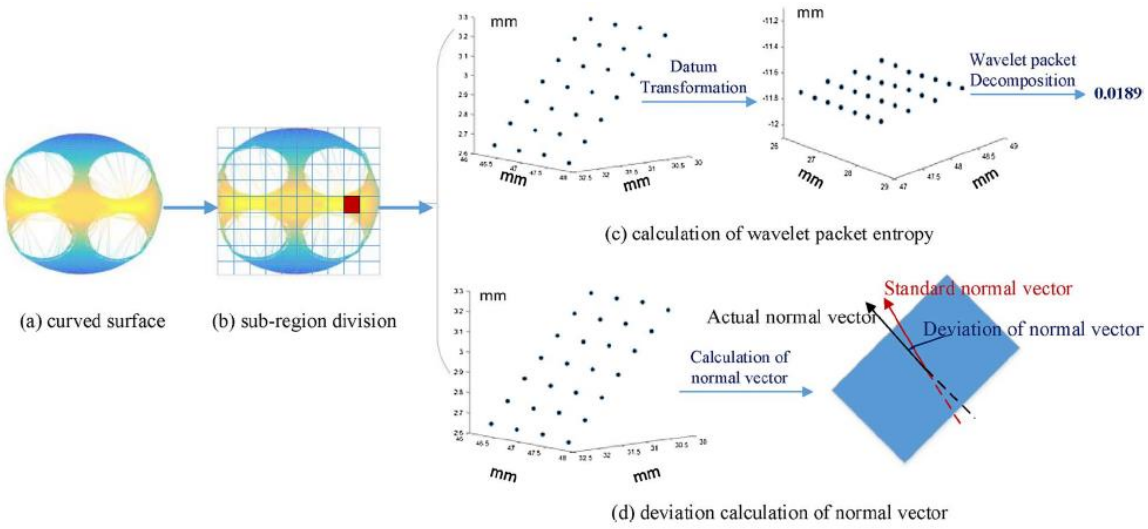


Figure 2.4: Evaluation of the defect detection by dividing the surface region into sub-regions and applying the wavelet entropy and normal vector. Reproduced from [67]

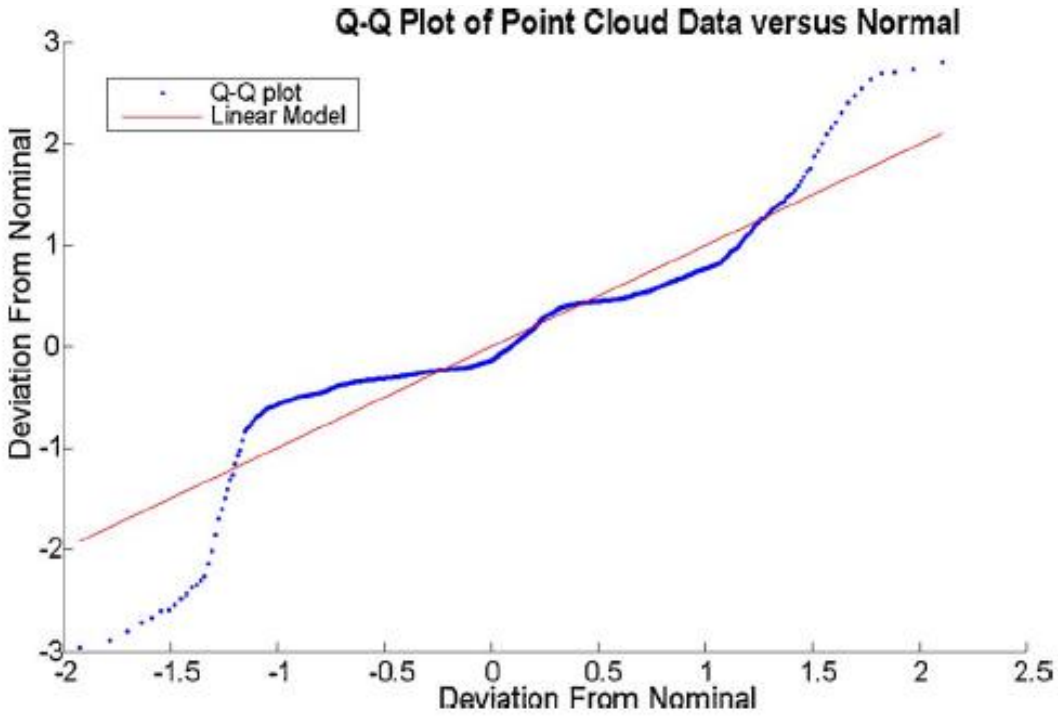


Figure 2.5: An illustration of an in-control Q-Q plot. Reproduced from [67]

2.3 Measurement Data Acquisition

An important part of surface reconstruction is the 3D data acquisition, as there is a direct link between how the data is acquired and the reconstruction algorithm [6, 8]. The acquisition

method determines the techniques and methods of surface reconstruction suitable for the generated data type and algorithm to reproduce the digital representation of the part [6]. This process involves preprocessing digitized points, curve net construction, surface fitting, post-blending, and trimming [19]. The capturing of high-resolution 3D range data of real objects is a lot easier, thanks to the recent advances in 3D scanner technology. 3D scanners have evolved over time with improvement in the number of points per second generated, response to different surface types, and precision to micrometre level [10]. Also, advanced programming dealing with measurement uncertainties from the environment and personnel, with reduced surface reflectivity effects, produces more accurate results [68].

Advances in the 3D representation of objects such as CAD/CAM applications and point cloud modelling have advanced ways of capturing 3D data for surface reconstruction applications. In applications like geometric modelling/processing, computer graphics/vision and medical imaging, capturing sampled points is an integral part of the process. The digital representation of real objects or artefacts is translated from the captured sampled points into models where visualization, analysis and manipulation of the data are performed to represent the original part. Information relating to the dimensional measurement of its geometric primitives are extracted for the redesigning or reverse engineering of artefacts [7, 8].

A challenging factor with surface reconstruction is the recovery of the digital representation of a part, usually a point cloud embedded with imperfections that can potentially affect the output result [8]. When the digital representation of an object is performed, attention is required during the recovery process as uncertainties can affect the integrity of the data. These include measurement errors due to the environment, missing data resulting from occlusion, faults with the scanning device, issues associated with the soft computing for data capturing, human error and preprocessing of the scanned data. These uncertainties can significantly influence the quality of the information obtainable from scanned data but are introduced with known parameters when generating data through computer simulation.

The acquired point cloud data can be either structured or unstructured. For the unstructured point set, only information relating to the coordinates of the points is obtainable, while the structured points give information relating to the coordinates of the points and their geometrical and topological information [69]. The unstructured data is also randomly

distributed compared to the organised distribution of structured data. In surface reconstruction, properties of the scanned surface are considered as well as generated points because this can influence the desired outcome and possible choice of a suitable algorithm.

The process of reconstructing a surface consists of several stages; the acquisition stage, registration, preprocessing, and the representation stage [70]. For data acquisition, surface points are generated using range scans from multiple angles then reconstruction is performed by applying a suitable algorithm. The reconstructed surface is converted into a polygon mesh called triangulation. Triangulation is when the points are interlinked to create a triangle where angles/vertices are shared. This triangulation creates a 3D model of the real part being visualised [69, 70]. The preprocessing stage is a process where noisy data and outliers are filtered out, input data is simplified, and data is segmented into regions (region growing) to represent the part. Other classifications for representing surfaces was suggested by Khatamain [69] using certain criteria such as surfaces containing actual elements of the data points, or estimation of their properties, or has sharp edges and corners. These criteria helped in classifying reconstructed surfaces into approximated versus interpolated and isotropic versus anisotropic surfaces. The interpolated surfaces are used in a situation of non-uniformity or sparse distribution of points which is a common uncertainty that affects scanning processes.

An illustration of the acquisition techniques is shown in Figure 2.6, indicating the various methods classified under the two main techniques of tactile and non-contact. The most varied method is the optical method with a wide range of applications. Both methods can be used as a composite system where an optical device can be attached to a CMM or robotic arm. 3D data acquisition devices include tactile Coordinate Measuring Machines (CMMs) and machines fitted with laser scanners [2]. Laser scanners can be blue [71], white [72], or red light scanner or light detection and ranging (LiDAR) [73].

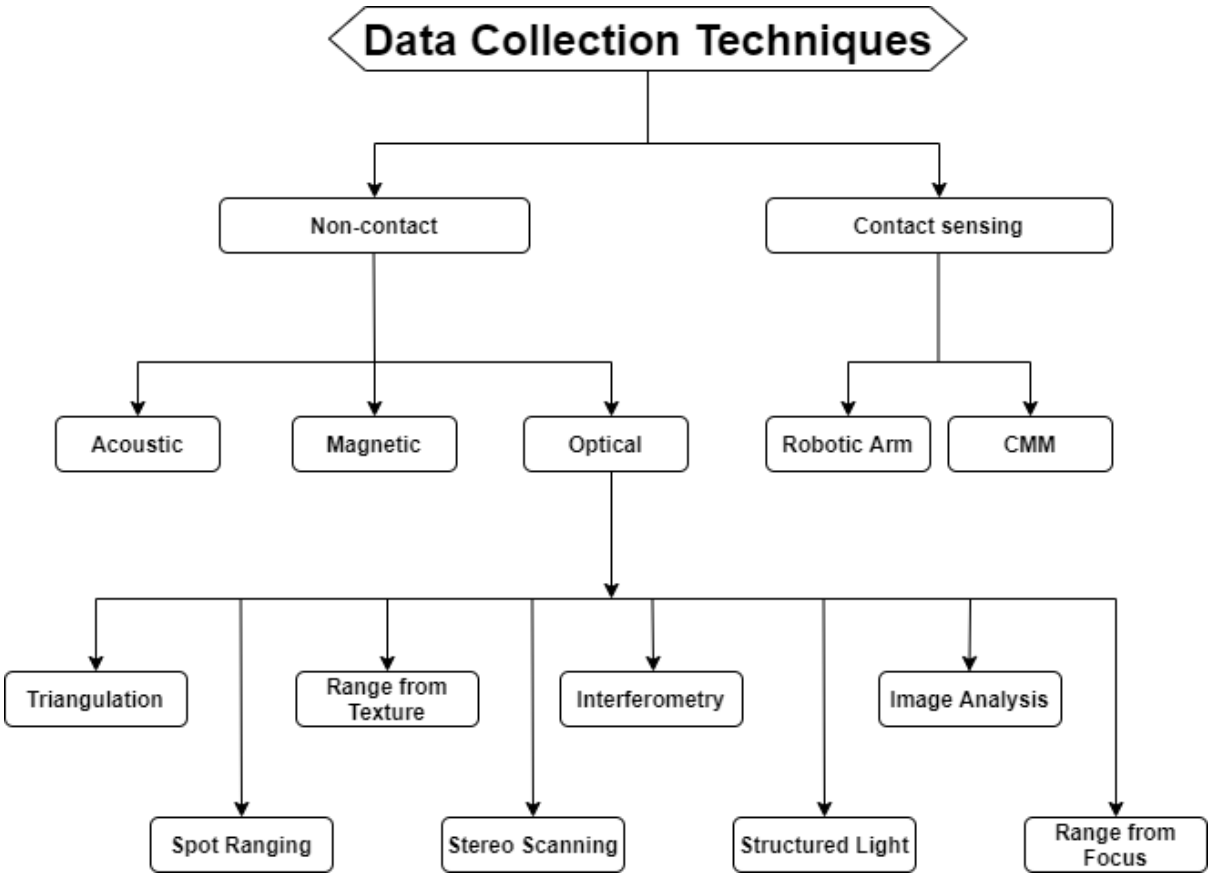


Figure 2.6: Data capturing techniques breaking down the several methods for both contact and non-contact techniques

White light scanners use white light to generate precise surface measurements. Its principle of operation is by projecting line shadows emanating through a 2D lens, and these lines are projected onto 3D surfaces. Then analysing the deviations in the 2D lines of the 3D objects using cameras. The blue light scanner is an improvement of the white light scanner in terms of how precise points are captured with improved accuracy, higher quality results, and demonstrates greater repeatability [71]. This is based on the narrower wavelength of blue light scanners, thereby allowing for enhanced filtering of interference from ambient light as compared to white light employing all visible spectrums at different wavelengths [74].

The tactile method operates on a different acquisition approach from non-contact, as contact with the surface is required when acquiring points. CMMs function by predefining a programme tool path in representing the part's features, and complex or freeform surfaces can be challenging. They produce low-density high accuracy data [11, 27]. The non-contact method captures points using laser lines at a predetermined distance by triangulating surface

points to represent the scanned model [9]. This distance affects the quality of the captured data with respect to the reflectivity of the surface and depth of capture.

2.3.1 3D Scanning

3D scanning involves using either touch-probes or 3D optical scanners. 3D optical scanners function identical to a camera having a conical visual field that captures data in its view. The scanner collects surface information and produces a 3D representation of the captured surface. The captured points indicate the distance of the surface from the scanner and the position of the points in the coordinate system. Dense data is often related to the non-contact approach to 3D scanning, while the contact approach can produce sparse or dense data after a very long time of rastering. The nature of the data depends on the step-over of the touch probe and other parameters during rastering, but the data is sparse compared to the millions of points from an optical system. Therefore, the optical system is a suitable approach for developing measurement data for this project.

2.3.1.1 Contact Scanning

Contact 3D scanning is a tactile method of using probes in contacting a surface either by touch probing or scan-probing. The probe could be attached to a CMM or AACMM, which provides the scanning mechanism, and this process is mostly used for tight tolerance measurements. The contact scanning process is mostly carried out on CMMs because they can be very precise compared to AACMM. Environmental factors can also affect the contact scanning process as they are very sensitive to environmental changes such as vibration, temperature, and part displacement. Although shopfloor CMMs exist, CMMs are installed in a measurement room to obtain better results, as shown in Figure 2.7.

Figure 2.7: Zeiss Coordinate Measuring Machine (CMM)Passive Non-contact Scanning

2.3.1.2 Passive Non-contacting scanning

Passive scanners function by detecting ambient radiation as they do not emit radiation. They detect visible light in radiations such as infrared, and they require no specialist equipment except for cameras and are inexpensive as they provide low accuracy measurements.

2.3.1.3 Active Non-contacting Scanning

Active scanners emit radiation in the form of light and detect this light as they pass through objects. The emitted light is reflected by the surface of the object, and the scanner captures information by recording the distance of the scanner from the surface, thereby producing points that are a representation of the surface.

2.3.1.3.1 Structured-Light Scanning (SLS)

Structured-light scanning (SLS) is the process of obtaining measurement information of an object for dimensional inspection and RE application. They are used in developing 3D representation, just like the laser scanning method. The recent advancements and innovative applications for capturing data have also contributed to an improved involvement of structured-light scanning for use in several industries. The system projects various lighting patterns or configurations (series of lines parallel to each other) and captures the distorted light patterns as it falls on the surface of the object using either sensors or cameras. Like laser scanning, this system uses trigonometric triangulation in projecting light patterns (that appears as a barcode) on a surface, and the camera(s), which is offset from the projector, studies the light pattern. Triangulation is used in calculating the distance to precise points on the object, and digital reconstruction of the object is done using three-dimensional coordinates.

The typical setup for performing SLS is shown in Figure 2.8, and this can be stationary or handheld. Two cameras are positioned at an angle obtuse to each other for an improved range of capturing, although they can be positioned at different fringe angles to the projector, as shown in Figure 2.9. When the setup is stationary, the object to be scanned is rotated either by literally turning the object or using a more recent turntable application. There has been a recent improvement on this setup where the projector and the camera(s) are incorporated into one device like the Artec scanners shown in Figure 2.10. In this case, the SLS operation is performed as a laser scanning operation with the object stationary, and the scanner can be handheld. This system can come in either blue light or white light and are susceptible to challenges faced by optical systems such as surface characteristics.

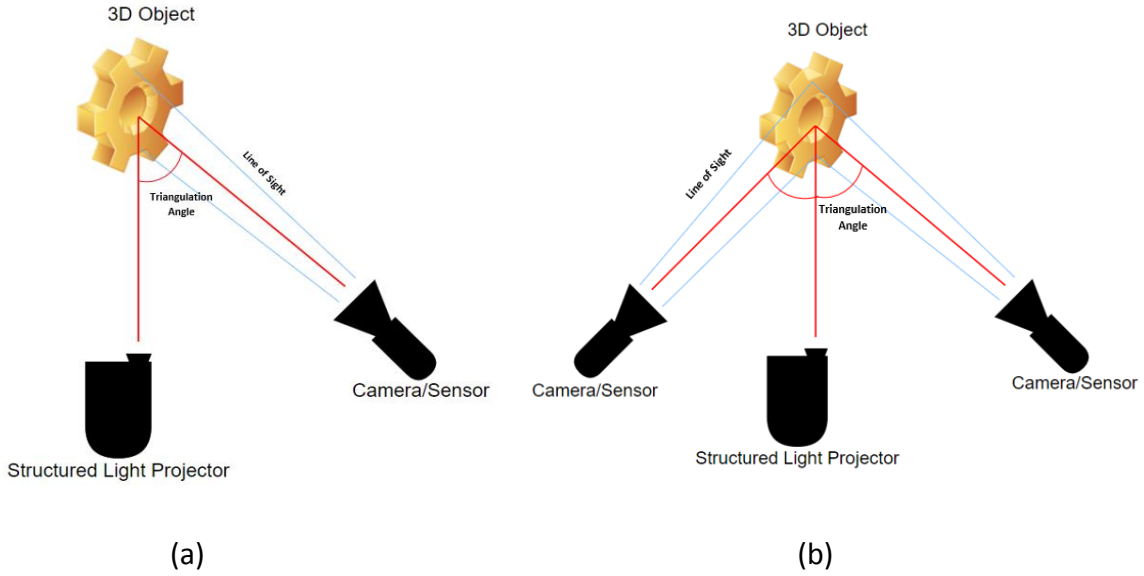


Figure 2.8: Typical setup of SLS (a) with one camera offset from the projector (b) two cameras positioned at an angle offset to the projector

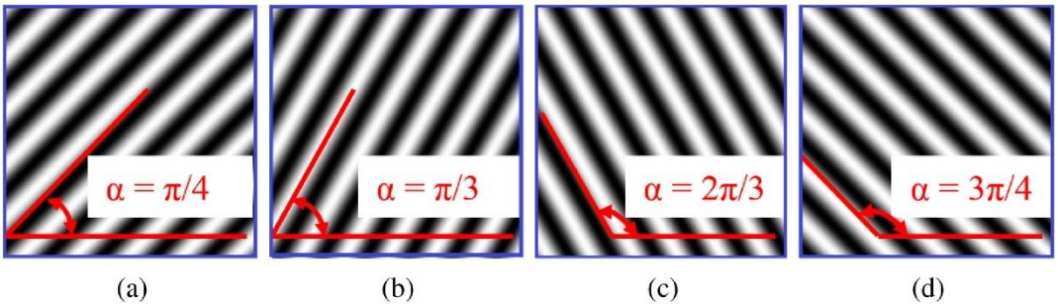


Figure 2.9: Patterns of different fringe angles between the projector and the camera. Reproduced from [34]

Figure 2.10: Artec spider and Eva scanning devices

2.3.1.3.2 Time-of-Flight

This type of 3D laser scanner uses a time-of-flight laser rangefinder to measure the distance of an object's surface by assessing the full circle of travel of a light pulse. When a light pulse emanates from the laser, the travelled time is computed before its reflected light is captured. The full circle of travel is when the light hits the surface and returns to the scanner, and the accuracy of the time-of-flight 3D scanner is reliant on how precise the travel time is measured. The working principle of the rangefinder is by identifying the distance of a single point in its range of view. A point at a time and different points are detected by adjusting the path of view of the rangefinder and measuring over 100,000 points per second. By turning the rangefinder or using a rotating mirror system, the direction of view of the laser rangefinder can be adjusted. This approach is often used because mirrors can be accurately rotated at a faster rate, and they are much lighter [75].

2.3.1.3.3 Triangulation

Triangulation 3D scanners project a laser on the target object, and the laser dot (flickering over the surface as it is not stable) is located using a camera. However, this is unstable depending on the distance of the laser to the surface, as shown in Figure 2.11. The laser dot, laser emitter and camera create a triangle to form a triangulation whose scale and shape is defined by, firstly, the distance a between the camera and laser emitter. Secondly, is the

definition of the laser emitter angle C_{ij} and thirdly, defining the camera angle B_{ij} using the laser dot position as in Figure 2.12. In some instances, the acquisition phase is enhanced by a laser line compared to a laser dot swept over the object's surface.

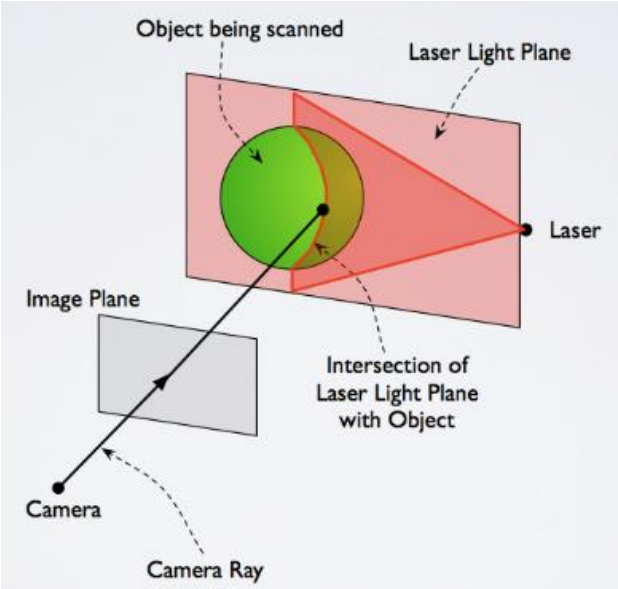


Figure 2.11: The basic structure of triangulation 3D scanning. Reproduced from [76]

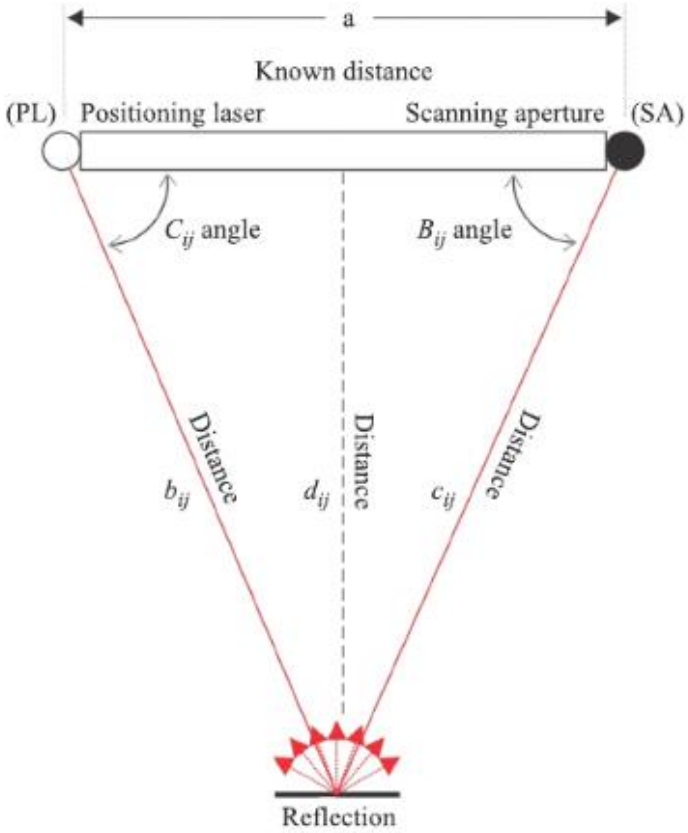


Figure 2.12: Dynamic triangulation of the triangulation 3D scanning process showing the different angles of the instruments. Reproduced from [75]

2.3.1.3.4 Modulated Light

3D scanning with a modulated light project a continuous shifting light onto the surface of an object creating a sinusoidal pattern. This pattern is reflected as it hits the surface, and a camera detects the pattern and computes the distance of travel of the light. The modulated light is utilised by the scanner in concentrating on the laser, avoiding other sources of light.

2.3.2 Articulated Arm Coordinate Measuring Machine (AACMM)

The Articulated Arm coordinate measurement machine (AACMM) is made up of multiple Articulated Arms with rotary encoders that are used to read the rotational angles or angular orientation of the Articulated Arms and calculate the coordinates of an object in three- dimensional space [11]. When the Arm is fully stretched, it can reach ranges of 0.5 m – 5 m depending on the model.

2.3.2.1 How AACMM Works

An important part of the AACMM is the device attached to the third joint away from the base of the Arm. This could either be a touch probe for performing tactile measurement or a laser scanner for performing optical scanning to capture dimensional information of a part. They are described in section 2.3.1. These devices have their specifications, properties, operation, performance evaluation and accuracy analysis. They cannot be used as a stand-alone device; hence, they must be attached to a CMM whose mechanism contributes to the performance and accuracy analysis of the probe and scanner. Therefore, most researchers have concentrated on the performance calibration [77, 78], error compensation [79] and accuracy improvement [80] of the AACMM. Measurement models were built with varying parameters and errors. A measurement accuracy evaluation was performed by Zheng et al. [81] analysing the 6 circular grating eccentricity error of the Arm. Mutilba et al. [77] investigated the performance calibration of the AACMM using the ASME B89.4.22 [78] standard in analysing the effective diameter test, single-point articulation test (SPAT) and volumetric test of the AACMM. They used the reference ball, trihedral seat, and Ballbar in performing uncertainty evaluation of the AACMM. During scanning, the system is little influenced by temperature but is important that experiment is performed in a temperature-controlled environment as this can affect the expansion coefficient of the artefact. Exposure affects the point cloud density, while missing data is significantly caused by depth of scan and an attempt to capture more points will result in overlaying data points.

As the lengths of each segment of the Arm are known, a scale is automatically created. A point can be located by using a known scale to measure angles. To determine the 3D position of the probe tip, angular positions and lengths of Arm segments are used. Different configurations of Articulated Arm CMMs can be classified according to their rotational axes at the joint assemblies. A 6-axis configuration is the most common, but there are other configurations. The diagram in Figure 2.13 below shows a 7-axis configuration that can be categorized as '2-2-3' because there are two rotation axes at the first joint, followed by two at the second, and followed by three at the third joint. Contact probes, non-contact probes, and multi-sensor probes may be available, depending on the configuration of the Arm.

When using a 3-D laser scanner, laser light is used for exploring objects. It employs a camera to look for the location of the laser line silhouette after projecting a laser line on an object. Points captured are points that fall in the camera's frame of vision on the laser line profile. These points appear at different positions depending on the distance of the laser as it strikes an object's surface. Because the points' location on the laser profile, the camera, and the laser emitter creates a triangle, this approach is termed triangulation [82].

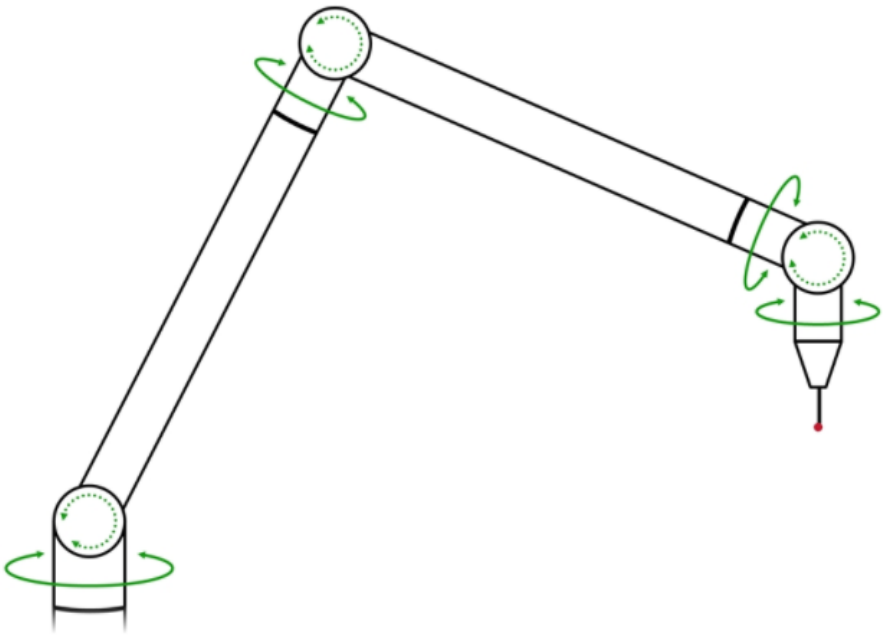


Figure 2.13: 7-axis configuration of an AACMM. Reproduced from National Physical Laboratory (NPL) e-learning

Articulated Arm CMMs were developed to perform a measurement in a convenient location, rather than having to bring the object to a measurement room. The portable Arm can be moved anywhere from within the machining floor to performing measurements off-site. Most commercial AACMMs have either 6 or 7 joints to provide 6 degrees of freedom movement. Its physical setup is like the human Arm consisting of the wrist, forearm, elbow, and shoulder.

2.3.2.2 Sources of Error

Articulated Arm CMMs can be accurate to within less than 100 µm, but only if there is a stable mechanical relationship between the Arm and the part, and if good practice is considered as discussed in the good practice section of 2.3.2.3. Of all the sources of error, the operator is the most common. Mechanical stress and angle measurement issues can have an impact on measurement quality. Inaccuracy is associated with extreme reach - when an Articulated Arm is fully extended or tightly curled [83].

Poor setup in the use of an AACMM will result in capturing of inaccurate data. A stable machining floor is an important factor, as are temperature, vibration, surface finish, ambient lighting and movement of the object [84]. A poor result can be obtained when artefact and scanner are vibrating at different frequencies or amplitude, and it is the best practice that both are positioned on the same platform, as shown in Figure 2.14. A stable base counters external vibration experienced during measurement. According to international standards, the ambient and artefact temperature should be close to 20°C [84]. Measurement using a laser line scanner requires that the object is kept at the same coordinate system as the scanner; when this is not the case, some capturing software can align separate scans, but this might introduce additional uncertainties. The Articulated Arm CMM used for this thesis has absolute encoders that require no referencing before measurement as it maintains positional information, but a calibration cycle is required when using touch probing. A typical setup of the experiment is shown in Figure 2.15.

Figure 2.14: Articulated Arm CMM (AACMM) with an attached laser scanner and a magnetic base for minimising vibration

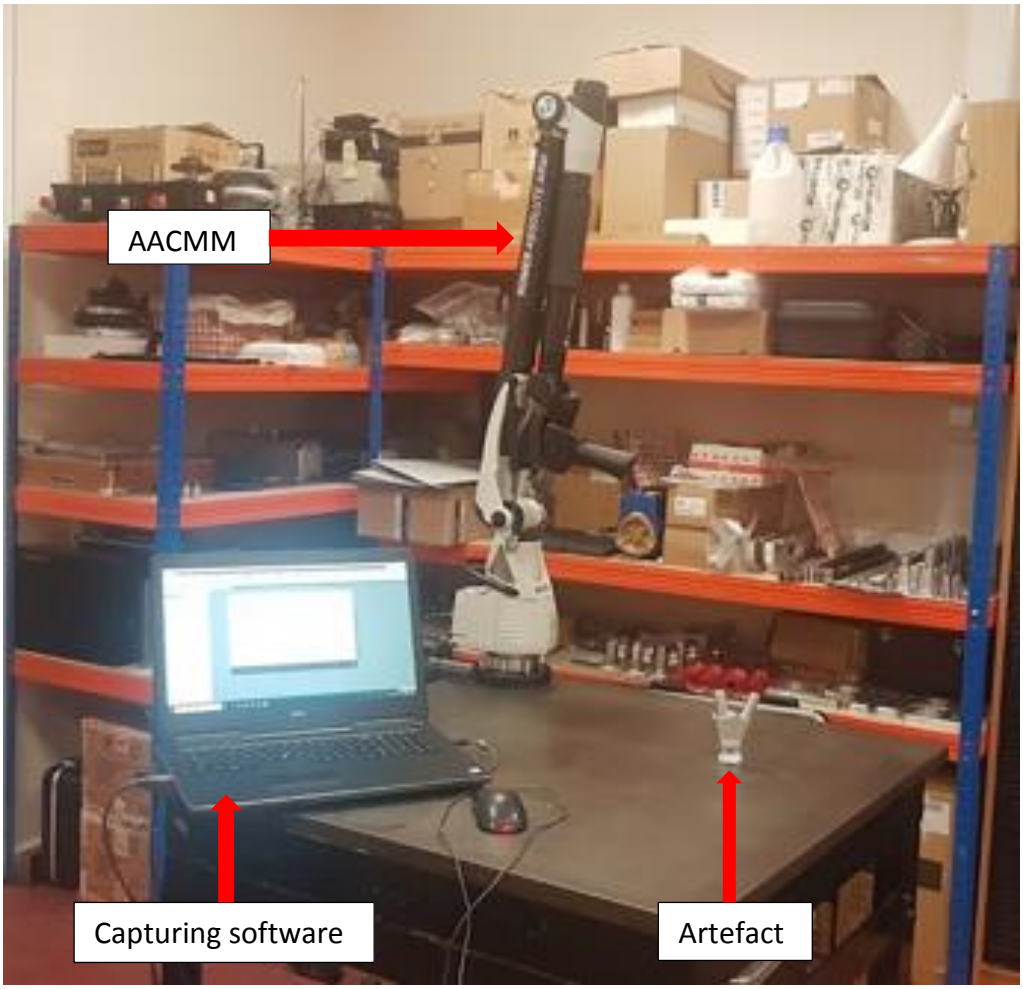


Figure 2.15: Typical setup of the Arm for simulating the same vibration in both AACMM and artefact

2.3.2.3 Good Practice for Performing Measurement with AACMM

When the joint is close to 90°, the encoders are most accurate as they have the greatest angle to distance moved ratio. To achieve this, the "elbow" of the AACMM is moved to 90° from the base to the vertical axis, as shown in Figure 2.16, thus leaving it parallel and right above the measuring surface to achieve maximum scanning accuracy. Therefore, for this 236 mm component, it should be centred at 624 mm from the base of the AACMM.

Figure 2.16: Picture illustrating best practice for aligning the artefact with the laser scanner. Modified from NPL [84]

The perfect position for placing the object to be inspected is a line projection directly downwards from the Arm's elbow to the measuring surface [84]. This position should allow the maximum range of Arm motion in connection with the inspected device. The elbow should also be kept close to 90° to maximise the resolution in the encoders, hence, it is important to adhere to a good practice procedure as stated below to minimise measurement uncertainties.

2.3.3 3D Laser Line Scanning and Accuracy of Data Captured

Laser line scanning is one of the active optical methods for capturing point cloud data. They can capture points from multiple features from a single scan having fine details of complex geometries. It can obtain information from surfaces unattainable by tactile methods. It functions like a structured light scanning (SLS), but it is more flexible in its structure and has high measurement speed. They rely on trigonometric triangulation to precisely capture an object and produces its 3D representation as millions of points, known as digitisation. They project a laser line on the surface of an object, and sensor cameras capture the reflection of points by perpetually recording the change in distance of the laser line to the surface. Since the distance of the sensor from the laser source is known, points measurement can be achieved by calculating the laser light reflection angle.

Laser line scanners function by scanning a surface at a predetermined distance, called the depth of scan, that determine the number of points generated by the width of the laser line. Laser line scanners produce data with high density but an accuracy lower than the contact method [11, 27, 85]. An indication of how they perform compared to a conventional CMM is shown in Figure 2.17. Laser line scanning is most desirable in measurements of complex freeform shapes and for large scale data capturing, although small components can be measured. In some situations, bringing a large component to a CMM might not be feasible, and optical systems have the improved flexibility of working in different environments but not without uncertainties. Laser scanning uses two sets of information to create point cloud data, one from the laser and the other from the sensor cameras. The collection of data is connected to an internal three-dimensional system at a predetermined distance, and as the scanner is swept over the surface, the scanner's position is tracked by encoders on each joint of the Articulated Arm CMM [86, 87].

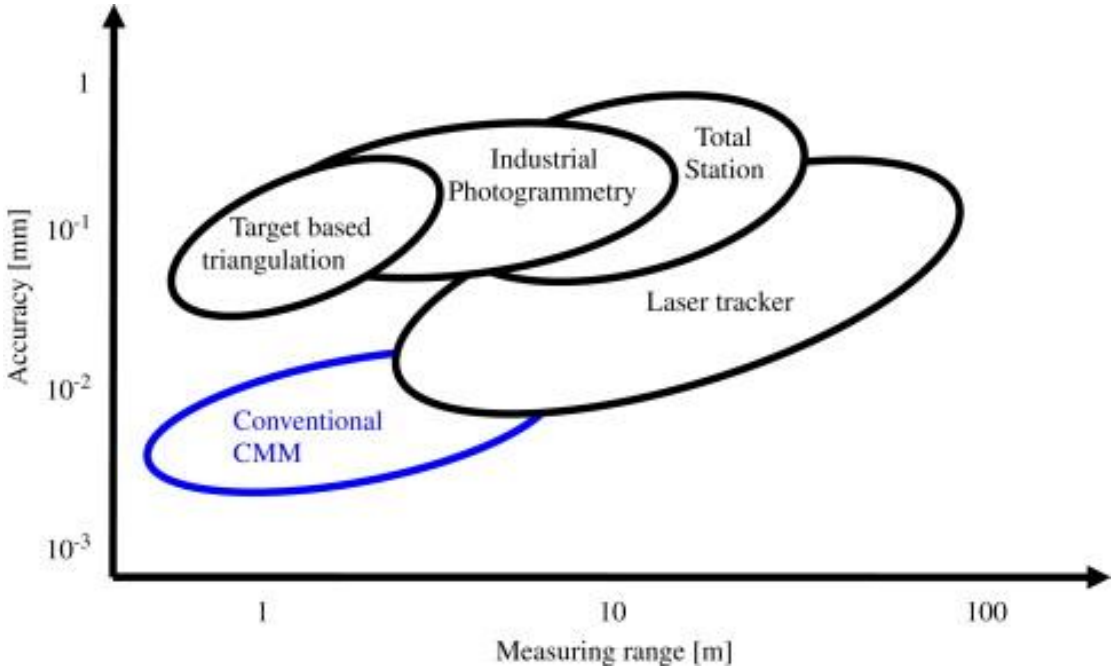


Figure 2.17: Optical CMMs in comparison with conventional CMMs measuring range and accuracy. Reproduced from [85]

2.3.4 Data with Noise

When working with point cloud data, noise is almost inevitable as it can be generated from various sources. This could be the temperature, vibration, surface finish, ambient lighting, artefact movement and inexperience of the personnel. Figure 2.18 shows a point cloud data with noise created from the surface on which it was positioned. This is not the only noise experienced as noise can be developed from outlier points due to reflectivity of the surface and ambient light and missing points due to personnel and occlusion. Figure 2.18 illustrates the noise in the data due to surface reflectivity and illumination, while Figure 2.28 in the summary section shows a situation of sparse or missing data.

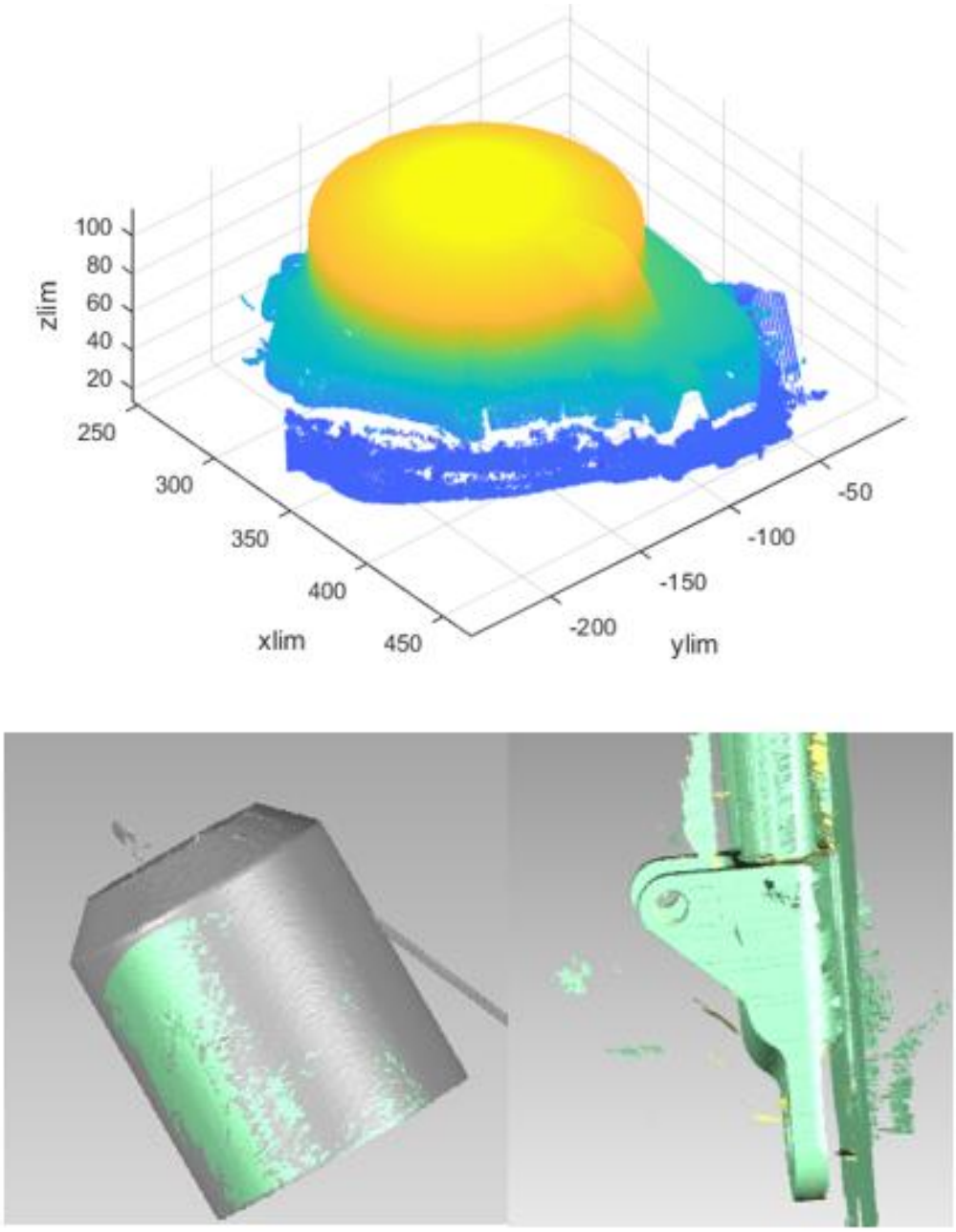


Figure 2.18: Showing how models are accompanied by noise

There are situations of overlapping data, and the preprocessing of such data is quite difficult as there is the risk of losing important surface information. In preprocessing, disconnected outlier points can easily be removed by setting a threshold or tolerance level using the distance of the outliers to the model. For connected outliers (connected to the model), it is an uphill task for cleaning such data as it requires a pain taking process of identifying at what locations these outliers are connected.

Furthermore, for overlapping data, the points' generation might be discontinuous where a set of points in a particular direction might be at a different plane to another set of points in an opposite direction. Since they do not meet, extra care should be taken when erasing supposed unwanted points as this might generate a horrid mesh with no surface continuity. When this happens, holes are created in the model and can be filled by hole-filling techniques, but the prediction of the surrounding curvature might be wrong. A preferred approach is by building bridges of proximity to the boundary, and a suitable hole filling technique can be applied depending on the surrounding curvature. Another contributor of noise to scanned data is capturing of the platform on which the part is positioned. To overcome this, a clipping plane can be created before performing scanning, and this deals with unwanted points being generated from the surface. Manufacturers of 3D laser scanners are continually improving the scanning experience of laser scanners. The number of points per second increases, which means more susceptibility to surface reflection and light, and newly developed scanners can generate over a million points per second with a more negligible effect of reflection. For a more reflective surface, capturing accurate data is quite challenging, coupled with the effect of light rays hitting such surfaces. To perform scanning on very reflective surfaces, it is good practice to treat such surfaces using a method known as etching or powder treatment.

2.4 The Concept of Edge Detection

As computer vision, machine vision and imaging are getting more innovative, certain parameters are used to define the edges of a part called boundaries, and over time, this has presented questions of fundamental importance for image processing. Therefore, edge detection has become a vital step in preprocessing for image processing applications [88]. The edges of an object define its boundaries in an image relative to the background or other objects within the image, occasionally having sharp discontinuity, as shown in Figure 2.19. But this is not always the case for all edge detection applications. Some applications can produce blurred edges making it difficult to differentiate between boundaries. Edge detection can classify boundaries depending on its homogeneous localisation within an image using such information as image intensity and texture. Therefore, edge detection is the process of detecting sharp discontinuities in pixel intensity within an image, defining the boundaries of objects and features in that image [89]. It is the process of finding an area in an image having a sharp change in intensity or colour where a

high value in frequency indicates a sharp edge and a low value indicates a shallow edge, as indicated in Figure 2.20. From the figure, the red line in the middle image indicates the edge created by the discrepancies in the frequency values of the image pixels. These frequency values also determine sharp or faint contrasts.

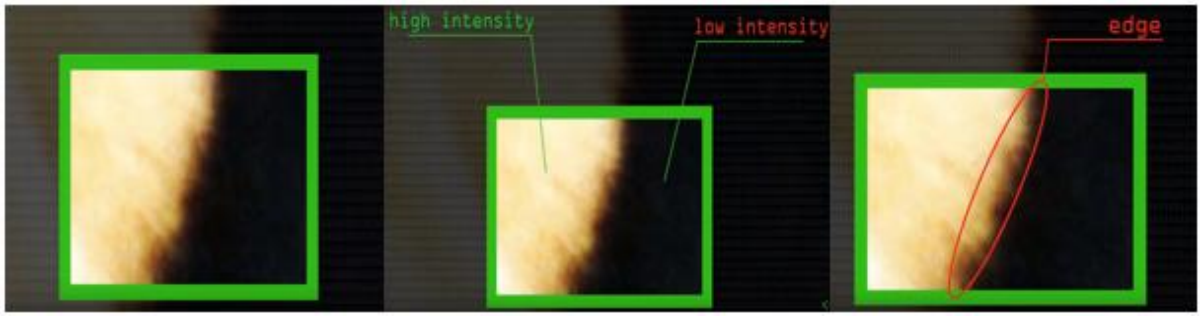


Figure 2.19: An illustration of the concept of edge detection existing between sharp variations in the intensity of an image. Reproduced from [90]

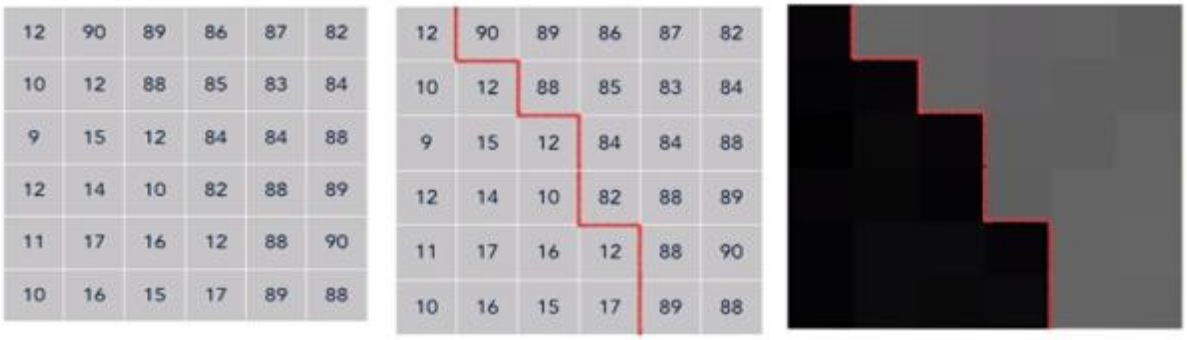


Figure 2.20: An illustration of edge detection in the pixel variation of an image

Priyam, et al., [91] described edge detection as a collection of mathematical methods to classify points in a digital image by identifying a discontinuity or contrast in the image brightness. This description is contrary to that of Papari and Petkov [92], who described edges because of human experience to mathematical definition. The primary aim is to detect the intensity variation in digital images as described by Qiuping et al., [93]. With this description, it can be said that edges exist between an object and the background, between two or more identical or distinct objects, between different regions on a surface, and is the most distinctive feature of an image. The suggested scheme of operation in performing edge detection includes smoothing, enhancement, detection, and localisation. Edge detection application can also be seen in section 5.2.2.

Smoothing is concerned with noise reduction without damaging the true edges, while enhancement utilises a filter in improving edges in an image. The third operation (detection) helps determine what edge pixels are identified as noise to be discarded and what edges should be retained. The precise location of an edge is determined by localisation using edge thinning and linking [94]. This localisation can be affected by blurring a grayscale image, although blurring aid in the removal of noise. Several edge detection algorithms, also known as operators, include Robert operator, Prewitt operator [91], Laplace of Gaussian (LoG), Sobel operator [95] and Canny operator [91, 96]. Each algorithm applies to a certain edge type using different criteria such as edge's orientation, structure, and sensitivity to noise response.

Because of its robustness and low sensitivity to noise, the Canny edge detection algorithm has been proven to produce better edge detection output than other algorithms by producing single point responses (returns one point for each true edge). During filtering, weak edges can be lost due to Canny's sensitivity [97] which has contributed to several enhancements on the Canny edge detection operator, such as field-programmable gate array (FPGA) implementation [88] for a cost-effective, robust Canny algorithm and gravitational field intensity was introduced to replace image gradient [97]. An improvement of Canny using local kernel smoothing was presented by Kuang et al., [94], which adapts the local neighbourhood to the local smoothness of the measured surface using the observed data. This improvement was due to the failure of the Canny algorithm in 2D Gaussian in removing noise and retaining the edges when a high level of noise is experienced. The local kernel smoothing correctly reduces noise in continuity regions and simultaneously preserves discontinuity [94]. Another improvement to the Canny algorithm is the Type-2 Fuzzy sets that automatically select the Canny algorithm threshold value, although the fuzziness influences the output performance of the threshold value.

The noise issue has been a challenging factor in image processing, and most algorithms for detecting edges are built on estimating the intensity gradient vector. This intensity gradient vector is sensitive to noise, and with the aim of censoring the noise, spatial averaging methods and differentiation methods such as Laplacian of Gaussian and detection of zero-crossing can be integrated together [98]. Although, there are different approaches to edge detection, such as the gradient techniques and the statistical methods, with less attention being focused on statistical methods. In statistical methods, the distribution of intensity values within the surrounding regions of a selected pixel is inspected to assess its classification as an edge. Despite its

unpopularity, statistical methods have been used by few authors such as Bovik et al., [99] and Yakimovsky [100].

2.4.1 Edge Detection Algorithms

The traditional edge detection algorithms are performed by detection of a maximal value for the first derivatives or zero crossings of the second derivative [97]. The first derivatives are classified as the Robert operator, Prewitt operator and Sobel operator, while the second derivative includes the Laplace of Gaussian operator and the Canny operator [97]. Roushdy [98] has mentioned that Boie-Cox, Shen-Castan, and Canny operators perform better than the LoG, which performs better than the first derivatives. These edge detection operators use a convolution mask in approximating the first or second derivative of an image. The edge detection process is usually divided into three stages. Firstly, the noise is reduced for better performance using a low-pass filter as the accompanying noise to the image is usually a high-frequency signal. With this method, there is a high possibility of removing edges as they give a high-frequency signal, but manipulating specific parameters provides a compensation between the reduced noise and conservation of edge information. Secondly, the edges are detected using a high-pass filter such as differential operator, and finally, the edges are localised in detecting genuine edges as compared to noise which gives similar signals using a technique known as thresholding [101]. The information of edges within an image is obtainable using convolution kernels, a 3x3 such as the Prewitt and Sobel, or a 2x2 kernel such as the Robert operator. The intensity gradient, together with a specified convolution kernel, is used in calculating the intensity discontinuities in identifying edges where a high local gradient is an indication of this discontinuity. Image production is usually accompanied by noise, and noise has been a challenging factor in the performance of edge detection algorithms and image processing techniques because of a reasonable level of noise sensitivity in most algorithms. The edges in an image produce high frequencies that aid detection, and the accompanying noise also produce either similar or higher frequencies, making them detectable by the edge detection algorithms as edges. The presence of noise degrades the quality of an image, making it difficult for the smooth operation of edge detection algorithms and other forms of image processing. In dealing with noise, thresholding methods such as mathematical morphology and hysteresis have been proposed for dealing with Gaussian noise. However, some noises such as Poisson noise, impulse noise, random noise, spike noise, speckle noise, might not be identified using these thresholding methods, leading to more

research in identifying and eliminating noise in an image. Such methods include filtering techniques such as adaptive, fir, median, linear and nonlinear filtering [102]. Some noise sources include the vicinity of the image, capturing device (inaccuracy in a camera, misaligned lenses, weak focal length, sensor noise and scattering), and ambient light, as well as incident lighting. Gaussian noise, which has a density approximation function equal to the normal distribution, is also recognised as the Gaussian distribution. A modified adaptive bilateral filter was proposed for the elimination of such noise [102]. The mean-squared error of various noises associated with image processing is shown in Appendix B Mean Square Error (MSE) of the Various Noises Associated with Image Processing with R, G, and B Values.

2.4.2 Gradient-Based Operators

The gradient-based operators (algorithms) are conceptually derived from the usage of both first and second-order derivatives of an image's gray level. The first derivative class of edge detectors works by detecting the maximum and minimum gradients in the first derivative of an image to match local image segments with certain edge patterns [89]. The magnitude of the gradient is used in processing first-order derivatives using convolution masks where the direction of the edge at a point is perpendicular to that of the pixel gradient. For the second-order derivative, two impulses on both sides of the edge are identified. The location of the edge can be accurately computed to the sub-pixel level by drawing a line between both impulses, and the point of intersection of the line with the zero axes is considered the centre of the edge. For sub-pixel accuracy, it is meant that zero-crossing can be at fractional pixel distance [98]. Assuming an image is expressed as a two-dimensional function $f(x, y)$ where x and y are spatial coordinates and f is amplitude, (x, y) defines the light intensity at that point and this is given as

$$\nabla f(x, y) = grad(f) = \begin{bmatrix} gx \\ gy \end{bmatrix} = \begin{bmatrix} \partial f / \partial x \\ \partial f / \partial y \end{bmatrix} \tag{2.1}$$

From the above equation, the vector gives the direction of a significant rate of change of f at location (x, y) whose value is given below as

$$mag(\nabla f(x,y)) = \sqrt{gx^2 + gy^2} \quad (2.2)$$

The orientation angle of the edge with relation to the pixel grid gives rise to the spatial gradient, which is given by

$$\theta(x,y) = \arctan\left(\frac{gx}{gy}\right) \quad (2.3)$$

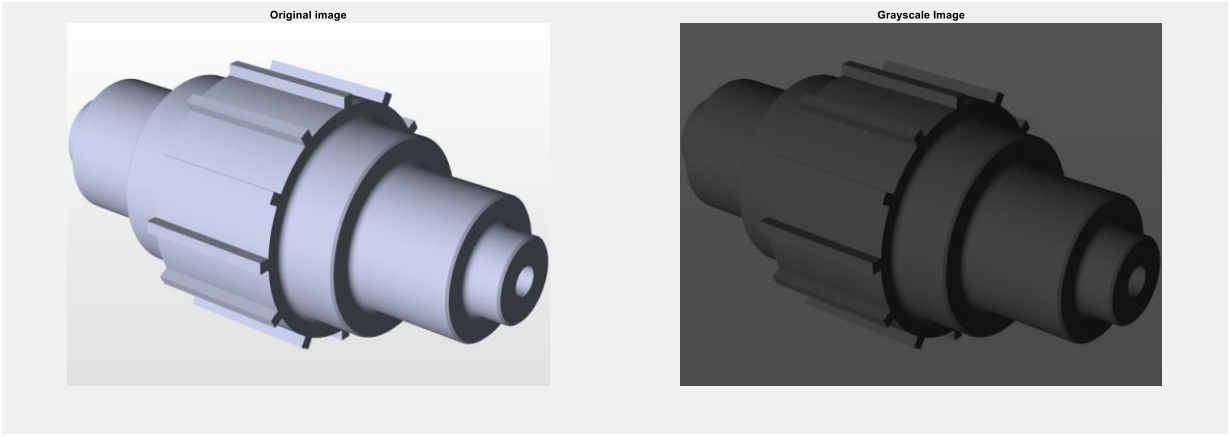
The above equations are used in computing the gradient of the whole pixel that exists within an image which is processed using small region pattern convolution. This convolution is the fundamental factor used by first-order derivatives in determining the direction of the convolution mask.

Table 2.2: 3x3 region of an image

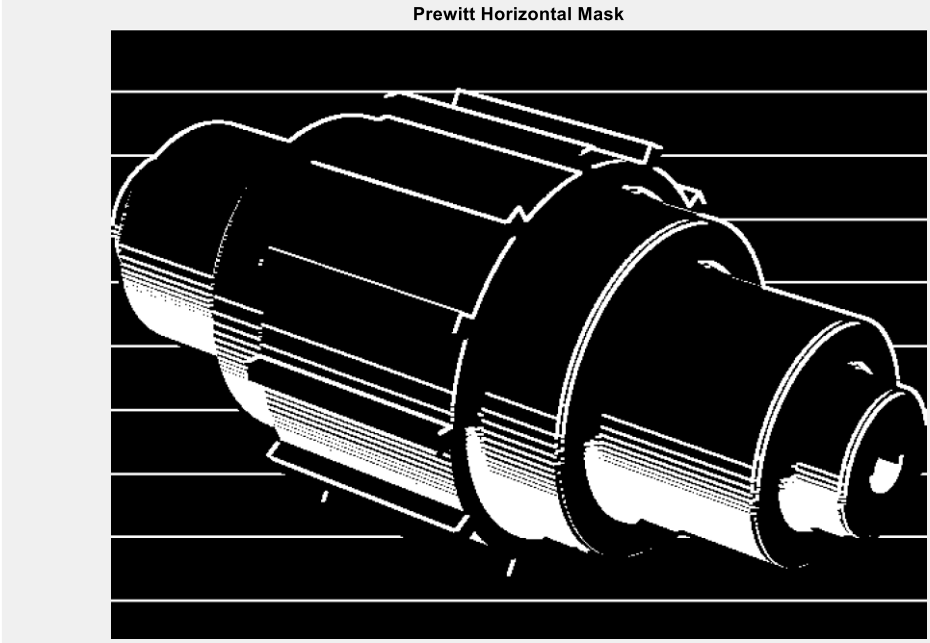
Z1	**Z2**	**Z3**
Z4	**Z5**	**Z6**
Z7	**Z8**	**Z9**

2.4.2.1 Prewitt Operator

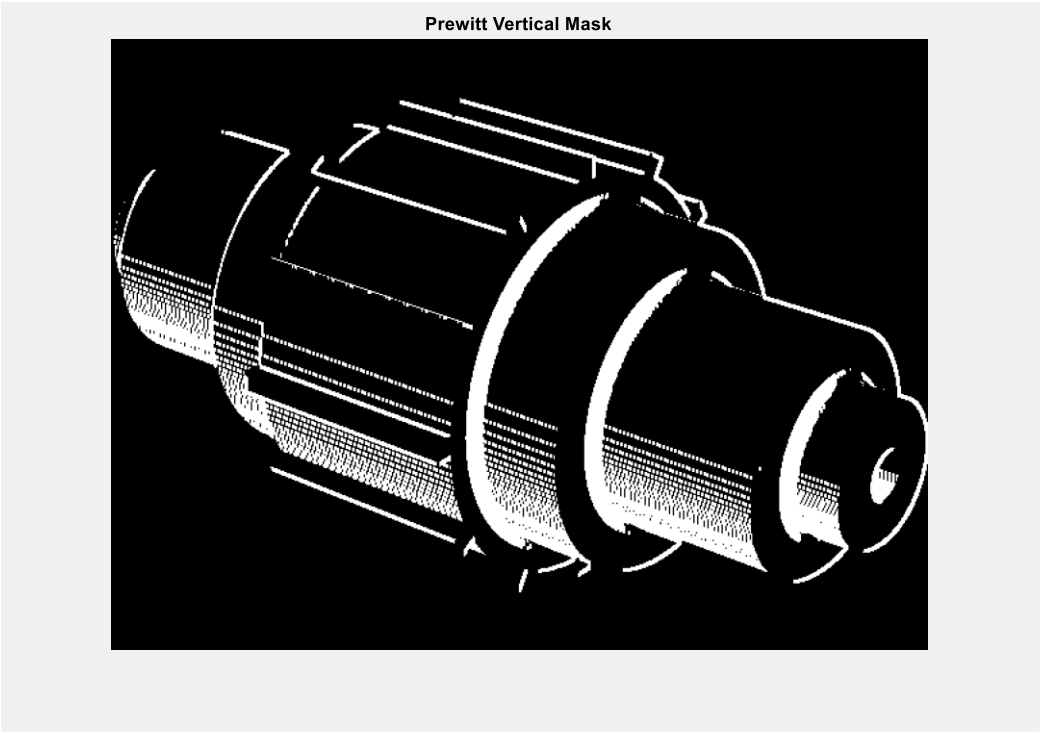
The calculation of edges by the Prewitt operator is by computing the discrepancy that exists between corresponding pixels intensities found in an image. It is classified as the derivative operator based on its gradient analysis, and it operates like the Sobel operator with a different kernel having a better performance. This operator performs detection in both vertical and horizontal directions and can be seen in the configuration of its kernels using a 3x3 mask in determining derivative values in both directions. An illustration of both vertical and horizontal directions is shown in Figure *2.21*. To extract the true edges, the gradient of both direction is computed, illustrated as the normalised mask of Figure *2.21*.



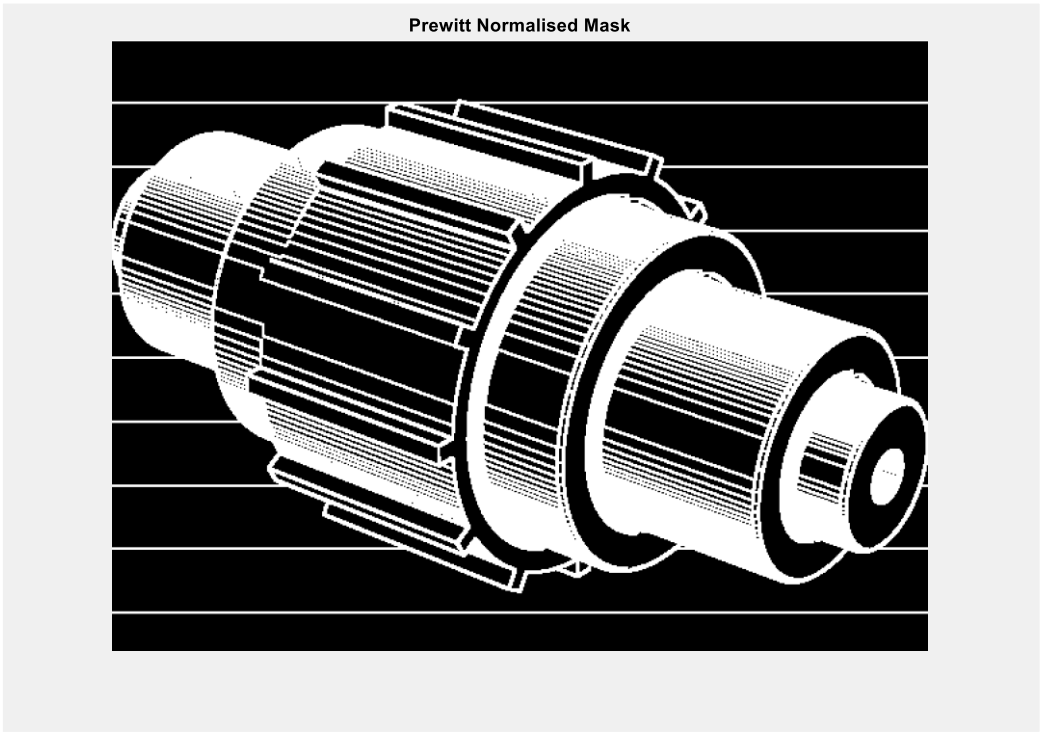
(a) (b)



(c)



(d)



(e)

Figure 2.21: Illustration of the mask convolution of Prewitt Edge detection (a) original image (b) grayscale image (c) horizontal mask direction (d) vertical mask direction (e) Normalised mask

Table 2.3: The Prewitt operator mask

-1	**0**	**1**
-1	0	1
-1	0	1

-1	**-1**	**-1**
0	0	0
1	1	1

$$gx = (z3 + z6 + z9) - (z1 + z4 + z7) \quad (2.4)$$

$$gy = (z7 + z8 + z9) - (z1 + z2 + z3) \quad (2.5)$$

The horizontal derivative approximation as equation 2.4 and vertical derivative approximation as equation 2.5 and the matrix structure is shown in Table 2.3.

2.4.2.2 Robert Operator

This operator uses a discrete approach in calculating the gradient of an image, and it is the most used tool for edge detection in computer vision and image processing. The gradient is calculated by summating the rectangles of clear contrast between transversely adjacent pixels, then the 2D point in a spatial gradient is computed. This operator is known as the directional mask based on gradient analysis, and this operator has got simplicity of operation for its small kernel size, less compatibility and sensitivity to noise. A 2x2 mask intertwined with the entire image using vertical and horizontal Robert masks produces edges in x and y directions.

Table 2.4: 2x2 region of an image

Z1	**Z2**
Z3	Z4

Table 2.5: Robert masks

-1	**0**
0	1

0	**-1**
1	0

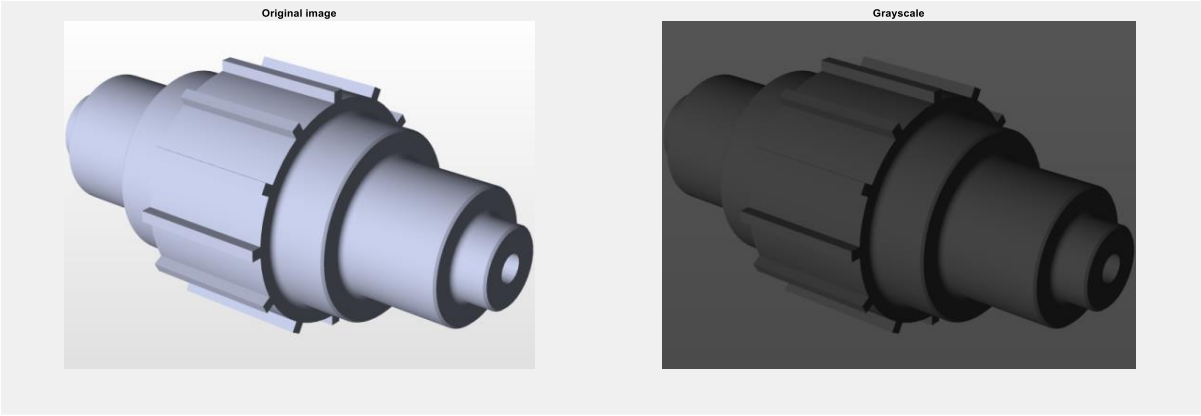
$$gx = (z4 - z1) \quad (2.6)$$

$$gy = (z3 - z2) \quad (2.7)$$

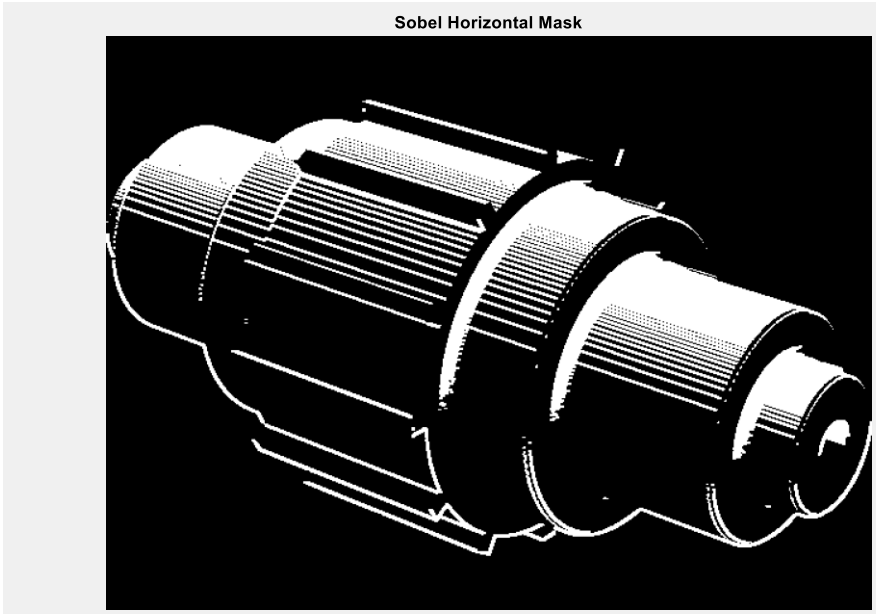
The horizontal derivative approximation as equation 2.6 and vertical derivative approximation as equation 2.7 and the matrix structure is shown in Table 2.4 and Table 2.5.

2.4.2.3 Sobel Operator

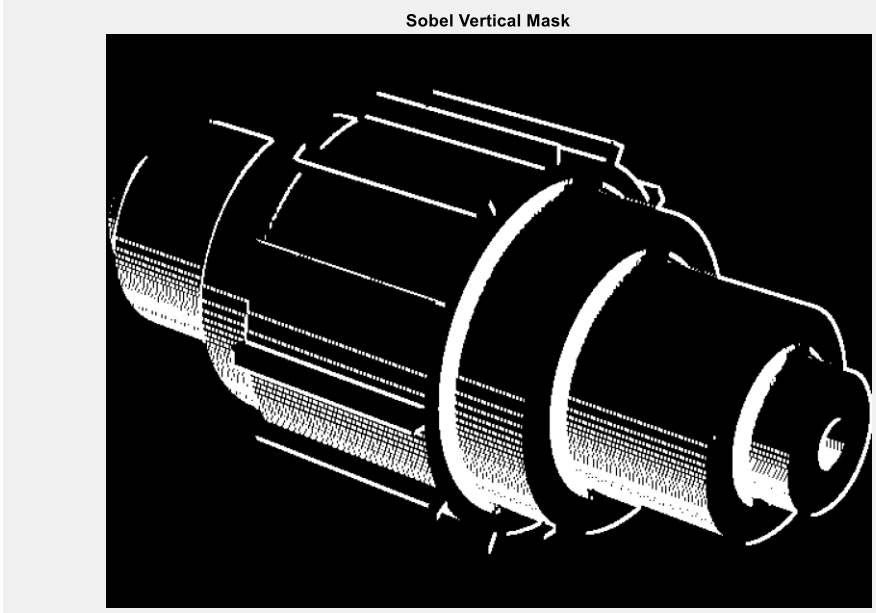
The Sobel operator exhibits characters like that of the Prewitt, and based on gradient analysis, it is known as the derivative mask but differs in terms of weight provided to the pixel value ('2' and '-2' in place of '1' and '-1'). The weight is at the edge region resulting in increased intensity and smoothing of the edges. The kernels of Sobel are designed to respond as much as possible to edges running in both directions connected to the separate pixel grid (vertical and horizontal) that produces different measurements of the gradient component in both orientations ($gx\ and\ gy$). An illustration of both vertical and horizontal directions is shown in Figure 2.22. To extract the true edges, the gradient of both direction is computed, illustrated as the normalised mask shown in (e) of Figure 2.22.



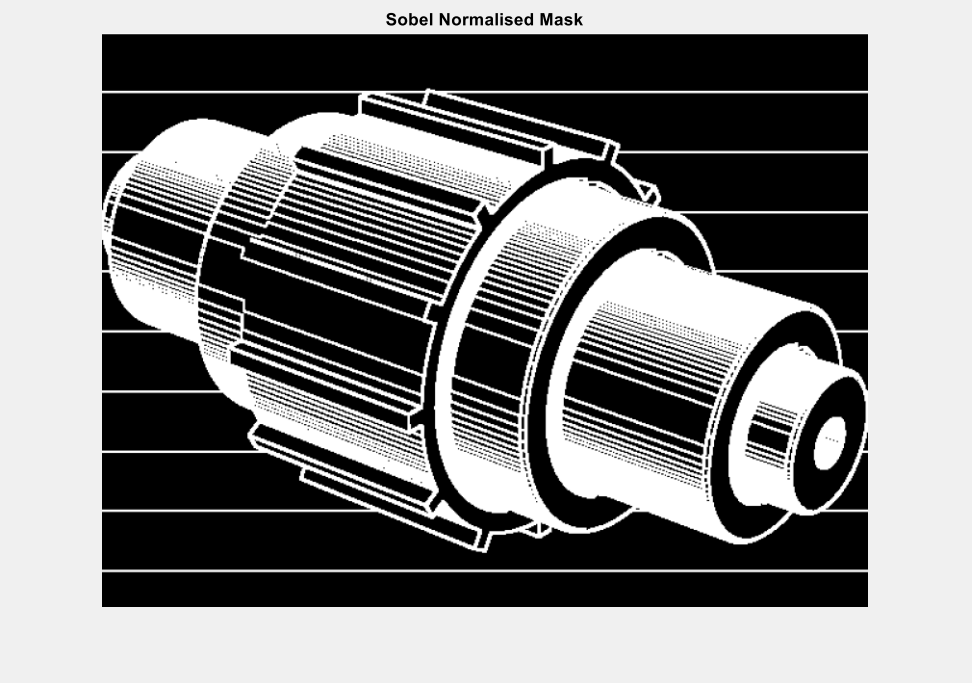
(a) (b)



(c)



(d)



(e)

Figure 2.22: Illustration of the mask convolution of Sobel edge detection (a) original image (b) grayscale image (c) horizontal mask direction (d) vertical mask direction (e) Normalised mask

Table 2.6: Sobel masks

-1	**0**	**1**
-2	**0**	**2**
-1	**0**	**1**

-1	**-2**	**-1**
0	**0**	**0**
1	**2**	**1**

$$gx = (z3 + 2z6 + z9) - (z1 + 2z4 + z7) \quad (2.8)$$

$$gy = (z7 + 2z8 + z9) - (z1 + 2z2 + z3) \quad (2.9)$$

The horizontal derivative approximation as equation 2.8 and vertical derivative approximation as equation 2.9 and the matrix structure is shown in Table 2.6. An example of the Sobel operator is shown in Figure 2.23.

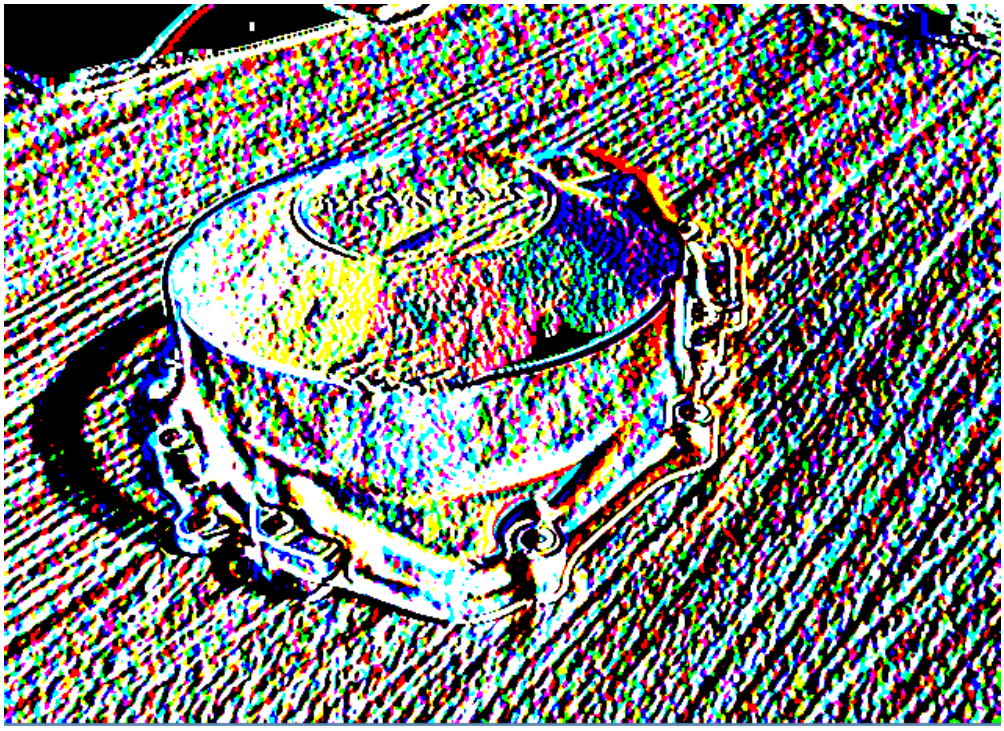


Figure 2.23: Sobel edge detection operator

2.4.3 Laplace of Gaussian (LoG)

The Laplacian of Gaussian (LoG) can be defined as a 2D isotropic measure of the second-order spatial derivative of an image that exhibits similar characteristics to a convolution filter. The derivatives for edge detection can be classified as first-order and second-order derivatives. The first-order derivatives are challenged with the task of computing functions to both extreme values of maximum and minimum, and second-order derivatives like the Laplace operator becomes desirable when checking the presence of zero-crossing points [103]. Laplacian functions in areas where regions with rapid intensity change are raised. The application of the Laplace operator produces poor quality imaging as the noise points are amplified, but this can be countered by applying a filter known as the Gaussian blur for a smoothing approximation that eliminates exposure to noise before applying the Laplacian filter. This process highlights edges of single gray level images as input and outputting binary gray level images.

The Laplacian $L(x, y)$ of an image having pixel intensity values $I(x, y)$ is given by

$$L(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \quad \text{where } I = f(x, y) \tag{2.10}$$

The Gaussian function is given as

$$G(x,y) = -e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.11}$$

Therefore, the Laplacian of Gaussian can be stated

$$LoG = \nabla^2 G(x,y) = \frac{\partial^2}{\partial x^2} f(x,y) + \frac{\partial^2}{\partial y^2} f(x,y) \tag{2.12}$$

$$\therefore LoG = \nabla^2 G(x,y) = -\left[\frac{(x^2+y^2)-\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.13}$$

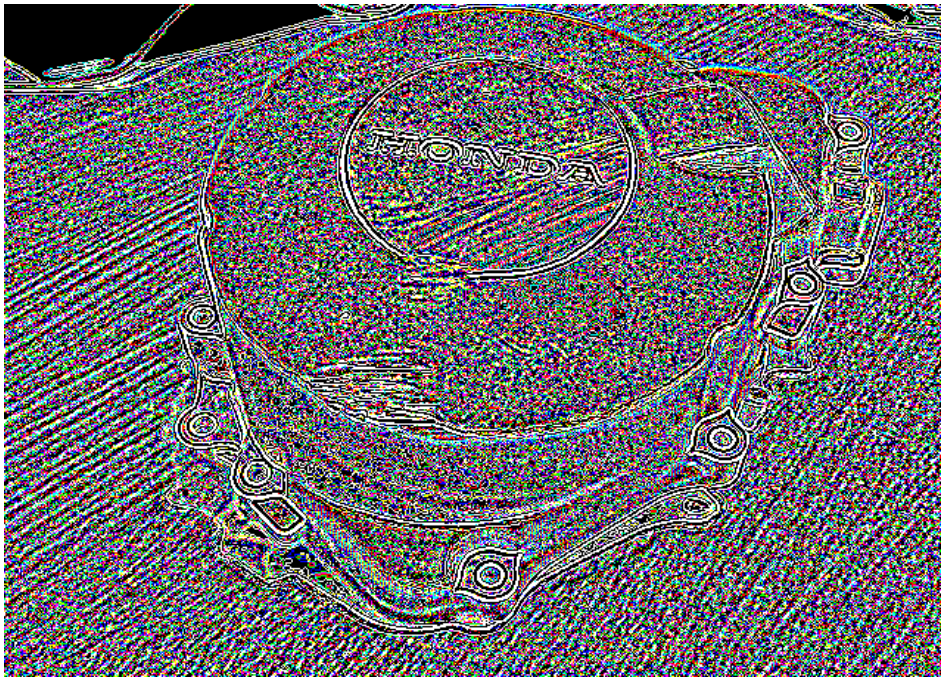


Figure 2.24: Laplacian of Gaussian edge detection operator

Figure 2.24 presents the detection of edges using the Laplacian of Gaussian algorithm. The edges are pretty visible, but the system detects other properties like the texture of the surface on which the object is placed and the texture of the object's surface. This gives a similar result when using the Sobel algorithm, as shown in Figure 2.23.

2.4.4 Canny Edge Detection Operator

The most robust and reliable operator for edge detection in grayscale image processing is the Canny edge detection algorithm. It requires three criteria for operation, firstly is the low error rate, which is an essential factor in achieving better system performance, as edges that appear in an image should be clear enough with no false response. Secondly is the localisation of edge points, which should be appropriately positioned to minimise the distance between the detector and centre of the true edge—finally, a single edge response. The final criterion was a necessary inclusion as the first two are not completely firm in output response, and the final criterion was included in the likelihood of multiple responses to an edge [96, 104]. This algorithm is a preferred operator as it presents two main goals of detection and localisation performance of a system. Existing between these goals is an uncertainty concept for systems dealing with noisy step edges, but generally, there is a direct relationship between both goals. An implication of this relationship results in the production of an impulse reaction having a specific exclusive shape in an optimal step detector. The relationship between detection and localisation can be modified when the spatial width of the detector is altered. A challenging factor in performing edge detection is the complication of distinguishing edge points and noise points in image processing which is suffered by other methods. The Canny's ability to locate edge points is quite satisfactory, and it has low sensitivity to noise because it uses a Gaussian smoothing filter in eliminating noise by smoothing an image [105]. This sensitivity allows for correct detection of edges as the system produces multiple responses from a single edge, and this is such a situation as to the application of the final criterion of operation [96]. Also, the effectiveness of Canny's performance is dependent on the mask size which is a direct consequence of the standard deviation σ of the Gaussian filter. A larger value of σ indicates an improvement of the algorithm's resilience to noise but subverts its ability to locate true edges. In contradiction, a smaller value of σ indicates better detection ability but decreases the algorithm's resilience to noise. One way to reduce this dependency is by adjusting the standard deviation parameter based on the noise characteristics of the image.

Regarding the sensitivity of Canny, Xiao and Hui [106] proposed an improved algorithm from a predisposal approach applied to corrupted images from the Gaussian noise environment. Their method combined the coefficient of the identified edge points and the distance judgement of the gray value and produced a comparison by embedding a predisposal step in

the Canny processing. Kuang, et al. [94] used a local linear kernel smoothing approach to adapt local neighbourhoods to the local smoothness of the measured surface as an improvement to the Canny edge detection algorithm in images corrupted by Gaussian noise. However, the algorithm performs poorly in situations with a very high noise ratio of almost half the maximum ratio. Other improvements on the Canny algorithm can be found in [88, 94, 97, 107]. The original Canny process uses a frame-level method for complex, time-consuming detection with consideration of hardware cost. An improvement of the Canny process was implemented by [88] using an FPGA in reducing computation cost and latency of the Canny process to improve the time interval between simulation and system response. Pixels were operated in parallel using a parallel implementation model to reduce latency [108]. The Canny operator can be implemented in the following steps [106]:

- The noise is eradicated using the Gaussian filter.
- The gradient in 4 different directions is computed, after which the detected point with its neighbour points in each direction is compared. A non-maximum in the gray value of a point is considered a non-edge point because it suppresses any pixel below the maximum threshold, and this step is called Non-Maximum Suppression (NMS).
- Confirmation of an edge point is obtained by comparing the gray value of the detected point and the thresholds, obtainable from accumulative statics values. It is expected that for an actual edge point, the gray value should be more than the higher threshold level, and non-edge points should produce values less than the lower threshold level. The points in-between both thresholds are characterised by comparison to the neighbouring points, i.e., a neighbour point is considered an edge if found in any of the four directions.

The last two steps are the reasons for the recommendation of the Canny operator in edge detection due to its non-maximum suppression and double threshold in selecting the edge points [104]. The double thresholding approach uses two thresholds, t_{max} and $t_{min,}$ for classifying the gradient into three groups.

- Gradients > t_{max} definitely an edge point
- Gradients < t_{min} definitely a non-edge point

- Otherwise, a deduced conclusion is dependent on the direction of the point and existing edge paths.

This double thresholding is used to determine what response is an acceptable edge and what edges are not, as establishing a threshold level when faced with a wide range can be challenging. Assuming a range of 0 to 255, determining what value is an acceptable edge will be challenging, and it is a good practice to create a range with 0 as no edge and 255 as the highest edge response obtaining. The three performance criteria for implementing Canny are

- Good detection with less probability of missing actual edge points and less probability of spuriously identifying non-edge points. This criterion directly responds to maximising signal-to-noise ratio as both probability factors decrease the ratio output functions.
- Good localisation where the identified edge points should be of close positioning to the centre of the true edge.
- A single response to each edge

2.4.4.1.1 Detection and Localisation Criteria

An important step in the Canny process is dealing with the signal-to-noise ratio and localisation. Assuming the impulse response of a filter to be $f(x)$, having an edge $G(x)$, and with the assumption that the edge is centred at $x = 0$, then the result from the filter to the edge at its centre H_G is computing using a convolution integral

$$H_G = \int_{-W}^{+W} G(-x)f(x)dx \tag{2.14}$$

Where the finite impulse response of the filter is restricted by the limits $-W, +W$. To obtain the root-mean-squared response corresponding to the centre will be given as

$$H_n = n_0 \left[\int_{-W}^{+W} f^2(x)dx\right]^{1/2} \tag{2.15}$$

Where the mean-squared noise amplitude is represented as n_0^2 and this over a per unit length range. The first criterion of the Canny algorithm (signal-to-noise ratio) is expressed as the quotient of the impulse response and the root-mean-squared response

$$SNR = \frac{\left|\int_{-W}^{+W} G(-x)f(x)dx\right|}{n_0\sqrt{\int_{-W}^{+W} f^2(x)dx}} \tag{2.16}$$

In dealing with the localisation criteria, the goal was to develop a system that increases with improvement in localisation and Canny proposed using a complementary system to the root-mean-squared distance of the identified edge from the centre of the true edge. Since the edge response from the operator $f(x)$ is classified as an edge at the local maxima, the first derivative is assumed to be zero. As edges are centred at $x = 0$, a local maximum for the response at this centre point should be provided in the absence of noise. Assuming the response of the filter to noise is given as $H_n(x)$, with its response to the edge as $H_G(x)$, and a local maximum at point $x = x_0$

$$H'_n(x_0) + H'_G(x_0) = 0 \tag{2.17}$$

Where $H'_G(x_0)$ is known as the Taylor expansion about the centre point, and it is given as

$$H'_G(x_0) = H'_G(0) + H'_G(0)x_0 + O(x_0^2) \tag{2.18}$$

With the hypothesis that $H'_G(0) = 0$, meaning that filter response has a local maximum in the absence of noise at the centre point, ignoring the expansion's first term.

$$H''_G(0)x_0 \approx -H'_n(x_0) \tag{2.19}$$

$H'_n(x_0)$ in Equation (2.41) is the Gaussian random quantity with a variance having a mean-squared value of $H'_n(x_0)$ which can be expressed as

$$E[H'_n(x_0)^2] = n_0^2 \int_{-W}^{+W} f'^2(x)dx \tag{2.20}$$

$$E[x_0^2] \approx \frac{n_0^2 \int_{-W}^{+W} f'^2(x)dx}{\left[\int_{-W}^{+W} G'(-x)f'(x)dx\right]^2} = \delta x_0^2 \tag{2.21}$$

Where the output δx_0^2 approximates the standard deviation of x_0. Then the localisation can be expressed as the reciprocal of δx_0^2 and is given as

$$Localisation = \frac{\left|\int_{-W}^{+W} G'(-x)f'(x)dx\right|}{n_0\sqrt{\int_{-W}^{+W} f'^2(x)dx}} \tag{2.22}$$

2.4.4.1.2 Thresholding

Thresholding or thresholding by hysteresis is a typical approach to obtaining the edge map of an object in an image as a post-processing method. Hysteresis thresholding uses a two-level

threshold, higher and lower thresholds $t_2 > t_1$. Pixels that lie in a region higher than the highest threshold t_2 the level is marked as an edge, and pixels lying between t_2 and t_1 are also marked as an edge and are connected to the edge, but pixels outside any of these regions (below t_1) are rejected. This method alleviates edge discontinuity issues by defining strong edges and retaining the corresponding frail edges and ensuring a degree of noise suppression. In threshold processing, an appropriate thinning technique such as the morphological algorithm can be used in obtaining an edge with a one-pixel width. Although thresholding can be pretty straightforward, the challenging factor is determining the appropriate value, mostly in performing simple thresholding compared to hysteresis thresholding (two-level threshold) [109]. In simple thresholding, it is difficult to determine what value is acceptable and what value is not. This difficulty with simple thresholding has led to the development of several methods for determining the threshold value, such as the unimodal thresholding method [110] that assumes a comparison between a more assertive population to a secondary population with the assertive population creating the main peak situated at the lower end of the histogram.

Thresholding is simply defined as having a function $T(x, y)$ and assigning values V_1 and V_2 as shown in equation 2.23, where these values depend on the value of the threshold t corresponding to the points (x, y) of the input image $f(x, y)$

$$T(x,y) = \begin{cases} V_1 & for\ f(x,y) > t \\ V_2 & for\ f(x,y) > t \end{cases} \tag{2.23}$$

The values V_1 and V_2 are referred to as the maximum or minimum of the image function and corresponds to 1 and 0. While the outcomes are positive, the hysteresis process considerably slows the overall algorithm.

2.5 Concept of Automatic Feature Recognition

It is a known fact that computer-aided design (CAD), computer-aided engineering (CAE), computer-aided manufacturing (CAM), and computer-aided process planning (CAPP) applications are feature-based. Features have become an essential factor when dealing with the design, modelling and reverse engineering of products, hence the need for concepts such as feature definition and automatic feature recognition. There have been several debates

about what constitutes a feature and how best to describe its characteristics. A definition by Martin [111] describes a feature “as a semantic group containing modelling atom which is characterised by a set of parameters that depicts an object which cannot be disintegrated, used in analysis relative to one or more activities linked to the design and use of the products and manufacturing processes.” Four requirements that a feature should have include (1) a feature has to be a physical constituent of a part, (2) should be mappable to a generic shape, (3) should have engineering significance, and (4) must have predictable properties [112]. With the investigation of various feature types, the form feature is the most common as they contain both shape and parameter information of a feature. Examples of these include holes, pockets, slots, and bosses. [113]. Various representations used to describe design features can simply be unfit for representing specific manufacturing processes. This may be tailored to the development of a component, resulting in differences in the manufacturing and design characteristics of the feature. This function correlates to a set of workflows needed for the creation of the component. This unresolved trade-off to the definition of a feature is due to the failure of concluding on a universal and globally acceptable definition, as well as a consensus on accompanying nomenclature by the research and academic community. Hence, a more formal definition of a feature is identified as a physical part of a product that can be mapped to a standardised shape and has functional engineering significance [114].

Application system such as CAPP provides functions that convert geometrical information of a part in a precise manner, defined by a CAD system into manufacturing information, and secondly, the representation of a practical process plans. These process plans include identifying the appropriate manufacturing process, selecting product tolerances, working piece size, setting-up planning, operation sequence and optimisation, cutting tools, cutting parameters and tool path generation through the development of CNC code. This process involves the integration of CAPP and CAD using three strategies based on the concept of manufacturing features, and they include

1. Design by feature (DBF)
2. Automatic feature recognition (AFR)
3. Interactive form feature definition

The design by feature infers that a manufacturing library exists with features adapted to the manufacturing process requirements and not the part function. Only features from this library

are used to form a product model. Automatic feature recognition is concerned with finding information of part representation characterising the specific types of production features. The strategies involved in this process aims at creating algorithms that recognises any possible manufacturing feature with less human interaction. And finally, interactive feature definition involves the user's interaction in selecting a set of manufacturing features, defining its recognition parameters and then performing the AFR in a CAD model or structure according to instructions [115]. Recent research has shown that this process can also be executed in mathematical software, which will increase the concentration of this project.

Automatic feature recognition (AFR) is a fundamental technique used in manufacturing for linking design and manufacturing, and features form the basis for this technique. AFR technique uses a geometric-model-to-application-specific method in identifying and designing models having various features employing several recognition principles. The development circle of a product involves CAD systems for designing, CAE systems for analysis, and CAM systems for machine-controlled manufacturing. Poor communication between systems results in loss of feature information when data is transferred between systems, hence, the need for effective integration of all processes required for the manufacturing of a conceptualised product. AFR has been proposed to bridge the integration gap existing between all computer-aided systems to reduce the time required and spent on data manipulation hugely. Two approaches are stated in Table 2.7 below, with their pattern recognition using logic rules and neural networks; these approaches are described in section 2.5.

Table 2.7: The classification of Automatic Feature Recognition approaches [116]

Form feature extraction		Pattern recognition
Geometric feature extraction	Form feature identification	
1. External approach 2. Internal approach	1. Hybrid Approach 2. State Transition Diagrams and Automata 3. Logic Rules and Expert Systems 4. Graph-Based Approach	Logic rules

	5. Syntactic Pattern Recognition 6. Hint-Based Approach 7. Cell-Based Volumetric Decomposition Approach 8. Convex Hull Volumetric Decomposition Approach	
	1. Graph-based approach 2. Face coding 3. Contour –syntactic approach 4. Volume decomposition	Artificial neural networks

The internal approach to feature extraction uses the application protocol interface (API) of the CAD software in gaining access to the topological and geometrical information of the part. The external approach is a situation of data communication between developing CAD software where the part was designed and an analysis or manufacturing platform. This data export is mostly done via a universally acceptable format. The challenge with the external approach is the risk of data loss or a derivative of the original data. From Table 1, the various feature recognition approaches have been segmented into logic rules and artificial neural networks (ANN) applications. The logic rules were formed from the geometric reasoning approach in recognising technological features. The ANN methods are favourable to the logic method as the logic approaches are limited by their inability to learn, limited range of recognition, and low speed hindering its practical application [115]. Each form feature identification method was described in section 2.8.

The logic based-rule approach for feature recognition uses a fundamental principle of pairing the identified structures in a component representation with known patterns in the knowledge base using a concept known as if-then rules. The importance of this rule is ensuring the unique form feature definition: this means that only one unique form definition must exist

(no two forms with unique definition) or a single form having more than one definition in the knowledge base. A proper assembly will guarantee an accurate and detailed feature identification [116]. This method is disadvantaged by its insufficient knowledge acquisition structure, and in situations where extracted form features cannot be matched with any pattern in the knowledge base, the system struggles.

2.6 Machine Learning for Damage Detection

Machine learning (ML) is an artificial intelligence (AI) application aimed at learning from data and improve their accuracy perpetually. Machine learning is a technique in data analytics that teaches machines to do naturally what human beings do: which is to learn from experience. Machine learning algorithms use computing methods for learning data with little or no dependence on a predefined equation as a model directly from data. Machine learning algorithms are trained to find patterns and features in large amounts of data to make decisions and forecasts based on new data. As the number of sample information in the learning increases, algorithms improve their performance adaptively, making more accurate decisions and predictions [117]. Two techniques for AI application are employed: the supervised learning functions by training a model with prior knowledge of both input and output data such that future outputs can be predicted, while the unsupervised learning analyses the input data to understand hidden patterns or structures. The machine learning structure is shown in Figure 2.25.

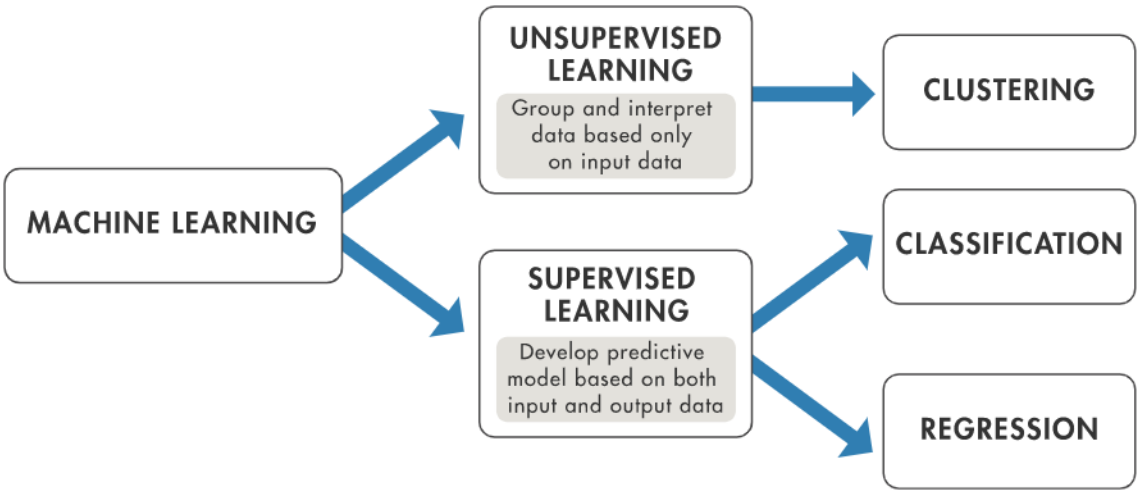


Figure 2.25: Machine learning techniques include both unsupervised and supervised learning. Reproduced from [117]

2.6.1 Supervised Learning

The construction of models for performing accurate predictions using supervised learning is based on providing information in the event of uncertainty. A supervised learning algorithm collects known input information and known data (output) responses and trains a model to create accurate forecasts to respond to new data. Classification and regression techniques are the known methods used by supervised learning in building predictive models. This type of learning uses datasets of input data together with its labelled data as a duo in training models [118]. Most supervised learning systems use a method known as transfer learning. Transfer learning uses already pre-trained networks in performing machine learning tasks, taking away the need to build a network from scratch. This learning method was employed for this project, and it will be further discussed in subsequent section 6.2.4.

2.6.1.1 Classification Technique

The classification technique predicts discrete responses by classifying input data. Medical imaging, speech recognition, image processing, and computer vision are some examples of applications. Computer vision is typically used for object detection and image segmentation. Support vector machine (SVM), boosted and bagged decision trees, k-nearest neighbour, Nave Bayes, discriminant analysis, logistic regression, and neural networks are examples of typical techniques for this application [117]. Neural network applications such as convolution neural networks and recurrent neural networks were used for this project due to the numeric

sequence nature of the data after performing sectioning (slicing) of the model and developed a sequence of slices.

2.6.1.1.1 Convolution Neural Network

The convolution neural network (CNN) was proposed as a classification algorithm for grouping areas of damage. This is a machine learning approach to damage detection using results from the 2D images of a model. This involves algorithms such as semantic segmentation or neural network algorithms like AlexNet, GoogleNet, and SegNet. The 2D profile data was used as training data with 80% as training set and the remaining data as the test set. When choosing an algorithm, accuracy and speed are the concepts to consider and, in some cases, the size of the network in the ImageNet database.

2.6.1.2 Regression Technique

The regression technique predicts continuous responses in situations of grouping (clustering) a set of elements with respect to their relationships, such as a change in temperature, load forecasting and continuous response pattern in a system. Typical applications include linear and nonlinear models, regularization, stepwise regression, boosted and bagged decision trees, neural networks, and adaptive neuro-fuzzy learning [117].

2.6.2 Unsupervised Learning

Unsupervised learning algorithms are used for generating new input data or for performing an evaluation of the hidden structures of a data set without been given classified responses. This method finds hidden patterns or structures peculiar to the data by extracting inferences from input data sets without labelled output [118].

2.6.2.1 Clustering

This is the most used unsupervised technique. It is used to find hidden patterns or groupings for the exploratory data analysis in data. Cluster analyses include analyses of the gene sequence, market research and identification of objects. Standard algorithms for performing clustering include k-means and k-medoids, hierarchical clustering, Gaussian mixture models, hidden Markov models, self-organizing maps, fuzzy c-means clustering, and subtractive clustering [117]. This method of unsupervised learning is illustrated in Figure 2.26.

Figure 2.26: Clustering finds hidden patterns in data. Reproduced from [105]

2.6.3 Transfer Learning

The process of retraining a previously trained network to identify a new collection of data is known as transfer learning. Transfer learning is a machine learning technique using a deep learning approach in which a model created for one task is used as the basis for a model on a different task. Considering the immense computation and time resources needed to create neural network models, it is a common practice when working with deep learning to use pre-trained models, as in Figure 2.27. These pre-trained models can be used as a start off stage when performing computer vision and natural language processing tasks. It is much faster and easier than training from scratch, and it can be used to improve system performance. The following parameters: pre-trained network layers, training data, and algorithm options are required for performing transfer learning [119, 120].

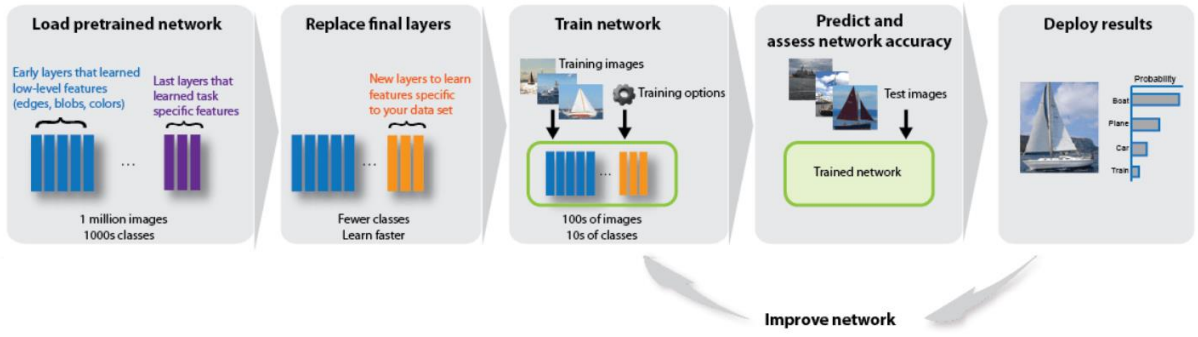


Figure 2.27: Structure of a pre-trained network for transfer learning. Reproduced from [109]

2.6.4 Review of Existing Works

Several non-destructive testing (NDT) methods have been developed based on machine learning and, in most cases, convolutional neural networks (CNN). Using lots of labelled images with negative samples (with defects) in CNN can be challenging without a corresponding positive sample for comparison. A supervised machine learning approach was proposed by Guldur and Hajjar [121] using underlying surface geometry and point cloud data as input data for a classification algorithm, a cascading fuzzy logic with image processing [122], convolutional neural network [123]. Zhao et al. [124] developed a novel framework for detecting defects by training positive samples. The basic identification method is to set up a reconstruction network capable of repairing existing defects in the sample and equate the input sample with the repaired sample to assess the specific region of defects.

A deep learning theory for detecting damage on a steel wire rope (SWR) was proposed by Huang et L., [56]. In this method, discriminant features were automatically extracted from optical images of SWR, and the layers in CNN were used to extract and classify features. Soukup and Huber-Mörk [125] used a database containing photometric stereo images of metal surface defects in training a CNN. These images were obtained by projecting different coloured-lighting sources on the metal surface and capturing cavities in a photometric dark-field setup. The images were classified using a model-based classification approach and extracting reflection properties to contrast non-defective images. Shihavuddin et al., [57] developed a deep learning-based automated damage system from high-resolution images of a wind turbine blade captured using a drone. The drone images were used to train a neural network algorithm to identify areas with potential future damage.

Although there have been machine learning applications for surface imperfections/damage detection, most of these applications are for training 2D images and using classifier algorithms in classifying extracted information into specified classes. From reviewed literature, performing damage detection and localisation on point cloud data using machine learning and unwrapping raw data method of this project is under-investigated. To limit the level of human influence, processes such as feature identification and extraction, profile monitoring, model sectioning, and machine learning was employed for damage detection. Damage detection on images and 3D visualisation using applications such as AI, segmentation algorithms, X-ray imaging, image processing such as object detection. The focus of the project

is to produce micrometre-level data from the point cloud using a model sectioning approach. The slices generated from model sectioning is used in training a recurrent neural network algorithm to identify slices with potential damage whose characteristics are different from the predictable good slices. This approach is an important step in implementing a proposed RE/SR framework that uses the damage detection information to identify regions of a model that require reconstruction

2.7 Review on Profile Evaluation and Extraction

Surface metrology has undergone a technological change in the last decade resulting in advanced manufacturing such as micro-manufacturing, nanotechnology, and improved surface quality. The need to make manufacturing more effective, economical, and less vulnerable to the effects of the environment on production and optimise efficiency prompted this technological change. According to Jiang and Whitehouse [126], size plays an essential role with respect to function, but this is evolving as planar technology and miniaturisation advances. This ensures that decimated data can be used for the same purposes as dense data, but care must be taken to preserve the object's key features. Hence, one reason why profile monitoring is as important as an areal analysis of a surface.

In their studies, Gardner et al. [130] and William et al. [131] have used the smoothing spline as a parametric model for profiling monitoring. Nonparametric types were also investigated by William et al. [127] when estimating profiles. According to Eric, et al. [128], spline estimation belongs to a broad class of nonparametric estimators with fewer constraints than parametric methods and exhibit undesirable properties, but they can be a viable alternative to the parametric model. Spline smoothing estimators will not provide a good model of a profile with specifically unsmooth features such as jumps or points of non-differentiability [128].

Several techniques are available for measuring optical 3D profiles, including stereo vision, fringe projection profilometry (FPP) and holography. The FPP is mainly used for its precision and ease of application as compared to other techniques. It can be implemented in two approaches, the first is the reconstruction of a dynamic profile from a single fringe arrangement, and the second is the use of techniques for high-speed phase shifting. Specific

applications such as inspection of machined parts, 3D reconstruction, and extraction of contour lines from an image are possible with the information obtained from profiling point cloud data [129].

For this project, a continuous wavelet transform was used in extracting information for training a machine learning algorithm. A wavelet function was utilised because it translates the response signal of the location of each point into a continuous transform along the time axis to produce a correlation that can be visualised through a wavelet coefficient. Another type of wavelet transform is the discrete wavelet transform suitable for multiresolution analysis by representing discrete input signal as an approximation in a consecutive manner. This is achieved by combining wavelet components with a smoothed signal component, hence, the reason for using a continuous transform to represent the data for training a machine learning algorithm.

2.8 Summary

This chapter reviewed the concept of surface reconstruction and the various algorithms that have been developed over time. The concept of data acquisition was also discussed as an integral part of the process of reconstructing a potentially damaged surface. It was also stated that noise is an inevitable factor when working with measurement data, and what factors are responsible for generating noise. How to mitigate this noise was discussed in subsequent chapters as part of the data acquisition process, a brief discussion on how measurement data are captured using a laser scanner and their accuracy in comparison to traditional CMM. Lastly, a brief discussion on the concept of computational geometry and the international standards relating to surface damage.

The gaps in the existing research are:

1. So many manufacturers of laser scanners are improving the scanning experience of using the laser line scanning method with a new scanner generating over 2 million points per second but are still being faced with challenges such as noise developed from reflectivity and ambient light. Although the accuracy of some portable CMM is within 40 microns, like in the case of the one used for this project, there are lots of other factors that contribute to uncertainties when performing surface scanning.

2. In addressing the issue of reflectivity, some products can be coated, which provides an extra layer on the surface. This should be considered when extracting measurement information such as nominal values as the coated surface may serve purposes other than dealing with reflectivity. The effect of coating a highly reflective surface is shown in Figure 2.28. The surface reflectivity can affect how defects are identified during scanning, resulting in surface treatment methods such as etching and powder treatment. Treatment with powder improved the visibility of the surface to the laser scanner as the powder grain settled on the edges of the damage. Another method for treating highly reflective surfaces is the dye penetration method, where the part is immersed in a special solution for 5-10 minutes, after which the solution is washed off. This solution settles on cracks and damage on the surfaces for easy identification. Other methods for treating surfaces include the fluorescent and magnetic test powder method.

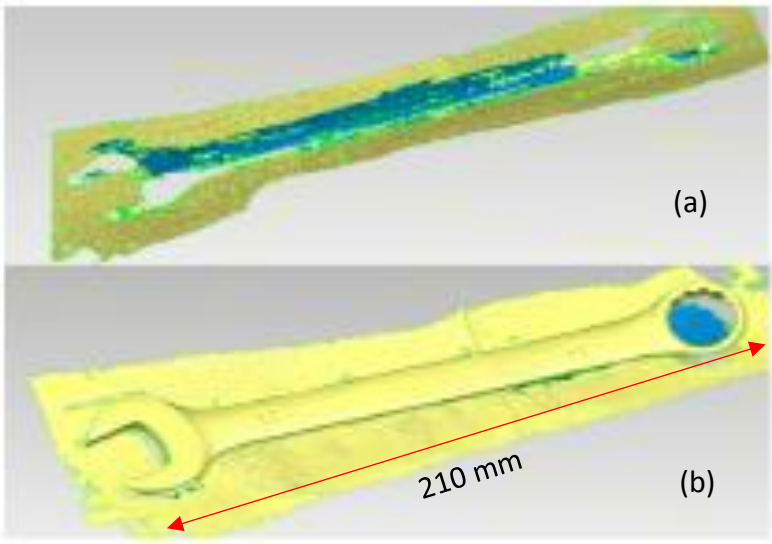


Figure 2.28: An illustration of thin coating a high reflective surface to acquire surface points (a) reflective surface without coating (b) with coating

3. The rationale behind the use of machine learning is because this technique offers a more objectified manner to accurately identify damage regions.. This will be further discussed in chapter 6, where several slices of a model using model sectioning method were developed and using slices of a good region to predict the damaged region. These slices provide micrometre-level information in a sequence-to-sequence format which formed the basis for investigating the slices as sequence data and not in

isolation of each slice, hence, the reason why a recurrent neural network like LSTM was employed. The use of this ML method means that a prior knowledge of a previous slice is required for learning to interpret the next slice. The classification from the LSTM neural network was subsequently applied to the model to extract slices with potential damage.

The concentration of this project is on damage detection and several methods and approaches have been proposed and implemented. Nevertheless, there has been ML application for damage detection and detection of surface imperfections, but most of these methods are on 2D imaging coupled with classifier algorithms. ML applications using point cloud as measurement data and generating micrometre-level information of a model is sparse, thence, the viability of the application for this project in detecting and localising damage.

Chapter 3 Methodological Approach to the Proposed Reverse Engineering/Surface Reconstruction Framework

This chapter presents the methodology and the different concepts used for this project. These concepts range from how the data was captured, preprocessed, and algorithms investigated, proposed, and implemented to achieve the project's aim of developing a RE/SR framework and identifying damage.

3.1 Methodology and Scope

The first approach was to identify techniques required for achieving surface reconstruction with more concentration on identifying areas of damage. The identification of regions of damage was implemented using a profile monitoring technique in performing a visual inspection. This technique used the rotational matrix concept to create a rotary movement of a model about an axis as its boundary profile is projected in 2D and analysed. The boundary profile was monitored, and potential damage was identified but knowing the location of this damage is required. Model sectioning was used to produce slices of a model to develop micrometre-level information in a sequence-to-sequence format. This format formed the basis for investigating the slices as sequence data and not in isolation of each slice, hence, the reason why a recurrent neural network like LSTM was employed. Investigation as a sequence data requires a prior knowledge of a previous slice for learning to interpret the next slice. These slices were used in training an LSTM neural network to perform the classification of the slices into specified classes, which will be discussed in section 6.2. The classification from the LSTM neural network was subsequently applied to the model to extract slices with potential damage.

The first advantage of using this approach is to explore features of a model at the micrometre level, whereas prior damage detection methods have primarily centred on images. Secondly, using point cloud as measurement data implies that potential damage can be investigated using the point distribution around the damaged region. This approach can aid the investigation of what is a damage and what is an intended design, although design intent was not covered in this work. The choice of using AACMM over conventional CMMs was for measurement speed, the number of points captured per second, and flexibility of use in capturing hard-to-reach areas.

From the proposed RE/SR framework, this thesis covered the damage acquisition process to identify areas of damage but did not investigate the proposed approaches after identifying damage as displayed in chapter 3. The proposed method of training a neural network was performed only on a cylindrical model and will require more analysis when applied to a different geometry. This thesis did not cover the aspect of surface reconstruction using Delaunay triangulation of a meshed surface.

Edge detection was investigated at the early stage to ascertain the suitable detection operator for the application of feature detection and extraction. The edge detection experiment found that the Canny operator performs better than most edge detection algorithms due to its sensitivity to noise and identification of false edges. Edge detection was applied to identify and extract features performed as a comparison of two methods: RANSAC using random consensus and the LSI method using linear interpolation. The values from both methods were compared to the standard United Kingdom Accreditation Service (UKAS) certification value of the calibration sphere and execution time. This method will be discussed extensively in chapters four and five.

Once features were identified, the proposed algorithm of using a rotational matrix for profile monitoring was used to monitor the feature's boundary profile. This application was more of a visual inspection requiring prior knowledge of part and providing less substantial evaluation. This level of evaluation led to further investigation using slicing for sectioning a model. Obtaining these slices provided a more in-depth analysis of the model to the micrometre level. Having several slices as a representation of the model and limiting human influence led to the use of machine learning. A proposed algorithm was developed for automatic classification of the slices into two classes, “good” and “damaged”, indicating where the damage starts and ends on the model, and this classification data was used in training the machine learning algorithm. Long-short term memory (LSTM) was employed due to the nature of the data being time-series data after unwrapping the raw data and as they were stacked in sequence. Machine learning is trained using both the extracted data and the classified classes specifying the required parameters such as layers and options. Accuracy of training and visualisation of the classification using confusion matrix was used to validate the automatic classification by the proposed algorithm. The investigation at the microns level distinguishes our work from existing literature that performed damage detection and

localisation using machine learning on images. This method will be investigated further in chapters six and seven with validation using measurement data.

The proposed framework for performing RE/SR of a potentially damaged object is presented in Figure 3.1. This framework covered the concepts of data acquisition, preprocessing acquired data, identifying features, detecting damage, intelligent dimensioning, and through to verification of the reconstructed artefact. The concept with the red borderline indicates the concentration of this project in detecting damage on a model.

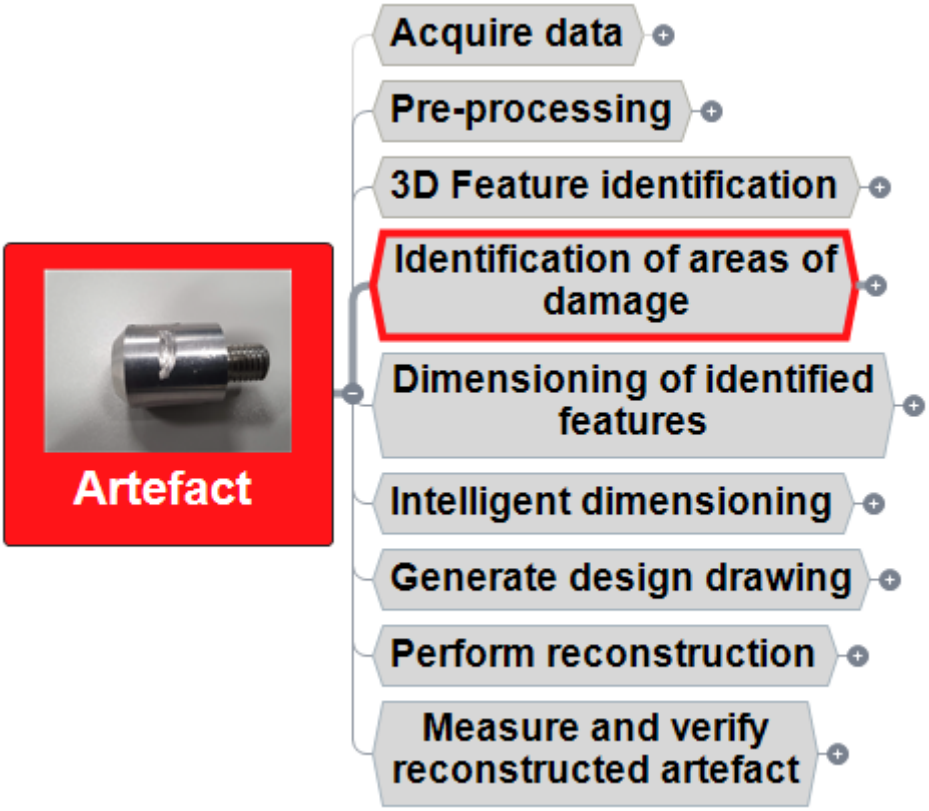


Figure 3.1: Reverse Engineering/Surface Reconstruction Framework

3.2 Acquisition of Point Cloud Data

The main instruments for performing RE are either a CMM, AACMM with a laser line scanner attached to it, or using photogrammetry. With a CMM, a 3D wireframe structure is generated when performing a tactile measurement, while for an AACMM, dense sample points were generated as a representation of the product. For this project, the process of generating dense sample points was used as it is faster compared to CMM data, produces dense data

set, is less costly when performing inspection of a large part, is more flexible than a fixed CMM and requires no pre-programmed routine as scans can be performed as at when required. For the detection of the damaged region, it was important to employ the Arm's functionality to capture internal and complex features, giving a textural representation for better understanding and visualisation. Accuracy cannot be left out. Although the CMM method is most desirable in a tight tolerance application, the accuracy of the Articulated Arm used is within a volumetric accuracy of 0.040 mm and can easily be transferred within the workshop.

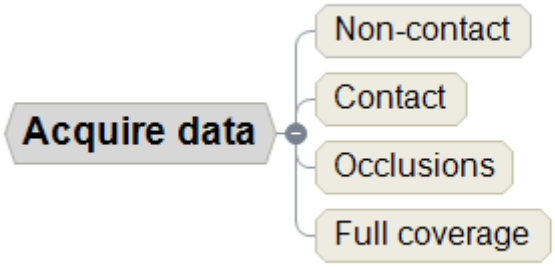


Figure 3.2: Methods of data acquisition, also discussing occlusions and full coverage of the scanner

When dealing with measurement data, an important concept is the concept of data acquisition, as this determines the techniques and methods suitable for data representation and algorithms to produce desired results for its representation(s) [7]. In combination with the characteristics of the scanned object, the data properties essentially distinguish the different levels of reconstruction approaches used in present times. This distinctive range of methods comprises (1) methods that assume a well-sampled point cloud that is generalized to arbitrary shapes and generates a watertight surface mesh, (2) methods that make weak assumptions on the quality of the point cloud, (3) methods that operate on specific classes of freeform shapes, and (4) methods which generate a non-mesh based shape representation [22]. The data acquisition technique determines the reconstruction approach suitable for the data obtained and the algorithm to produce the desired outcome. This process involves preprocessing digitized points, curve net construction, surface fitting, post-blending, and trimming [19].

The popularly used devices for data acquisition are the Coordinate Measuring Machines (CMMs) and CMMs with attached laser scanners. The CMM is classified as a tactile (contact)

system, while laser scanners are an optical (non-contact) system, as shown in Figure 3.2. The optical system can be divided into active and passive optical systems, and laser scanners are active optical systems. An example of a passive optical system for data capturing is photogrammetry, where one or series of cameras are positioned strategically to capture data. The digitisation process becomes time-consuming when using the CMM to obtain an initial set of points on complex or freeform surfaces. To use CMM in recovering the features of the workpiece, a predefined programme tool path must be specified in advance. Compared to the optical sensing system, this may pose a distinct disadvantage on the part of the tactile system.

The optical sensing technique uses a laser line at a predetermined distance called the depth of scan that determines the number of points generated by the width of the laser line. Surface points from multiple features are acquired on a single scanning path, making the laser line scanning very sensitive to unrelated problems to the tactile system, such as the influence of surface colour, shininess, transparency and in some cases, the texture of the surface. These differences have led to the development of a multi-sensor system configuration described by Durrant-Whyte [130] and Li et al. [17]. This multi-sensor system proves that both systems (tactile and optical) can compensate each other; elements having a high tolerance should be tested using high-precision probing techniques, whereas optical means are employed for elements with a lower tolerance application. High tolerance elements should be measured using high-precision probing techniques, while elements with lower tolerance can be measured using optical means.

Implementing the CAD model is strenuous when working with objects of complex shapes such as the human organ and freeform shapes having a difficult surface continuity between features as this will require much effort and expertise. This difficulty has led to the development of intelligent feature recognition and segmentation algorithms for collecting surface information [17]. For an automated process, most of these procedures must be taken to produce a surface free from deformation as complex continuity conditions exist between adjacent regions of points [19]. The input data to a surface reconstruction algorithm has X, Y, and Z coordinates defining its location in three-dimensional (3D) space. Another characteristic of the points is their vectors. The data obtainable from using a fixed CMM can be likened to an AACMM with point distribution and density difference. There are situations where an

optical device can be attached to a CMM but without the flexibility experienced with an AACMM.

For damage detection application, the acquired data is a large set of points representing the object surface (or point cloud in x, y and z coordinates) and an indication of the location of the points in three dimensional (3D) space [7]. When Point cloud is produced using laser scanners, noise and outliers sometimes accompany them. Therefore, it is necessary to filter off unwanted properties from the data. Another way of producing point cloud data is by mathematical simulation, and they are clean from noise except for the situation where noise is simulated into the data. Several factors contribute to the noise contamination of the point cloud data, and these factors can affect the integrity of the data. Factors include measurement errors in terms of the limitations of the sensors in dealing with environmental induced noise like the Microsoft Kinect and time of flight cameras, LMS-200 sick laser mounted on a sweeping unit [131], built-in noise of the scanning device, mounted structure on which the Articulated Arm sits, surface topology of the object, external lighting [132] and missing data due to human error. The scan was performed using a handheld scanner attached to a 7-axis Articulated Arm CMM mounted on a firm tabletop on which the object was placed. This operation, to an extent, tackles the uncertainty of mismatched data in the event that the object or Arm is moved from the initial registered position. The area between the base of the scanned object and the table was missing some data that was preprocessed/filtered, and reconstructed. For scanning small items, it should be elevated using a stable platform positioned on the same table as the Arm.

3.2.1 Non-Contact Method

The non-contact method is also known as an optical method of data acquisition, relying on the properties of light to achieve measurement, and they generate a large amount of data as points representing the captured surface. This method involves the use of optics for data capturing, and it includes a laser line scanners, which can be categorised into Vision systems, CT scanning, Structured Lights, Photogrammetry and Radar systems. This method produces hundreds of thousands of points that could be either structured (organised) or unstructured (randomly distributed) [133].

3.2.2 Contact Method

This is also known as a tactile method of data acquisition. This is usually on a CMM with a stylus directly contacting the surface of the object being measured. This method generates data with less density and high accuracy. They are applicable to systems with a tight tolerance level. They generate points to represent an object which are not a point cloud but points that could be equally spaced depending on the application.

3.2.3 Occlusion

This refers to the obstruction of the range of view of the laser line of the scanner. In computer vision, it involves object tracking by identifying object movement in an environment [134]. When performing scanning operations, certain regions of an object - due to object complexity - cannot be seen by the laser line, thereby resulting in missing points. It is crucial to make sure such missing data does not affect the point representation of the artefact.

3.2.4 Full Coverage

This contrasts with occlusion and is made possible by the automatic stitching of several scans by the laser scanner where the entire surface of an object can be captured for its digital representation. Points were continuously being captured as the scanner scans over the surface of an object, reaching into compact regions of freeform models.

For this project, the proposed approach is to reverse engineer models using optical systems and perform verification using the tactile method. The optical system provides good spatial coverage and is faster, but after generating a model from it, a tactile measurement that could be more accurate or lower uncertainty can be used to verify the model using an independent measurement system. However, the slow nature of the tactile system might not be significant if RE a small object.

3.3 Preprocessing

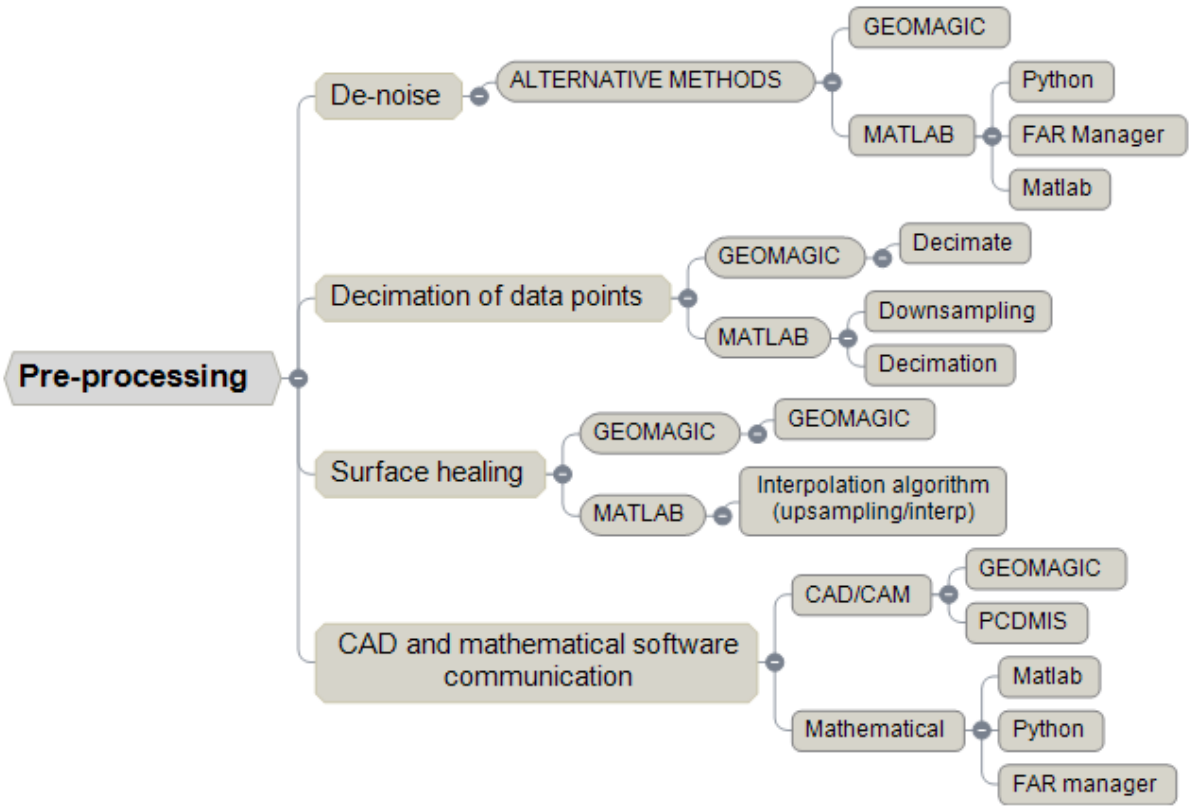


Figure 3.3: Preprocessing stage of the RE/SR framework

Measurement uncertainties are essential elements of consideration when sample points are captured as a digital representation of an object as they are projected as noise. A better representation is achieved by preprocessing the measured data to produce a dataset sometimes not entirely clean but contains fewer uncertainties. This can be achieved using the techniques specified in Figure 3.3. Dealing with a large dataset is another identifiable factor when acquiring data through optical means, as an enormous number of points are generated as the laser line scans over a surface. The AACMM with an attached scanner used for this project generates 720 000 points per second at a point spacing of 0.011 mm and volumetric accuracy of 0.040 mm. This large data can be downsampled with the risk of losing important model data, so extra care is required when reducing sampled points. This can be done using different de-noising methods, and for this project, the downsampling/decimation method proposed by [42, 43] was used. During scanning, the laser line captures the reference plane

on which objects are placed, and this can be useful in some applications, but they generally contribute to noisy data and enormous point generation. A better practice is to initiate a clipping plane, preventing the eventuality of such estimated reference planes.

3.3.1 De-Noise

Noisy data are inevitable, and before commencing reconstruction, it is important to de-noise a dataset. As part of the motivation for this project, data integrity is a significant challenge in working with data gathered via a non-contact method. This dataset contains noise, outliers and disconnected points that must be dealt with in most applications as they directly affect the outcome of processes and applied algorithms. There are situations where outliers and noisy data are allowed within the dataset to test the robustness and behaviour of algorithms. Although de-noising is a good practice when working with dense data sets, extra care is required not to lose important information and the data is reduced to an amount that is still a digital representation of the captured object.

3.3.1.1 Decimation/Downsampling of Data Points

High-density 3D scanned models often include noisy data and outliers, making it hard to perform detection and extraction of features. This can be due to scanning devices producing thousands of points per second. Although having thousands of points can be desirable by the user as most regions on a surface can be captured, scanners also pick up unwanted points. This means that preprocessing of measurement data has become an integral part of working with such data. Filtering and downsampling point cloud data are essential steps in preprocessing 3D data to derive meaningful information. In decimating point cloud data, the spatial resolution of the points is reduced not to lose important feature information but to reduce the data to a workable size and still a representation of the feature [135]. Standard filtering methods such as median and mean filtering have been mostly used for filtering and downsampling of noisy data. [131]. The voxel grid and bilateral filtering are other widely employed methods. The bilateral filtering reduces noise with consideration of corners and edges using Gaussian functions but performs poorly with outliers [136]. In [131], a downsampling technique based on Growing Neural Gas (GNG) was suggested, and this system retains the 3D model topology and simultaneously calculates the GNG network over the raw points and can deal with outliers of input space. As a preprocessing step, downsampling of the sampled points was applied to the system by decreasing the sampling rate, thereby

removing noise and irregularities in the data. For example, the sampling rate of input data K is reduced from the first number of the defined sampled number in each column. This approach is a sample-based processing procedure in which the number of samples in each column of the K matrix is viewed as a separate channel.

3.3.2 Hole Filling Technique

There could be sparse underlying point data during the scanning process, and holes are created in the triangular mesh of a model when missing data occurs. Hole-filling is a method of filling gaps in a polygon model by detecting holes and constructing a polygon mesh in an organised form over the detected hole. The hole filling process shown in Figure 3.4 is applicable to regions of smooth flat surfaces and very challenging with complex regions of high curvature. The difficulty of high curvature surfaces has resulted in the development of techniques that apply to various levels of surface curvature. Depending on how highly curved or flat a surface is, an appropriate technique can be used to better represent and avoid deformities that could be generated when holes are wrongly filled. The wrong filling of holes can change the original geometry of the model, and in the case of remodelling, an unwanted shape with no design intent and a specific dimension will be developed. There are three techniques for filling holes, and the use of each technique depends on the curvature of the surrounding mesh in creating a watertight mesh with the newly created mesh having a C^1 continuity [137, 138].

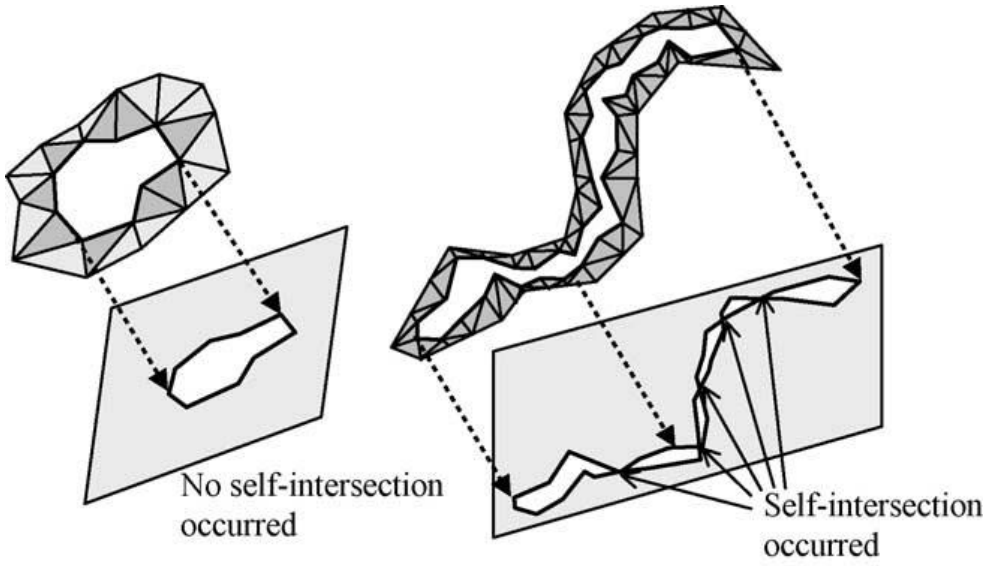


Figure 3.4: Projection of polygon holes showing simple and complex hole. Reproduced from [125]

Curvature: this technique indicates that the newly generated mesh for filling the hole must align with the curvature of the surrounding mesh when creating poly-faces, as shown in Figure 3.5.

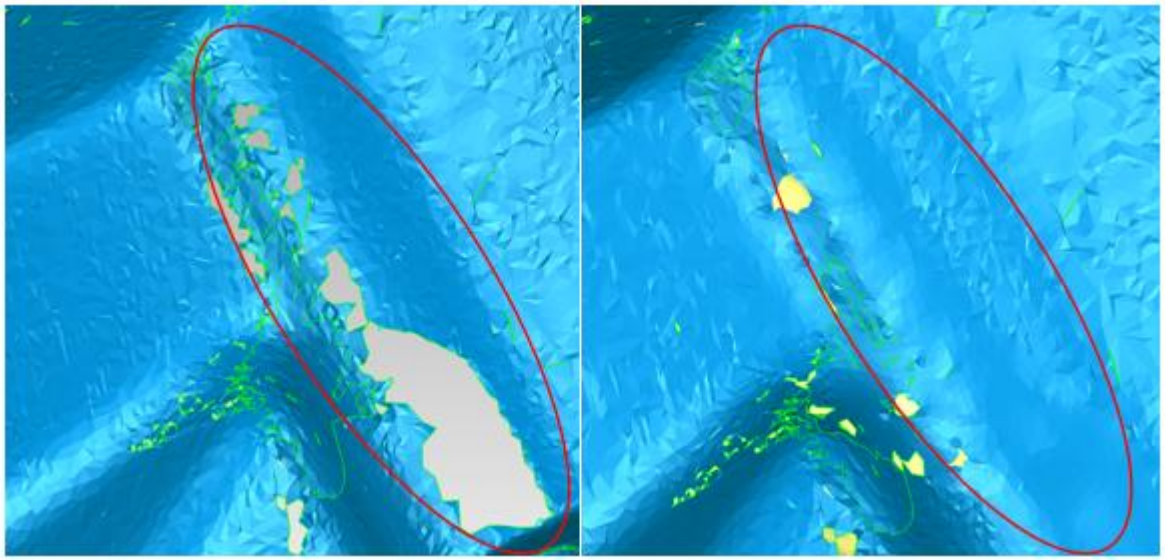


Figure 3.5: Hole-filling using the curvature technique

Tangent: this indicated that the new mesh must align with the curvature of the surrounding mesh but with more tampering.

Flat: this technique produces poly-faces which are usually as flat as the surrounding mesh, as shown in Figure 3.6.

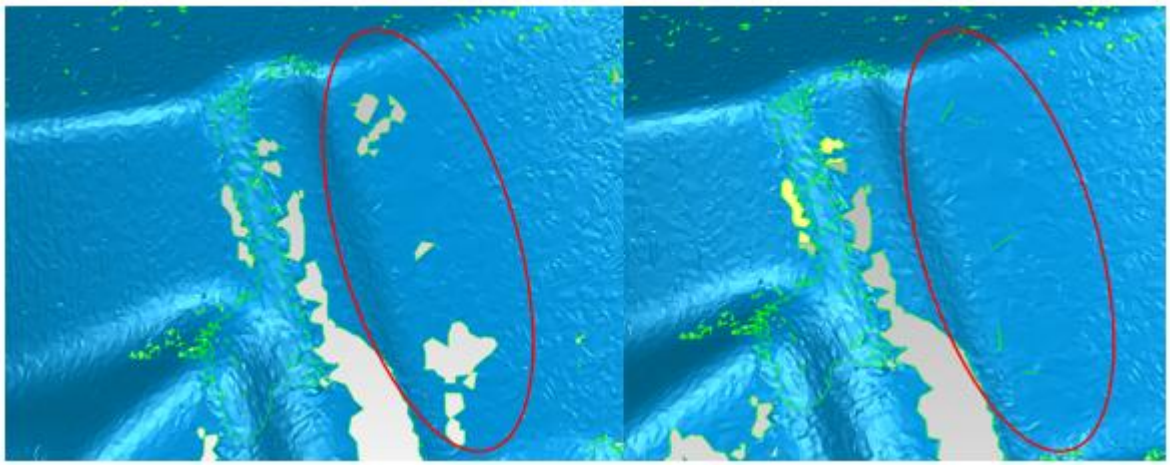


Figure 3.6: An indication of how the flat hole-filling technique is performed

Several algorithms for filling holes in a polygon model have been developed, and as mentioned earlier, they are easily applicable to smooth surfaces but not robust for high curvature surfaces. The aim is to file holes by creating new triangles that blend perfectly with the existing mesh of the model. A piecewise pattern for automatically filling complex polygonal holes was developed by Jun [139], where a complex hole, as shown in Figure 3.5, is filled by dividing it into simple holes, and each hole is filled with a planar triangulation method until the entire hole is filled. The issue with this method is the use of planar triangulation, which might not be the right fit for a highly curved surface, and such application would require a system capable of performing prediction of the intended shape using the boundaries of the surrounding mesh. The approach used for this project is similar to the work by Jun [139] but by creating bridges, thereby reducing the size of the complex hole. And for surfaces of high curvature, one of the three techniques (curvature) earlier mentioned was used to predict the nature of the intended surface using surrounding mesh and of certainty that the bridge blends perfectly. The small holes were also filled depending on the surface curve to create a watertight mesh with the surrounding meshes, as shown in Figure 3.6.

3.3.3 Outliers and Disconnected Points

Outliers in a model are produced due to factors such as surface reflectivity, ambient lighting. They are points produced that are not part of the true surface of a model, and they are randomly distributed with a density smaller than that of the sampling density of the model. Figure 3.7(d) below shows that outliers can also be produced in a structured manner as a high-density clustering of points. This behaviour is due to scanning devices having a multi-view stereo system generating points based on view-dependent assumptions, thereby resulting in false consistency. Outliers and disconnected points should not be mistaken for noise, as shown in Figure 3.7(c) either in a categorical way via detection or in a definite manner via robust approaches [139].

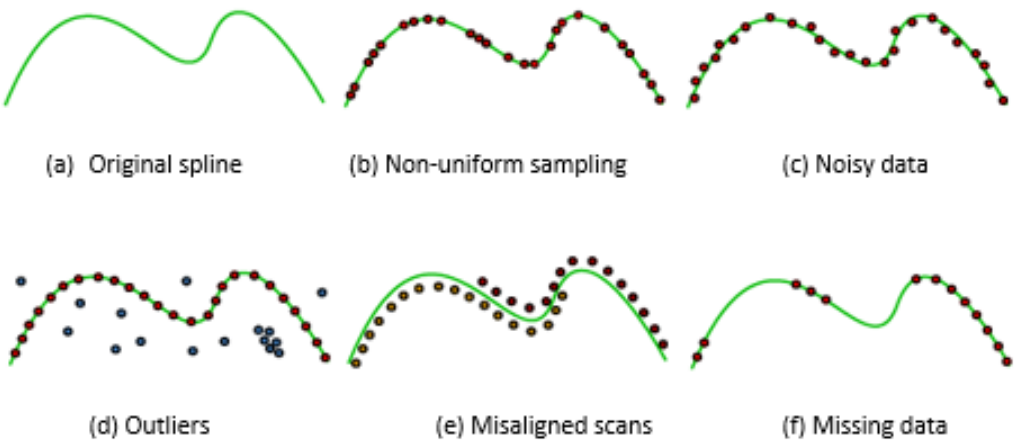


Figure 3.7: Different forms of point distribution in a sampled data. Reproduced from [22]

3.3.4 Clipping Plane for Scanning Operation

To deal with unwanted data in the form of noise and disconnected points, a clipping plane is used to prevent the capturing of the surface on which the part is positioned. This surface is unnecessary in most cases, and a clipping plane hides the points behind the plane from the laser scanner. This process helps in reducing the number of points generated and, at the same time, the memory space required and aids faster processing.

3.3.5 Measurement Data Communication

The preprocessing of point cloud data is usually performed within CAD software. IN the case of preprocessing using a mathematical application, the initial difficulty is the file format communication. This has been an ongoing challenge in the CAD/CAM community and contributed to creating a universally accepted format such as the STEP and STL formats. Once data is exchanged between a CAD and mathematical application, it is observed that some non-coordinate system properties hinder this communication. Data communication between specific CAD software applications has become an ongoing problem for CAD/CAM applications for manufacturing design, reverse engineering a product and for research purposes. It is complicated to use this data in a separate programmer than the producing programme if a similar software framework creates the point cloud data. This is also valid for most CAD systems where communication is complicated between generating software and processing software due to the compatibility of the file format. Most CAD software providers create their own format, which in some cases cannot interact with a different software

platform except it is exported as a generally accepted file format. Because of this, a popular model is known as the STEP, commonly appropriate for many CAD applications, was developed. The Stereolithography (STL) format is another widely accepted file format, and this format also communicates with non-CAD programmes such as mathematical simulation software such as C, C++, MATLAB, and Python. Most users of CAD models or point cloud data occasional performs interaction of CAD to mathematical tools but mainly within the CAD/CAM community.

For this project, we investigated the difficulty of exporting the CAD models to a format acceptable by mathematical tools for preprocessing, edge detection, surface imperfection detection. When built and exported as STL, CAD models are relevant in comparison with binaries for exporting data as an ASCII file because mathematical software has the potential of detecting its vertices. In addition to this, the tolerance for the ASCII file should be increased, providing additional point distribution in a mathematical application for model visualisation. A low degree of tolerance makes it very difficult to process as the points were sparsely distributed. During the experimental stage, scanning was performed, and when the raw data was exported – having XYZ coordinates – the data showed that the scanning process generated certain unwanted lines. Each time the scanner was clicked, sets of organised/unorganised points were generated depending on the specification, and at the end of the experiment, several sets of scans were obtained reflecting each clicking of the scanner, but these scans were put together to give a bigger picture of the scanned object. This implies that for each scan, new sets of points were generated, and the individual scans are separated by a line which is neither a vector nor a coordinate point and thereby, mathematical applications cannot read such data except preprocessing is performed in detecting and eliminating such properties of the data. Although this does not affect CAD applications in any way, the non-vectors or non-coordinate points are unidentifiable by mathematical applications making such data communication unfeasible.

With this challenge, preprocessing of the data is required, and an algorithm for detecting the non-numerical characters was developed. This algorithm functions by first detecting these characters by searching each line of the data and reproduces the data by eliminating the non-numeric character during the reproduction. The reproduced data is then saved as a format readable by mathematical applications.

3.4 3D Feature Identification

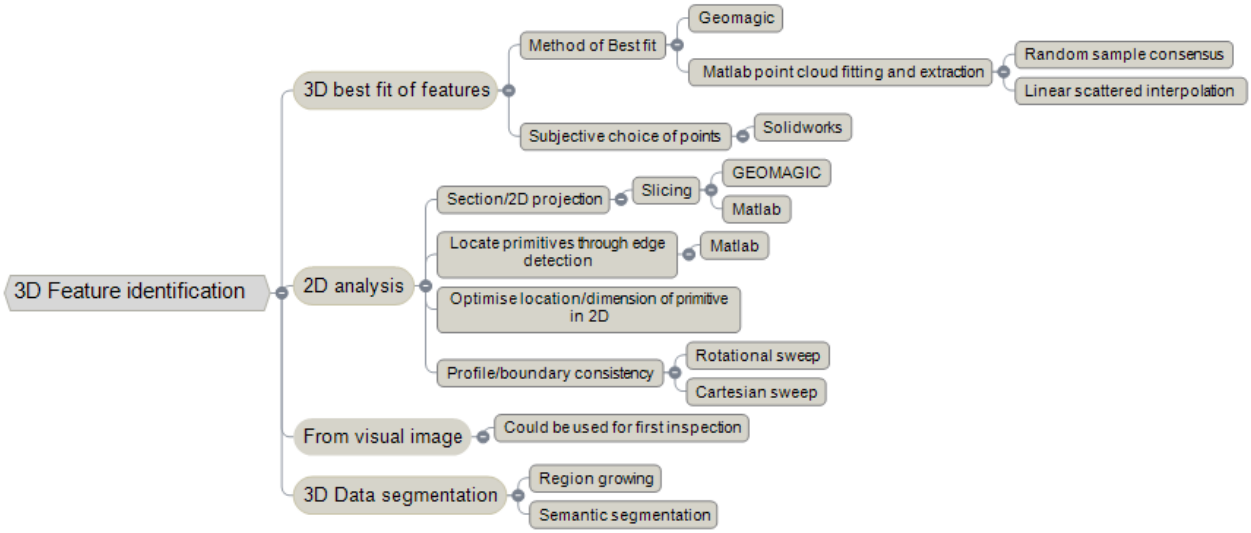


Figure 3.8: Processes of feature identification and extraction

The identification of features is an important aspect when performing surface reconstruction of a model, as complete knowledge of the geometries associated with the model is required. This application as described in Figure 3.8, can be essential for enhancing the integration efficiency that exists in the CAD community and can reduce the time and cost of machine-controlled manufacturing processes. This integration is crucial in establishing proper communication between design (CAD) and manufacture (CAM) for downstream applications like computer-aided process planning (CAPP). Although much of the CAD method is generally understood to be a compositional process of generating geometric shapes from primitives, the CAM processes are regarded as decomposition, which identifies and extracts surfaces and features on models.

3.4.1 3D Best-Fit of Features

The process of performing the best fit of a set of points is an intriguing part of feature identification and extraction. Another name for the best fit is optimal data alignment, and it can be described as the process of exploring the best possible ways for transforming measurement data together with their coordinate system into a polygon model using the best transformation matrix. A good application of best fit in product manufacturing and remodelling requires CAD manipulation where point cloud data is rendered and rotated so it is symmetric with the CAD model. This is usually used for checking product conformity to

specification by comparing the point cloud of the manufactured part to its CAD model, checking for deviations and manufacturing accuracy. This process is known as best-fit alignment in the Geomagic CAD software used for this project. This process manoeuvres an object in sharing physical space with another object, as shown in Figure 3.9.

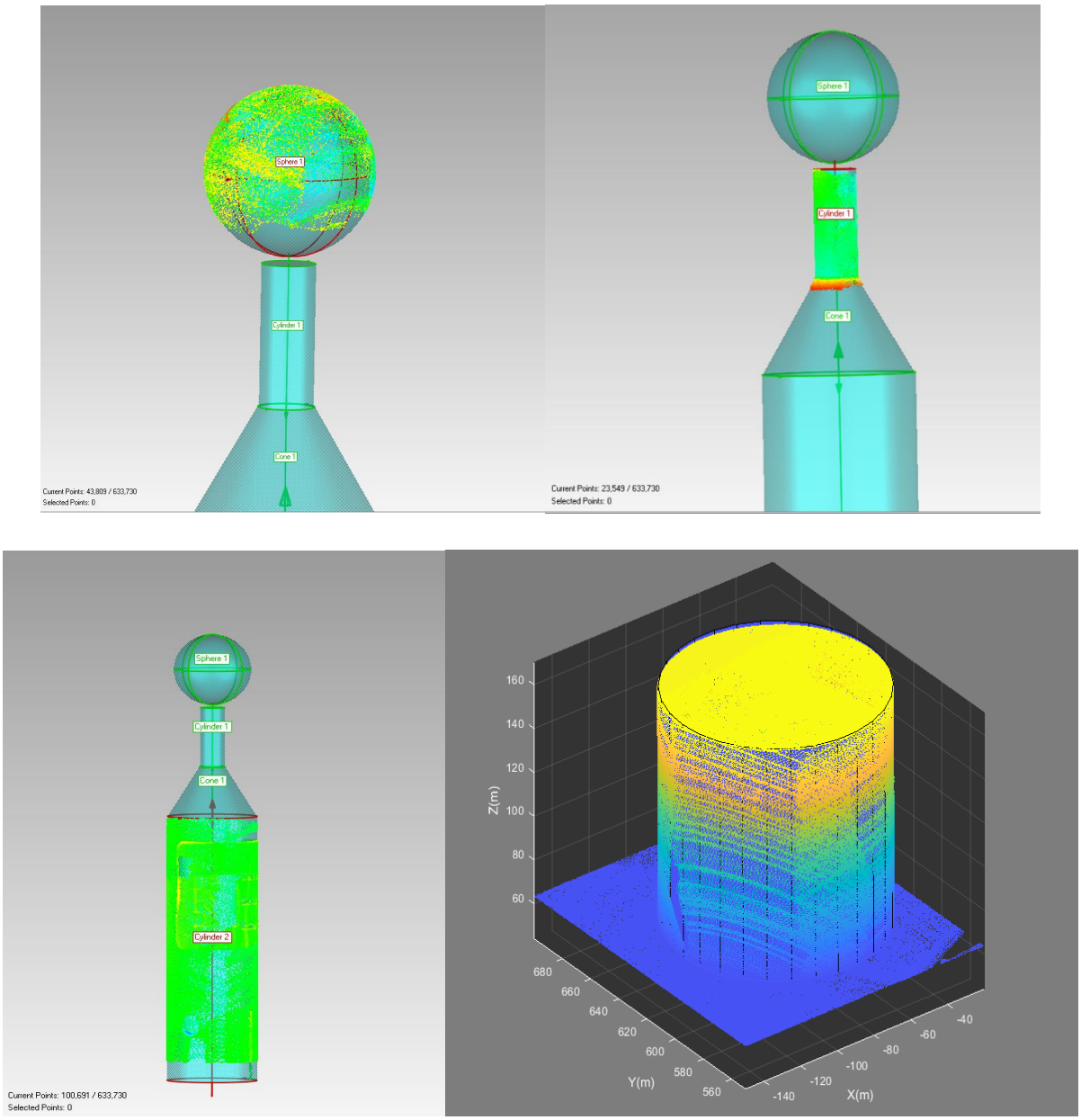


Figure 3.9: 3D fitting to a set of point cloud

The fixed object (reference object) could be the CAD model, and the floating object (test object) – which is the point cloud – is rotated to be correctly aligned to the fixed object. Best-fit alignment can also be performed between two CADs or two-point cloud models and operates in two sections:

- Initial gross alignment where both objects are roughly aligned using a fewer number of points of comparison
- Fine adjustment alignment where alignment is optimised using a much greater number of points of comparison

The number of points of comparison is used in identifying a similar region within a model, and it is important that this is not misplaced as it might result in misalignment of the objects. So once a particular region is selected on the fixed object, it is vital to select an approximate region on the floating object for the best result.

Fitting or aligning points perform Best-fit to a mathematically perfect feature by assessing the selected region, finding the best possible feature inside the region, and aligning all points to a model. A challenge to this function is the selection of points. If the selected region does not correlate with the desired feature, fitting of the points will be attempted, but the result will be undesirable, and this is experienced in situations of manual selection where outlier points or points relating to a separate model might be selected. To further explain this, assuming a distribution of points represents a sphere, and an attempt to fit a cylinder to such points will be undesirable. More recently, automatic feature fitting functions has been developed with good accuracy for aligning points. A practical application of an automatic best-fit for this project was applied to a set of data points with point distribution related to different shapes all in one model. The difficulty faced with such an application is identifying points correlating to the desired feature as the model is composed of several other features. For best practice, the desired feature is specified, and the approximate location of the feature is also specified (referred to as Region of Interest (ROI)). This eliminates the manual process of selecting a region and the possibility of selecting points that might not correlate with the desired.

The results obtainable from performing best-fit includes the model maximum length, average error, and RMS error. The maximum length indicates the greatest distance moved by any point during alignment, and the average error is the average deviation of all points of comparison – the smaller the value, the more precise the alignment. The root-mean-square error shows the error of all points of comparison, and the smaller the RMS error, the more precise the alignment.

3.4.2 2D Analysis

For feature identification, 2D analysis can create a profile view of the model compared to the areal view. Although an areal view is more desirable, viewing a model in 2D can be very helpful in understanding the boundary structure and dimensions and possibly detect any variation in the outer boundary. This is more like a simplification technique for models by creating section planes using nonlinear analyses in situations where geometry and model features are symmetric. A 2D analysis was performed on some of the models used for this project. A profile plot of a cylindrical shape is a 2.5D projection of the model, and when the model is rotated about the XY-axis with the Z-axis is fixed, a 2D profile is produced due to the sweeping function of the rotational matrix. This application presents a 2D analysis of the model used to identify possible damage on the model's boundary. A similar application is creating several slices of the model about the Z-axis with a predefined spacing (can also be done in X and Y axes). These slices set up a 2D analysis of the model from which image processing information can be extracted. Other applications of 2D analysis are in the extraction of surface texture parameters like waviness and roughness in obtaining information such as flatness deviation, profile height, roughness depth, roughness profile depth, arithmetical mean deviation, waviness height and RMS value of the surface roughness.

3.4.3 From Visual Image

From visual image introduces the concept of physical inspections of the part. When analysing the surface of an object, a CAD model is usually developed except for other forms of surface analysis, such as surface texture with respect to surface parameters like waviness and roughness. Performing analysis on the CAD model sometimes requires prior knowledge of the object, which can be done by visual inspection. Visual inspection also equips the personnel with the good understanding required when incorrect results are obtained.

3.4.4 Data Segmentation

Segmentation is performed mainly on dense triangular meshes generated from scanned data. Triangular meshes are usually optimised, having non-uniform vertex distribution where big triangles represent flat or almost flat regions, while many small triangles clustered together represent highly curved regions. These are unwanted surfaces as they appear as spikes and are due to incorrect filling of holes.

Segmentation can be described as the process of creating subdivisions of a model into a set of regions that do not superimpose on each other and, when combined, produce the entire model. These sets of regions correlate to objects and their important parts. The input to this system is usually a preprocessed image with an output that depicts the regions found within the model. Once the segmentation is done, the regions can be represented, analysed, described, and classified using feature extraction and visual pattern recognition methods. Segmentation algorithms are mostly developed based on discontinuity and similarity of features or a combination of both systems. Although these algorithms are implemented with a common goal, they differ in image type (such as binary, gray, colour), mathematical structure (such as morphology, statistics, graph theory), features (intensity, colour, texture, motion), and appropriate technique (top-down, bottom-up, graph-based) [140]. Segmentation algorithms have been organised according to [140] as follows:

- Intensity-based methods are dependent on how pixels are positioned, and a good example is thresholding. These methods are also known as non-contextual methods.
- Region-based methods rely on the condition between a pixel and its neighbours in terms of adjacency and connectivity. They are also known as contextual methods. Examples include region growing and split and merge.
- Other methods are based on texture, edges, and motion.

The intensity-based methods are the most straightforward methodology for performing segmentation, and they require the pixel statistics of an image in the form of a histogram. They use these statistics in determining what pixels should be grouped like objects and what pixels be labelled as background, and the most used method in this category is the thresholding method (image, global, optimal, and local). Thresholding is widely used in image processing for its intuitive properties, simplicity, and ease of implementation. It gives better control of how the application performs segmentation. The region-based method does not consider pixels as part of an object, and it considers the relatedness amongst pixels in determining which are of the same region (belong to an object) or pixels that are not.

3.4.4.1 Region Growing

Region growing is a segmentation process of grouping models of similar geometry properties into a specific group. It is part of the region-based method of segmentation. The aim is to

segment features into meaningful regions and, by meaningful region, is described as a region with specific geometric properties common to all models. These properties could be surface curvature and normal, geometric entities, features, and a similar structure in terms of features (vertexes and edges). For image processing, this method uses a bottom-up approach starting from a pixel and grows a region that encompasses the image provided the newly developed region satisfies a homogeneity criterion. The key factors for performing region growing include the choice of similarity criteria, the selection of seed points and the definition of a stopping rule as stated by [140].

3.4.5 RANSAC Estimation and Linear Scattered Interpolation

Random sample consensus (RANSAC) and linear scattered interpolation (LSI) were two methods of feature detection and extraction investigated in this project. There are many features detection and extraction methods from research on automatic feature recognition, but the reasons for choosing these two systems will be discussed further in chapter 5. The RANSAC method performs the extraction of shapes by creating marginal sets from the point cloud and constructs primitive shapes from these sets. RANSAC aims at defining captured points based on two input data classifications, the outliers and inliers. They function by specifying the region of interest that contains inlier points representing a specific primitive. The LSI method uses an interpolation function in determining the properties of an irregularly spaced data point in terms of differentiability, continuity, and smoothness. The LSI is one of the three interpolants, and the linear interpolant was preferred for its speed and ease of application on primitive shapes. Both methods were compared in terms of their efficiency in prediction accuracy and speed.

3.5 Identification of Areas of Damage

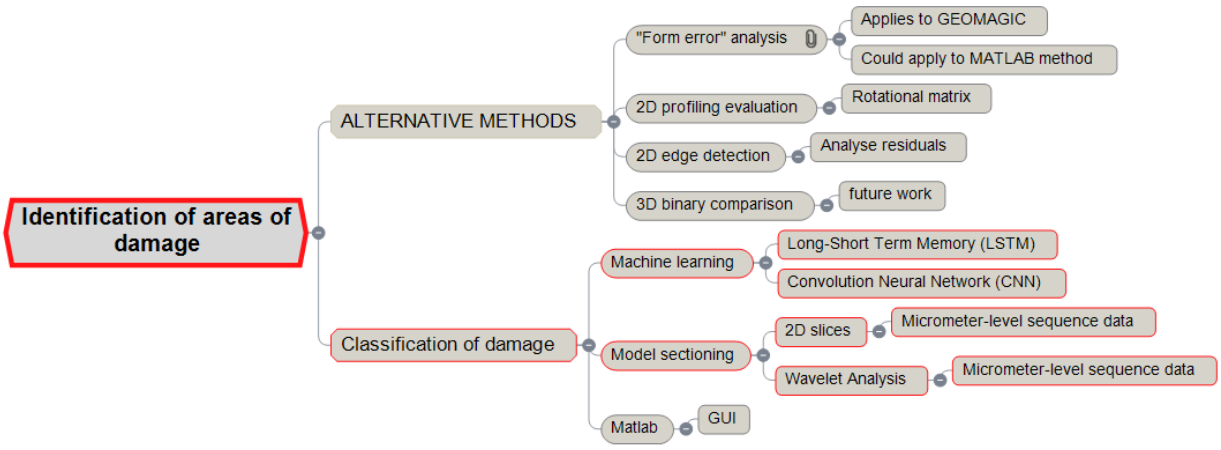


Figure 3.10: The concepts and proposed methods for detecting damage

An essential part of reconstructing a potentially damaged surface is the identification of damage present on the surface, and this process is illustrated in Figure 3.10. It presents alternative methods to achieve the identification and classification of slices from model sectioning. The identification is quite a complex analysis regarding determining what actual damage is and what is not, and what type of damage are we dealing with? What appears to be a damage or surface imperfection according to ISO 8785 might be part of the design intent or actual damage, and how can we evaluate this? These questions become highly compelling when performing a 2D analysis of the whole model; although 2D provides vital information, it might be challenging to ascertain what damage is and its intended design. A simple but not the most efficient way to dealing with this challenge is by visual inspection of the part or having prior knowledge of what the part looks like. As visual inspection is not efficient but important, it has become necessary to develop an accurate and efficient method for detecting potential damage automatically. Also, the classification of these damages in relation to causes and could be due to machining error, scratch, dent, or deformation due to several reasons. Although it is not left to us to determine what inflicted the damage, an understanding of what type of damage we are dealing with is an important part of this process.

To identify areas of damage, this project employed a 2D analysis approach compared to 3D visualisation because, with 2D, the boundary structure of the surface can be obtained by creating a profile view of the whole model. Using this profile, a new application using a

rotational matrix was developed. Although the rotational matrix is not a new concept, it has not been used on profiles for detecting damage. Another approach was creating several slices of a model with a predefined distance to produce 2D plots from which damage can be seen but by visual inspection. Moreover, how do we ascertain that it is actual damage or a design intent feature? This will be discussed further in chapter 6 when feature profiles are extracted.

3.5.1 "Form Error" Analysis

Form error is an analysis of point cloud data when performing feature fitting. In feature fitting, a geometric feature is fitted to a point cloud using the average number of inlier points representing the specific geometry. These points are generally selected, and the appropriate geometry is fitted to the selected points but not without the outliers. The fitting function uses as many points as possible that fall within the model's dimension, but outliers can be seen to influence the form error of the function, although the form error value does not affect the fitting function.

3.5.2 2D Profiling

Profile monitoring is used in describing the boundary structure of a machined part with the identification of parametric models [141]. The machined part boundary can be described by parametric models, which can be differentiated by predictable behaviours of the model having natural variability. These predictable behaviours are known as the manufacturing signature, which provides a profile image of the machined part, and the manufacturing signature is the standard pattern that differentiates all process-made features. They are used to plan appropriate profile monitoring procedures. A parametric model known as the smoothing spline for profiling monitoring was used in work by Gardner et al. [142] and William et al. [127]. Non-parametric forms of estimating profiles were also used by the latter [127].

Eric, et al. [128] argues that the use of spline belongs to a wide range of non-parametric estimators with fewer constraints than parametric methods with undesirable properties but can provide a feasible alternative to parametric models. A spline smoothing estimator does not provide a good model for a profile with specifically unsmooth characteristics, such as jumps or points of non-differentiability [128]. The information obtained from point cloud profiling is used in performing other applications such as analysis on the machined parts to 3D reconstruction and extracting the contour lines in an image.

3.5.3 2D Edge Detection

This is an important part of image processing as the edges in an image represent the outline of objects or features within the image. They are primarily used in boundary detection and feature recognition. Given a grayscale or textured image, the edge detection algorithms convert them into a binary form that produces gradient images and, in some cases, edge maps, which gives information of the original image content.

3.5.4 Point Distribution

This has to do with how points obtained from a scanning process are positioned or spaced. This positioning or distribution depends on the surface type, how light is incident on the surface, and how the scanner's camera captures points. For example, a machined part with a surface finish with the spindle movement seen will produce a different point distribution than a finely finished surface, cast, or 3D printed surface. Sometimes, points are captured as a cluster, while others are sparsely spaced, and some overlap with others. Occlusion of the range of view of the laser line can also result in a different form of point distribution and the effect of light on the surface. The reflectivity of the surface is also a factor that directly affects how points are distributed. When a surface is scanned, the cloud points are distributed such that the points correspond to the surface, which is known as point distribution given a shape.

3.5.5 Long-Short Term Memory

Long-Sort Term Memory (LSTM) is a Recurrent Neural Network (RNN) with connections to the previous state, enabling it to retain previous information to perform classification. Sequence data are examined in context and coupled with the knowledge of the previous data; the network can make an improved association when making decisions.

3.6 Dimensioning of Identified Features

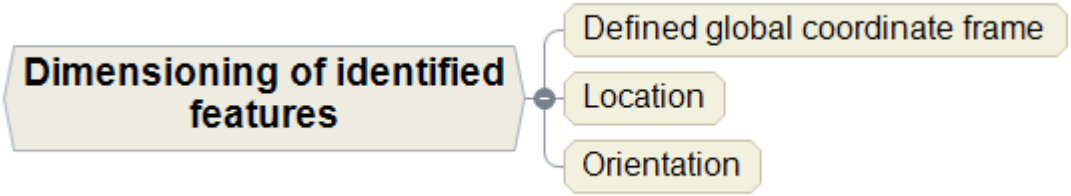


Figure 3.11: Proposed techniques for feature dimensioning

The identified features are actual objects with dimensions, and when their 3D information is obtained, it is expected that the 3D models have similar values to the real objects. When feature recognition is performed, the extracted feature is measured to compare its dimensional values to that of the real object. The closest dimension of the 3D model to that of the real object indicates how accurate the algorithm performs extraction and the accuracy of the scanning process. The proposed technique for performing dimensioning is shown in Figure 3.11.

3.6.1 Location

As a geometric application, the location of features within a dataset is an important piece of information, mostly when performing feature extraction. For RANSAC and LSI algorithms, where a dataset contains several other features, to recognise and extract specific features, the location must be detected, termed the region of interest (ROI). This location information helps the algorithm constrain the search, making it fastest as the algorithm tries to identify points that correspond to the desired shape. Not having this information will produce an undesirable result where the system might take a long time because it searches for corresponding points or fits features to the whole model.

3.6.2 Orientation

This has to do with how the axes change with respect to each other. A slight change in the angle of an axis will give a different result as to what is required. For example, if a cylindrical model is plotted in 3D with all three axes at 90 degrees to each other when carrying out 2D analysis such as slicing, it is expected to obtain a perfect circle when sliced in the Z-axis. Assuming the model is reoriented slightly from 90 degrees – either more or less than 90 degrees, the outcome of the slice will be an oval shape compared to a circle. This analysis is important for the integrity of the data generated from the slices for possible use in machine learning. Having to produce several 2D slices with non-uniform centralisation will be detrimental to the quality of the result obtained, defeating one of the objectives of this project.

3.7 Intelligent Dimensioning

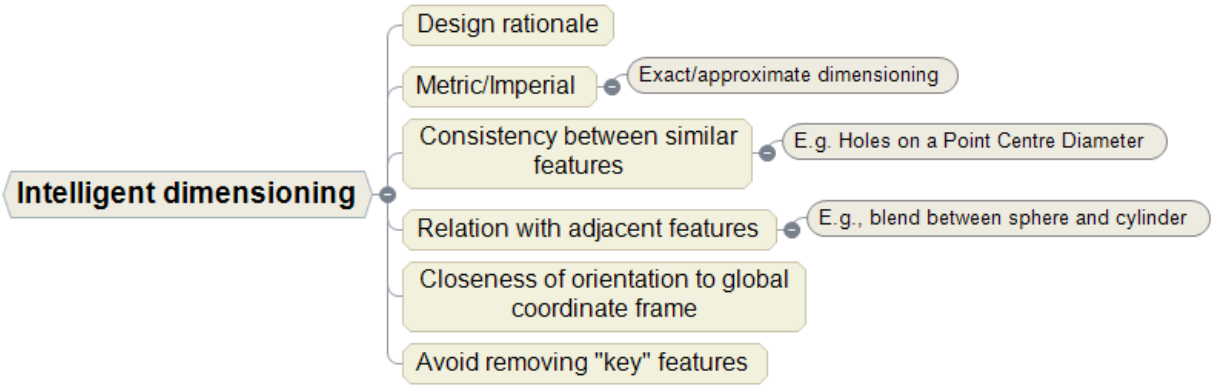


Figure 3.12: Proposed methods for investigating extracted measurement information

Intelligent dimensioning was proposed to investigate the concept of design rationale that explores the explicit reasons behind design decisions. Also investigating unit of measurement for extracted dimensions – are dimensions exact? And if approximated – are dimensions rounded up to the nearest whole number value while examining what unit of measurement with respect to either metric or imperial units. This approach should explore the relationship with adjacent features in freeform or complex shapes as shown in Figure 3.12. For example, the transition of geometric properties where a sphere sits on a cylinder or a cylinder on a cone. Extra care is required to maintain a good level of point density, avoiding the removal of key features during data processing.

3.8 Generate Design Drawing

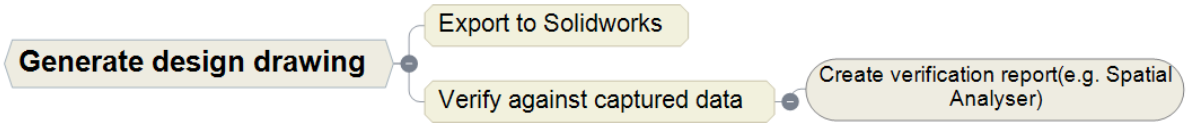


Figure 3.13: Generating design drawing for performing reconstruction

Figure 3.13 illustrates the proposed approach of generating a design drawing of a model. This approach could employ both contact and non-contact methods as described in the data acquisition of section 3.2.1. Design drawing can be generated by meshing point cloud data in

Geomagic and exporting it to Solidworks to produce a design drawing, as shown in Figure 3.14.

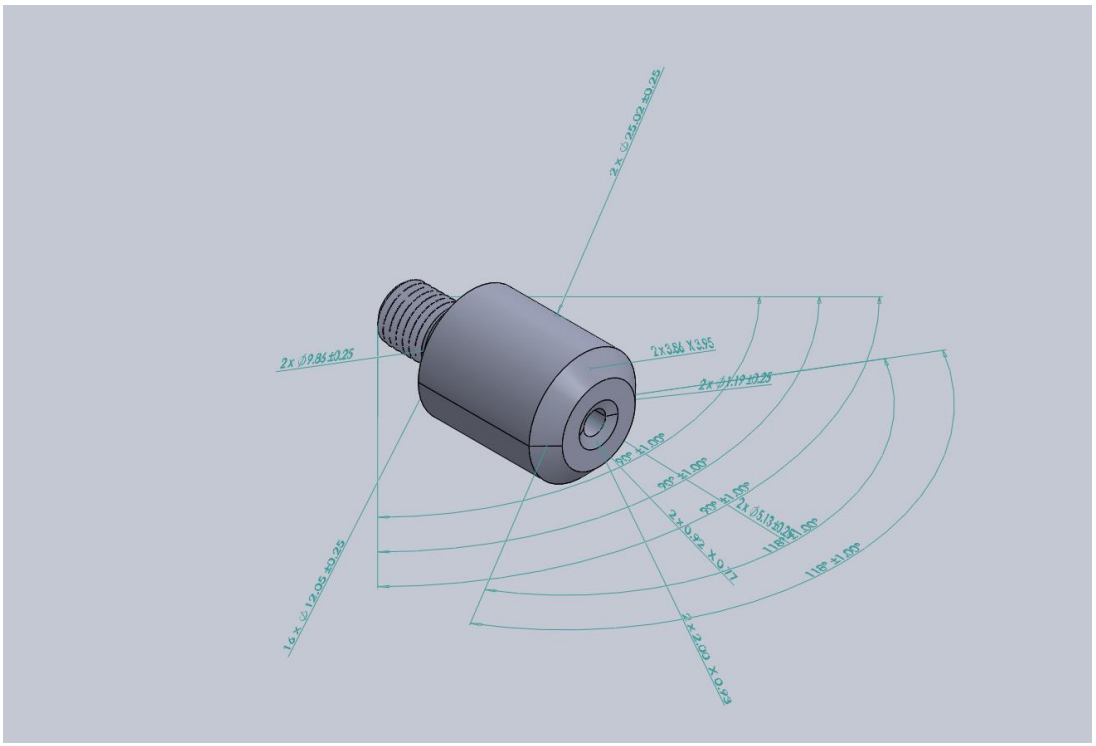


Figure 3.14: Solidworks design and dimensioning of a model

3.9 Measure And Verify Reconstructed Artefact

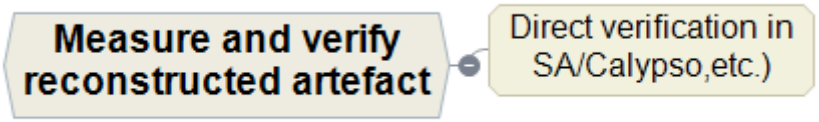


Figure 3.15: Verification of reconstructed surface

The RE/SR framework includes verifying the idea at the end of the process against the manufactured part using some of the methods suggested in Figure 3.15. The verification certifies that the initial idea behind the design correlates with the real part in terms of dimensions, functions, and appearance and highlights maximum deviations to ensure an accurate digital representation of the real part. An exception to these factors is when a part is modified, remodelled, and improved, as RE can improve the functionality of a part.

3.10 Summary

This methodology chapter described the methods in the proposed SR/RE framework for achieving the aims and objectives of this project. The framework was developed using a mind-mapping application indicating areas of concentration, methods that have been implemented, and alternative approaches to the proposed methods. Also, methods of further works to achieving the final goal of performing reconstruction and verification such that the reconstructed model is a direct representation of the real part.

Chapter 4 Edge Detection and System Optimisation for Image Processing

This chapter investigated the application of edge detection algorithms for the identification of edges within the frame of an image. A description of the edge detection concept was presented in the literature section, but this section will concentrate on using the Canny edge detection operator.

As shown in Figure 4.1 below, is a situation where the ROMER Absolute Arm CMM was used to scan over an object (Honda engine cover) to collect point cloud data from its surface in X, Y, and Z coordinates using an ultra-wide laser strip of up to 150 mm and an integrated RS4 laser to capture 752,000 points per second. Being a part with a combination of both freeform and geometrical shapes, the concentration for this application on the geometrical shapes, and this has a diameter of 236 mm for the outer cylinder. PC-DMIS was used to automatically capture the sampled points of the object, which is then projected as a 3D model as the laser line runs over the surface. As the laser line scans over the surface, the system has an automatic stitching function that captures points continuously as it is aware of the object's position in the coordinate frame.

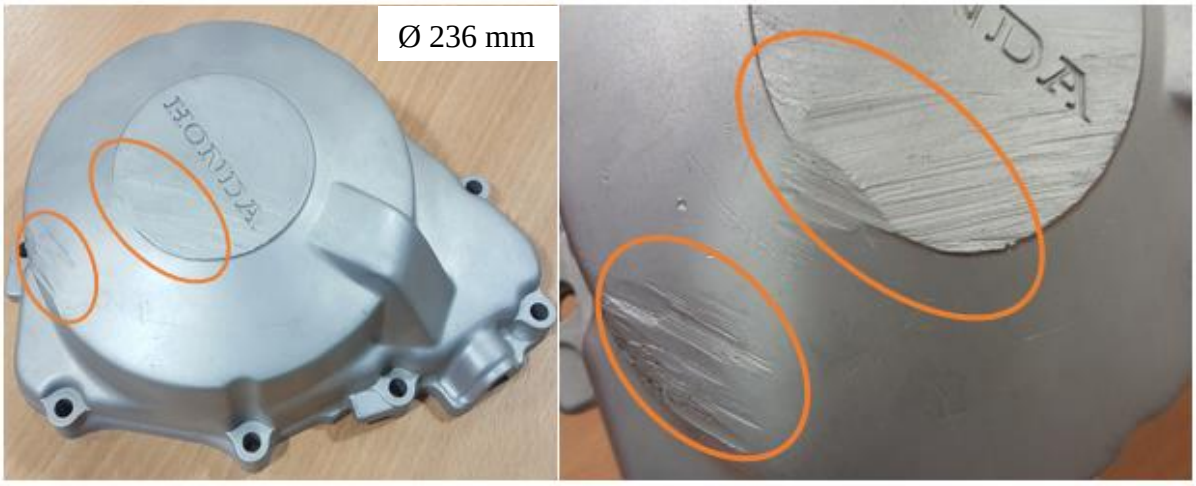


Figure 4.1: Sectional view of a damaged object with a cylinder positioned on a cone showing the intensity of damage creating a blend between both features

Using GEOMAGIC STUDIO 12 for reconstruction, the 3D model was translated into a CAD module. When imported into GEOMAGIC, the point cloud data was captured as XYZ-file, cleaning up was

performed on the surface before translating into a wrap file (polygon mesh). Here the CAD module is translated into its Delaunay triangulation. The software helps to restore boundaries and fill holes with a little geometric forecast, but the process requires huge human influence. When digitising has been performed and reconstruction of the Voronoi, the circled region was very challenging for reconstruction using the Voronoi of the surface points. This commenced the investigation of edge detection and automatic feature recognition in an attempt to create a boundary between both geometries (cylinder and cone). The initial damage to the object can still be seen after reconstruction, as shown in Figure 4.2 below. The damage on the side was successfully reconstructed, while edge detection and feature recognition are needed on the top. This also depends on the degree/intensity of the damage.

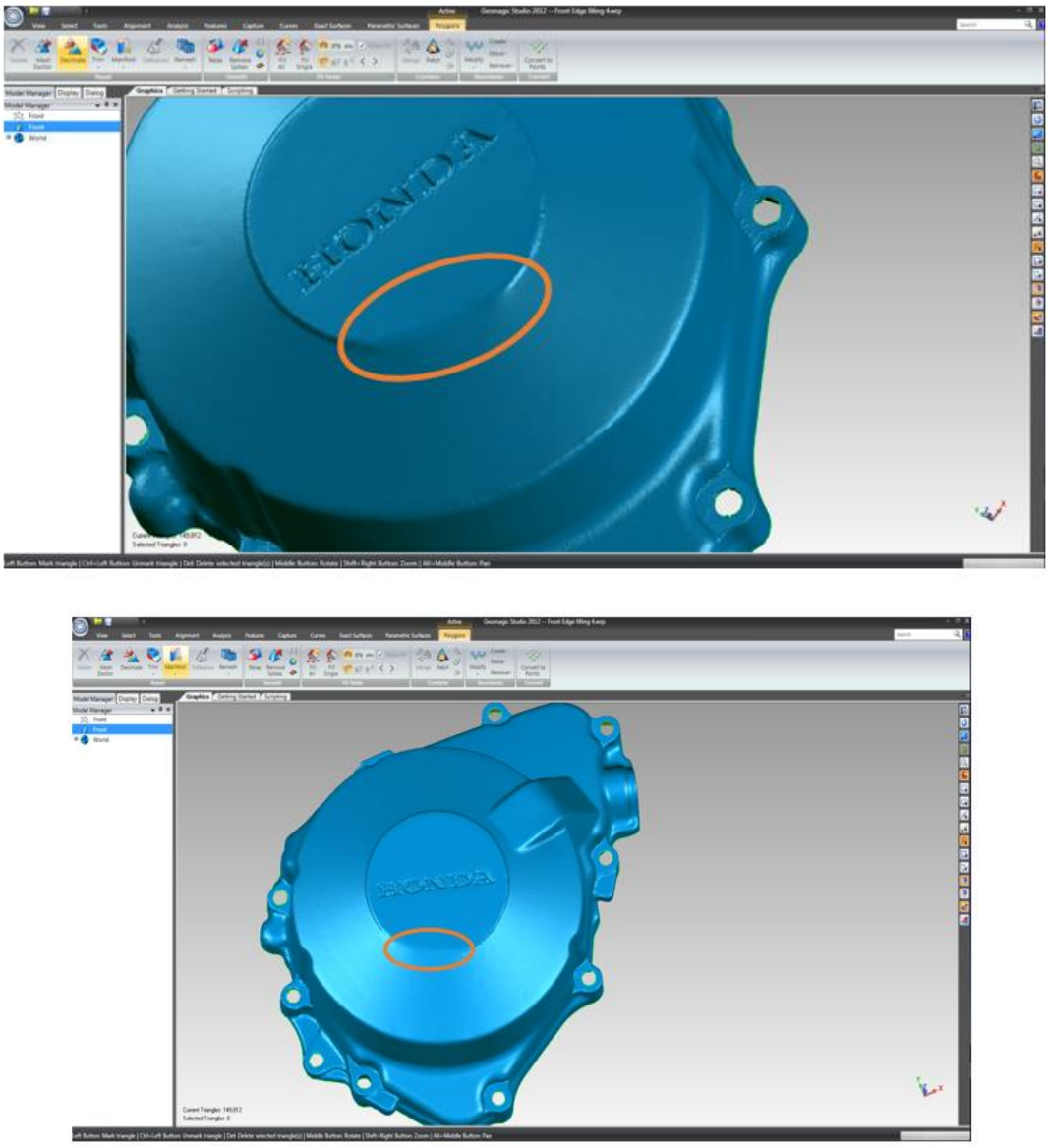


Figure 4.2: CAD module of the scanned object

4.1 Canny Edge Detection Operator

Figure 4.3: The Canny edge detection operator process

The Canny process of edge detection, as shown in Figure 4.3, begins by first thinning of the edges within an input image to the size of a pixel (one pixel wide) because it is more concerned with the location of the edge and not how thick the edges are. It then uses a hysteresis thresholding method known as two-level thresholding. Considering Sobel's orientation, an assessment is done to determine if the edge is a local maximum for every pixel (meaning it is more significant than its neighbours). For example, examining a single pixel x in the image shown in Figure 4.4 below, the Sobel X and Y operator is performed on it to produce gx and gy, and the magnitude of the edge is computed together with the magnitude of its neighbours to determine if x is bigger or smaller. Canny takes the information from the orientation of the edge based on the output of the Sobel operator $\arctan(gx/gy)$ to determine if x is more prominent than its neighbours. This is applied to the whole image, which produces thin edges, and the obtained information is vital to the peak of the centre of the required response. The mask convolution used was the Sobel vertical derivative as its output is the input to Canny and this can be found in Table 2.6.

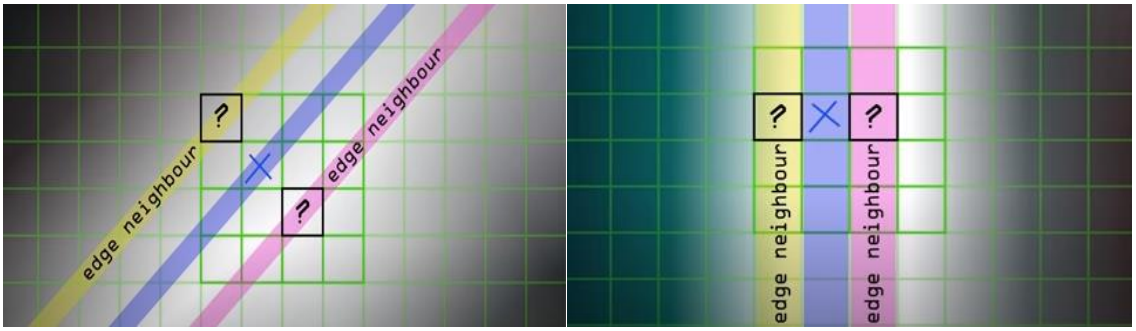


Figure 4.4: Showing how the local maximum of a pixel relates to its neighbour in both X and Y directions. Reproduced from [90]

The second stage is to create an image with dominant edges ignoring other factors around it, which could be an edge not up to the local maximum as they produce weak responses and appear as noise. A two-level hysteresis threshold is used to determine what edges are important and what edges are not. For example, in a threshold range of 0 to 255, with 0 as no edge and 255 as the strongest possible edge, the task is to determine what value is considered good from this range. A low value could produce edges with the weak response having lots of noise, and a high value could produce very sharp discontinuities discarding other important edges connected to the good edge but fall below the threshold.

Any edge above the top threshold is defined as a good edge, and any edge below the bottom threshold is automatically discarded, and any edge between both thresholds and is linked with the edge above the top threshold is acceptable, as illustrated in Figure 4.5. This preserves the core edges required for image processing applications. The output from the Sobel is taken as input to the Canny operator, thereby making it better or produces valuable information for image analysis. Like most operators, the input image is converted to grayscale to identify boundaries due to clear variation in the pixel of the edges when in grayscale.

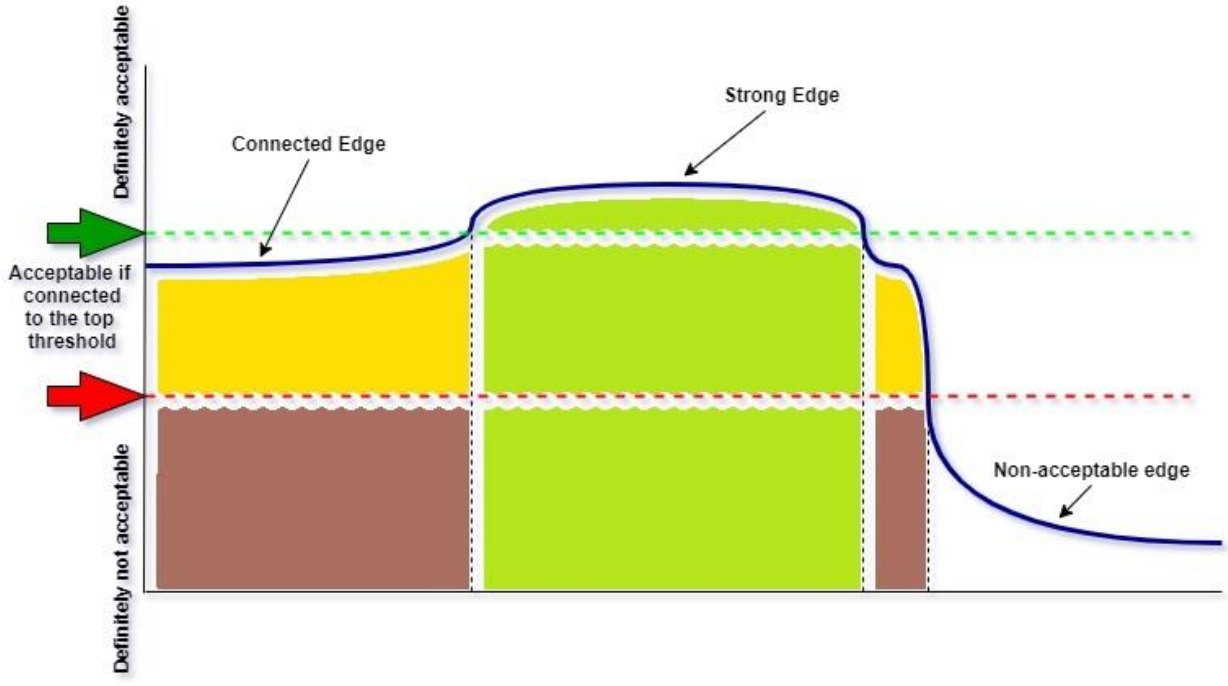


Figure 4.5: Hysteresis thresholding showing both acceptable and non-acceptable edges. Redesigned from [90]

The Canny algorithm is focused on an optimal edge detector provided some parameters are met. These parameters include classification of most edges by decreasing the error rate, maximising the localisation of an edge by labelling edges in proximity to an actual edge, and edges are labelled a single time if a single edge is present for a minimal response. The optimal detector that meets these parameters can be approximated using the Gaussian function as stated below

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{4.1}$$

$$\frac{\partial G(x,y)}{\partial x} \propto x e^{\frac{-(x^2+y^2)}{2\sigma^2}} \quad \frac{\partial G(x,y)}{\partial y} \propto y e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{4.2}$$

The image is smoothed by convolving with a Gaussian filter, and the gradient is obtained by passing the result of the image smoothing through a convolution operation to produce a Gaussian derivative in both directions. The convolution operation is expressed as

$$I'(x,y) = g(k,l) \otimes I(x,y) = \sum_{k=-N}^{N} \sum_{l=-N}^{N} g(k,l) I(x-k, y-l) \tag{4.3}$$

Given that $g(k,l)$ is the convolution kernel, the original input image is $I(x,y)$ with its filtered form as $I'(x,y)$. The size of the convolution kernel is expressed as $2N+1$. Both Gaussian mask and their derivatives can be separated, meaning the 2D convolution can be simplified [94].

Generally, the assessment of Canny algorithm performance densely relies on the adjustable parameters σ and threshold limits, where σ represents the standard deviation for the Gaussian filter and controls its size. A greater value of σ is directly proportional to a greater value in the size of the Gaussian filter, and this means that more blurring of the image is required for noisy images, and more prominent edges can be found. Inversely, a smaller value of σ means a smaller Gaussian filter, thereby restraining blurring and keeping very fine edges. These parameters can be adjusted to adapt to various environmental effects having distinct noise levels. It is supposed that the bigger the Gaussian noise, the less precise the edge's positioning is [94]. The thresholding application is shown in Figure 4.6.

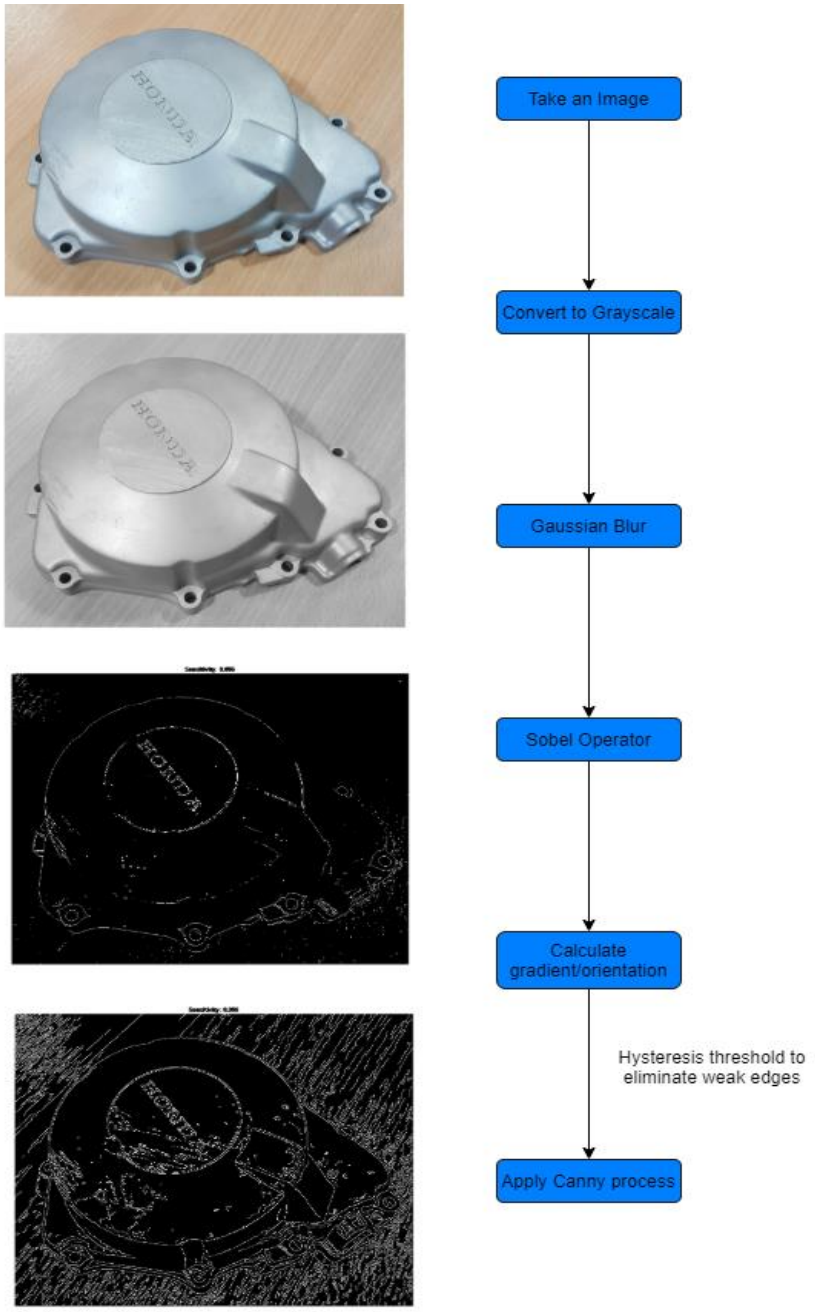


Figure 4.6: The Canny process of edge detection using hysteresis thresholding from the input from Sobel

4.2 Python Opencv for Edge Detection

The reconstruction process of the CAD model presented in section 4.1 was quite challenging as the damage at the top of the model was intense, and it was difficult for the system in predicting the geometry. A cylinder sits on a cone-like geometry from the model, and with the huge damage, there is a blend between both features. Several edge detection algorithms were implemented in python OpenCV with a live feed camera in detecting the edge around

the damaged section. The edges within the view of the camera were detected as the visualisation progressed. The presence of a light source creating different light intensities can either make the system lose some edges or make some edges more visible resulting in the detection of non-edges. The non-edge detection is an excellent example of the second criterion of Canny to good localisation of edge points resulting in a reduced distance between the detector and centre of a true edge. The sensitivity of Canny is shown below in Figure 4.7 through to Figure 4.9 as actual edges are detected by returning a single point response.

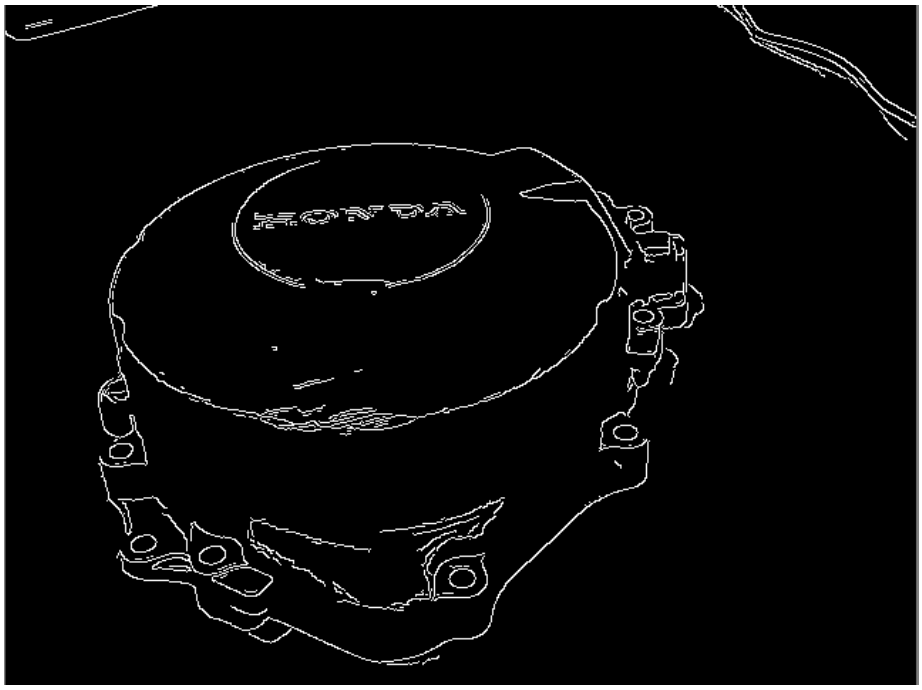


Figure 4.7: Canny Edge detection operator from a live feed camera

Figure 4.8: Canny edge detection with the effect of an external light source from a live feed camera

Figure 4.9: Canny edge detection with sensitivity to noise from a live feed camera

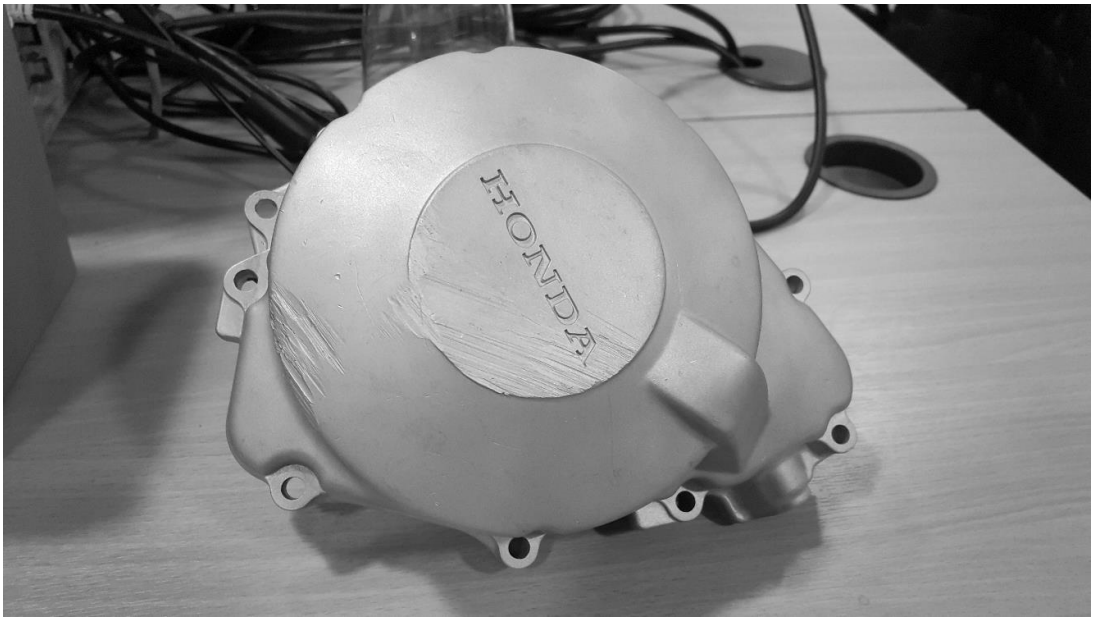


Figure 4.10: A captured image of the damaged object

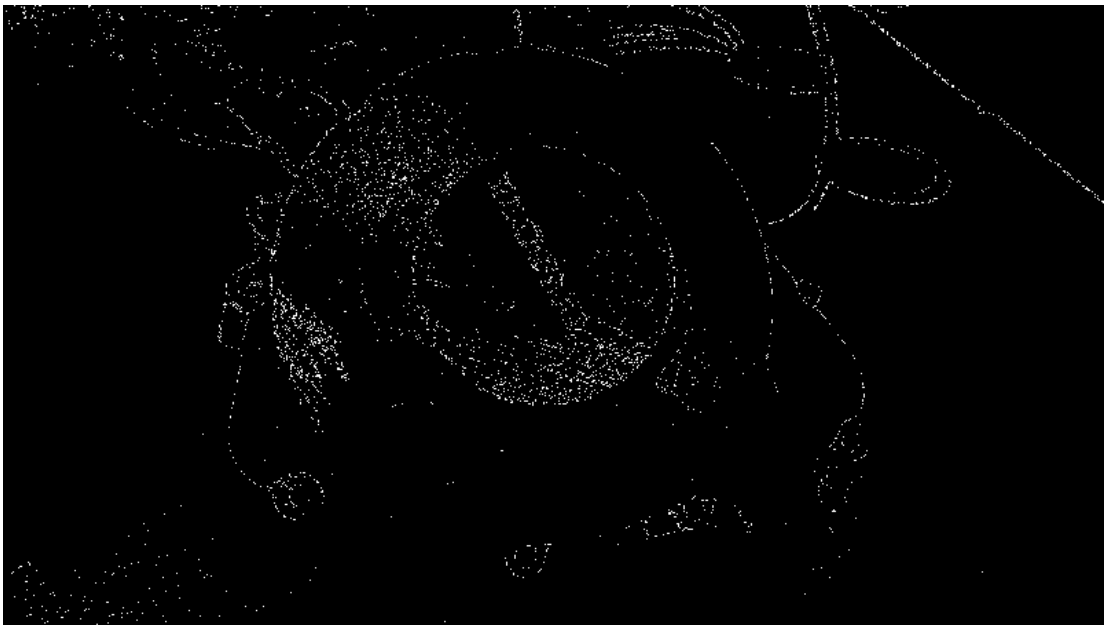
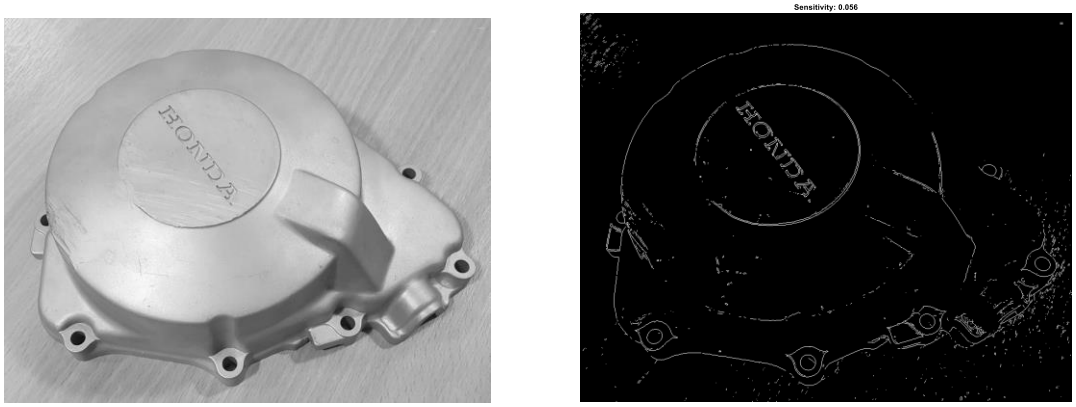


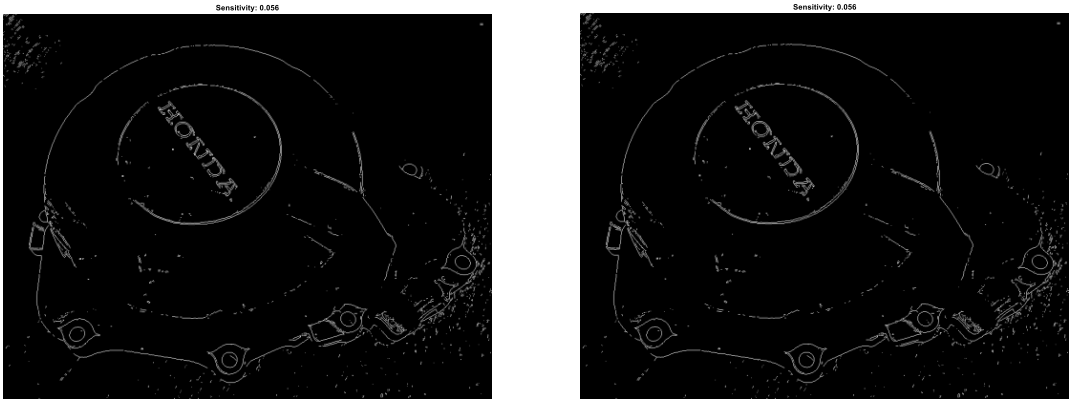
Figure 4.11: Canny edge detection algorithm of a captured image of the damaged object

The image in Figure 4.10 is the captured image of the damaged part, and this image was used in MATLAB for performing edge detection using the Prewitt edge detection algorithm. Although some edges were not clearly detected, as shown in Figure 4.11, the detected edges appear as an array of points distributed along that edge part created by the difference in intensity level in the image. The missing points are the undetected edges resulting from a very

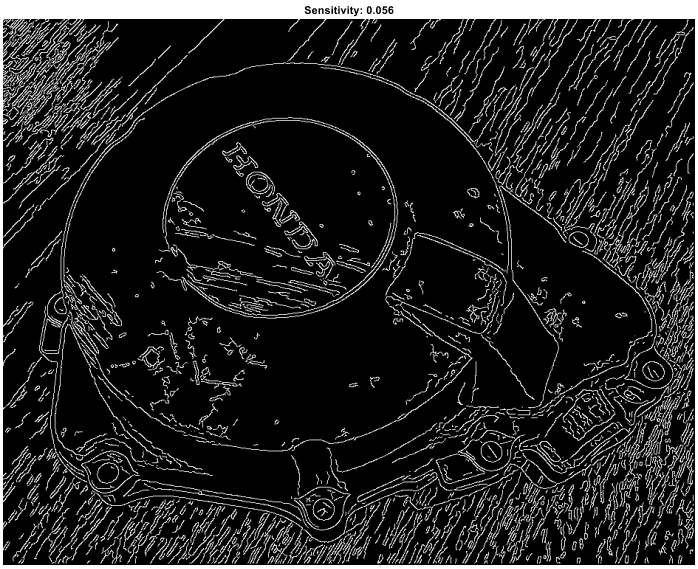
low-intensity variation that cannot be detected with low sensitivity algorithms. In this case, the lost edges are because of the shadow created by the effect of a light source. Using the image of Figure 4.10, further investigation was performed using other edge detection algorithms, as shown in Figure 4.12. Edge detection by first-order derivatives such as Robert, Sobel, and Prewitt produced very similar output responses with less sensitivity, thereby losing some edges. The sensitivity of the Canny algorithm can be seen in Figure 4.12(e), where the system detected all possible edges alongside the surface on which the object was positioned, as shown in Figure 4.12(e-g). Some elements on the surface of the object were also detected as noise, and this is due to the surface texture, as very unsmooth surfaces could create an intensity easily identifiable by Canny as an edge which explains its sensitivity to noise. The noise was taken out when the object was placed on a supposed smooth surface, and the edges could be clearly defined, as shown in Figure 4.13.



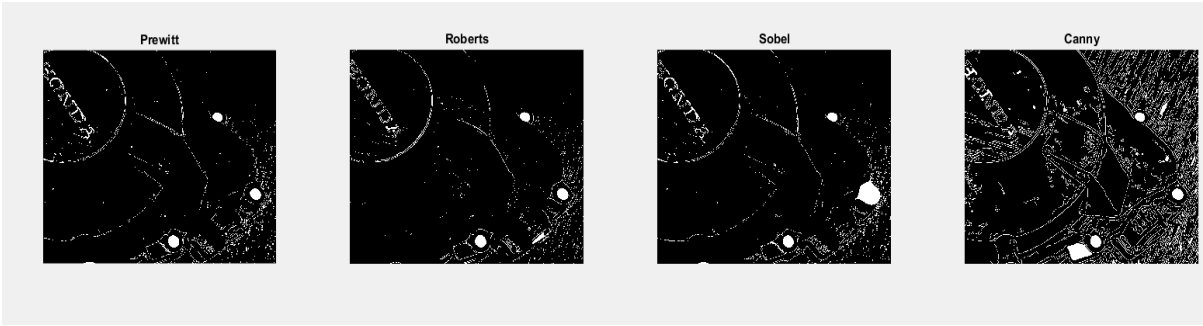
(a) (b)



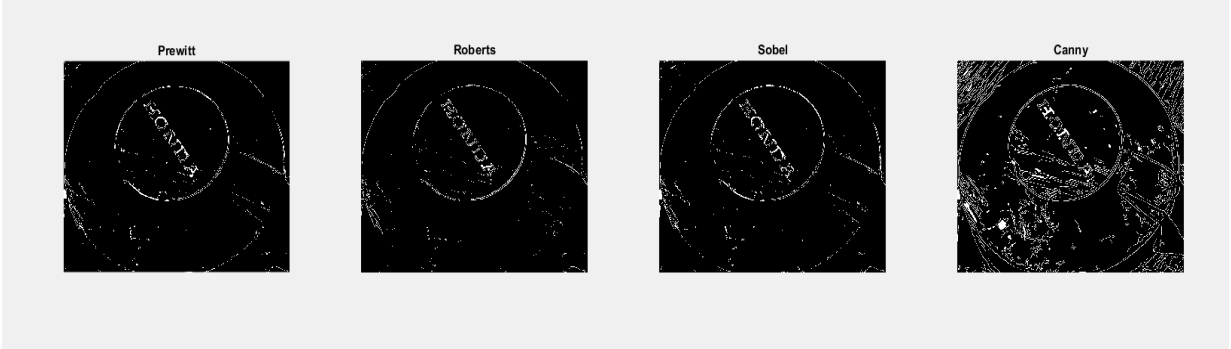
(c) (d)



(e)



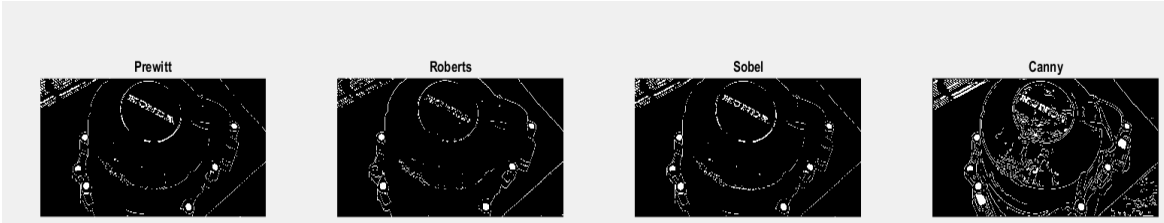
(f)



(g)

Figure 4.12: (a) show shows the captured image of the damaged part (b) edge detection using the Prewitt algorithm (c) edge detection using the Robert algorithm (d) edge detection using the Sobel algorithm (e) edge detection using the Canny algorithm (f) comparison

(a)



(b)

Figure 4.13: Edge detection on a smooth and non-reflective surface

4.3 System Optimisation

The optimisation of systems demonstrates that process-controlled operating systems can be enhanced. An optimisation is improving the productivity or output performance of a process, product, system, or procedure. This process could make a system more robust, stable, better performance response and accuracy, and reduce running time and cost [143]. Optimisation can be based on theory, algorithm, or application. Based on this classification, there is a relationship between theory and algorithm, and likewise, a relationship between algorithm and application. Optimisation determines the optimum and minimum of a deterministic mathematical function, sometimes having independent variable restrictions that could influence optimisation algorithms' performance. These variables could be several decision variables, constraints, and features of both the objective function and the constraint. The objective function is described as a function of the design variable.

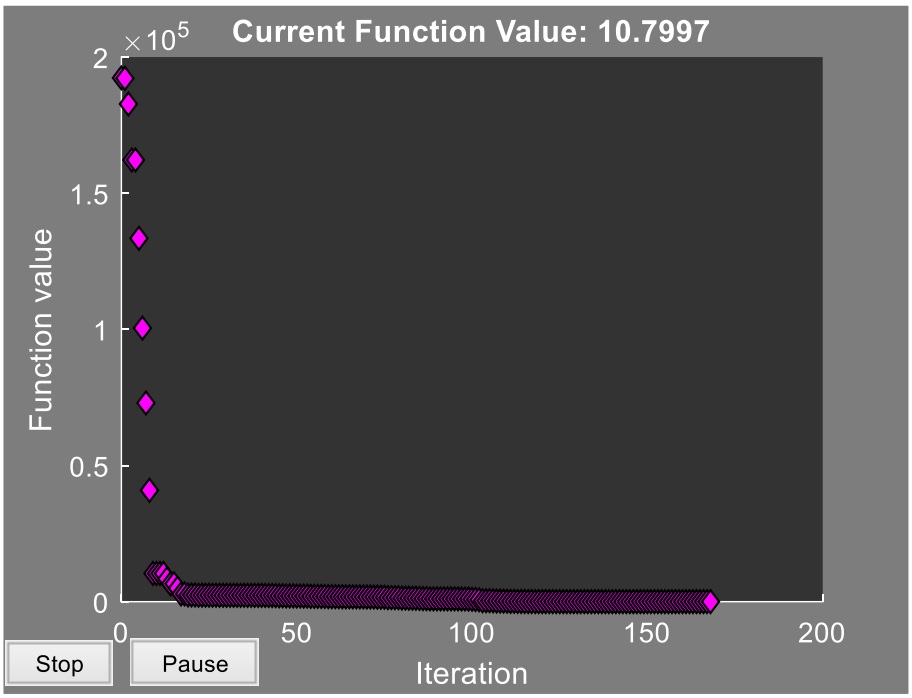
Several problems can arise during system operations, and most of the issues can be modelled as optimisation problems. These problems are solvable by creating a plot of the objective function to find the least constrained or unconstrained variable function. Optimisation problems are classified as constrained and unconstrained optimisation problems. The Linear Programming Problem (LPP) is considered an optimisation problem if its objective function and constraints are linear functions of the design variable. These problems are found in industrial sectors with a constant desire to improve the performance of existing or proposed processes. To create a solution for optimisation problems, certain factors such as numerical performance, robustness and ease of computer implementation are analysed to obtain an efficient algorithm. These are necessary when developing software for optimisation algorithms with efficiency and reliability.

4.3.1 Simplex Optimisation

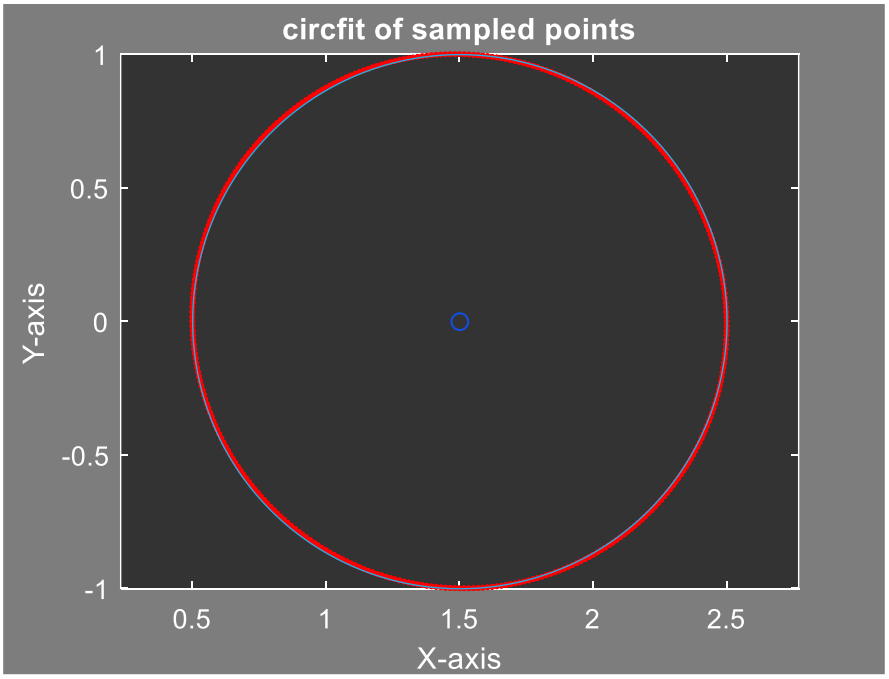
4.3.1.1 The use of FMINSEARCH for System Optimisation

An optimisation is described as a mathematical method of determining the minimum value of a function obtainable by minimising the objective function and variable describing the function. These variables are known as design variables. FMINSEARCH is a MATLAB function for solving unconstrained optimisation problems. By unconstrained, we are referring to systems that do not require derivative information in performing optimisation. To investigate the behaviour of this optimisation function, a system having a predefined set of points was

generated by using the parametric function for a circle, and a circle of best fit was fitted through these points using a MATLAB function CIRCFIT. Optimisation can be classified as both a univariant and multivariant system when similar factors permit the accessibility of the difference in interaction occurring between variables. To accomplish this, the initial guess (factor) of one or both variables can be altered. This change in the initial guess helps the system, with no derivative information but an initial estimate, in identifying the minimum scalar function having n variables. From Figure 4.14, the system was applied to a set of points evenly spaced with no noise in finding the best fit of a circle to the points. The goal was to plot a circle of best fit through these points having a specified centre and minimising the effect of outlier points. It was beneficial in achieving this goal as this concept was applied when fitting features such as cylinder, cube, and sphere to a set of point cloud data. In testing system robustness, the same process was applied to a set of points with simulated measurement noise, as shown in Figure 4.15, and the system was able to fit a circle to the noisy data. Additional analysis was performed to further assess the system's robustness by simulating missing data, and measurement dent was computed.

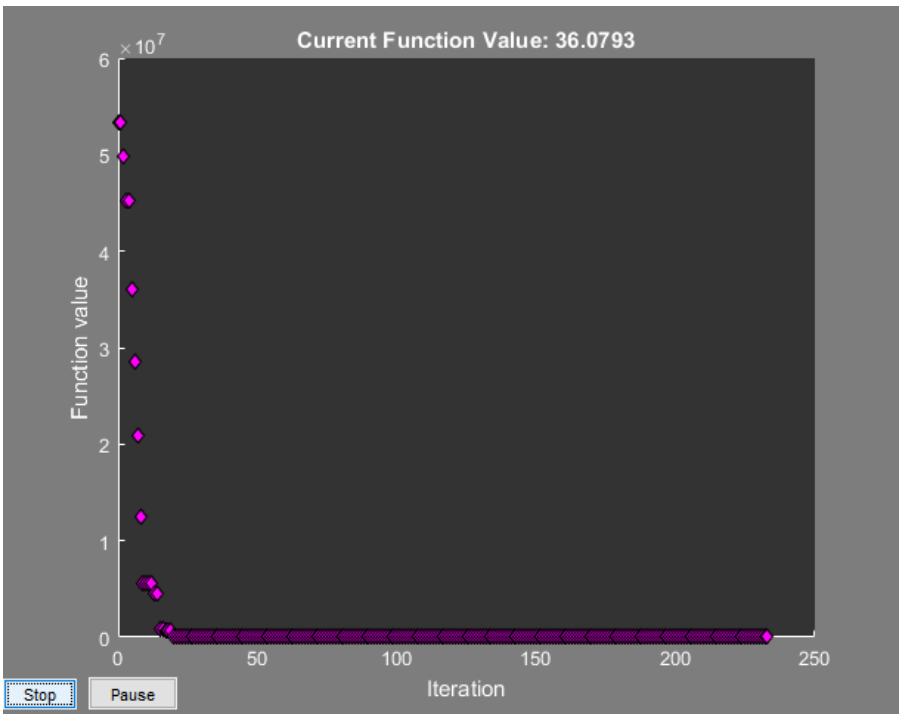


(a)

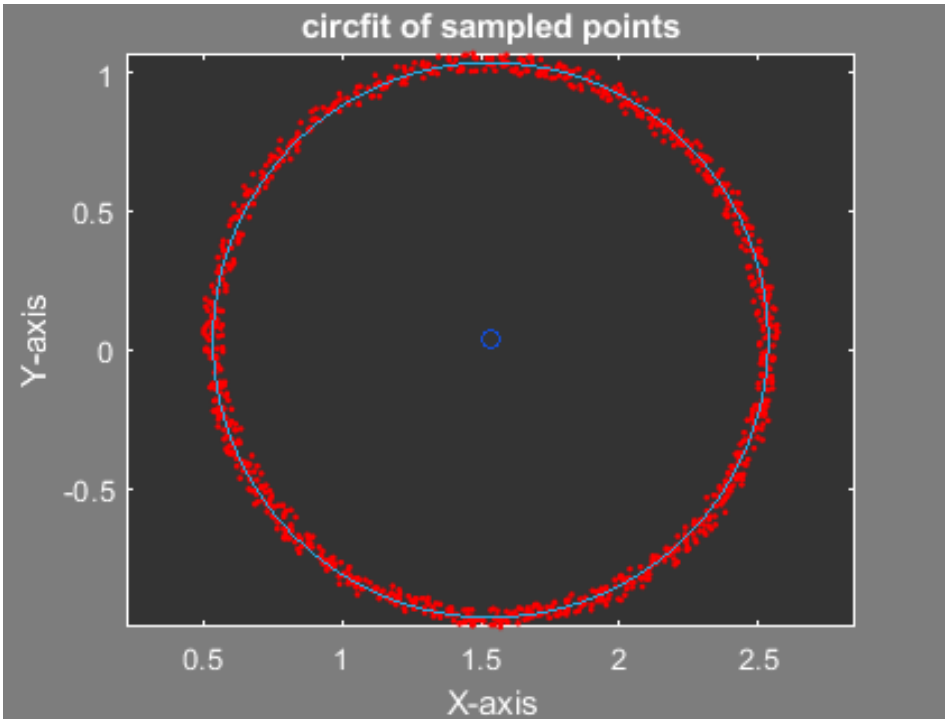


(b)

Figure 4.14: plots showing (a) optimisation plot function of the system without noise and (b) the circfit function showing how the circle fits the generated points.



(a)

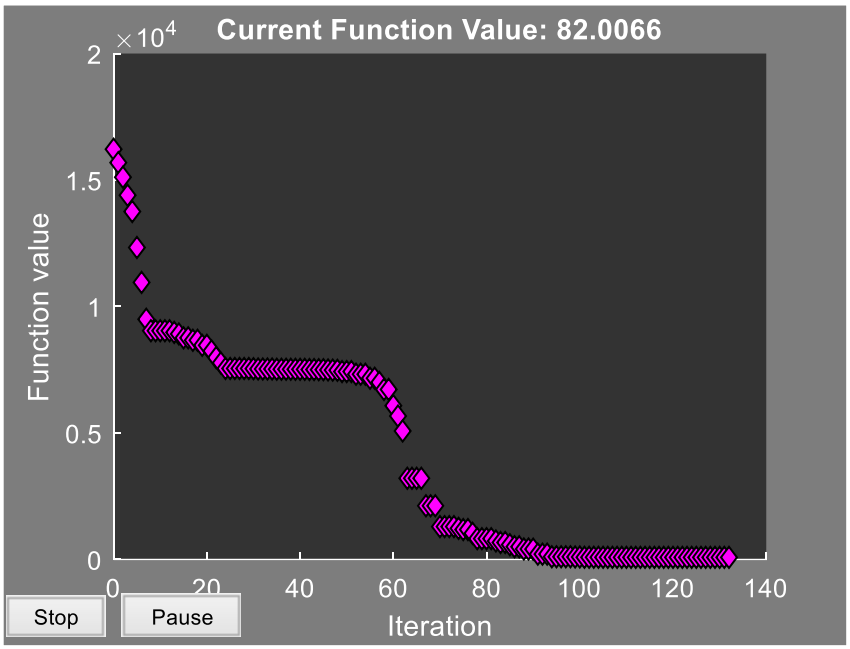


(b)

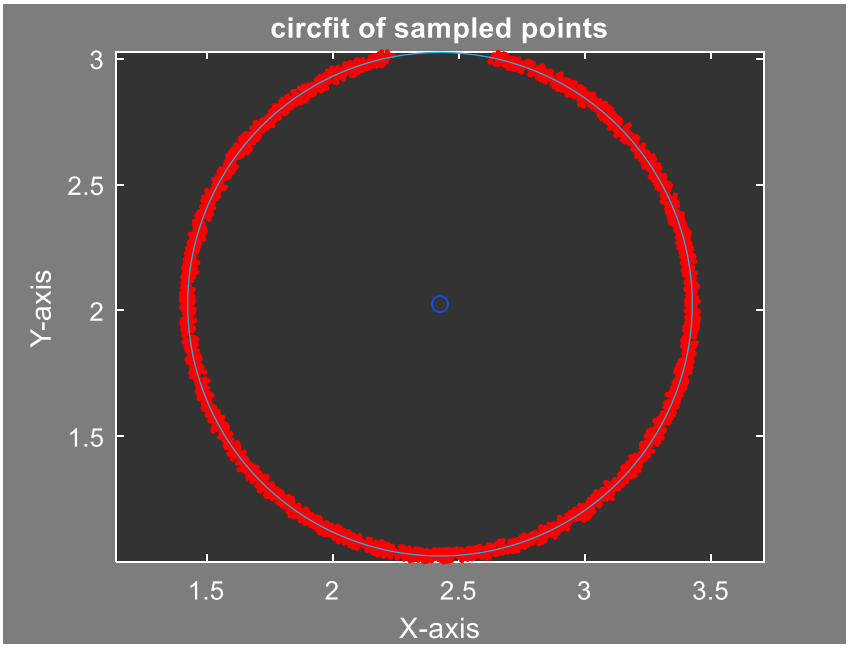
Figure 4.15: Plots showing (a) optimisation plot function of the system and (b) the circfit function showing how the circle best fits the generated points with measurement noise. The centre of the circle is perfectly aligned to the centre of the sampled points.

From Figure 4.15(a) and Figure 4.16(a), the optimisation plot function indicates how the measurement density produces an element of stress on the algorithm, and this is reflected in (b) as missing data in the measurement data. Despite the missing data, the system optimises the simulated points. A simulated dent was added to the data to test the system's robustness further, as shown in Figure 4.17. The simulated dent was aligned to the edge of the missing section, and by mathematical calculations, the angle was measured using the radius of both circles and the measured distance from where their centre and radius intercepts. The system's noise to radius ratio was calculated where the amount of noise visibility is proportional to the radius. When a little noise is applied to a system with a large radius, it produces a system that appears like a perfect circle, but the noise level can be seen when zoomed in. All possible outcomes for different scenarios can be found in Table F.0.1, Appendix F MATLAB Code for Simplex Optimisation Adapted from MATLAB. The table showed the system's robustness with the presence of noise, dent and missing information and identified the feature.

Bringing in the concept of the damage on the Honda cover, the cylindrical shape has a bulging section, and that can pose as noise when trying to fit features to the points to obtain a perfect geometry. Since the surface area of the bulge to the whole model is very minimal and from the previous discussion of the radius to damage ratio, optimisation of the system was not affected by this amount of damage. The optimisation was also used when a sphere was fitted to a set of point clouds in Geomagic CAD application. Based on this analysis on the optimisation process, an observation was made when the diameter of the sphere fitted to points generated from the calibration sphere of the Arm was compared to the calibration certificate by UKAS; this was within 0.0315 mm. This value indicates how well the system can fit a feature to a point cloud data having outliers and form error value can be affected. The outliers do not affect the fitting operation as the system optimises the maximum number of inlier points in the data.

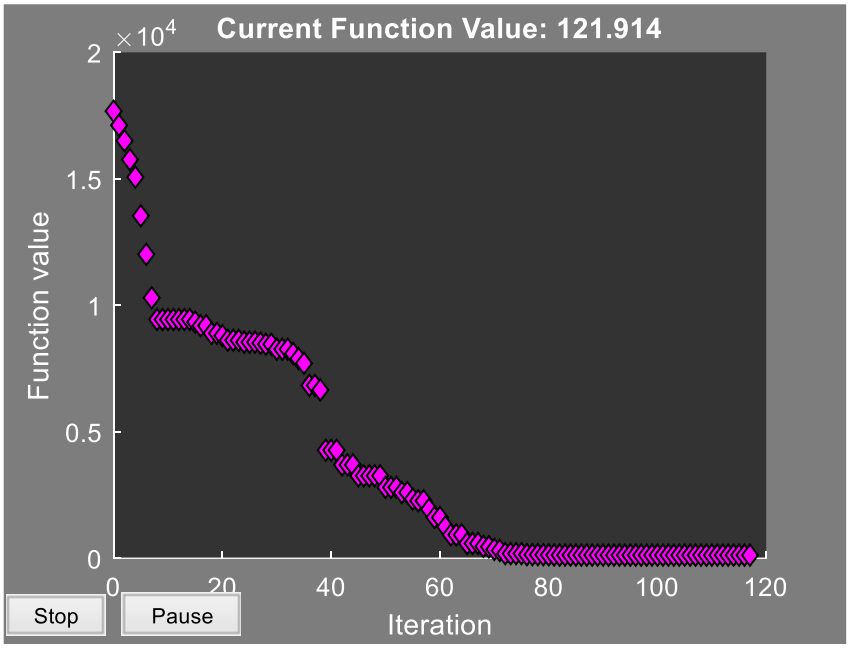


(a)

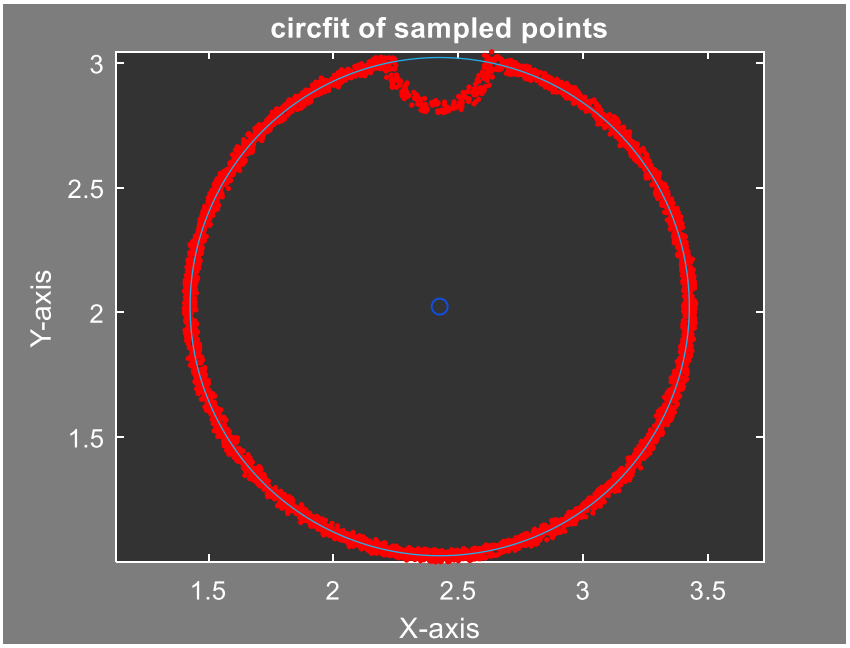


(b)

Figure 4.16: Plots showing (a) optimisation plot function of the system and (b) the circfit function showing how the circle best fits the generated points with missing data. The centre of the circle is perfectly aligned to the centre of the sampled points



(a)



(b)

Figure 4.17: Plots showing (a) optimisation plot function of the system and (b) the circfit function showing how the circle best fits the generated points with measurement noise. The centre of the circle is perfectly aligned to the centre of the sampled points

4.4 Phase Stretch Transformation (PST)

Phase Stretch Transformation is a method of transforming digital images by simulating electromagnetic wave propagation through a diffractive medium with a dispersive (frequency-dependent) dielectric function such as refractive index. The phase of the transform has properties that make it easy to detect image edges and sharp transitions. It uses an all-pass phase filter coupled with a definite frequency that relies on dispersion to imitate diffraction. Like edge detection, it is used for object detection and classification by relying on the symmetry of the dispersion profile and can be explained using dispersive eigenfunctions or stretch modes.

This method first smooths the original image using a localisation kernel and then passes through a phase operation called Phase Stretch Transformation (PST). In the frequency domain, PST uses the 2D-phase function for the image. The phase count used on the image is dependent on frequency, i.e., a more significant number of phases is used on an image with a higher frequency. As image edges have higher-frequency characteristics, the PST emphasises the image edge details by adding higher frequency characteristics in more phases.

Thresholding the PST output phase image will aid in extracting image edges, which using morphological operations can process the image further [144]. An application of PST on PCD is shown in Figure 4.18, showing how some definite edges on the original image are not detected. However, it was able to detect edges that appear to have a higher resolution, as shown with the two straight horizontal lines in the figure. Nevertheless, edge detection algorithms appear to perform poorly with PCD.

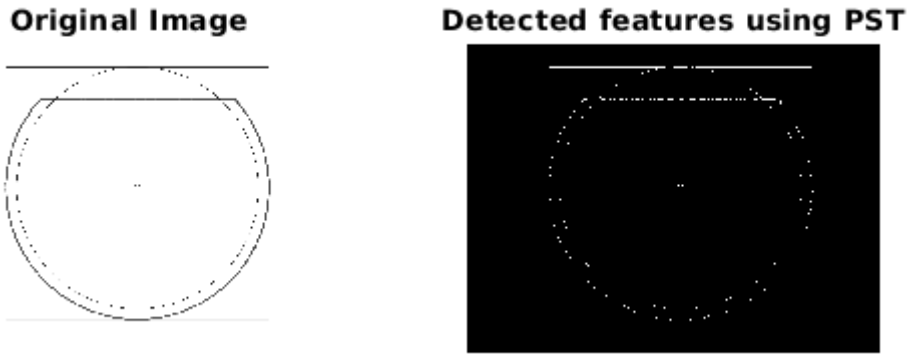


Figure 4.18: Edge detection using phase stretch transformation

4.5 Summary

This chapter investigated the concept and different types of edge detection operators, proving that the Canny algorithm is best due to how it handles noise and its response to produce definite edges. This application will be used in Chapter 5 in order to decompose complicated parts into areas of identifiable primitive geometries. Also, the simplex optimisation algorithm will be employed in chapter six when creating slices of a model in performing the best fit of the points.

Chapter 5 Automatic Feature Recognition (AFR)

5.1 Brief Description of Two AFR Methods

Machined surfaces contain various connected, expressive regions known as features that have a specific structure or topology significant to manufacturing and design. The technology of feature recognition has become a widely used application for CAM. This has resulted in developing two major approaches: feature recognition using CAD models and design using geometric features [145]. The development of the CAD modelling technique for geometric modelling has allowed the design and representation of complex features for different industrial applications, such as aircraft components, mould and die components, reverse engineering, animation, production/design, gaming, and research applications. Primitive surfaces are composed of more than one distinct work part from a machining point of view, using a few processes. Once machining occurs, the different machining components must be known; else, a sufficient surface area will be challenging to create. Also, for intended concavities which may appear as damage due to the light they scatter in relation to the reference surface, it is important to establish a distinction between an intended design and a damage. This can be performed by inspection using a method known as rotational matrix presented in section 6.3 of this thesis, where the geometric continuity of a model can be assessed as its boundary profile is rotated about an axis. For an intended concavity, the light scattering is expected to produce ordered points when the laser beam hits the surface as compared to the scattered points of a localised damage. Other methods of assessing surface imperfections as described in ISO 14997:2017 include dimensional and visibility methods.

It is assumed that the use of feature extraction on machined surfaces with machining patches and cloud-point data allows for analysis, understanding of how surfaces function and the purpose behind their designs. However, some classic algorithms [146-148] have constraints on the features' type and surface topology. When surface points are captured (point distribution), a slight variation in the continuity of patches on machined and moulded surfaces may occur. Recognition speed was not the focus of previous works [149, 150], but algorithms with computational time specific to the model have been developed, which may play an essential part in some applications such as reverse engineering, automation, and robotics handling. In this project, two algorithms were compared for their stability and accuracy in extracting features, namely the Random Sample Consensus (RANSAC)

and the Linear Scattered Interpolation (LSI) algorithm. The RANSAC was designed to identify primitives, and in keeping with our recent experience, LSI is a modern primitive detection technology, and the identification speed is compared to RANSAC. Measurement data is used as experimental data for evaluating these algorithms.

A point cloud fitting function pcfitcylinder has been found to best fit a cylinder to point cloud data with a maximum permissible distance from the inlier points to the cylinder. A region of interest (ROI) is specified to restrict the search to enable the system to run faster, and the system uses the inlier points in the ROI to best fit a cylinder to the cloud points of a model. The function uses a robust M-estimator Sample Consensus regression system (MSAC) [151]. This system uses other adjustable parameters such as orientation constraints as a 1-by-3 reference orientation input vector, absolute angular distance, linear indices relating to both inlier and outlier points and mean error of the distance of the inlier points to the model. The data were obtained by using an RS4 scanner attached to the Articulated Arm. Before importing into MATLAB, cloud points were captured and preprocessed using recommended experimental setup. This is shown in Figure 5.1 below.

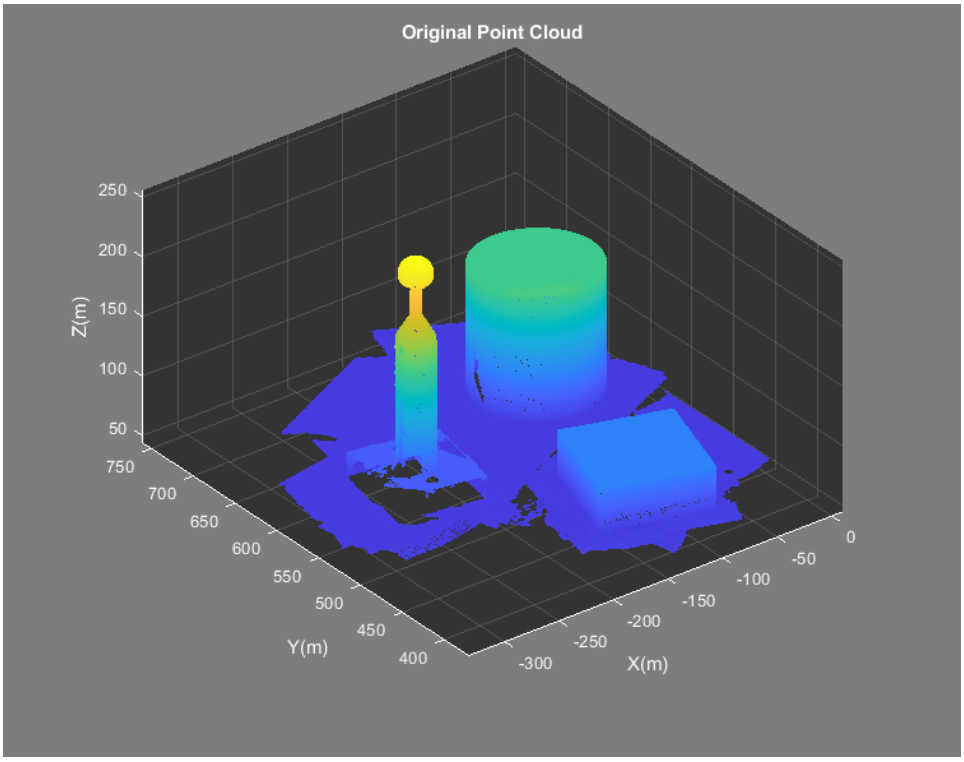


Figure 5.1: Measurement data of different geometric primitives

The ROI is defined using a point cloud specific function known as pcshow as an initial projection of the point cloud representation in the cartesian plane. Hence, a human contribution is needed to estimate the area within which the inliers of the desired feature can be found from the entire point cloud. A large ROI will increase the time it takes to run the programme and can be reduced by automating the process of specifying the ROI. The pcfitcylinder provides measurement information about the model after the fitting/extraction operation. Information relating to the position and direction of points in X, Y, Z coordinates, radius and the centre point of a model can be extracted using this function.

Further validation is necessary to check with all these parameters how best the system fit the model accurately, and this can be performed on the actual artefact with a CMM. Another concept to consider is when an artefact comprises of different geometrical shapes like the one in Figure 5.2 below. In such a situation, the fitting function and ROI can identify and accurately fit a desired geometry to the specification. This model has two different sections of cylindrical geometry; one links the base while the other holds the sphere. A random fitting of a cylinder to this model will most likely happen on the cylinder with the larger area mass, as shown in Figure 5.3, and the Z-axis must be specified in the ROI for feature fitting on the smaller cylinder as they both have the same centre point as shown in Figure 5.4. This is because the function requires point cloud normal, and when not specified, it fills it with six points as a representation of the local cylinder. It finds this local cylinder by computing the maximum distance from an inlier point to the cylinder, and in most cases, it is expected to perform this computation on models containing more points over a large area.

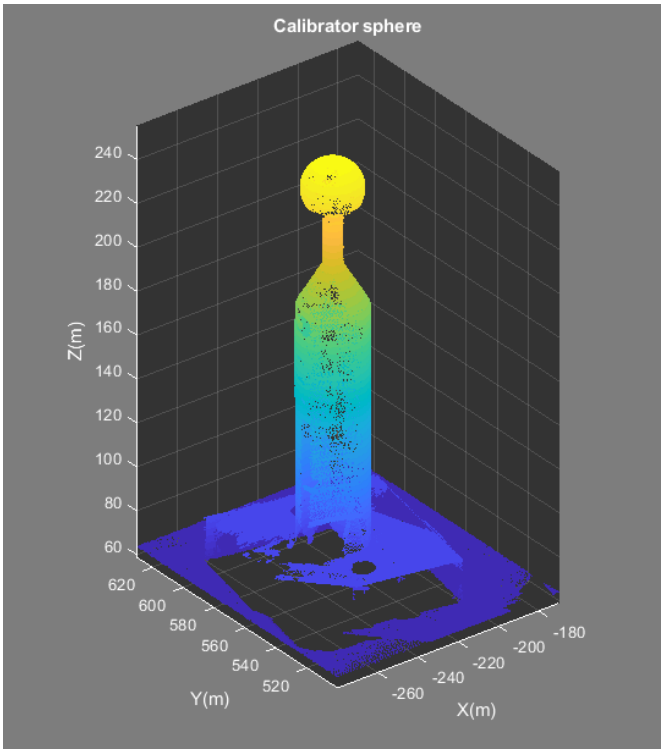


Figure 5.2: Point cloud of the Articulated Arm calibrator having different primitives in the whole artefact

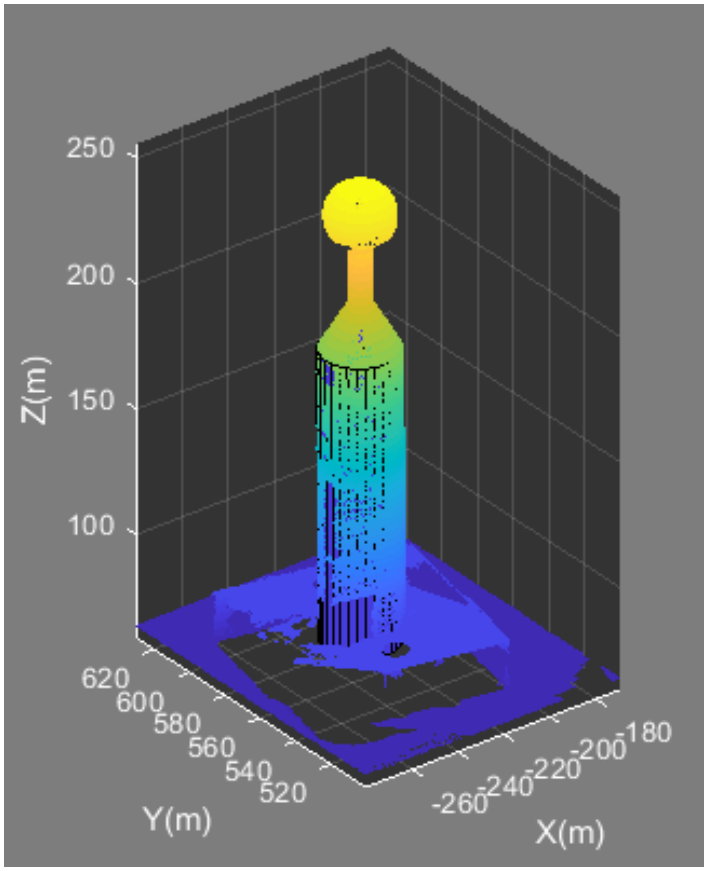


Figure 5.3:Showing how the function fits a cylinder to the point cloud

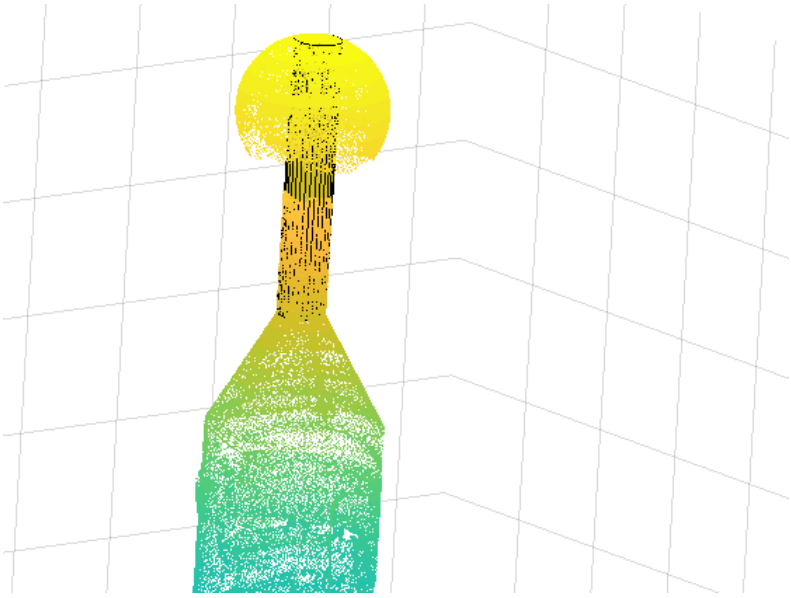


Figure 5.4: Showing how the function fits a cylinder to the primitive holding the sphere

Dimensional measurement with a micrometre, CMM and laser scanning was achieved on the Arm calibration sphere. MATLAB and Geomagic analysis was carried out after dimensional values were generated from fitting a model to points. The sphere has a UKAS calibration certificate with a diameter of 25,39995 mm in 2017 and was used for other methods as the reference value. All three methods had 0.315 mm as the highest discrepancy with cylinder 1 (with the sphere attached) and 2 (fixed at the base). The highest discrepancy was achieved in cylinder 2, with a 0.2655 mm difference between Geomagic and MATLAB. These discrepancies are caused by factors of uncertainties, such as a human error in using a micrometre and the quality of the cloud point data from the Articulated Arm and the factors that influence measurement using the Arm. Figure 5.5 shows the colour map of how the function fits a feature to a point cloud within tolerance. Further analyses show that the feature is 0.315 mm up to the sphere, but the error in its form is 0.2144 mm. Figure 5.5 shows the colour map of how the function fits a feature to a point cloud within tolerance. Further analysis shows that the feature is 0.315 mm up to the sphere, but the form error is within 0.2144 mm. This form error was a result of noise – some points fell outside the ROI. However, the accuracy of the fitting function has less been affected because the inliers in the point cloud have been optimised.

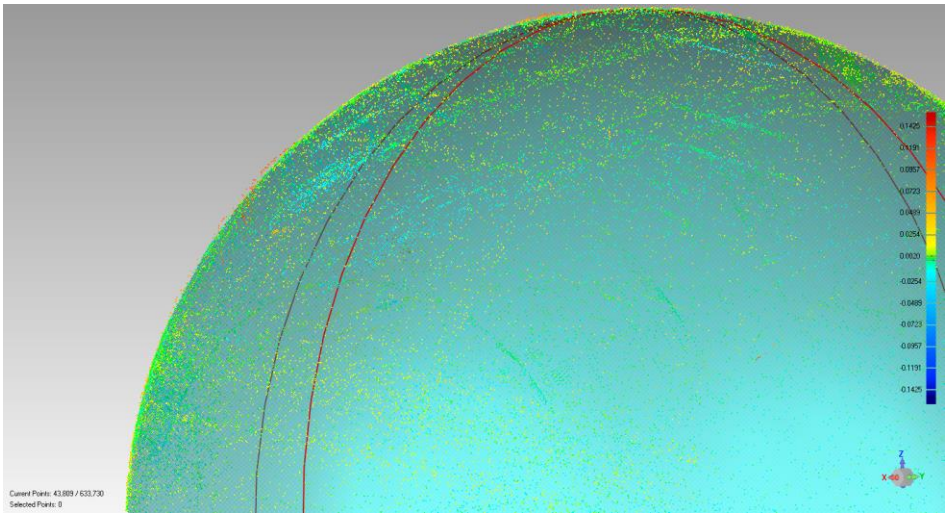


Figure 5.5: Colour map showing tolerance level of the fitting feature to the point cloud

5.1.1 Random Sample Consensus

Random Sample Consensus (RANSAC) algorithm is used to randomly draw marginal sets from the point cloud and construct the corresponding primitive forms. Fischler and Bolles [152] introduced it for the first time in 1981. This approach simulates parameters associated with a noise-contaminating data model in fields like image and signal processing. The RANSAC algorithm is primarily concerned with identifying the points identified by two categories of input data: outliers and inliers. The RANSAC algorithm requires two stages, known as the hypothesis and the test phase.

The hypothesis phase uses its strategy for minimal data sampling points. This means that the parameters of the models are determined by comparing them with the entire collection of data with the minimum sampling data points obtained from the sampled random data. The minimum sample collection is determined in the second process. Points closest to the model are preferred, while points farther away are refused. The Minimum Sample Set (MSS) is a configuration from all points listed in the Consensus Set (CS). The algorithm is continually applied until it comes under a certain threshold for all points in the CS. The best CS points are chosen to represent the final configuration based on the approximate number of iterations. The points selected are known as inliers. One advantage of the RANSAC algorithm is that it is possible to suit models saturated with noise and its methods allow the user to monitor the extraction mechanism of the algorithm more effectively.

The MLESAC or MSAC algorithm and PROSAC algorithm, are some of the improvements required to increase the computational speed of the RANSAC algorithm. The initially developed MLESAC or MSAC algorithm [141] can use M-estimates to estimate the expected inliers based on the CS data. It is calculated that the likelihood of CS occurring is the better solution. For this investigation, the MSAC algorithm (a version of the RANSAC algorithm) was used to derive cylindrical shapes using its parametric function from an AACMM dataset. The Guided-MLESAC algorithm was an enhancement of the MLESAC algorithm suggested by Tordoff and Murray [153]. Compared with iterative random samples, this algorithm resolves MLESAC's limitations by estimating the likelihood using guided input information. This approach is identical to the process in this paper where extraction/assembling of a component is carried out on a model with multiple shapes, and not specifying the desired shape produces undesired results as the system tends to apply the parametric function of the selected shape to the whole model producing undesirable results.

5.1.2 Linear Scattered Interpolation

The linear scattered interpolation (LSI) approach can be used in different fields, including machine vision, digital graphics, and face recognition. Given a set of irregularly distributed data

$$P_i = (x_i, y_i), \quad i = 1, \ldots \ldots . m \tag{5.1}$$

The spacing of irregular data points Pi over real and scalar values given by Fi matches $Fi = Fi(xi, yi)$ for an integral function in which $F(x, y)$ provides an interpolation function as

$$\overline{F}_i(x_i, y_i) = F_i \tag{5.2}$$

All points are considered not to be collinear and distinct from each other. It is stipulated in this text that the dispersed points known as Pi are in a plane, and $\overline{F}_i$ is a bivariate function. The interpolation function $\overline{F}_i$ allows multiple properties such as differentiability, consistency or perhaps smoothness to be obtained. The interpolating function can be represented in forms such as proceedings, implicit, and explicit. These forms generate a significant influence on the interpolating function, producing problems such as encoding, rendering, probing, and assessing the interpolating content known as the interpolant. No method is optimal since there are special characteristics for each type of interpolating method. The interpolating function, for example, may be rational, piece-wise bivariate or piece-wise bivariate

polynomial. Numerous algorithms for the scattered data interpolation have been developed. Many systems include linear triangular and tetrahedral interpolation, linear multi-valued triangular interpolation, Clough-Tocher Systems (Cubic Triangular Interpolation), triangle-based blending, Shepard algorithms known as inverse distance weighted methods, radial-basis functions, and natural neighbour interpolation methods. Furthermore, the most straightforward approaches are linear triangular and tetrahedral interpolation and can be used conveniently along with the scattered interpolation data. These are the techniques used on the measurement data for this research as they comprise primitive shapes such as cylinder, sphere and cuboid, easily extractable without complex edges or holes [154].

5.2 Experimental Results, Evaluation and Comparison

The Articulated Arm Coordinate Measuring Machine (AACMM) unit is commonly used to physically measure an object in many industrial fields to represent its geometrical characteristics. By accurately documenting the X, Y, Z coordinates, points are generated and analysed by different methods for extracting features [155]. AACMM uses 3D scanning techniques to create points, typically known as a point cloud. In preventing loss of essential data on corners or sharp edges, a digital representation of the material must be retained during the reconstruction of the surfaces. Feature recognition methods must therefore be adequately used to extract primitives with specified geometries from machined parts. Cylinder 1, cylinders 2, sphere, and cuboid are the simple shapes derived from Figure 5.6.

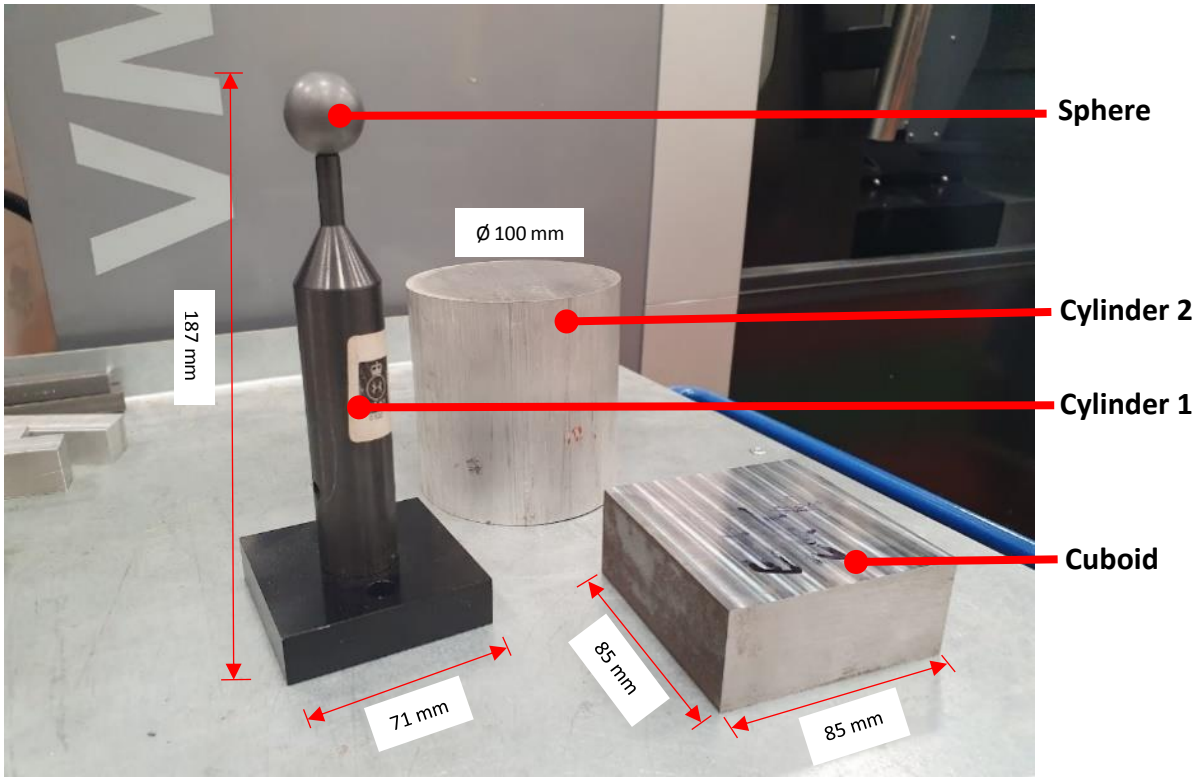


Figure 5.6: Original set up of the modelled part

5.2.1 Implementation of Random Sample Consensus

In evaluating the working principle and viability of the RANSAC, a collection of 3D data points from the AACMM device was used to assess the extraction process's precision on a primitive feature. Figure 5.7 displays the density and projection in the X, Y and Z coordinates of the original point cloud data. Using mathematical software such as MATLAB for analysis, the point density is measured by the index level for each model with the maximum index of 250 in RGB interpolating colour-space. This colour-space represents the linear mapping of the points. The maximum point-to-cylinder distance (maxDistance) is set at 100 mm or 0.1 m, the sampling points are calculated. The parameter orientation constraint or reference vector is set as a 1-by-3 vector. The region of interest (ROI) in which cylinder-1, cylinder-2, sphere, and cuboid are placed in the original model must be defined to allow the RANSAC algorithm to work more quickly and consider all the potential inliers, restricting how the algorithm finds the inlier points. This is achieved by performing a projection of the 2D contours of the whole model to indicate the exact position of each shape, as shown in Figure 5.7. Failure to define an ROI means that the RANSAC algorithm operates on an empty vector and attempts to extract or fit

a feature to the whole model as outliers and inliers cannot be separated, and the purpose of its implementation is therefore defeated. And this is one of the limitations of the practicability of RANSAC.

The ROI of every shape is a direct input parameter for the search-restrictive index variable. These are the linear indices representing the inlier points in the specified ROI and the cloud input points, which are polygonised at this level. The model contains the radius, centre of the sphere and point cloud position. The sphere's location, as shown in Figure 5.3, which is positioned at the same position as cylinder 1, was determined by 2D Contour. It gives an exact position for each model on the X and Y-axis. The Z-axis orientation is obtained when the image is projected in 3D and represented either in the ZX or ZY axes. The RANSAC algorithm is used for the detection and extraction of primitives by the removal of outlier points and the consistent enlargement of the set of points using a feasible number of inlier points.

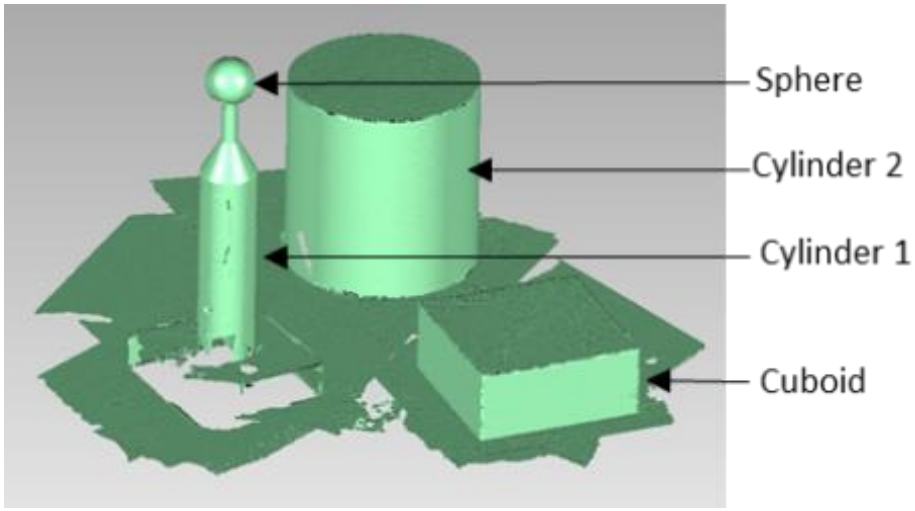
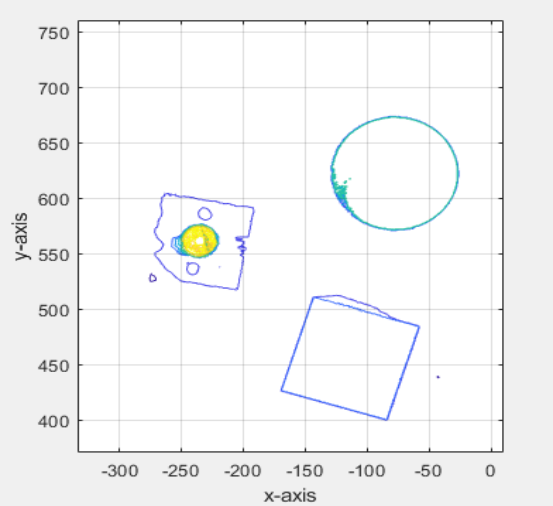


Figure 5.7: Original point cloud data of the model projected in a CAD software

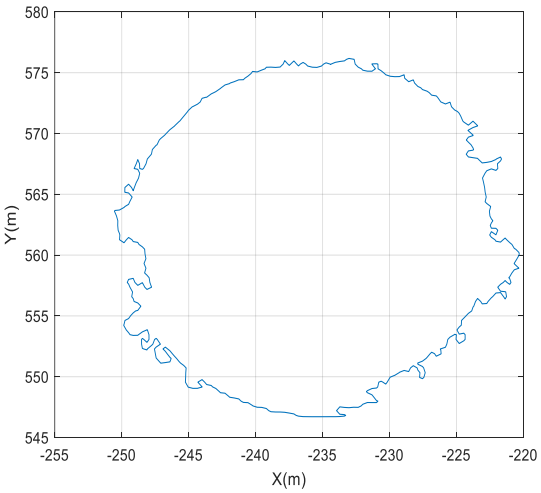
5.2.2 Implementation of Linear Scattered Interpolation

The LSI is one of the three different interpolants for the extraction of features whose applications are dependent on the type of primitive shapes. These interpolants include nearest, linear, and natural interpolant, but the linear interpolant was used for this research because they are suitable for models with simple geometric primitive shapes and are also one of the fastest methods. Other interpolants may be employed for more complex and freeform

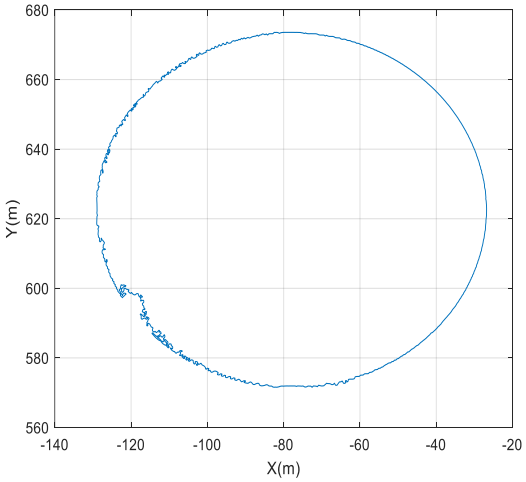
shapes. The LSI automatically detects each shape by extracting its index boundary variable from the original point cloud shown in Figure 5.7 and using edge detection, the 2D contour of the entire model can be seen in Figure 5.8(a), while (b) and (c) shows the contour of cylinder 1 and 2 as derived respectively from the whole model in (a), and (d) is the extracted contour of the cuboid. It is done using the number of scattered sampling points to determine the sampled point position, and the outliers are not affected by its performance.



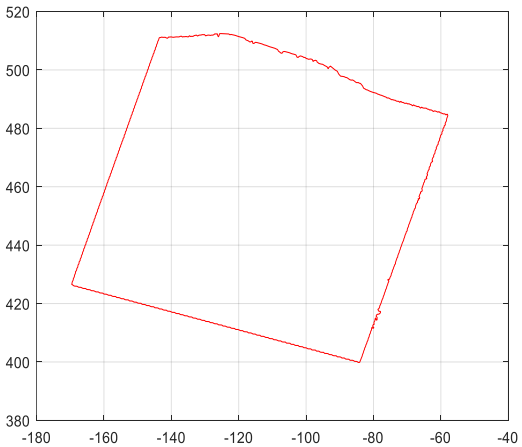
(a)



(b)



(c)



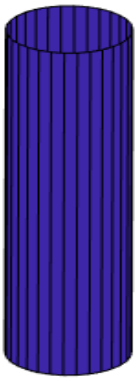
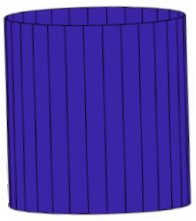
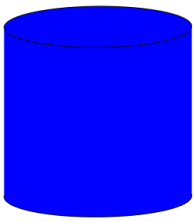
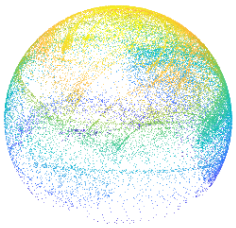
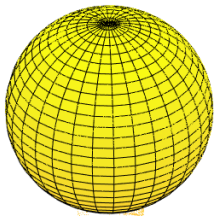
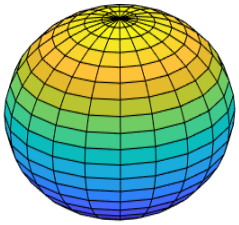
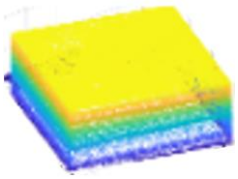
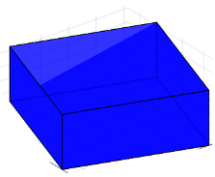
(d)

Figure 5.8: Edge detection of (a) whole model (b) cyliner1 and (c) cylinder2 (d) cuboid

5.3 Comparison of RANSAC and LSI Algorithms

In Table 5.1, column (a) illustrates the result of feature extraction using point cloud from a model. To perform the extraction, the fitting function by the RANSAC method, as shown in Table 5.1, Column (b), requires a substantial percentage of the points that reflect a geometry of the shape within the ROI defined. For the algorithm to predict its geometry and perform feature fitting, the points extracted in column (a) relative to the representation of each shape must be sufficient. The investigation into the robustness of the RANSAC process was performed by modifying the maximum distance of points to the sphere, thereby reducing the number of sampled points to evaluate the system's performance fitting with a minimal number of points. The system could accommodate a minimal number of points (but the geometry must be maintained) by fitting a sphere to the points, although the point distributions were not considered, and the measurement uncertainty could have consequences. The feature extraction using the LSI approach can be seen in Table 5.1, column (c). This approach specifies the point position measured, and after detection, the derived features in the model are constructed automatically from their respective contours. After the sample position has been determined, the LSI algorithm extracts the boundary of the feature through edge detection and projects the 3D image of the detected feature automatically employing derived contour shape matrices. RANSAC and LSI was applied to cylinder-1, cylinder-2, and the sphere. RANSAC was also able to detect the cuboid. The LSI method was unable to extract the cuboid from the point cloud due to the discarding of the duplicate points when the scattered Interpolant function was used. In using RANSAC, the parameter that differs is the ROI for all features as they have different locations and dimensions, except for cylinder-1 and the sphere having the same the location because the sphere sits above cylinder-1. The LSI typically uses the edge detection information which also means they have different locations and geometry.

Table 5.1: Feature extraction/fitting projection of both methods using Matlab

Feature Extraction	RANSAC	Scattered Interpolation
(1a)	(1b)	(1c)
		
(2a)	(2b)	(2c)
		
(3a)	(3b)	(3c)
		
(4a)	(4b)	(4c)
		No feature for cuboid using LSI

Compared to the UKAS certification, the result of sphere extraction is a 0.02 mm error for RANSAC and 0.09 mm for LSI, which shows the accuracy of both systems with a well-known measurement standard to validate the system's performance with predictive accuracy. The execution time for the sphere is very minimal compared to other models due to the number of points. Both methods used measurement data having uncertainties related to the AACMM and laser scanner. This is illustrated in Table 5.1. The UKAS do not have verified cylindrical and cuboid dimensions as the objects were collected as test materials from the machining workshop. The findings were compared to results obtained using the Geomagic studio 2014.3.0.1781 64 bits CAD application, using a particular detection approach called best-fit, in which the outliers are overlooked, and the detected inliers are averaged. It is likened to RANSAC since the inlier points detected are determined by selecting the detection region, and the algorithm performs the best fit of points in that region, ignoring any "form error" of the model as shown in Table 5.1. Although both methods compare closely to a conventional CMM, this is also dependent on the accuracy of the scanning device coupled with user experience and level of mitigated uncertainties. The accuracy assessment is an essential element in reverse engineering applications. Due to its surface area, but within 0.12 mm, the LSI caused a somewhat larger error to the larger cylinders in Table 5.2. Drawing from the comparison of both methods with respect to accuracy, execution time, and how they are affected by outliers, a performance table illustrating their relation to curve fitting and surface reconstruction is presented in Table 5.3. This tables illustrates that RANSAC is a good application for tight tolerance application for both curve fitting and surface reconstruction. Also, despite using outlier as part of its classification algorithm, it not affected by it once the ROI is specified.

Table 5.2: Radius comparison of the features obtained using different systems of measurement

Features	**RANSAC**	**LSI**	**CAD software**	**UKAS/CMM Verified**
Sphere	12.71 mm	12.60 mm	12.70 mm	12.69 mm
Cylinder 1	14.79 mm	14.73 mm	14.79 mm	14.76 mm
Cylinder 2	50.96 mm	51.01 mm	50.88 mm	50.89 mm

Table 5.3: Performance table showing how both methods of feature detection are assessed in relation to curve fitting and surface reconstruction

	Curve fitting			Surface reconstruction		
	Accuracy	**Execution time**	**Outliers**	**Accuracy**	**Execution time**	**Outliers**
RANSAC	✓	×	✓	✓	×	✓
LSI	×	✓	✓	×	✓	✓

Recognition may be vital in some applications, but a comparison of recognition speeds could refer to automation and robotic handling, in which recognition is as important as the time between each operation. Due to its fitting function and the number of points required to detect and extract a feature by the algorithm, it was observed that RANSAC takes longer periods compared to LSI. If the cylinder has more points, it takes much longer than the sphere with fewer points, as shown in Figure 5.9. The speed needed for extraction using RANSAC is a direct consequence of the distribution of the point density of the feature and the computational process. The LSI method operates at a faster speed of 10 seconds across all 15 epochs for the sphere, as shown in Figure 5.10, as this approach automatically recognises features using a 2D contour profile through edge detection and projections of its 3D image. However, there was an observation that the LSI was slightly less accurate from Table 5.2, with the sphere producing an error of 0.09 mm to the UKAS value.

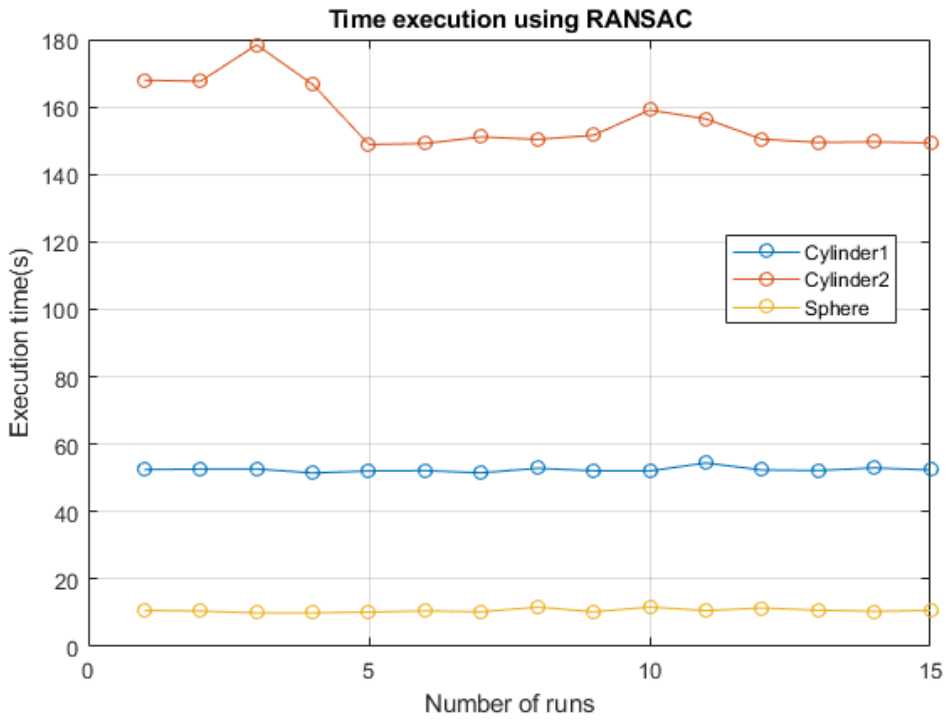


Figure 5.9: Computational time for executing RANSAC

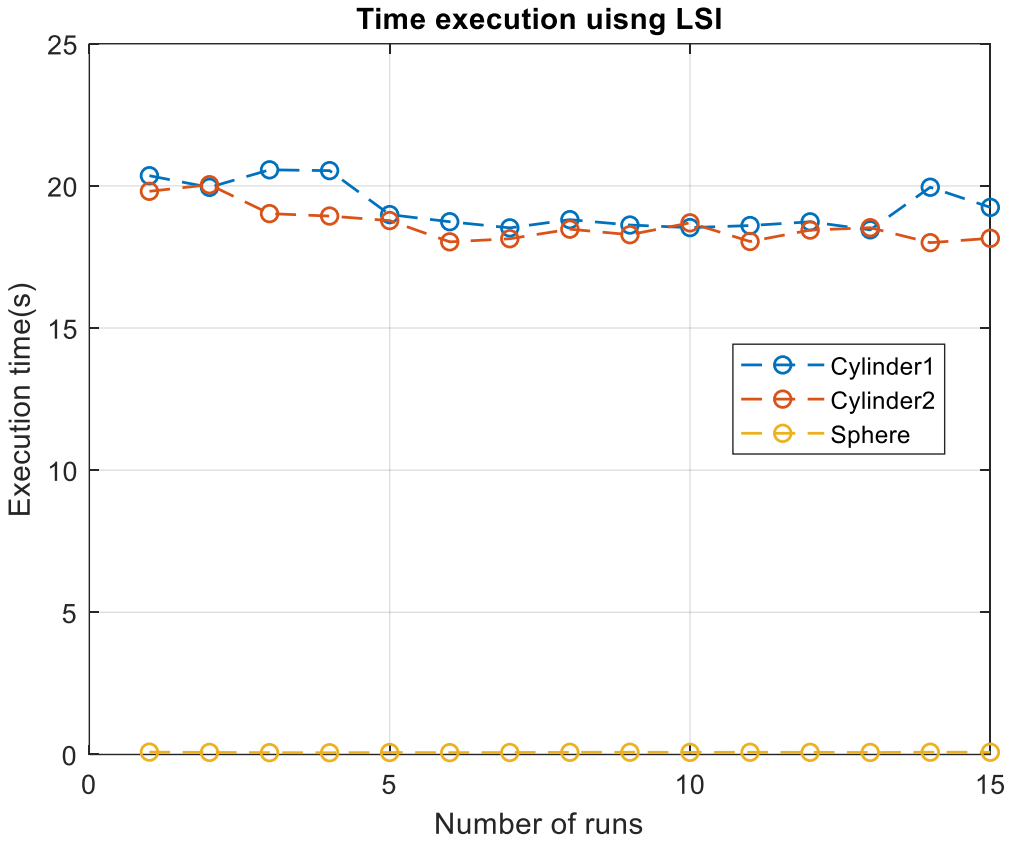


Figure 5.10: Computational time for executing LSI

5.4 Summary

This chapter investigated the concept of Automatic feature recognition (AFR) and a brief description of the various feature recognition methods of a rule-based pattern as stated in the literature chapter. The process of fitting a feature or primitive shape to a set of cloud points was analysed with a particular interest in an interpolation method and a sample consensus method. Both methods were compared to ascertain how best they compare to a high tolerance measurement system such as the CMM, also evaluating their accuracy, execution time and performance with outliers.

Chapter 6 Feature Profile Extraction and Extraction of Micrometre-Level Point Cloud Data for Damage Detection

Profile analysis is the process of defining parametric models to describe the boundary structure of a machined component [141]. The machined part's boundary can be described by parametric models characterised by predictable and natural variability behaviour. A concept known as manufacturing signature refers to this predictable behaviour, which provides a profile view of the machined component. It is the uniform pattern that identifies all the features machined within a process. They are used to create the requisite for profile analysis procedures.

As part of an object's surface analysis, developing a digital representation of the part is very important to obtain its dimensional properties and 3D model. Techniques based on Statistical Process Control (SPC) are used for profile analysis of machined models producing well-established and understood borderline contours of the part. Profile analysis can be limited in terms of visualisation as compared to areal analysis for surface texture because areal analysis provides a 3D visualisation of the surface properties concentrating on the root mean square parameter as in ISO 25178-2 [156]. But this does not fit the purpose for micrometre level application. However, the primary surface, waviness, and roughness of a surface can be analysed in the profile system [126], it is quick and offers a simple approach to analysing the surface properties of a model. This process involves interpreting a system's characteristics by linking a response variable measured along with the appropriate values of one or more explanatory system variables [157].

6.1 Description of Measurement Data Capturing

The experiment was carried out in a controlled environment with a steady ambient temperature, and the AACMM was mounted on a stable platform to counteract any vibration that might influence the integrity of the measured data. Extra care was taken to ensure the part was clean from dirt and positioned in a fixed position to counter vibrations from the machining environment. The AACMM used is a 7-axis Absolute Arm and attached to the Articulated Arm is an RS5 laser scanner (specifications are stated in Table C.0.1 under

Appendix C Reverse Engineering Tools), and scanning was performed at a steady pace within good capturing distance monitored by the RDS system with respect to ISO 10360-7 [158].

The Arm's accuracy is measured according to a variable L which indicates the length of the Arm at which the measurement was performed. A higher value of L indicates that the Arm is measuring at a larger measurement distance, and the accuracy increases when the value of L decreases. At the maximum length of the Arm, the volumetric accuracy is 0.038 mm, roundness uncertainty is 0.025 mm and uncertainty of measurement at 0.025 mm at a temperature of 20° C ± 1° C. For best practice, the joints should be positioned near 90° as the encoders are most accurate, having the largest angle to distance moved ratio. Positioning the Arm's elbow at 90° to the base vertical axis produces a parallel projection from the base to the surface being measured, and a downwards projection from the elbow produces the preferred positioning of the artefact. This is discussed in section 2.4.2 as this gives the maximum range of movement of the arm's elbow relative to the artefact. Points are automatically registered on the capturing software as the laser line runs over the surface of the part.

A significant challenge to how the scanner captures points is the reflectivity of the surface and surface material. The laser scanner generates 725,000 points/seconds at a minimum point spacing of 0.011 mm. When the laser line runs over the surface of the part, certain factors influence how surface points are generated, and this includes the laser scanner in the case of overlaying points where the averaging and blending process happens, the surface texture and colour, the milling pattern and surface finishing. These factors determine the line spacing between points and points generating patterns depending on the geometry of the part at a particular region, i.e., at regions having curves and freeform more points are generated compared to a flat surface. Points generated from a machined part will produce a different pattern from parts moulded/cast and 3D printed.

In 3D representation of a part, one major challenge has always been file transfer formats which aid in communication between software. This factor is to be considered as data can be generated using specific software and analysed in a different software application; hence, it is vital that there is a good level of interoperability (interfacing) between applications. The point cloud data was obtained using a CAD capturing software as shown in Figure 6.1. Preprocessing was performed to deal with noisy and unwanted data points before exporting

them as a text file. This file format means that the data can be imported into mathematical software for further analysis. The text file was imported into MATLAB and can be visualised using the computer vision toolbox.

Figure 6.1: Digitisation of parts for damage detection analysis using profile monitoring

6.2 Model Sectioning for Damage Detection

From the analysis in section 6.1, the visualisation of the part has been performed by rotating vectors, and the damage can be seen on the profile by a relative change in the B-spline. Further analysis investigating the location of the damage at the microns level is required, and this will be discussed in this section. First, we will discuss the use of simulation data generated in MATLAB with simulated damage for evaluating the performance of the proposed sectioning method. This gives the flexibility of adjusting parameters and comparison after extracting information. This will be followed by validation of the method using measurement data captured from a part with either machining damage or purpose created damage for this project.

6.2.1 Simulated data

Point cloud models were developed mathematically in MATLAB R2020a to replicate the cylinder model used for this project. One has a random distribution of points as shown in

Figure 6.2, and the data is identical to data obtainable from capturing point cloud using AACMM. The other data, as shown in Figure 6.3, is an ordered point cloud generated by specifying parameters such as the number of points as 10,000 points with a radius of 5 mm, a height of 10 mm, line spacing of 0.1 mm, and location of damage at 3.4 mm on the Y-axis and between 4.6 mm and 5.6 mm on the Z-axis. Also, another damage at 4.8 mm on the Y-axis and between 9 mm and 9.78 mm on the Z-axis. For the random distribution data, parameters such as height at 10 mm, the radius at 5 mm, resolution of 0.1 mm, point spacing of 0.011 mm, damage start angle at 0-degree, damage start end angle at 90 degrees, damage level at 0.5 mm, and damage z-height at 1.5 mm. This data replicates the point distribution similar to measurement data and provides the flexibility of adjusting these parameters both for generating several models and generating more data for training a deep learning algorithm. Details of this can be found in Appendix H LSTM Training of First Sets of Data Producing False Classification Due to Preprocessing Methods of Generating and Classifying the Slices. The intensity and location of damage can be simulated to meet the purpose, improve the performance of the algorithm, and emulate real damage on a part's surface. The model's slicing images for both simulated data are combined and classified into two classes for training a convolution neural network (CNN).

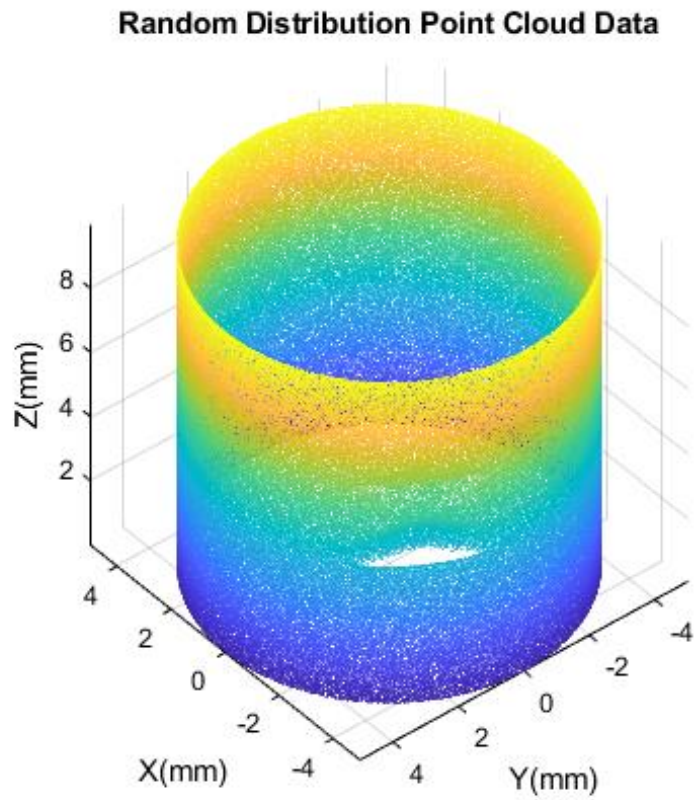


Figure 6.2: Simulation data _ random distribution point cloud

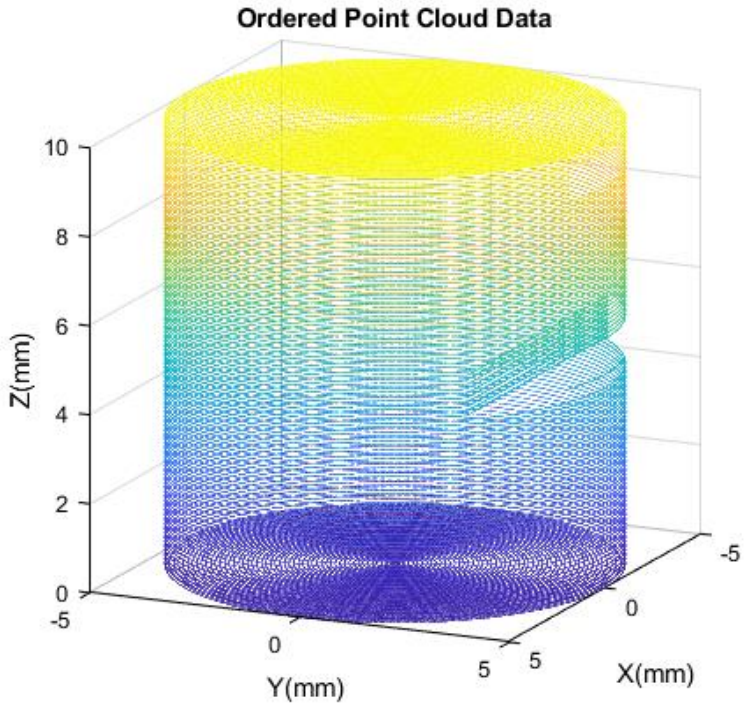


Figure 6.3: Simulation data _ ordered point cloud

6.2.2 Slicing Experiment and Classification

Data preprocessing when working with simulated dataset is not necessarily required as the data is produced with specified parameters with no data contamination as encountered with measured data, except for simulated damage or noise. Several slices are created on the model in the Z-axis direction by using the maximum and minimum points to produce an equal-sized 2D projection. Visualising the slicing data in both X or Y directions will produce little or no information showing any possible damage using the contour of the part. This is because the points distribution appears to be even when projecting in X or Y directions. One way to find out is to intensify the damage as more points are created on areas with bumps or dents on the surface. Although this is not the case with this model (when projected in the Z-axis), the points distribution is less on damaged areas and cannot be distinguished from the whole surface when projected in the X or Y-axis. When working with a cylinder, a 2D projection will produce a circle when viewed from the ZX-direction, as shown in Figure 6.4 and Figure 6.5. These slices contain points, and a circle fitting algorithm is applied to determine how well these points fit to a perfect circle, as discussed in section 4.3.1.1. Furthermore, a bounding box is applied to find the smallest measure of an area that fits all the points in a slice to speed up the computational process. This is indicated as the red square boxes in Figure 6.4 and Figure 6.5. These boxes continually change position with the random distribution data due to the continuous alignment of the points by the algorithm, but it appears steady with the ordered data because of its form. A classification algorithm is developed to classify the slices into two classes of “good” and “damaged” by specifying the region of damage and classify slices outside of this region as “good”. This is implemented before unwrapping each slice to produce an ordered numeric sequence data. This numeric sequence data supports the aim of dealing with the data at the microns level as compared to images. With this data, the nominal radius of the model was obtained. This can easily be achieved with simulated data as the parameters are already known but can be challenging with measurement data with no reference information; rather, the nominals are estimated. In Figure 6.5, the top row is a representation of the damaged slices of the ordered dataset. It is not bigger in diameter as it appears compared to the good slices in the bottom row, but it is a function of the bounding

box finding the smallest area and using the circfit discussed in section 0 to fit a circle of best fit through the points.

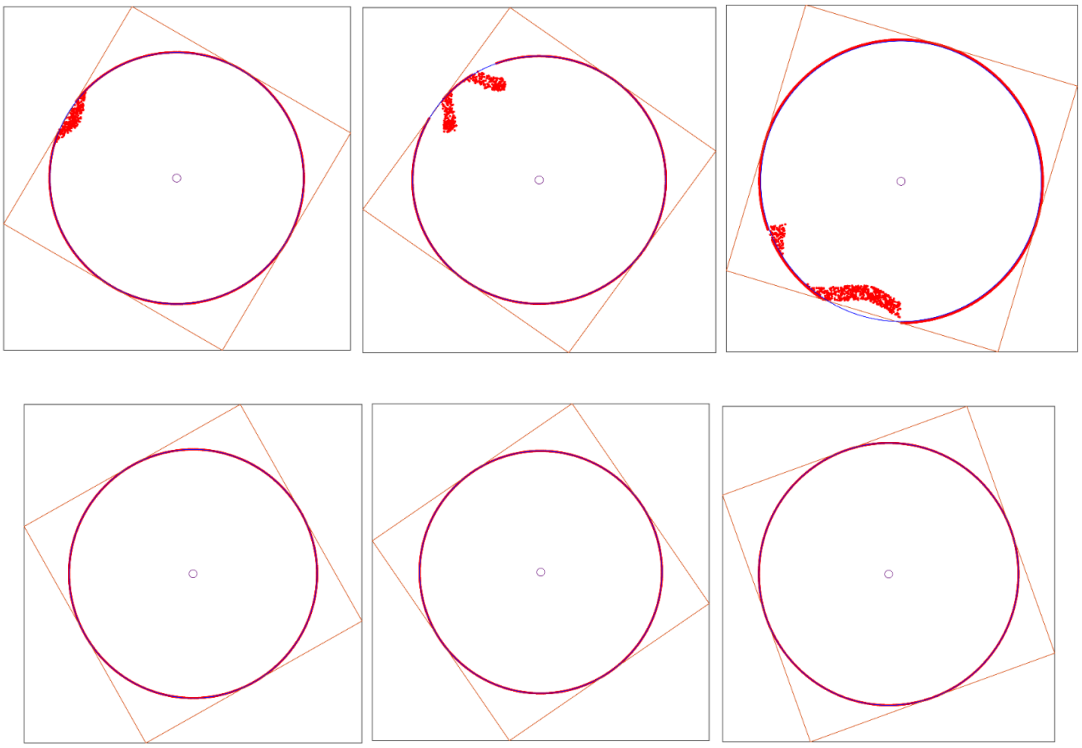
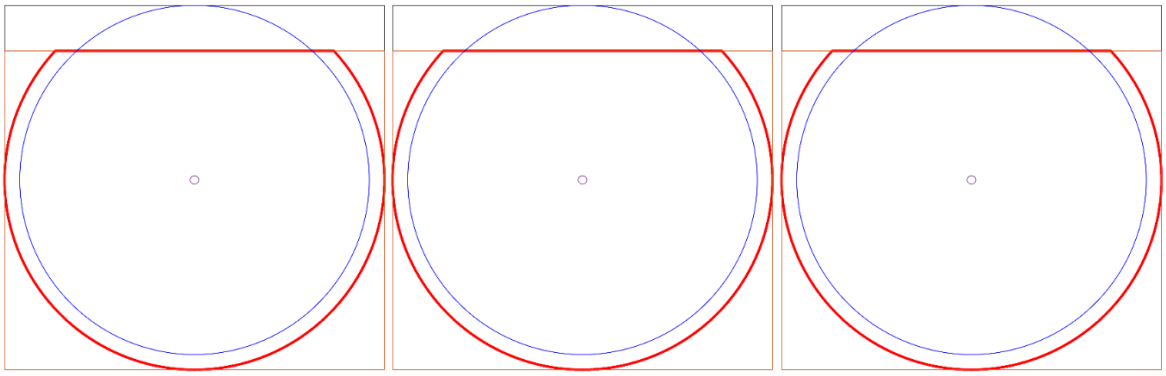


Figure 6.4: First row: damaged slice at different locations; second row: good slices for the random distribution data



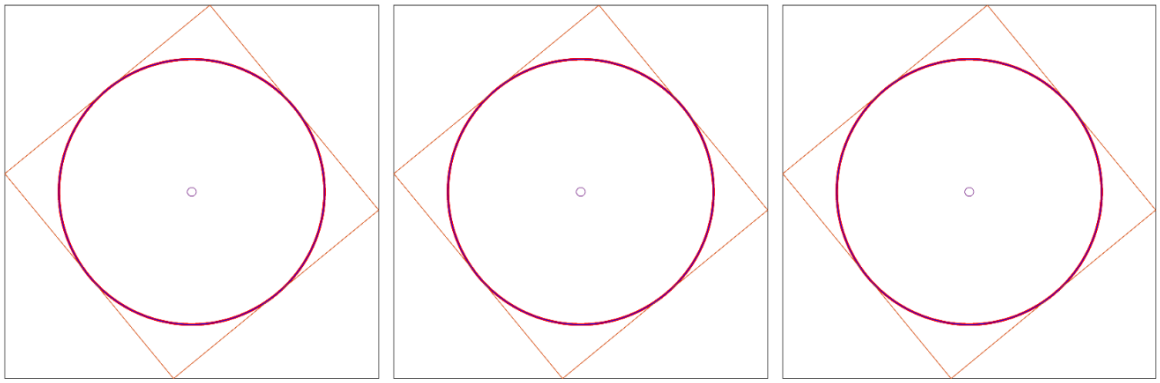
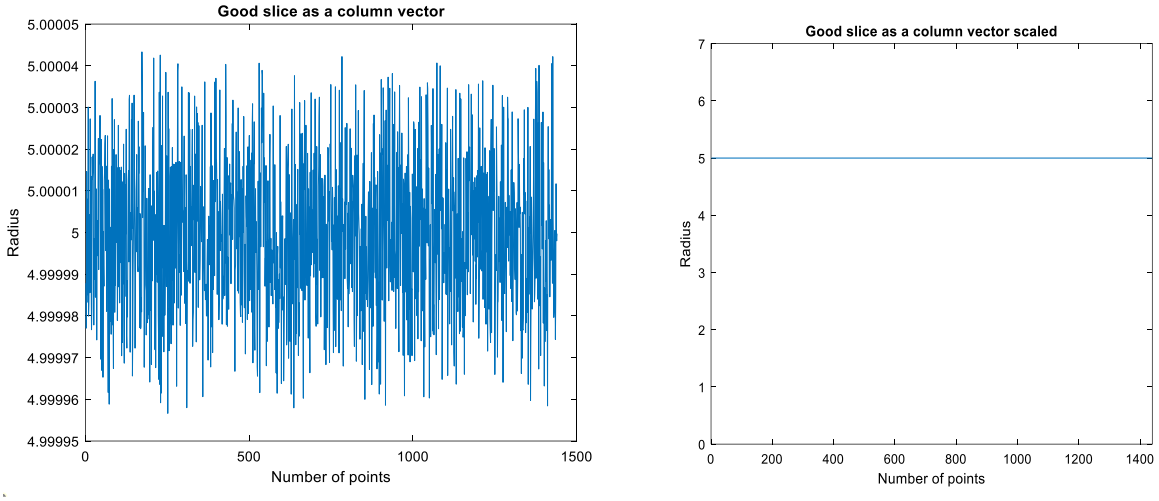


Figure 6.5: First row: damaged slice at the same location but different Z-axis height; second row: good slices for the ordered data

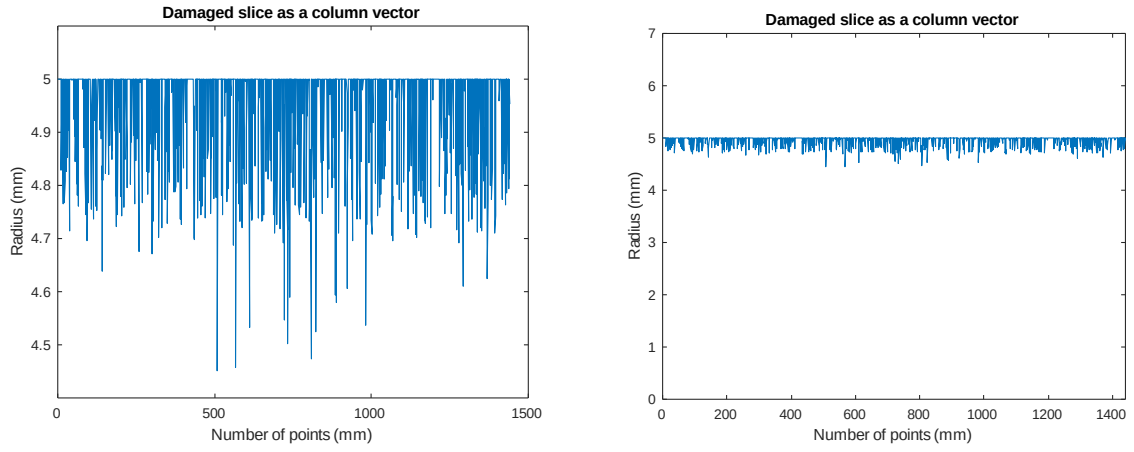
The unwrapping of the slices provided a different dimension to how the data can be analysed, and the displacement of each point at microns level can be obtained. Figure 6.6(a) shows a plot of a good slice with the position of each individual point, and Figure 6.6(b) shows when the plot is scaled to accommodate the maximum number of points. The randomly distributed point data replicates the scanning data but with known tolerance as the amplitude of some points is a few microns off the known radius of 5 mm, as shown in Figure 6.6. This is not always the case with scanned data. The plots were investigated to understand how the individual slice relates to potential damage. With the data being a vector, the data appears to have the exact damage distributed over the whole slice when plotted as a row vector, as shown in Figure 6.8, but the variation between good and damaged can be clearly seen when plotted as a column vector. On a damaged slice, the damage has a specific position relative to the whole slice, but when visualised, the damage appears to be distributed over the slice. This required further investigation, and it was observed that during data generation, the algorithm was wrongly sorting the points as in Figure 6.7 by distributing the damage throughout the length of the slice. Training the LSTM with such data, it was observed that the LSTM was learning all data as being good as in Figure 6.9 hence producing false-positive classification, and this will be discussed further in section 6.2.4. This can be compared to situations where intended features may be recognised as damage. In such situations, parameters that define the intended design can be used in specifying a threshold level and the light scattering of points will produce well defined boundaries that produces C^2 continuity to the reference surface. It is expected that the radius value for each point be closely matched. Further preprocessing of

the data will produce a normalised dataset where feature variations can be explicitly visualised. For regions of the surface with intended features, much variation in radius value is not expected to occur and this can help the LSTM prediction coupled with specification of the network's hyperparameters. Also, specifying the samples size of each slice as described in section 6.2.5.



(a) (b)

Figure 6.6: (a) A good slice plotted as a column vector (b) A good slice plotted as a column vector and scaled



(a) (b)

Figure 6.7: (a) A damaged slice plotted as a column vector (b) A damaged slice plotted as a column vector and scaled

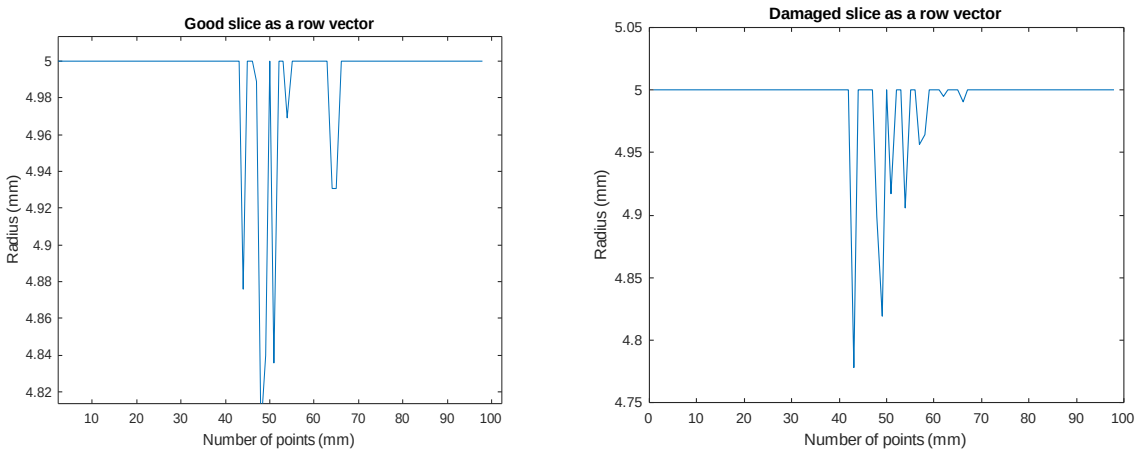


Figure 6.8: Both good and damaged slice as a row vector

6.2.3 Nominal Radius Estimation

When reverse engineering blind with no reference information of the artefact involved, it is required that the nominals be estimated. Nominal estimation could be on the surfaces, the points, or geometric dimensions. In this case, we estimated the radius using the linear least-squares fitting method on the unwrapped slices, and this should produce an average value and the line of best fit at the intercept on the Y-axis. The values obtained compared well with measurement information from a CMM and analysis on the original scanned data in MATLAB.

6.2.4 Long-Short Term Memory

The nature of the slicing data is a sequence of several slices where one slice is stacked above another slice until the whole model is completely sliced. This provides a data set in sequence form having two classes in the case of this project, and accurate classification can be performed using information from a previous slice; hence the application of recurrent neural network architecture is required.

Once slices are classified into two classes of “good” and “damage”, time-series data is generated alongside its categorical cell array, and this data is used for training the LSTM. The input data was partitioned into training and test datasets for cross-validation using 70% for training and holding out the rest for testing. The neural network was trained with eight layers because of an extra LSTM layer to make the network deeper with an output mode set to sequence and accompanied by a dropout layer to prevent overfitting. The first layer is the sequence input layer with the number of features set as the size of the training dataset, a

double LSTM layer, a fully connected layer with an input of two classes – good and damaged, a SoftMax layer, and classification layer.

Using the data from the ordered point cloud produced training progress with a steady response due to the specified values and non-numerical (NaN) characteristics in the data. This required some normalisation of the data to extract responses from the training process, hence the classification result. An initially trained LSTM was learning all input data as good notwithstanding the input classification, and there appeared to be an issue with the simulated data generated by the automatic classification. With further investigation into the data, it was found that when plotting the bad slices, the damage spreads over the whole number of slices of 1078 x1440. Doing a plot function of one slice as a row vector appears to be a wrong visualisation approach, but when plotted as a column vector, identifiable data to what is expected can be visualised.

Another issue observed with the data plotted as a column vector and visualising all data in entirety coupled with an investigation of the data content, the damage was identified on all supposed slices, which appears to be wrong as the damaged region is relatively small compared to the whole model. The wrong categorisation of the points and other factors such as data preprocessing might have contributed to the LSTM learning the data wrongly, and this is shown in Figure 6.8. More indications of this are further shown in Appendix H LSTM Training of First Sets of Data Producing False Classification Due to Preprocessing Methods of Generating and Classifying the Slices. However, this training produced accuracies of over 70 % in some training sessions, as in Figure 6.9, there appeared to have a large amount of false positive and false negative classification as in Figure 6.8 and Figure 6.10. These false classifications are also a result of the LSTM learning all input data as good or misunderstanding the data structure contained in each classification or due to the fewer training data set or network architecture. Another situation of misclassification is shown in Figure 6.11, indicating a false positive/negative prediction of both classes.

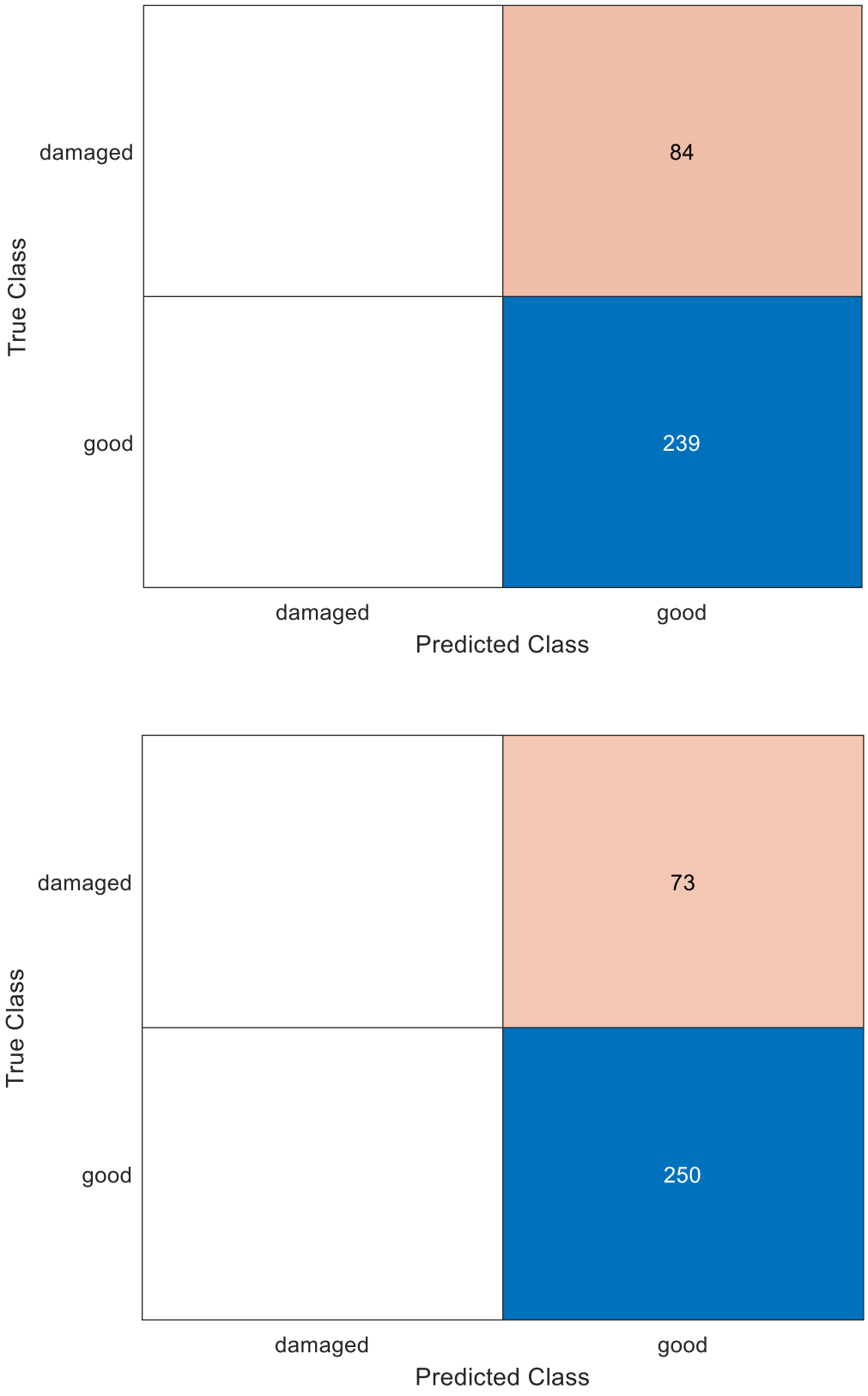


Figure 6.9: Confusion chart indicating the classification result of the LSTM

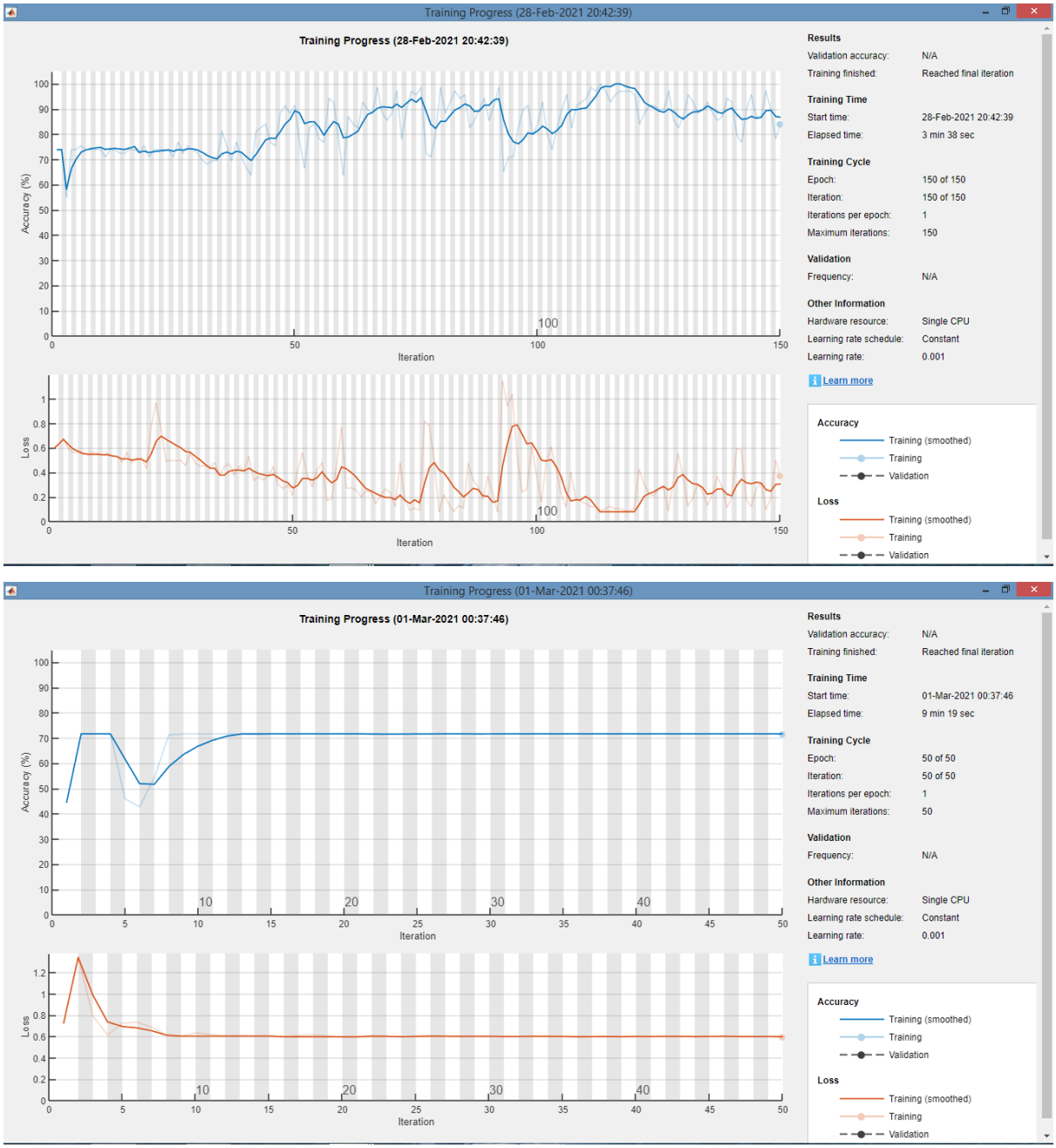


Figure 6.10: Training progress of the LSTM

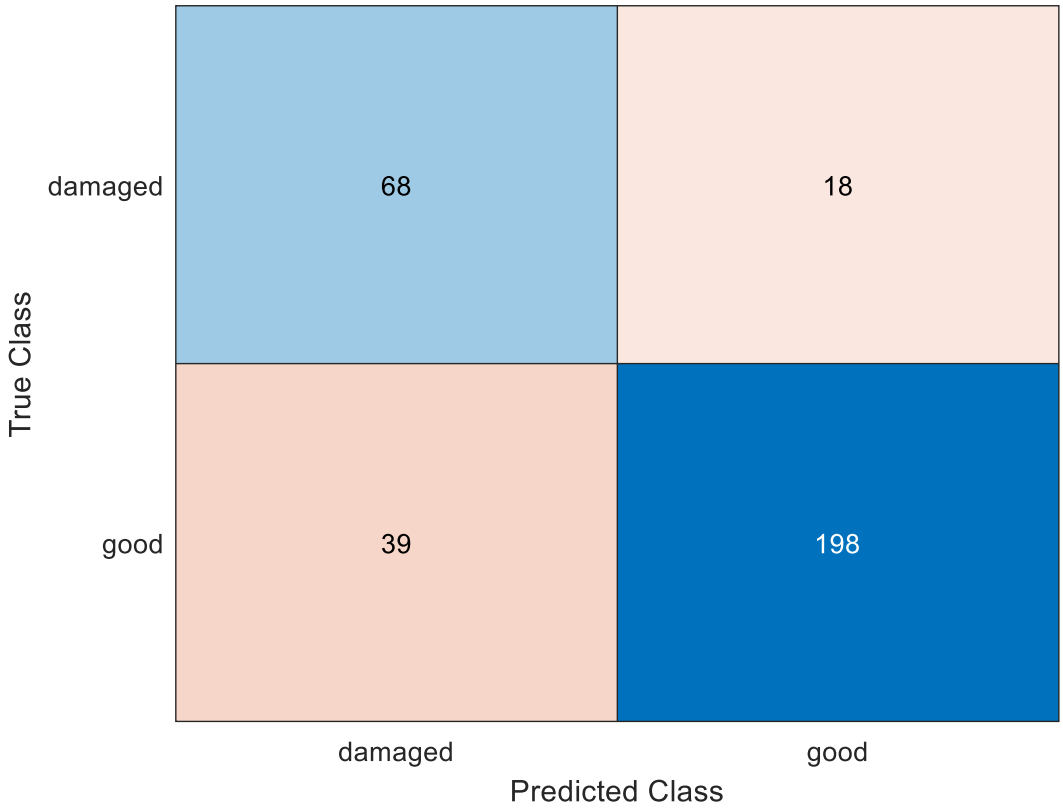


Figure 6.11: LSTM training indicating false positive/negative classification

6.2.5 Generating Training Data for LSTM using Wavelet Transform

The data analysed in section 6.2.4 produced unsatisfactory results when trained with LSTM, and this was due to the data preprocessing and the algorithm's extraction of the points. This led to the use of wavelet transform in extracting meaningful features for analysing the time-series data obtainable after slicing. A continuous wavelet transform was used in extracting information for training a machine learning algorithm. The raw data is a set of unordered PCD captured by light scattering as a laser light hit the surface of a model. The approach is to create slices of specified resolution and unwrap each slice to obtain points contained within each slice. This process presented a data in the form of a time-series data which resulted in the use of wavelet in performing time localisation of points contained in each slice. The wavelet function translates the response signal of the location of each point into a continuous transform along the time axis to produce a correlation that can be visualised through the wavelet coefficient. This produces a dataset of the number of points in each slice coupled with their radius data. This data is further preprocessed by performing interpolation to

produce a normalised dataset where the sequence length of each slice is made equal. A dataset with equal sequence length is not enough information for training an LSTM. The axis size of each slice in the dataset can be different, hence, the resolution of each slice is regularised by specifying the sample size. For this thesis, the sample size is set to 1° per sample to identify the slightest variation in the points of each slice. This approach can also aid in the data preparation for intended features that may appear as damage by inspection.

A simulated damaged cylinder was developed by specifying parameters such as model's dimensions, damage location, the intensity of damage and scanner noise to replicate measurement data. Slices are produced by computing the entropy surface geometry variation using a continuous wavelet transform (CWT) to produce coefficients representing each slice. The analytic Morse wavelet together with the symmetry parameter is used to obtain the CWT. The minimum and maximum scales are automatically determined using the energy coefficient spread of the wavelet in frequency and time, as shown in the wavelet coefficient of Figure 6.12. If the input data is real-valued, the wavelet transform is expressed as a 2-D matrix with each row corresponding to one scale. The column size of the wavelet transform equals the length of the input data. Figure 6.12 through to Figure 6.15 illustrates the slice extraction process using CWT to produce angles of each point, the radius of each slice, and project the response of the energy coefficient of each slice. The total number of slices is obtained by dividing the height of the model by the specified slice resolution and not the mean or aggregating a series of slices.

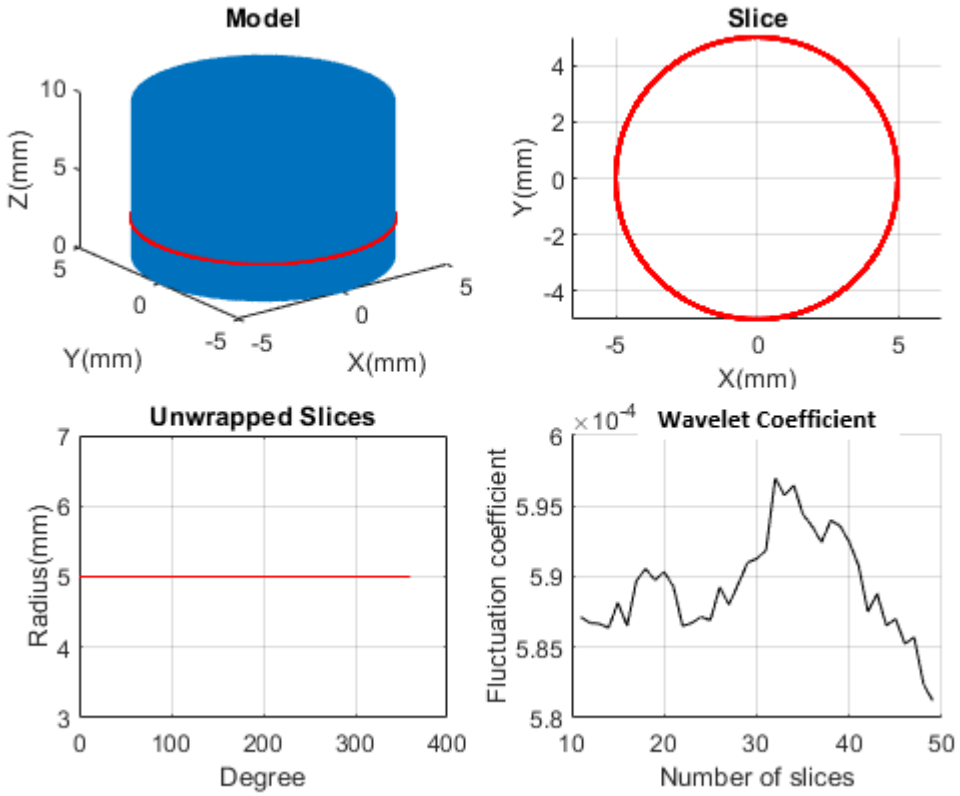


Figure 6.12: Generating slices using CWT showing early stage of the process and wavelet coefficient showing number of slices 2.5 mm from the base

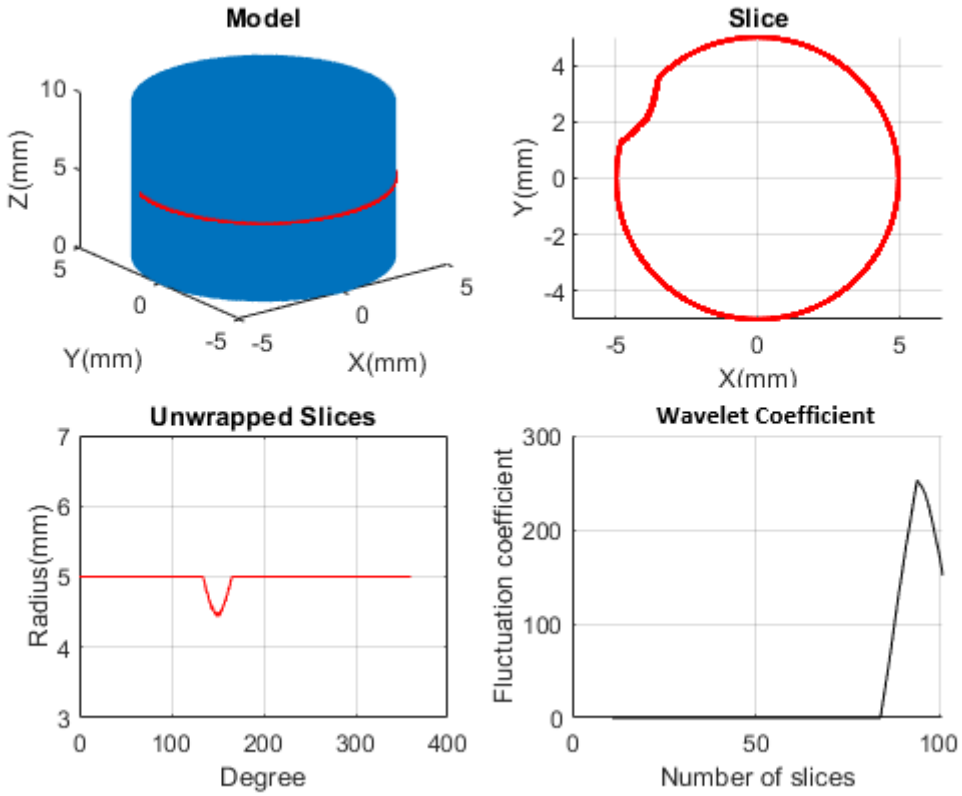


Figure 6.13: Generating slices using CWT showing early detection of the region of damage and wavelet coefficient showing number of slices 5 mm from the base

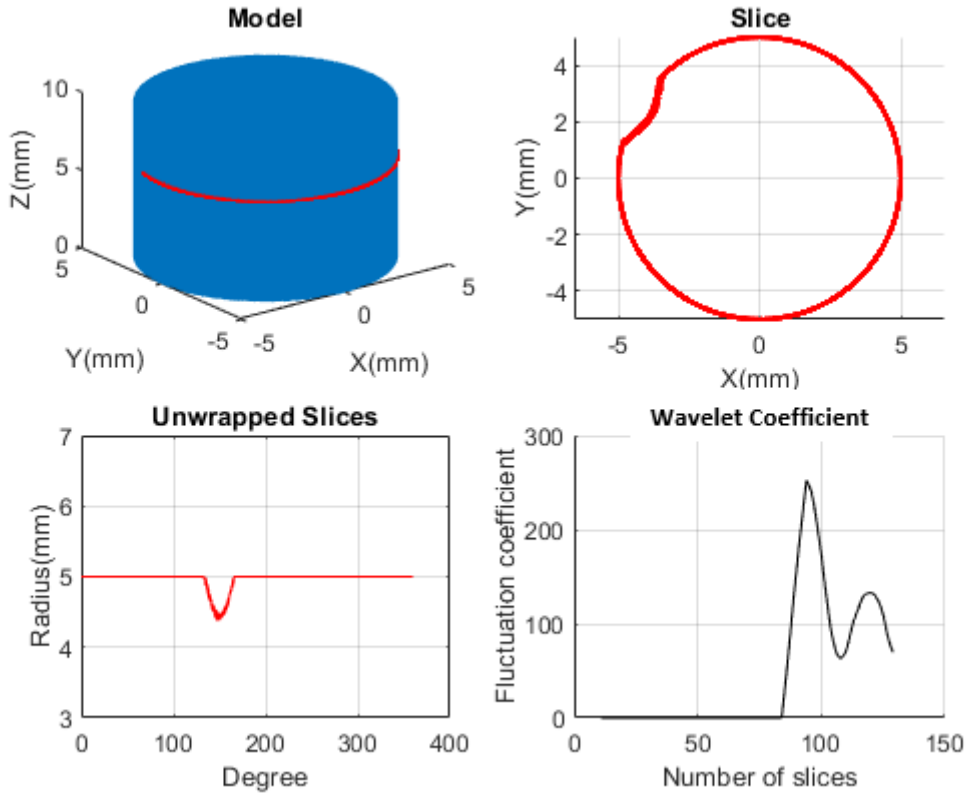


Figure 6.14: Generating slices using CWT representing the region of damage in the Morse CWT form and wavelet coefficient showing number of slices 7.5 mm from the base

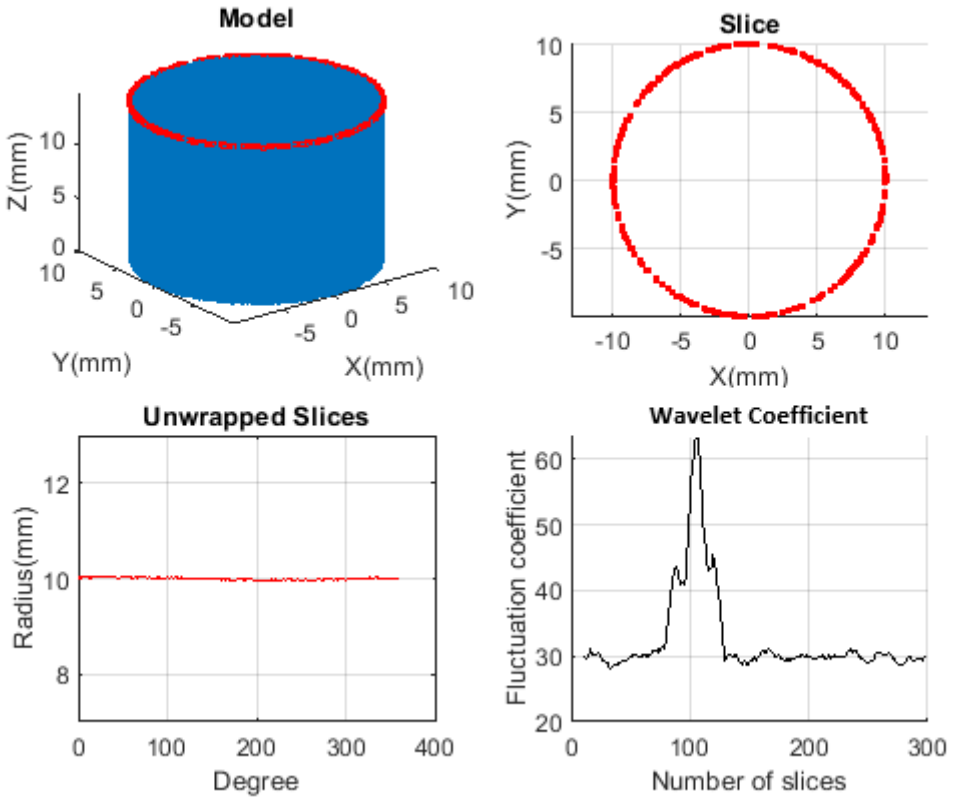


Figure 6.15: Generating slices using CWT showing final stage of the process

The output radius of each slice from this process varied and was further preprocessed using an interpolation algorithm, and the data is normalised to improve its properties in preparation for training. This is shown in Figure 6.16, where the top plot is the radius data before performing normalisation having unequal values for all slices, while the second plot is the regularised and interpolated data. Due to the unequal nature of the sampled numbers, the edges of the region of damage could not be clearly established. From the top plot, each column represents the data points captured from each slice as the location of the points are random. Each slice has a random number of points due to the randomly distributed nature of the point cloud. Regarding data preparation for LSTM, it is preferred that the data format for each slice be equal, i.e., the length be equal in the sequence. However, making the sequence length equal is not enough for a prepared data, but the resolution is regularised using an interpolation algorithm. This regularised data was used for training the LSTM.

The simulation of damage on the model can be randomised as well as specified by the user. For the experiment of Figure 6.16, the damage start angle was specified at 50˚, with a damage height of 5 mm representing the centre of the damage on the height axis. The computation of the algorithm used the damage start angle coupled with the slice resolution in evaluating the width of the damage seen as the blue patch. The randomised nature of the algorithm is illustrated in Figure 6.17 through to Figure 6.20, where the damage location changes with different damage start angle, height, and scanner noise.

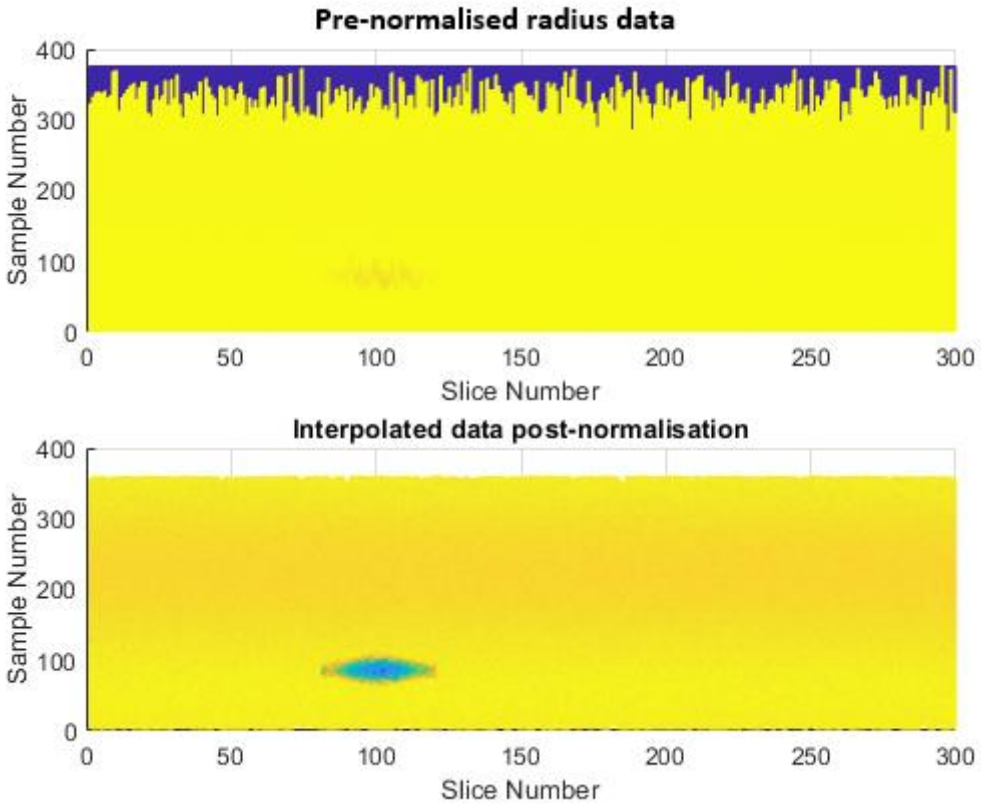


Figure 6.16: Before and after plot of the normalisation of the radius data

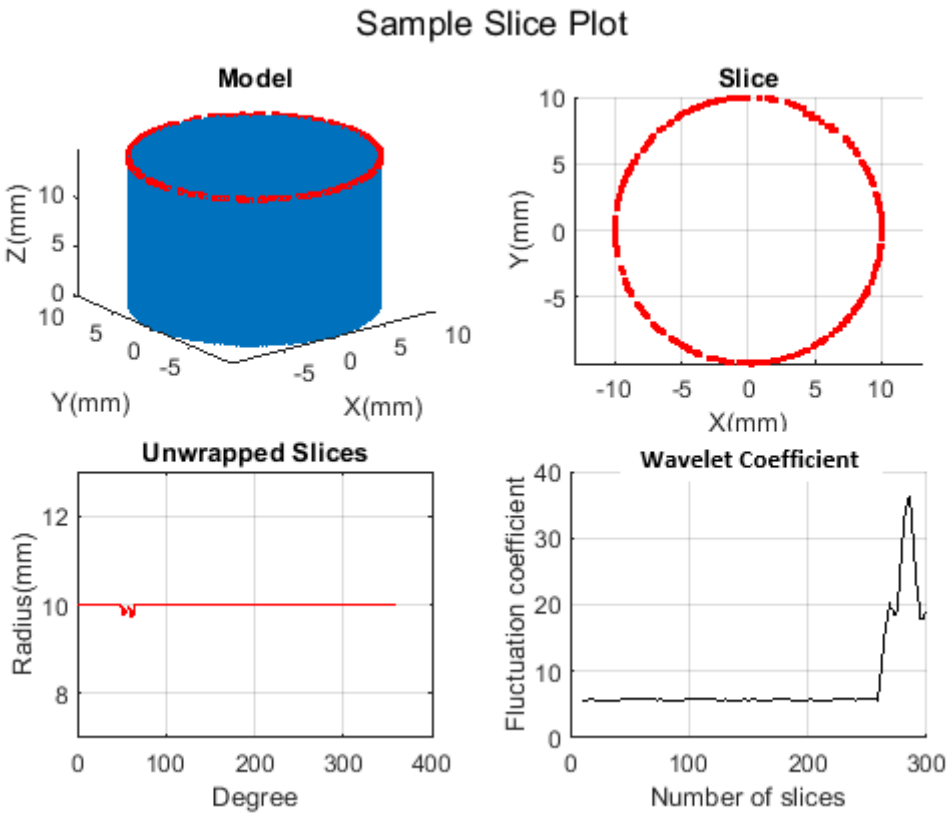


Figure 6.17: Generating slices using CWT using random parameters

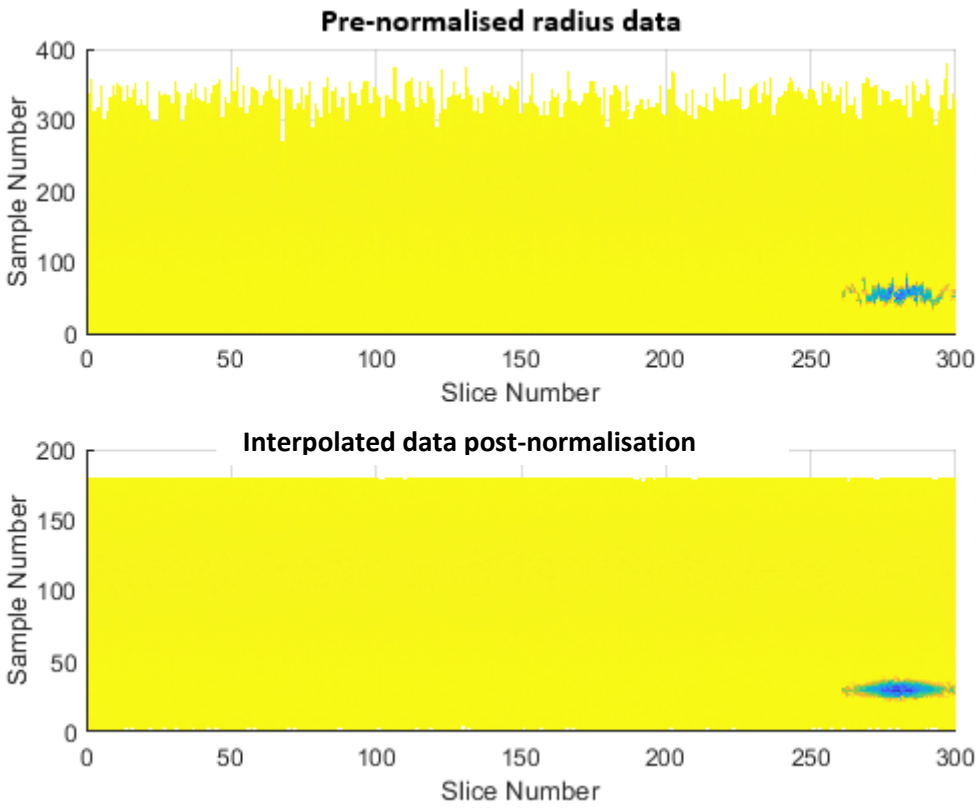


Figure 6.18: Regularised radius of random slice generation

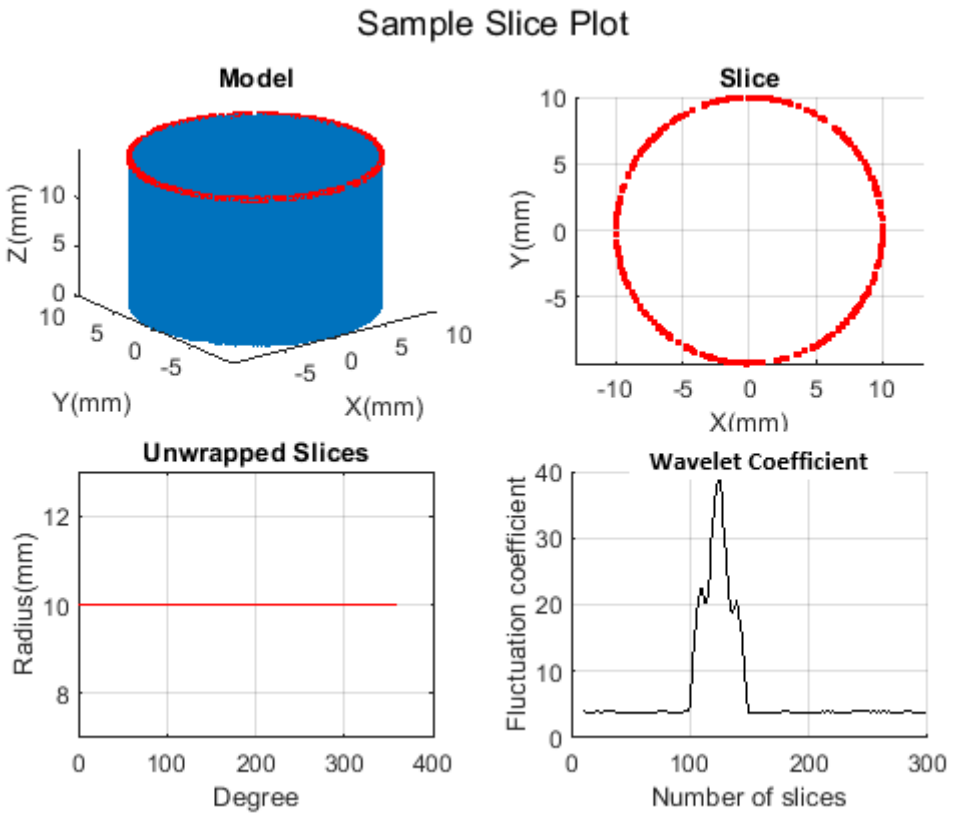


Figure 6.19: Generating slices using CWT using random parameters with change in damage location

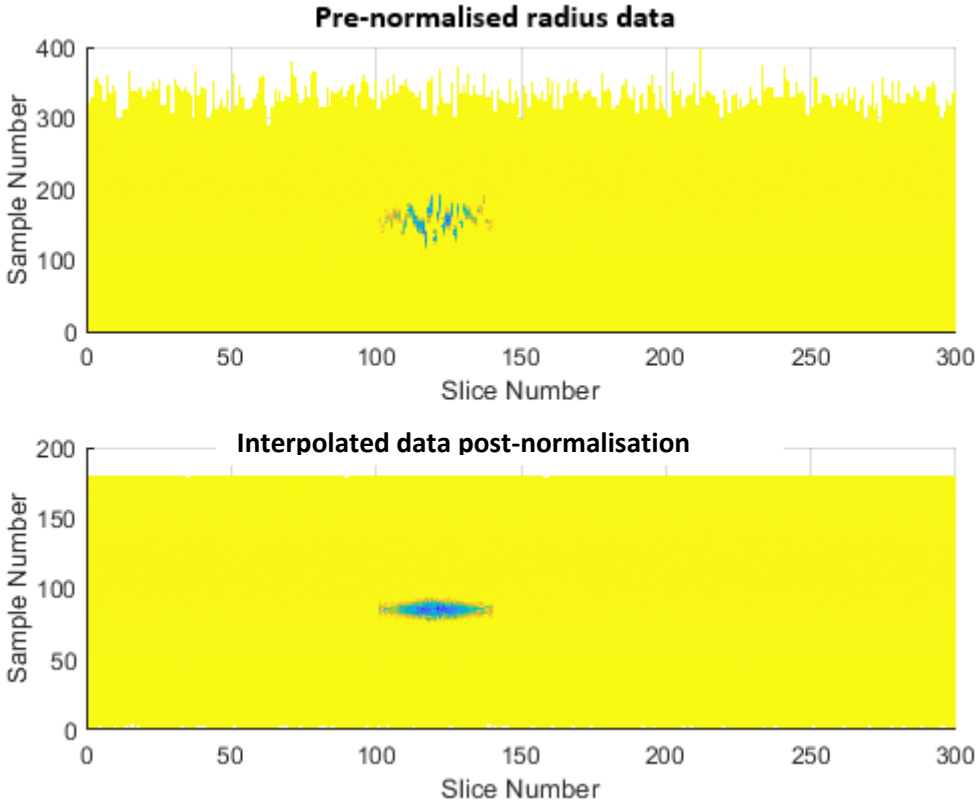


Figure 6.20: Regularised radius data randomly generated with change in damage location

For the scanner noise, the self-learning algorithm does not work if the noise hits the threshold of 0.05 mm detecting points distribution of symmetry value of 4. It is only suitable for instruments with noise levels lower than the threshold. If the damage is at a similar level to the scanner noise or superficial, the self-learning algorithm will not be able to find the damage. This was observed to be a cut-off threshold for situations where the algorithm would work and where it has no chance of detecting damage.

When the noise is increased, the damaged location can be clearly seen using the wavelet transform. This algorithm has a limitation of responding to the noise level. This depends on the magnitude of the damage being detected as the noise level will make no difference if the damage is massive, but it can become challenging with small-sized damage. This is illustrated in Figure 6.21 and Figure 6.22, where the coefficient denotes a point distribution of symmetry value range of 4, implying that no distinctive damage was detected, but there is damage present in Figure 6.22 just right above the 100 mm mark on the X-axis. Figure 6.23 illustrates the training process of the LSTM using the improved preprocessing method of wavelet

transform with a training accuracy of 99% with less overfitting. And Figure 6.24 shows the confusion chart of the classification result after training.

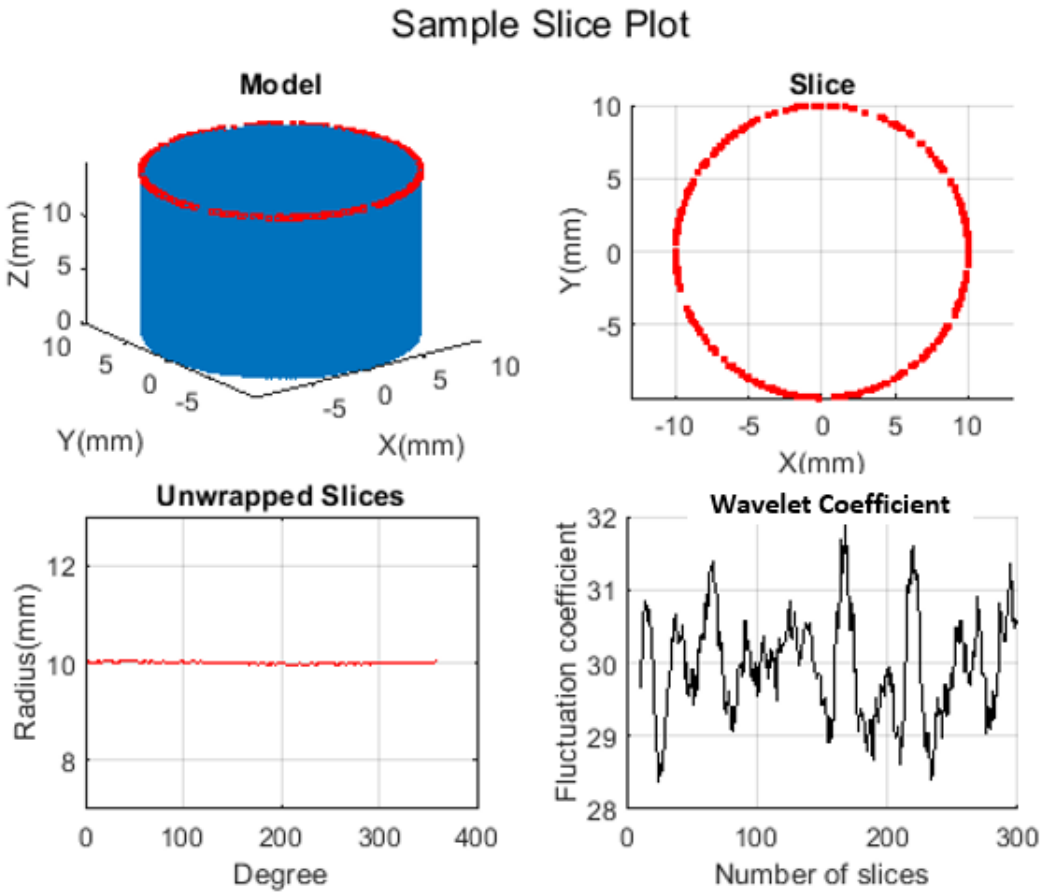


Figure 6.21: Generating slices using CWT showing system response to noise level considering magnitude of damage

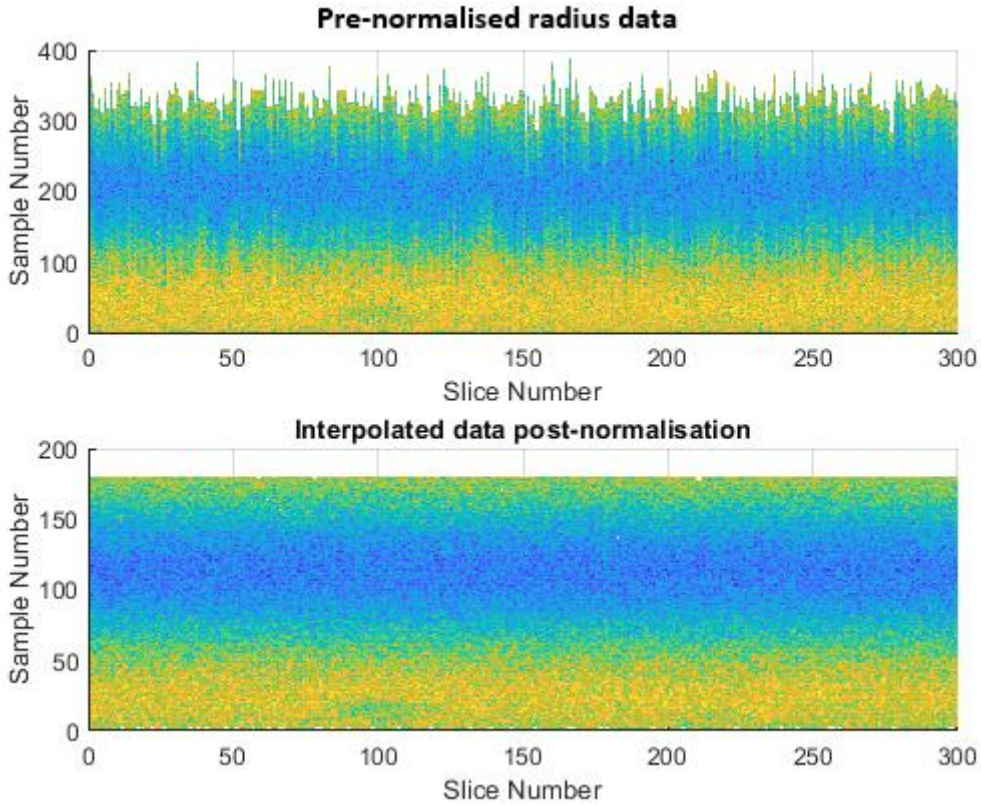


Figure 6.22: Regularised radius data of small damage

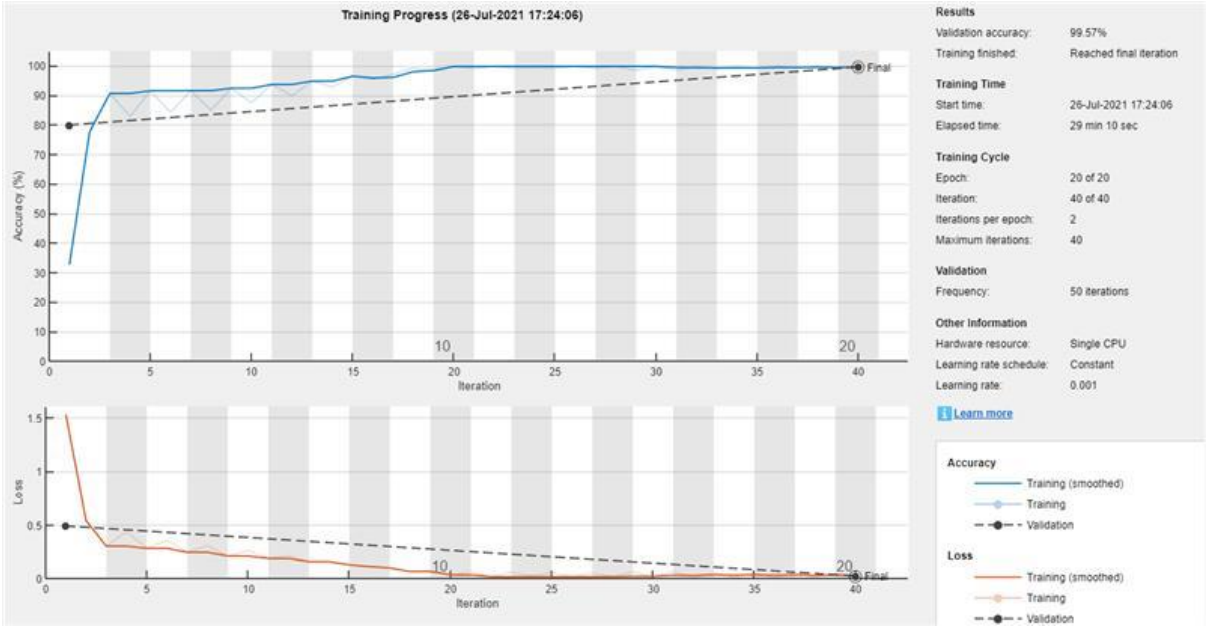


Figure 6.23: Training progress of the LSTM using wavelet transform data

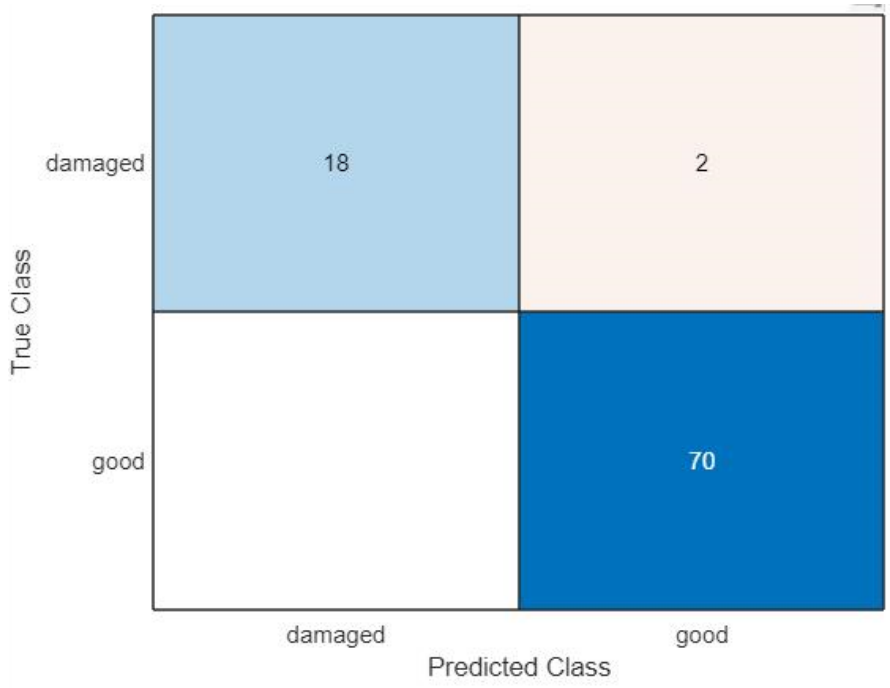


Figure 6.24: Confusion chart indicating the classification result of the LSTM using wavelet transform data

6.2.6 Application of LSTM Approach to Damage Detection

The core of the proposed approach after training the neural network is to test that the result from the network performs well when applied to a model. This was done by testing the network's classification result on a model to identify and extract the damaged slices with their location on the Z-axis of the model. The process starts from importing the data through to extracting information and is as follows

- Import a model that is void of noise and outliers after preprocessing but retains the induced damage
- Perform sectioning of the model to produce slices with a specified resolution, unwrap slices and automatically classify them into specific classes of 'good' or 'damaged.'
- Perform further preprocessing to improve the data structure in preparation for the LSTM
- The LSTM is trained with the slicing data to perform its classification of the model with continuous monitoring of its result and accuracy
- The LSTM classification data from the trained network is applied to the model to classify every slice, starting from the bottom of the model with that of the

classification data. The range of the damage is identified with actual values on the height (Z-axis) of the model. In developing NC codes for the milling operation for reconstruction, these values can be used in the CAM process of developing tool path, although this is out of scope for this project

Once the trained network and classification result is imported, the model coupled with how each slice is created on the model is visualised. Parameters such as height, starting point, axis limit, and assigned variables are specified. A loop is created to iterate the slice resolution on the model as shown in the flowchart of Figure 6.25, and once the circumference of a current slice is different from the previous, it correlates it with the network result to determine what category is the current slice. This process is continuous until the circularity of a current slice changes relative to the previous and is correlated with the network result to ascertain the class of the slice. With this approach, the range of where the damage starts and ends can be extracted with actual measurement information, as shown in Figure 6.26 and Figure 6.27. From the flowchart, the process starts by importing a noise free model, extraction of slices, normalisation of data, LSTM training, and application of data from LSTM network on the PCD model, outputting values that indicate the sizes of both simulated and predicted regions of good and damage. The rectangles to the left of the diamond and hexagonal shape provide more explanation to both steps.

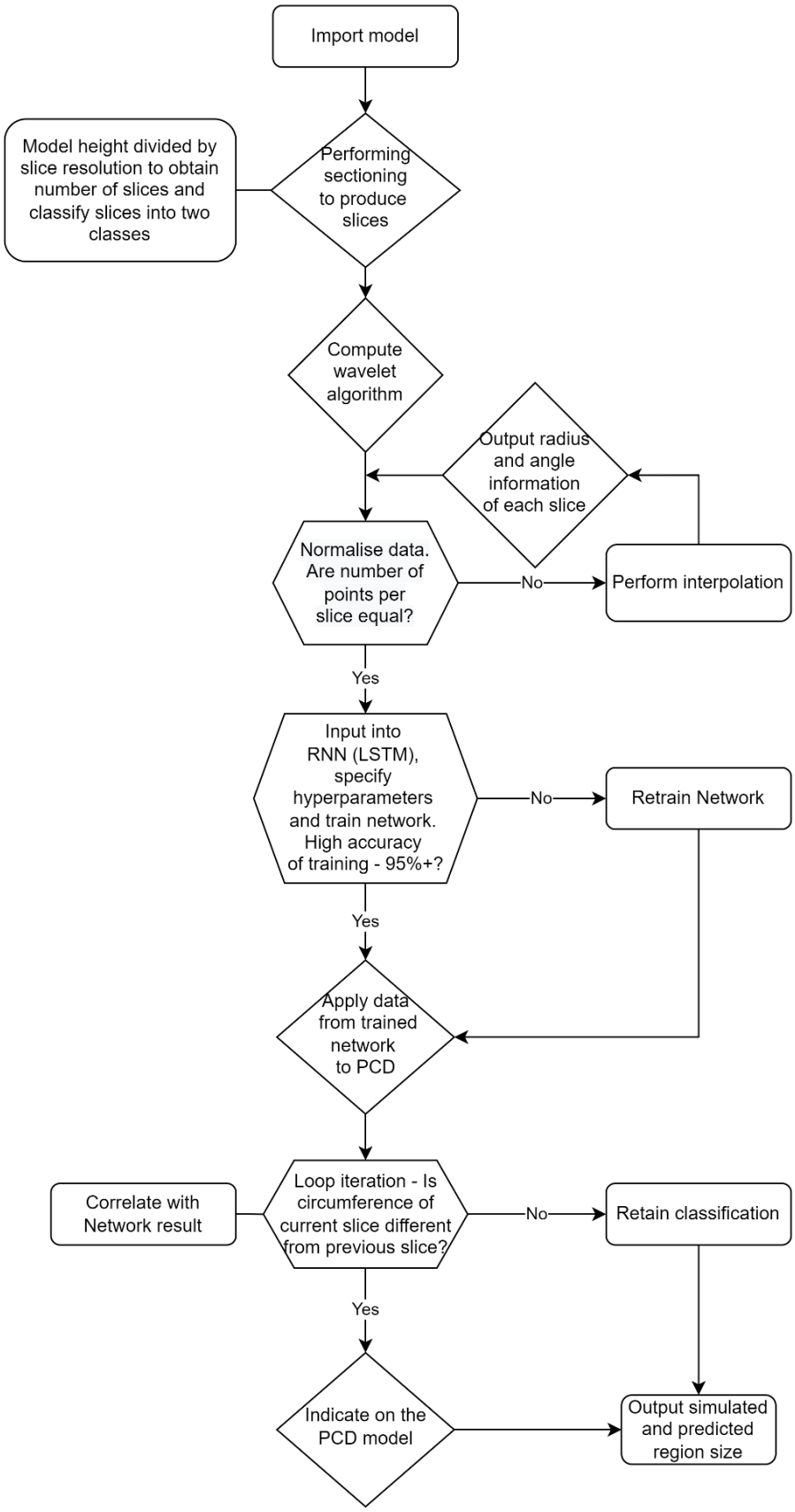


Figure 6.25: Flowchart indicating the process of applying LSTM training data to PCD model for damage identification

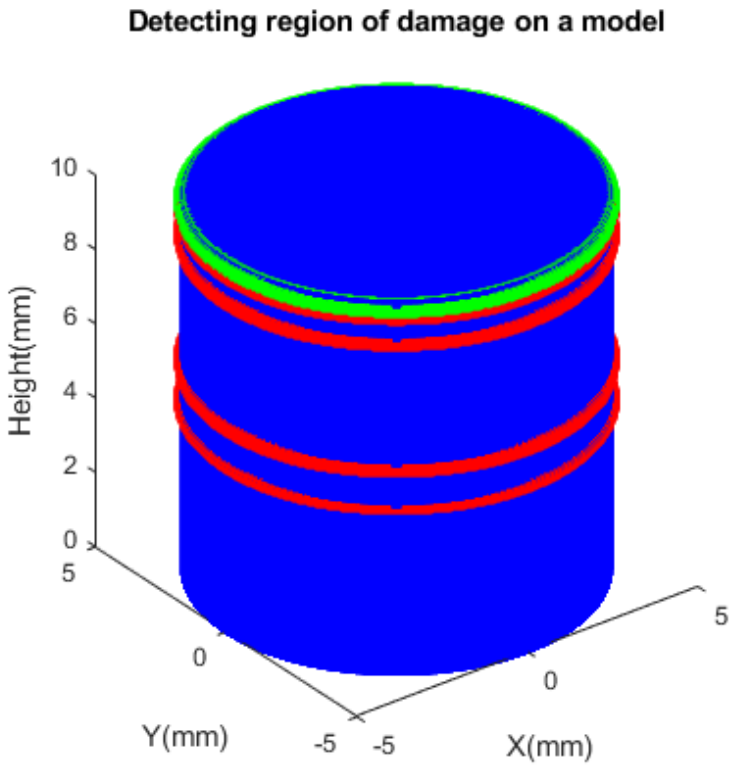


Figure 6.26: Application of LSTM result on ordered data

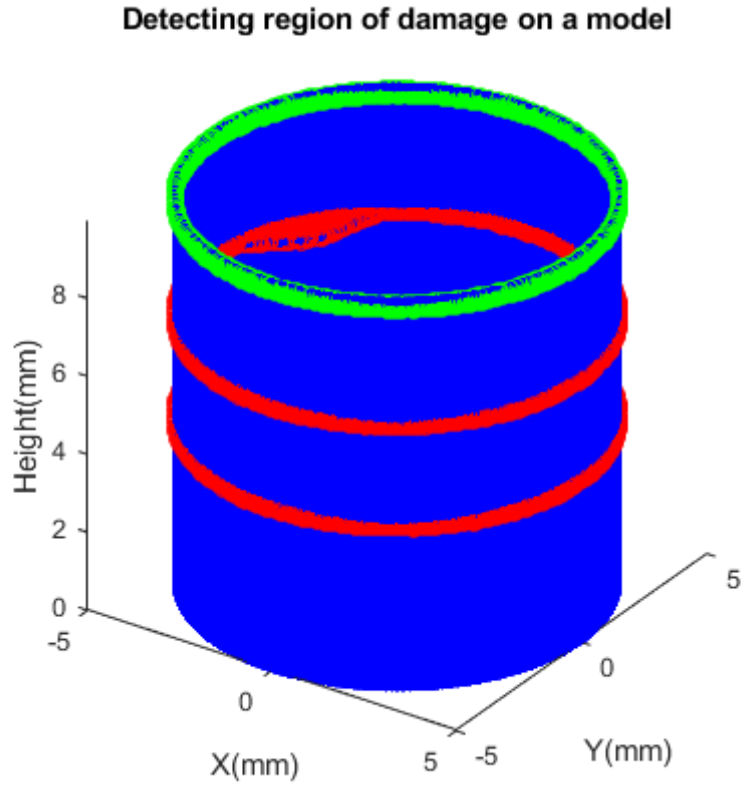


Figure 6.27: Application of LSTM result on randomly distributed simulation data

The region of damage after application of the LSTM result on the ordered data is illustrated in Figure 6.28 and Figure 6.29, respectively. It indicated that the damage started at 4.50 mm to 5.50 mm for the first region from the bottom, and the second region ranged from 9 mm to 9.70 mm. The green line at the top indicates there are good slices with the remaining 0.30 mm to the top. This method also performed well with randomly distributed points that replicate actual measurement data. In the random distribution data, the damage slices range from 4.20 mm to 6.80 mm on the Z-axis.

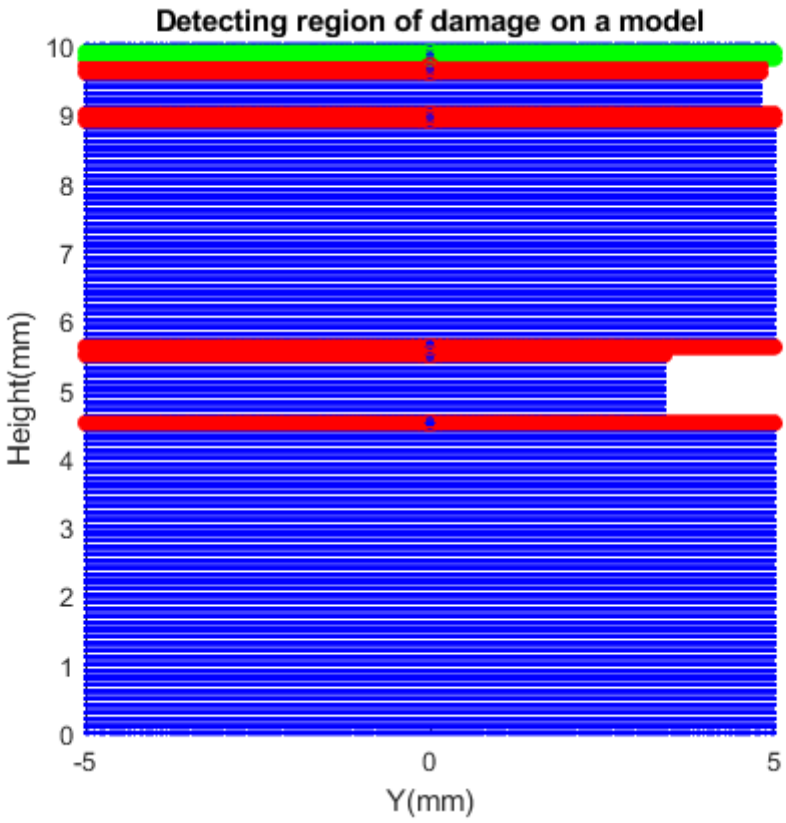


Figure 6.28: Visualisation of the damaged region on the ordered data set

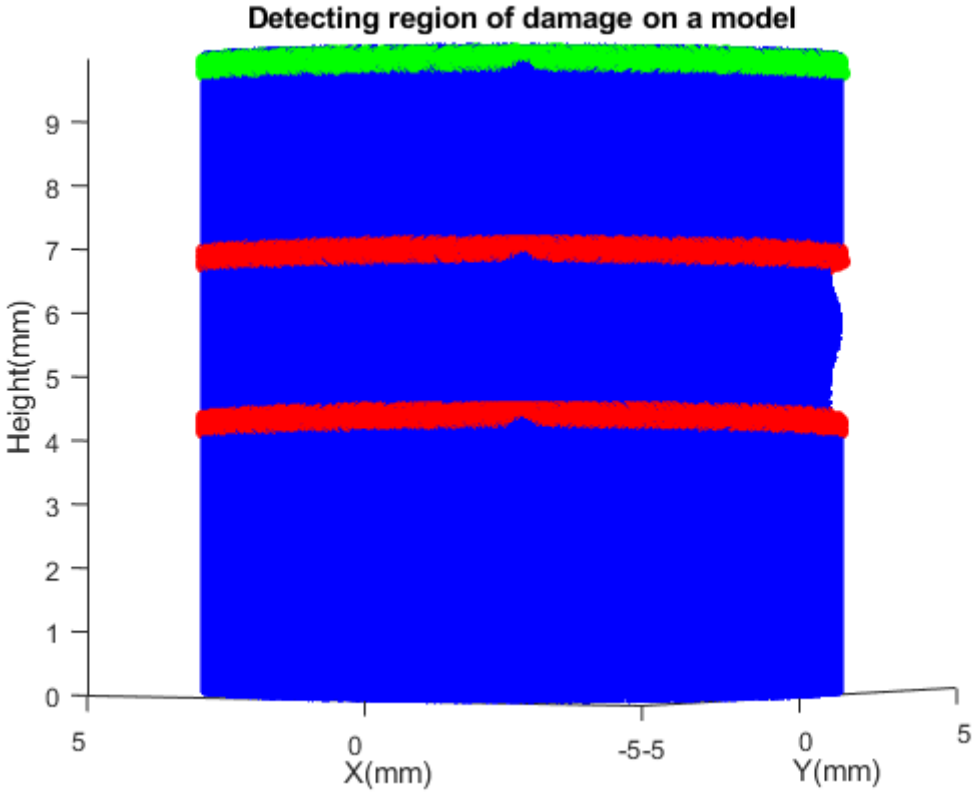


Figure 6.29: Visualisation of the damaged region on the random distribution data set

Further assessment of the developed damage detection algorithm is to test the behaviour and accuracy of the system when applied to models of different parameters. The slice resolution for this application is 0.05 mm. The resolution can be made coarser to improve calculation time but with less precise location information. However, resolutions below 0.05 mm will produce slices with fewer points and can eventually lose the structure of the model after slicing. It is therefore recommended that, for a fully automated system, the algorithm in this thesis could be used to hone in on the areas of interest and instruct the measurement system to take additional points for that region.

After performing network testing, the procedure described in the first paragraph of this section was implemented to extract the slice location where the damage region starts and where it ends. Although the end of the damage region was not indicated in the table, it can be calculated by subtracting the slice resolution from the location where the good region starts. For example, in **Error! Reference source not found.**, there are five regions of potential

damage indicated by the pair of red lines, these pairs specify slices where the damage starts and ends, and the next slice is a good slice from the classification.

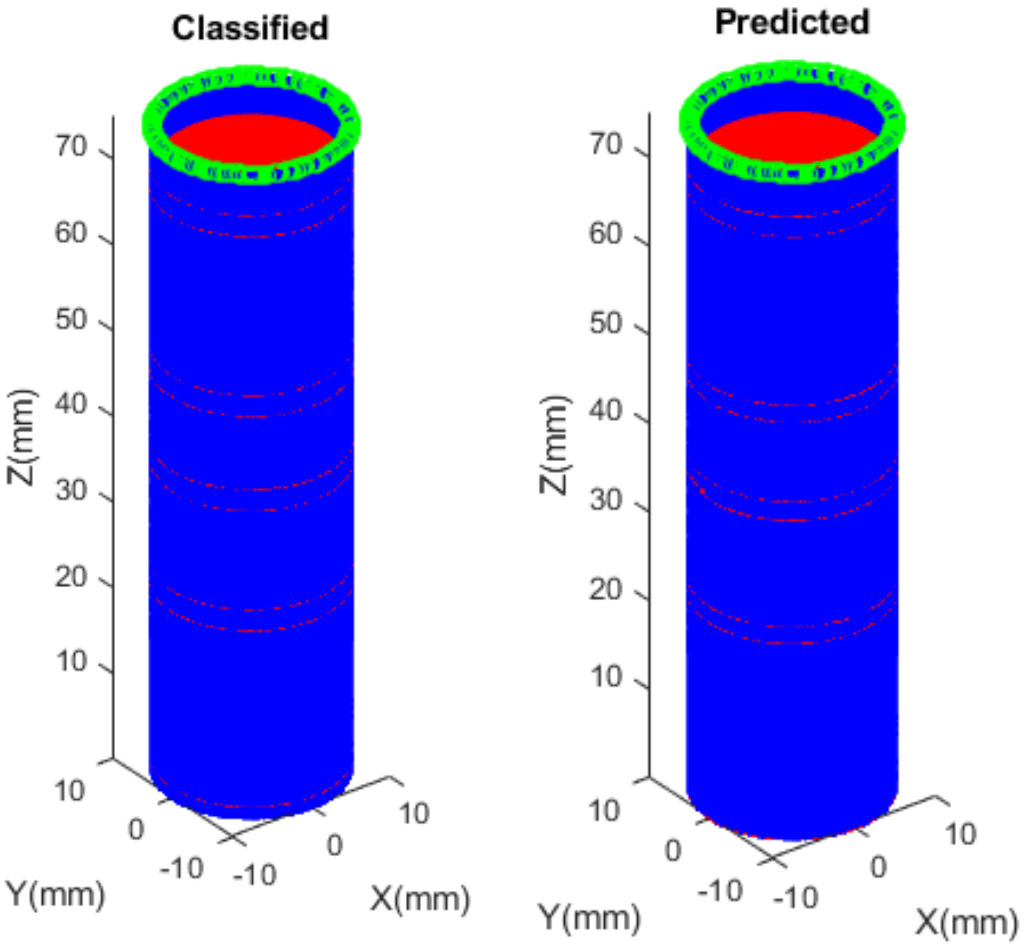


Figure 6.30: Comparing the classification of slices from both the simulated data and the LSTM prediction

Presented in Table. 6.1 through to Table 6.3 is the error of the LSTM prediction from the classified slices. Table 6.1 illustrate the prediction error of six different networks when applied to the model on which they have been trained. The locations of damage were randomly generated, trained, and the prediction was tested on the same models. The simulation column was specified as the standard (known value) of which the prediction was compared. For networks with no values at the start indicates that the damage started at the bottom of the model. The "classification accuracy" column shown is the network's accuracy after applying the prediction result. This is calculated by equating the network's classification to the training data and performing a right array division of its sum by the number of elements in the training data. The values in the simulated and predicted columns are the locations that

indicate where the damage regions starts and where the good region starts. The residual error was calculated by computing the difference between the prediction and known values. These networks are 8-layered networks with variations in their dropout probability and training options such as maximum epochs and learning rate.

For **Error! Reference source not found.**, five different networks were tested on cloud point data independent of the training data to test its accuracy of performing prediction when presented with an external data set. The first three networks were tested on models having similar parameters as to trained models, while network 4 was tested with a model bigger in size than the network's trained model. The reverse is the case with network 5 being tested on a model smaller than the network's trained model. This was to evaluate the networks behaviour and ability to perform prediction when presented with different data types, and their accuracy after classification is shown in the last column. Network 4 appears to have not detected damage in the first two regions from the bottom of the model during prediction; however, it has an accuracy of 98% during training. This missed detection of damaged regions is due to the size of the training data for that network. However, this situation also affected testing network 5 on a smaller model. Table 6.3 illustrates the testing of a single network on several data types having various parameters. The network is an 8-layered network with a double LSTM layer to make the network deeper, and each LSTM layer is accompanied by a dropout layer of dropout probability of 0.2 to prevent overfitting. The prediction accuracy of the network was observed to remain constant despite the different models being tested. This accuracy is equivalent to the initial training accuracy of the network.

Table 6.1: Error calculation of the machine learning prediction of the location of damage using the same trained models for testing on the same networks

Experiments		Simulated	Predicted	Residual Error	Classification Accuracy	Generated data with randomly generated damage location/model parameters specified
LSTM1	Damaged start	7.05 mm	7.15 mm	+0.1 mm	82%	
	Good start	9.50 mm	9.10 mm	-0.4 mm		
LSTM2	Damaged start	1.05 mm	1.20 mm	+0.15 mm	73%	
	Good start	3.50 mm	3.50 mm	-		
LSTM3	Damaged start	1.49 mm	1.64 mm	+0.15 mm	91%	
		15.15 mm	15.30 mm	+0.15 mm		
	Good start	1.50 mm	1.65 mm	+0.15 mm		
		17.50 mm	17.40 mm	-0.1 mm		
LSTM4	Damaged start	-	-	-	91%	
		15.05 mm	15.15 mm	+0.1 mm		
	Good start	1.5 mm	-	-1.5 mm		
		16.50 mm	16.35 mm	-0.15 mm		
LSTM5	Damaged start	0.10 mm	-	-0.1 mm	94%	
		24.05 mm	24.10 mm	+0.05 mm		
	Good start	2.50 mm	2.35 mm	-0.15 mm		
		26.50 mm	26.40 mm	-0.1 mm		
LSTM6	Damaged start	4.05 mm	4.10 mm	+0.05 mm		
		29.05 mm	29.15 mm	+0.1 mm		
		39.05 mm	39.05 mm	-		
		53.05 mm	53.10 mm	+0.05 mm		

		71.05 mm	71.10 mm	+0.05 mm	95%
	Good start	6.50 mm	6.20 mm	-0.3 mm	
		30.05 mm	29.75 mm	-0.3 mm	
		41.50 mm	41.30 mm	-0.2 mm	
		55.50 mm	55.45 mm	-0.05 mm	
		73.50 mm	73.25 mm	-0.25 mm	

Table 6.2: Error calculation of the machine learning prediction of the location of damage using different models for testing different networks

Experiments		Simulated	Predicted	Residual Error	Classification Accuracy	
LSTM1	Damage start	4.05 mm	4.50 mm	+0.45 mm	94%	
		17.05 mm	17.60 mm	+0.55 mm		
	Good start	6.45 mm	5.85 mm	-0.6 mm		
		19.50 mm	18.80 mm	-0.7 mm		
LSTM2	Damage start	-	-	-	94%	Rows with no values indicate that the damage started from the bottom of the model
		22.05 mm	22.25 mm	+0.2 mm		
		36.05 mm	36.15 mm	+0.1 mm		
		47.05 mm	47.20 mm	+0.15 mm		
		68.05 mm	68.15 mm	+0.1 mm		
	Good starts	1.5 mm	0.10 mm	-1.4 mm		
		24.50 mm	24.10 mm	-0.4 mm		
		38.50 mm	38.25 mm	-0.25 mm		
		49.50 mm	49.10 mm	-0.4 mm		
		70.50 mm	70.40 mm	-0.1 mm		
LSTM3	Damage start	5.05 mm	5.59 mm	+0.5 mm	94%	No Values indicates the end of model
		28.05 mm	28.70 mm	+0.65 mm		
	Good start	7.50 mm	7.10 mm	-0.4 mm		
		-	-	-		
LSTM4	Damage start	4.05 mm	-	-4.05 mm	98%	Empty rows indicate no damage detected.
		29.05 mm	-	-29.05 mm		
		39.05 mm	39.30 mm	+0.25 mm		
		53.05 mm	53.30 mm	+0.25 mm		
		71.05 mm	71.20 mm	+0.15 mm		
	Good starts	6.50 mm	-	-6.50 mm		
		30.05 mm	-	-30.05 mm		
		41.50 mm	41.15 mm	-0.35 mm		
		55.50 mm	55.10 mm	-0.4 mm		

		73.50 mm	73.20 mm	-0.3 mm		
LSTM5	Damage start	-	11.25 mm	+11.25 mm	96%	
		20.05 mm	20.20 mm	+0.15 mm		
		36.05 mm	36.15 mm	+0.1 mm		
	Good start	15.05 mm	11.85 mm	-3.2 mm		
		22.50 mm	21.30 mm	-1.2 mm		
		38.50 mm	38.10 mm	-0.4 mm		

Table 6.3: Error calculation of the machine learning prediction of the location of damage using different models for testing a single network

Experiments		Simulated	Predicted	Residual Error	Classification Accuracy	
LSTM Testing 1	Damage start	6.05 mm	6.15 mm	+0.1 mm	96%	
	Good start	8.50 mm	8.35 mm	-0.15 mm		
LSTM Testing 2	Damage start	4.05 mm	4.15 mm	+0.1 mm	96%	
		29.05 mm	29.10 mm	+0.05 mm		
		39.05 mm	39.15 mm	+0.1 mm		
		53.05 mm	53.35 mm	+0.3 mm		
		71.05 mm	71.25 mm	+0.2 mm		
	Good start	6.50 mm	6.30 mm	-0.2 mm		
		30.05 mm	30.20 mm	+0.15 mm		
		41.50 mm	41.40 mm	-0.1 mm		
		55.50 mm	55.05 mm	-0.45 mm		
		73.50 mm	73.20 mm	-0.3 mm		
LSTM Testing 3	Damage start	-	-	-	96%	The first damage started at the bottom
		22.05 mm	22.20 mm	+0.15 mm		
		36.05 mm	36.15 mm	+0.1 mm		
		47.05 mm	47.25 mm	+0.2 mm		
		68.05 mm	68.20 mm	+0.15 mm		
	Good start	1.50 mm	0.70 mm	-0.8 mm		
		24.50 mm	23.70 mm	-0.8 mm		
		38.50 mm	38.35 mm	-0.15 mm		
		49.50 mm	49.10 mm	-0.4 mm		
		70.50	70.05 mm	-0.45 mm		
LSTM Testing 4	Damage start	4.05 mm	4.25 mm	+0.2 mm	96%	
		17.05 mm	17.15 mm	+0.1 mm		
		6.45 mm	6.05 mm	-0.4 mm		

	Good start	19.50 mm	19.10 mm	-0.4 mm	
LSTM Testing 5	Damage start	2.05 mm	2.15 mm	+0.1 mm	96%
		21.05 mm	21.15 mm	+0.1 mm	
		34.05 mm	34.10 mm	+0.05 mm	
		53.05 mm	53.25 mm	+0.2 mm	
		60.10 mm	60.15 mm	+0.05 mm	
	Good start	4.50 mm	2.90 mm	-1.6 mm	
		23.50 mm	23.40 mm	-0.1 mm	
		36.50 mm	34.95 mm	-1.55 mm	
		55.50 mm	55.05 mm	-0.45 mm	
		62.50 mm	62.25 mm	-0.25 mm	

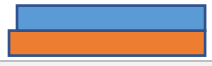
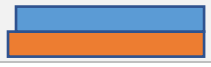
Isolating a few networks to investigate the size of damage, both simulated and predicted damage region sizes were calculated to investigate the level of deviation of the prediction, and this is illustrated in Table 6.4. The blue bar represents the simulated region size, while the amber bar represents the size of the predicted damage region. For the first network, the size of the prediction on the damage region shrank by 0.5 mm to the simulated region on both margins. It detected a damage slice 0.1 mm after the simulated region and detected a good slice 0.4 mm before the simulated. For network 3 damage region 1, the total deviation of the prediction against the simulated was 0.1 mm. The prediction detected damage before the simulated as shown on the left margin and shrank by 0.05 mm on the right margin.

These deviations occur at the transition between good and damaged regions due to the nature of the points, the slice resolution, and the accuracy obtainable from training the neural network. At the transition point, the structure of the slice can be difficult to differentiate as the location of the points are beginning to change. This is not the case at the centre of the damage as the structure is distinctive. An approach to further investigate this region is to reduce the resolution at the transition point, but this is not without its challenge. Reducing

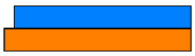
the slice resolution will reduce the number of points in a slice, and this might not be enough to represent the structure of the model when training the LSTM.

Table 6.4: Deviation of the neural network prediction with the blue bar representing the simulated range and the amber bar represents the predicted range

Experiments		Simulated Region Size	Predicted Region Size	Residual Error	Margin Visualisation
Net1	Damage	2.4 mm	1.9 mm	-0.5 mm	
	Good	5.5 mm	5.9 mm	+0.4 mm	
Net3	Damage	2.4 mm	2.15 mm	-0.25 mm	
	Good	6.5 mm	6.65 mm	+0.15 mm	
Net6	Damage	4.05 mm	4.10 mm	+0.05 mm	
		29.05 mm	29.15 mm	+0.05 mm	
		39.05 mm	39.05 mm	-	
		53.05 mm	53.10 mm	+0.05 mm	
		71.05 mm	71.10 mm	+0.05 mm	
	Good	6.50 mm	6.20 mm	-0.3 mm	

		30.05 mm	29.75 mm	-0.3 mm		
		41.50 mm	41.30 mm	-0.2 mm		
		55.50 mm	55.45 mm	-0.05 mm		
		73.50 mm	73.25 mm	-0.25 mm		

Experiments			**Simulated Region Size**	**Predicted Region Size**	**Residual Error**	**Margin Visualisation**	
Net 1	Damage		2.4 mm	1.9 mm	-0.5 mm		 Simulated range Predicted range
	Good		5.5 mm	5.9 mm	+0.4 mm		
Net 2	Damage		2.4 mm	2.15 mm	-0.25 mm		
	Good		6.5 mm	6.65 mm	+0.15 mm		
Net 3	Damage	Region 1	2.35 mm	2.30 mm	-0.1 mm		
		Region 2	2.4 mm	2.25 mm	-0.15		
	Good	Region 1	21.5 mm	23.7 mm	-2.2 mm		

		Region 2	3.5 mm	3.6 mm	+0.1 mm		

To perform cross-validation of the performance of some LSTM networks, external data independent from the trained model were used in testing an already trained network to access its accuracy of classification when fed with a different model from which it was trained. Table 6.5 below shows the classification accuracy of eight (8) trained networks tested with a different model with the plot of all eight networks shown in Figure 6.31. Most networks performed well to a minimum of 79% except for network 7 with low accuracy due to training parameters producing a misclassification of the data set, hence its accuracy.

Table 6.5: Accuracy of cross-validation of eight LSTM networks using different models

	Data1	**Data2**	**Data3**	**Data4**	**Data5**	**Data6**	**Data7**	**Data8**
Net1	99%	84%	96%	93%	90%	79%	83%	92%
Net2	84%	98%	90%	89%	93%	85%	97%	94%
Net3	99%	97%	98%	97%	94%	84%	98%	94%
Net4	98%	98%	98%	98%	97%	86%	98%	95%
Net5	97%	98%	97%	96%	98%	84%	97%	95%
Net6	99%	98%	98%	90%	94%	95%	97%	98%
Net7	65%	72%	60%	62%	75%	68%	70%	73%
Net8	99%	84%	98%	95%	93%	83%	92%	95%

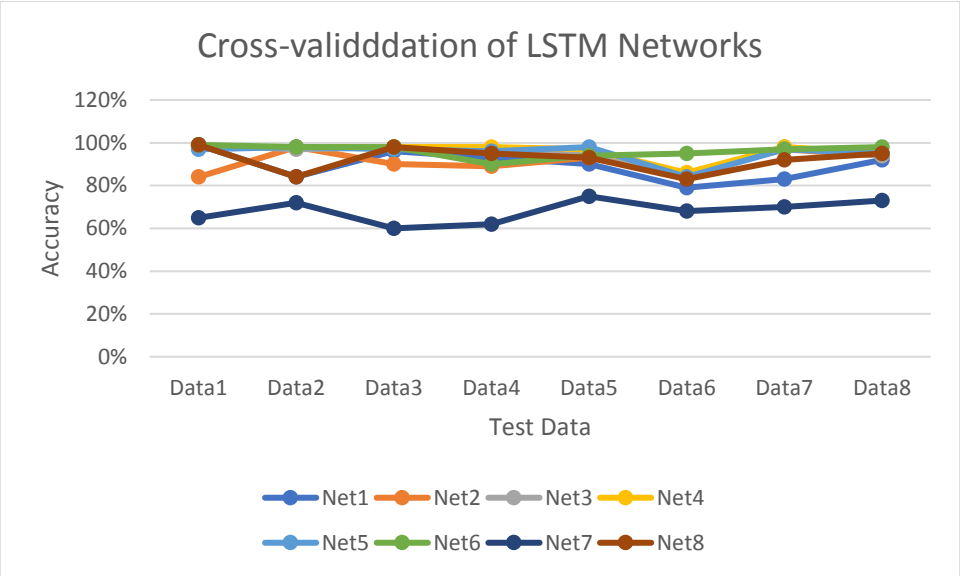


Figure 6.31: Cross-validation of LSTM Network

6.3 Profile Analysis

The sectioning was one way of extracting data for training the LSTM neural network. This method produced slices of specified dimensions along the height axis of the model. A new approach was developed for extracting the profile of the model for training the LSTM.

Figure 6.32 depicts the approach that has been developed. Preprocessing aids in the reduction of the scanned point set as well as the elimination of noisy data and redundant points. This procedure may generate a collection of data that may still be dense and contain redundant points. The dense data is further reduced to achieve system efficiency by using sampling modules to remove redundant points and obtain an improved data collection. A data reduction technique as discussed in section 3.3.1.1 was used to reduce the number of points while maintaining enough points to represent the model for accomplishing this approach. The scan is then profiled by centralization along the X and Y axes. The scan direction of view is altered, and the model's profile is projected from the centre point. The rotation of axes or a model relative to fixed axes are two conventions that can be accomplished in a rotational matrix, but the axes are rotated relative to the model for this project.

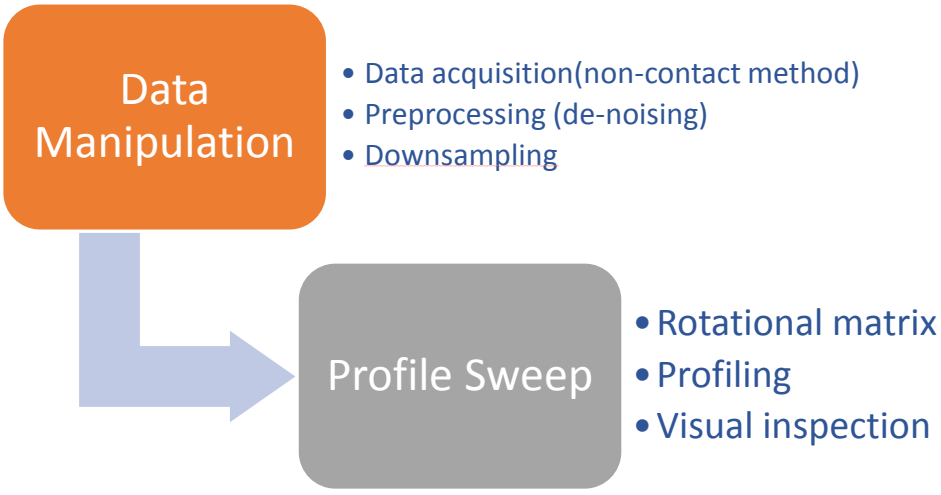


Figure 6.32: A proposed approach for performing profile analysis

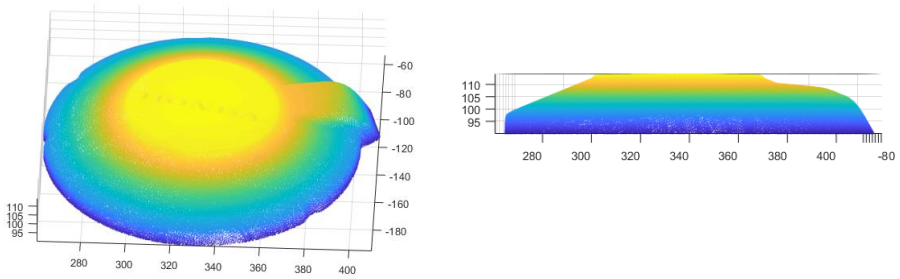


Figure 6.33: (a) top projection of the model, (b) side projection of the model

The projection of the point cloud from Figure 6.33 was plotted from 90 mm to the top, leaving out the base to make the application run faster with the necessary data required for analysis. The concentration for this work is on the region with the damage. This approach is particularly important to save time and use less computing power as other regions might not be relevant to achieving the desired result, resulting in the processing of a massive amount of data not needed. Figure 6.33 depicts the damaged part of the model. Further preprocessing was performed by decimating the points in the model, thereby eliminating outliers and noise, resulting in a much simpler model with extra care to preserve the topology of the model's surface. As shown in Figure 6.34, a rotational matrix about the Z-axis was applied to the model by centralising the X and Y axes using coordinate transformation and rotating θ about the z-axis. This is a transformation matrix used to rotate an axis about a given point, and the centre of a cartesian coordinate frame is used as the rotation point. The rotational matrix functionality is applied to vectors which lead to rotating vectors while coordinates axes are fixed. This concept of having a rotating vector at a fixed coordinate is known as active transformation.

Given a vector of position x_o, y_o at an angle θ and radius r, using Pythagoras theorem,

$$\cos\theta = \frac{x_o}{r}\ , sin\theta = \frac{y_o}{r}$$

$$x_o = rcos\theta, y_o = rsin\theta \qquad (6.1)$$

If the vector is rotated counter clockwise (θ is positive) at an angle θ' to give a new position of x_1, y_1, the angle of the vector at this position is $(\theta + \theta')$

$$x_1 = rcos(\theta + \theta')\,, y_1 = rsin(\theta + \theta') \tag{6.2}$$

Using trigonometric identities

$$x_1 = rcos(\theta + \theta') \tag{6.3}$$

$$= r(cos\theta cos\theta' - sin\theta sin\theta')$$

$$= rcos\theta cos\theta' - rsin\theta sin\theta'$$

$$x_1 = x_o cos\theta' - y_o sin\theta' \quad , y_1 = rsin(\theta + \theta') \tag{6.4}$$

$$= r(sin\theta cos\theta' + cos\theta sin\theta')$$

$$= rsin\theta cos\theta' + rcos\theta sin\theta')$$

$$y_1 = y_o cos\theta' - x_o sin\theta' \tag{6.5}$$

$$x_1 = x_o cos\theta' - y_o sin\theta'$$

$$y_1 = y_o cos\theta' - x_o sin\theta'$$

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} cos\theta' & -sin\theta' \\ sin\theta' & cos\theta' \end{bmatrix} \begin{bmatrix} x_o \\ y_o \end{bmatrix}$$

(6.6)

The real 2x2 special orthogonal matrix shows how the x-y plane rotates by θ measured in the positive X-axis in a counter clockwise rotation.

$$R(\theta) = \begin{bmatrix} cos\theta' & -sin\theta' \\ sin\theta' & cos\theta' \end{bmatrix} \tag{6.7}$$

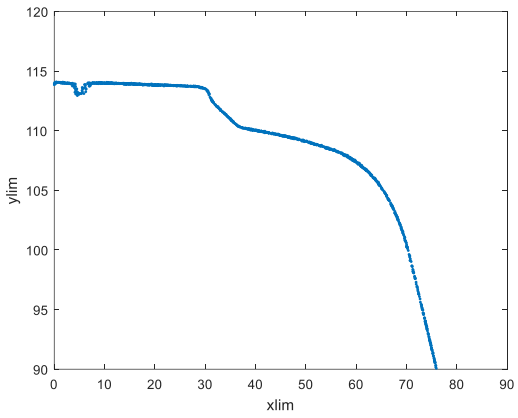
In three-dimensional space, the matrix is represented by a real 3x3 orthogonal matrix, and the rotation of the model is about one of the 3-axes of a coordinate system. For this model, the rotational matrix is rotated about the Z-axis in Equation (6.10)

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{bmatrix} \tag{6.8}$$

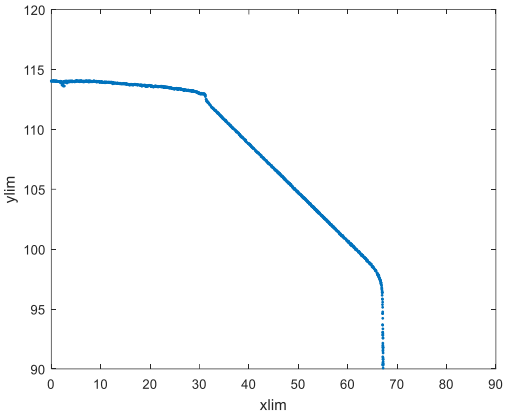
$$R_y(\theta) = \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix} \tag{6.9}$$

$$R_z(\theta) = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (6.10)$$

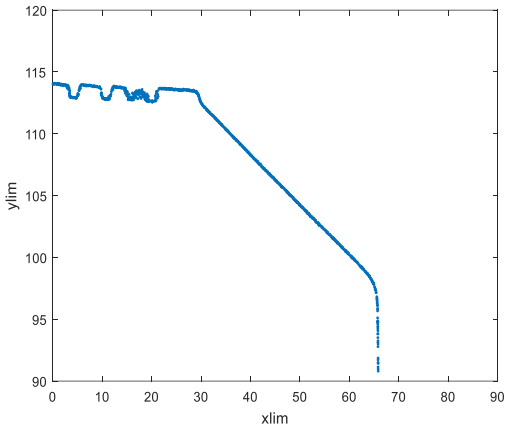
A 2D profile line of the model is drawn to illustrate how the matrix sweeps through it, revealing its boundary. The matrix sweeps in a counter clockwise direction as θ is positive, and a negative θ will sweep the matrix in the opposite direction. The profiling of the model in Figure 6.33 is shown below in Figure 6.34. The profile in Figure 6.34 (a) and (c) indicates a good region of the model, while (b) and (d) at the top displays some deviations in the profile, indicating a damaged region. Again, there could be questions about the profile (c), but due to prior knowledge and visual inspection of the part, this profile is assessed as good, and the deviations at the top are the text engraved onto the part. This conclusion provides less valuable evaluation when applied to parts with complex geometries, hence the need for further investigation into more ways to draw concise and reasonable conclusions.



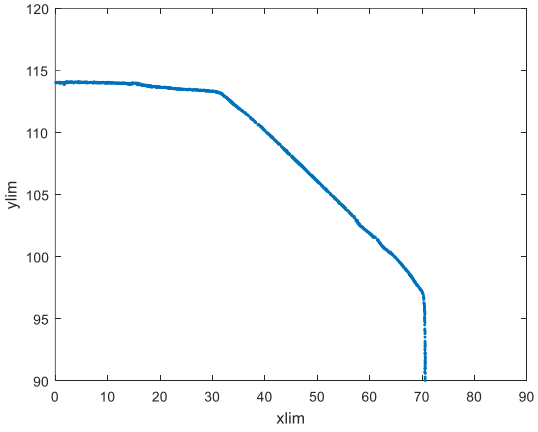
(a)



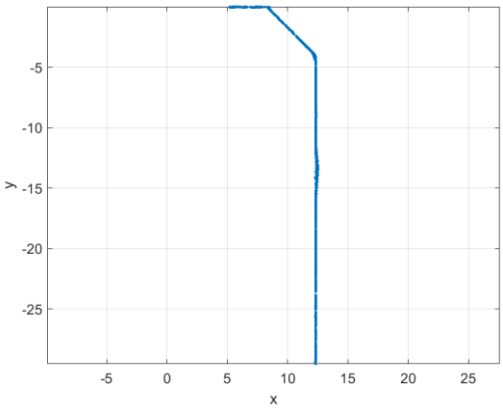
(b)



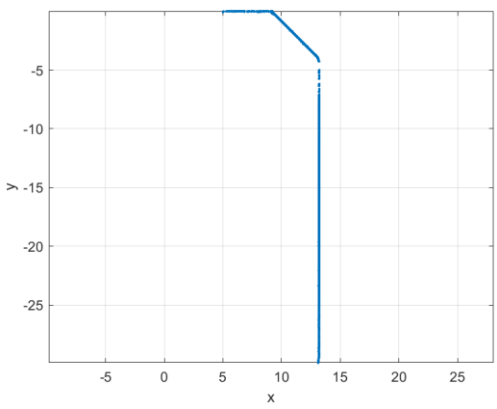
(c)



(d)



(e)



(f)

Figure 6.34: Profile of the model at different sections

Further analysis into profiling is shown in Figure 6.35. In situations where there is no prior knowledge or visual inspection of the part, the analysis can be deployed in deducing valuable information and conclusions, such as the region on the model where the damage from the profile display is located. This method pinpoints the profile at every point on the data and is plotted as a 2D profile in a subplot shown in Figure 6.35, and this verifies the work by Kai and Kemao [129] and other researchers described in section 2.7.

Most profile evaluations are based on fringe projection using phase changing interferometry and Fourier transform profilometry [159, 160]. These methods only create fringe profiles on the surface of a part that is unsuitable for the goal of this project in detecting and localising damage; hence, further analysis is required. Therefore, this proposed method is another approach for getting the data rather than sectioning; this has been shown with these graphs below that more information that can be used for training the LSTM can be generated.

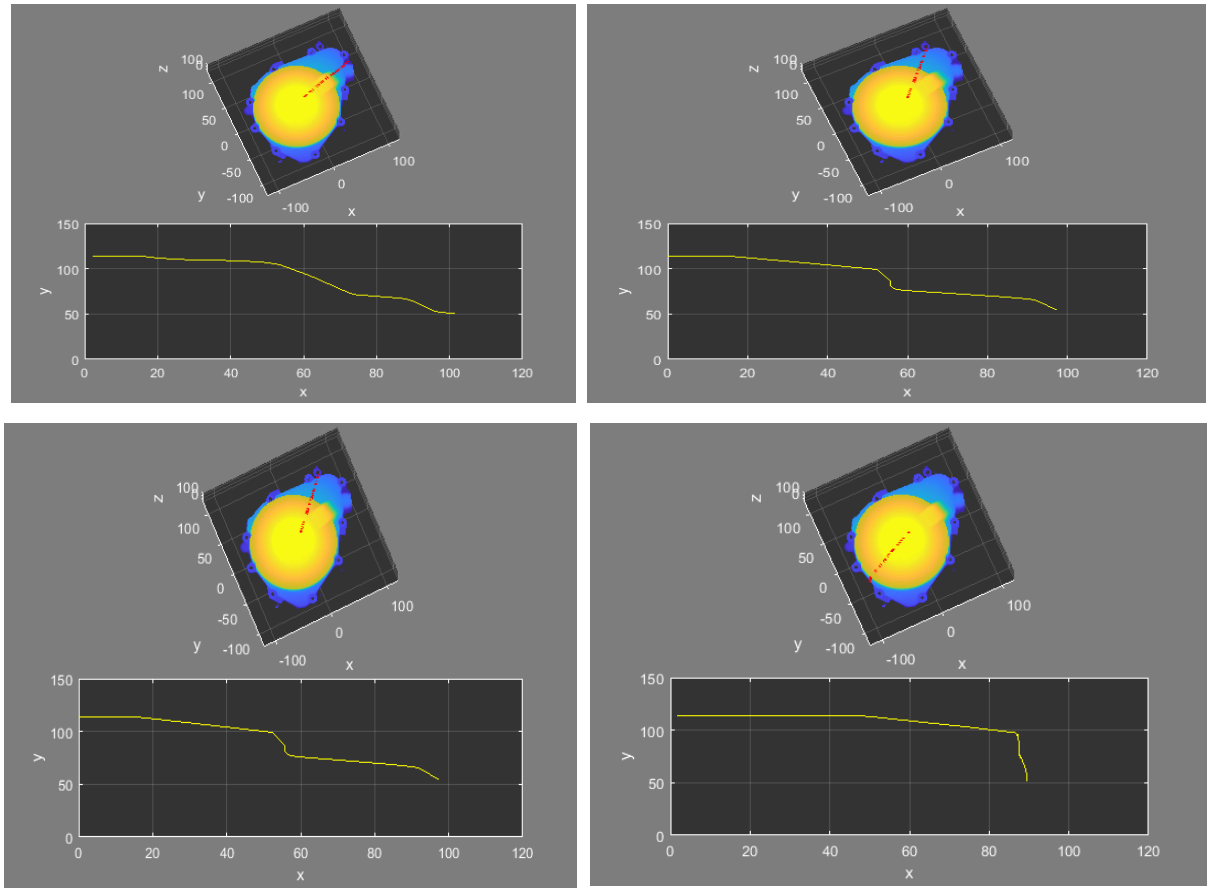


Figure 6.35: Plots showing the matrix sweep of the model and a subplot displaying its profile as the matrix moves over the surface of the model

6.4 Summary

If the noise is increased, the wavelet algorithm can find responses from the data indicated by the peak in amplitude of the wavelet coefficient. It has a limitation of responding to the scanner noise but incorporating LSTM and the filter will produce identifiable responses. But most importantly, the system response depends on the magnitude of the damage being detected. With a massive damage the amount of noise will make no difference, but a very small damage becomes problematic for the algorithm when it is below the threshold of the slice resolution. An example is the reduced number of points per slice below the slice resolution making it quite challenging for the neural network.

To reiterate, RNN was employed due to the sequence-to-sequence nature of the data, and this was not without its challenges. One major challenge encountered was the problem of overfitting during training. RNN generally deals with timesteps and when these timesteps is

very long, it becomes saturated, and overfitting happens. To overcome this, LSTM can use the previous relevant information of a network together with input data in making improved predictions. Having input data with not enough distinction between classes can be challenging for RNN. A misclassification of the data was experienced when the network was seeing everything as one class due to the damaged feature being distributed over the whole slice when plotted as a row vector. The random nature of an input data into RNN can be quite challenging. For the case of this project, the initial slice data contained random number of points per slice and preprocessing was required to improve the data format for each slice equalling the sequence length. However, the equal sequence length is not enough for optimised performance of the network, hence, it is necessary to regularise the resolution of each slice by interpolating the dataset. Intended features can be recognized as damage when training an RNN and which can be challenging. A generalised challenge of an RNN is the vanishing gradient problem where a network becomes untrainable when the loss function in relation to the weights becomes very small.

One of the high points of an RNN is that it produces reliable prediction with the right format of input data.

Chapter 7 Validation of the Proposed Method using Measurement Data

7.1 Measurement data

To validate the algorithms proposed and implemented on the simulation data in sections 6.2 and 6.3, a simple cylindrical model in Figure 7.1 was scanned to obtain point cloud data. In performing this measurement, all necessary procedures were implemented as described in section 6.1. A stable base to minimise vibrations, correct positioning of the part relative to the Arm, the ambient light and temperature, are factors considered when performing this measurement. Hence, this experiment was performed in a controlled environment. A cylindrical model was used as compared to the quasi-cylindrical model of section 6.1 due to its simplicity. Having decided to work with a simple geometry of known geometric properties and profile, then work up the ladder to models with very complex geometries, which will be discussed as future work recommendations.

Figure 7.1: Original part to be scanned

The part was held with a 3 Jaw chuck as shown in Figure 7.2 to make certain that the part is stable, as the movement of the part during scanning will produce inaccurate continuous stitching of the surface points. The 3 Jaw chuck is mostly used for holding parts when turning and milling; however, it can be utilised for measurement, as in this case, making sure the part is steady. The 3 Jaw chuck also helped with elevating the part to enable capturing of more points at the bottom region as the size of the part is relatively small. Point cloud data was captured using a Hexagon RS5 scanner attached to a 7-axis Hexagon Romer Arm 85 AACMM. Capturing was performed using 3D Systems Geomagic Studio 14 and New Kinematics Spatial Analyzer capturing software which are commercially available applications. The digitisation of a model with a diameter of 25.02 mm and a height of 35 mm excluding the threaded section is shown in Figure 7.3 with two induced damage. One damage is on the side with a width of 4.05 mm and a depth of 0.80 mm. The other damage is located at the bottom, with a depth of 5.06 mm from the measured surface of the part. After capturing, the 3D model was inspected making sure that important features of the part were captured. The point cloud data was exported as raw data retaining its original co-ordinate points, and this data was imported into MATLAB for preprocessing in preparation for model sectioning and LSTM neural network training.

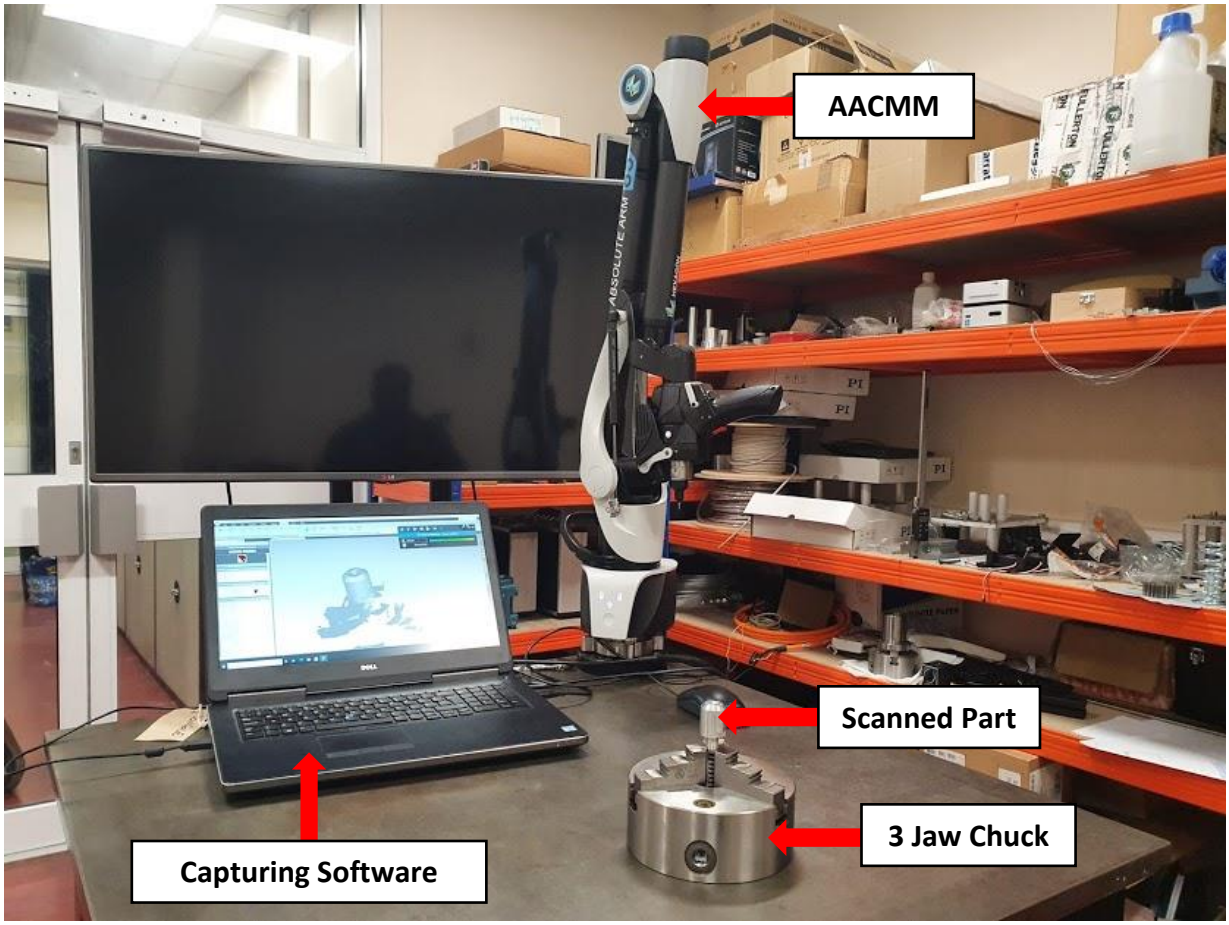


Figure 7.2: Measurement setup of the Arm, capturing software and scanned part

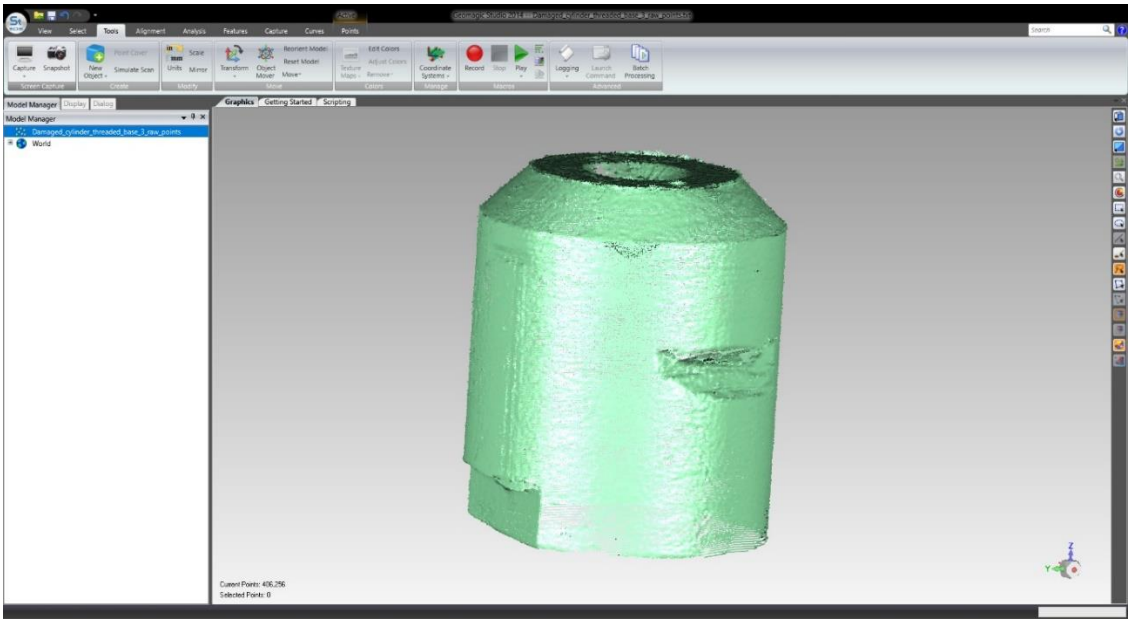


Figure 7.3: Digitisation of the model in Geomagic Studio 14

7.2 Model Sectioning using Wavelet Transform

Measurement point cloud data was inputted into the algorithm, and preprocessing was performed to eliminate outliers, and the model was centralised. The point cloud was downsampled to remove overlaying points to make the computational process faster. Model sectioning was done using wavelet transform to produce slices of the model, as discussed in section 6.2. A limitation of the slice extraction process when working with measurement data is that adjustments of the computational process are required to accommodate the nature of the data. This is because the point distribution after processing could be an unsuitable input for the mathematical operation of the algorithm, for example, interpolation of the data set while retaining the model's structure. Measurement data can sometimes produce points that are not unique, i.e., duplicated and overlaying points from the scanning process can affect the computation. The goal is to get the measurement data as organised as possible, like simulated data. To achieve this, the data is downsampled and the point cloud interpolated, making sure not to create duplicate points.

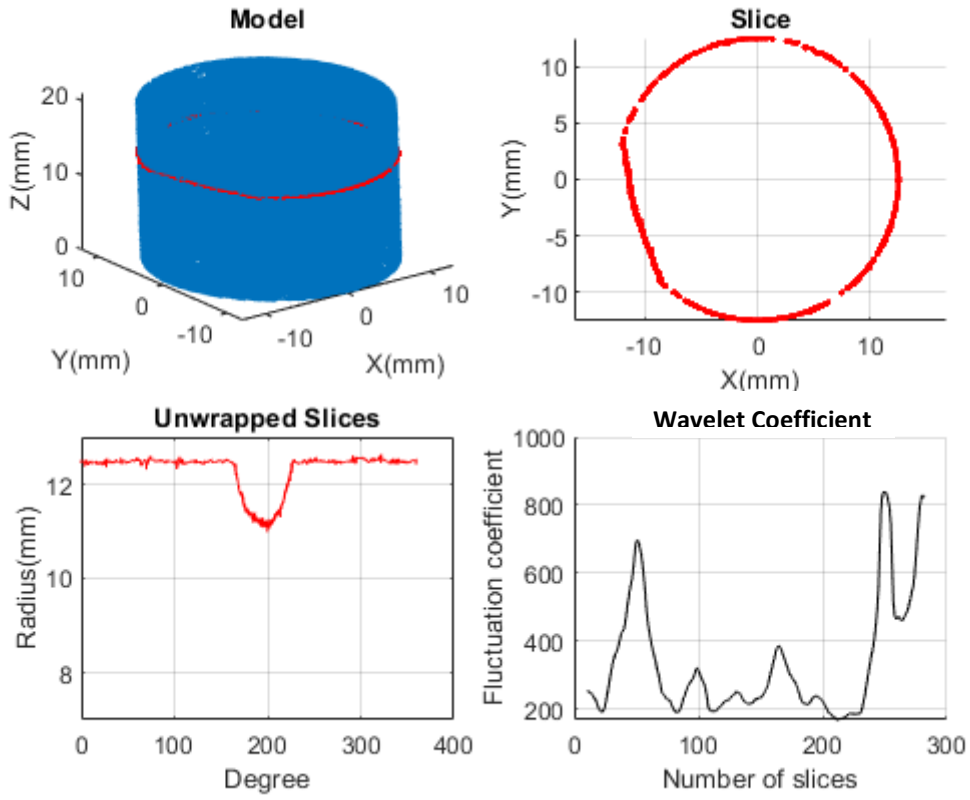


Figure 7.4: Generating slices using CWT showing early detection of the region of damage

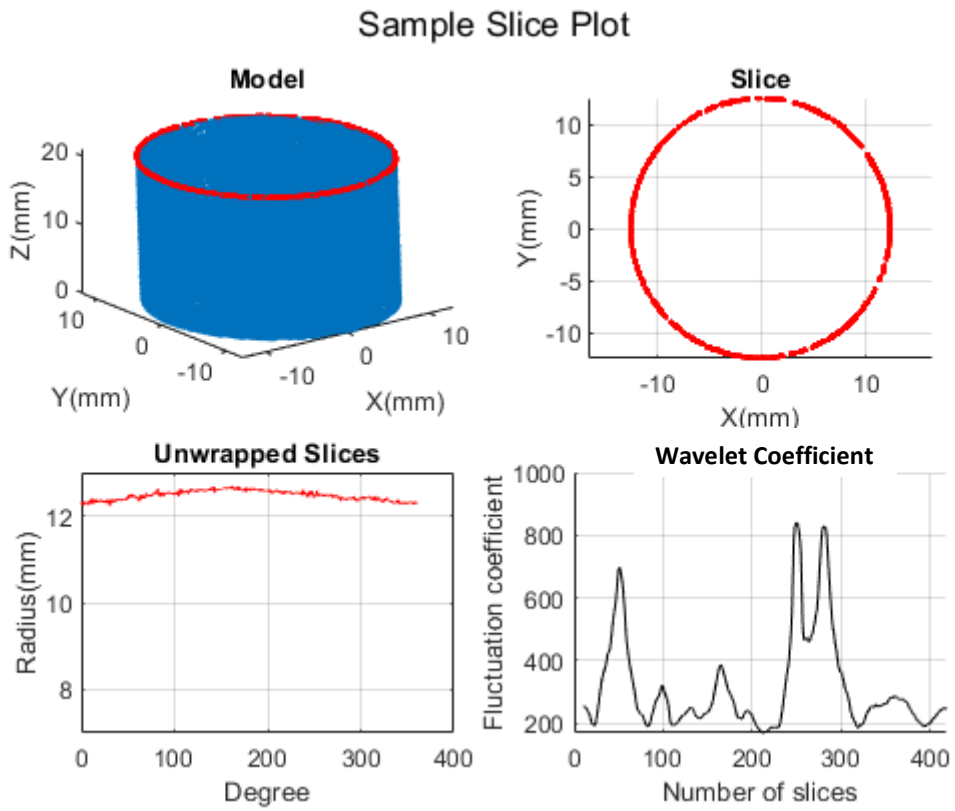


Figure 7.5: Generating slices using CWT showing final stage of the process on measurement data

The slice generation process shown in Figure 7.4 is using wavelet transform showing early detection of the region of damage, and Figure 7.5 shows the final stage of the process. On the unwrapped slice plot, damage appears to be less than 2 mm in size, and the wavelet coefficient shows the number of slices where damage is identified. However, the damage of this size can be challenging to identify using the radius data for training the LSTM neural network. This data is shown in the top plot of Figure 7.6 but can be faintly seen after normalisation of the data as the two highest amplitudes of the coefficient plot of Figure 7.5 indicates the damage region on the bottom plot of Figure 7.6. This can be seen above the slice number of 100 for the first damage and between 330 to 400 for the second damage on the

X-axis. There is also another damage being detected between slice number 300 to 350 on the X-axis and below sample number 50 on the Y-axis.

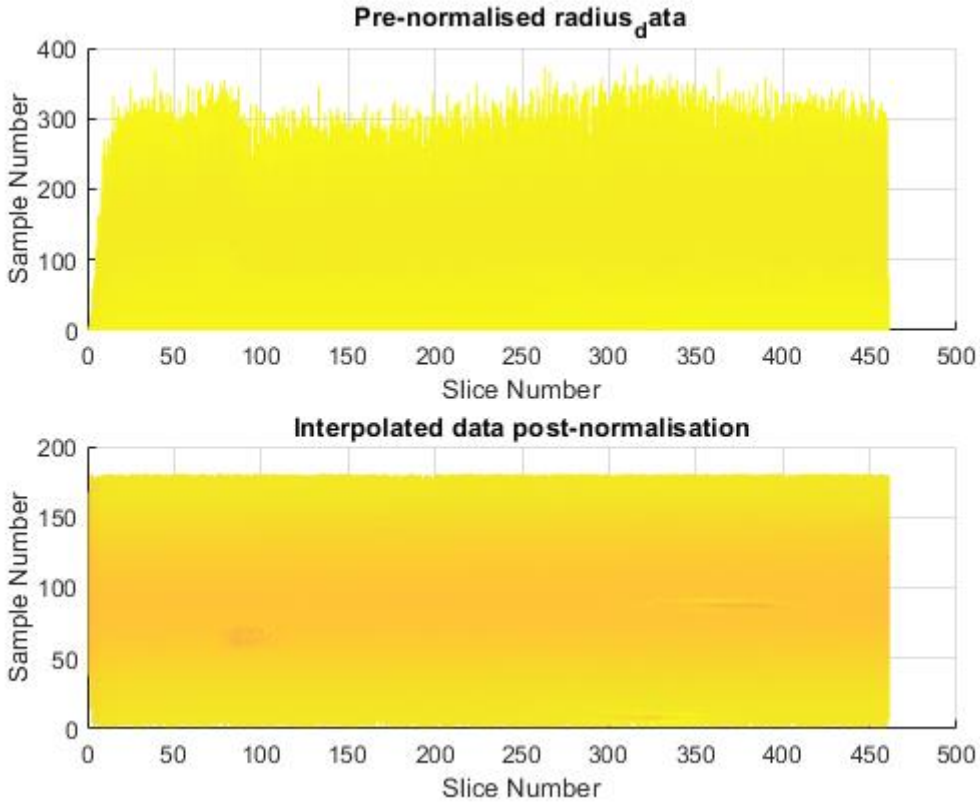


Figure 7.6: Before and after plot of the normalisation of the radius data of measurement data

Figure 7.7 and Figure 7.8 shows the slicing of the part described in section 7.1 having obvious damage at two locations. The coefficient plot of Figure 7.7 indicates the slices with damage, and this correlates with the radius data plot of Figure 7.8 after normalisation. This data is then used for training the LSTM neural network to perform prediction. The procedure is expected to be identical to the application of simulated data except for necessary adjustments to accommodate the nature of the measurement data.

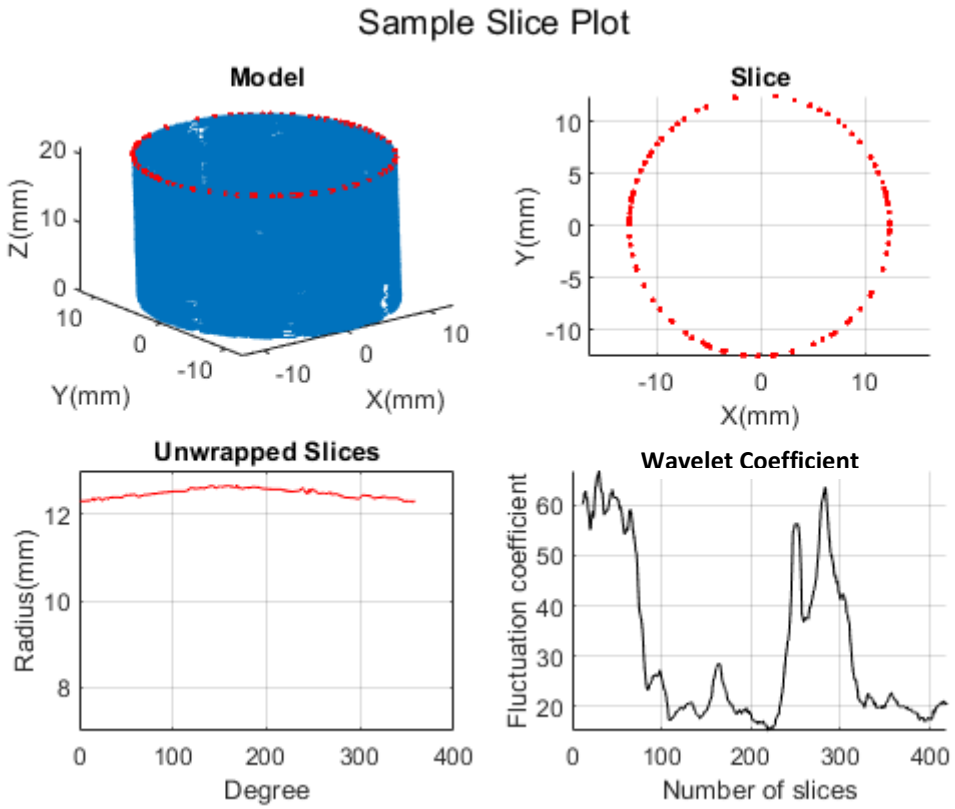


Figure 7.7: Generating slices using CWT showing final stage of the process on measurement data with intense damage

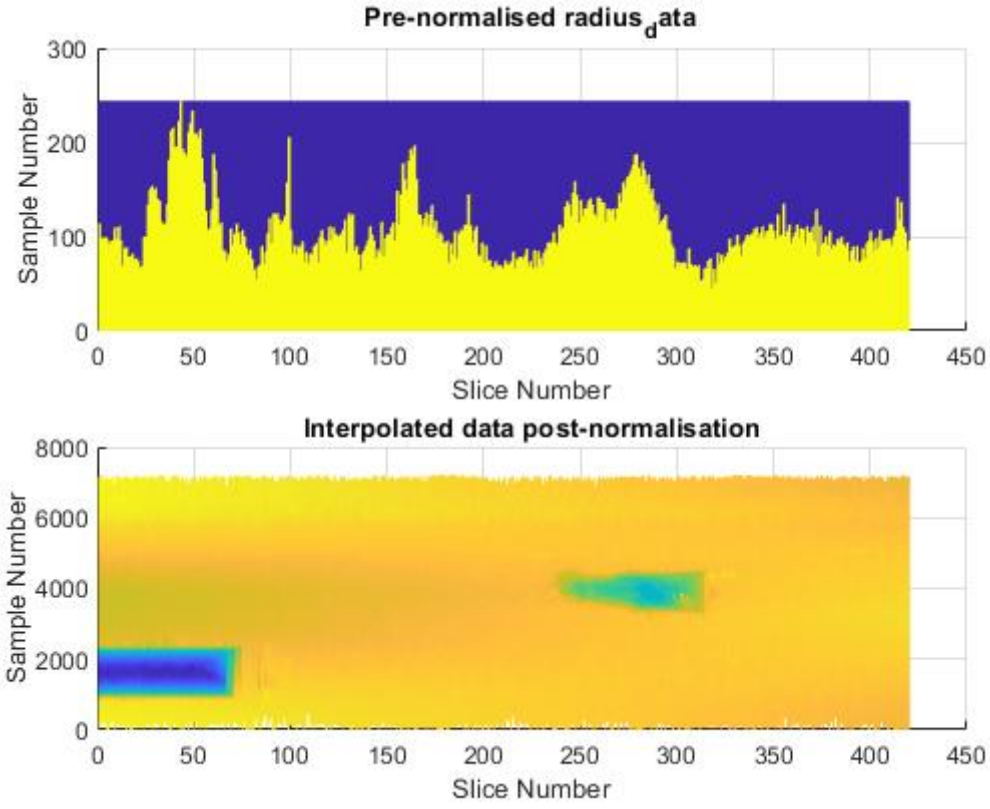


Figure 7.8: Before and after plot of the normalisation of the radius data of measurement data with intense damage

7.3 Summary

This chapter presented the validation of the algorithm proposed in Chapter 6 using raw measurement data captured with an AACMM in an experimental setup described in section 2.3.2.3. The procedure validated the analysis discussed in chapter 6 using an obvious damage from the reference surface to overcome one limitation of the algorithm posed by scanner noise. During this experiment, it was observed that the algorithm responds different to raw measurement data as compared to simulation data on which it was developed, requiring some adjustments. These adjustments include parameter specification and the non-unique nature of measurement data containing duplicate and overlaying points, hence, requiring downsampling and interpolation. One limitation of the algorithm is performing computation of huge dataset.

Chapter 8 Conclusions and Further Work

8.1 Summary

This thesis investigated the concept of detecting and localising damage on a surface as a step to performing surface reconstruction of a potentially damaged surface with no access to design information. Despite working with no reference design geometries, this project used the concept of reverse engineering for this reason, through a proposed framework in generating measurement information of an artefact, in this case, primitive shapes. This involves a digital representation of the artefact's surface, preprocessing, identification of geometries, identifying regions of damage and reconstruction using the estimated nominal values to infer the part's dimensions. The nominal values cannot be obtained when reverse engineering blind and with no access to the original design documentation, so an estimated nominal value of the radius was achieved using a linear fitting method. With the estimated nominal values and extracted measurement information, potential defects on the part's surface were analysed and classified.

The digitisation representation used in this thesis is in the form of point cloud data (PCD). Measurement data were obtained using a laser scanner attached to an Articulated Arm CMM. Most analyses performed in this thesis were based on PCD except for edge detection performed on images, as this performed poorly on PCD (see section 4.4). One of the project's objectives was to develop a system that can deal with problems of missing or sparse data and non-numerical characters caused during capturing. This was a necessary part of the work but was not intended to provide novelty for the thesis. Nevertheless, it did highlight some issues that other researchers may also face such as system interoperability between the data from a CAD application, like Geomagic studio 12, into a mathematical software such as MATLAB. There was an error in the raw data that was identifiable when using MATLAB due to non-numerical characters but appropriate for the CAD software. This was due to a parse error in the code that produces the raw points. An algorithm was developed to identify and eliminate non-numeric characters from the raw data and producing a data set readable by MATLAB (see section 3.2.5)

In the early stage of the research, the Delaunay triangulation of a damaged artefact was reconstructed in Geomagic studio 12 by connecting points to form triangulations. This

operation can be very onerous, depending on the geometry of the region. The model in question (Honda engine cover) is a combination of several geometric features coupled with freeform shapes. Together with the intensity of the damage, this freeform nature made it very difficult to perform Delaunay triangulation. Depending on the intensity of the damage, reconstruction of intense damage was quite challenging as there were no distinctive boundaries around the damage, making it difficult to identify the geometries about such regions. Edge detection was brought in for tracing the edges and relating them to the geometries of the artefact, in this case, a combination of a cone and cylinder. The cylinder was sitting right on the cone, and the region with high intense damage created a blending of both geometries. Different algorithms were implemented in both Python and MATLAB to establish a boundary and produce a defining edge. They both proved that the Canny edge detection algorithm is the most recommended as it converts images into grayscale and uses a Gaussian filter in smoothing of the edges to produce a single point response. The algorithm, when implemented in MATLAB also proved that the Canny algorithm has a better performance in detecting edges compared to other algorithms due to the computational process of using hysteresis thresholding in determining true edges in an image. When edge detection was implemented in python, algorithms such as Sobel and Laplace had the texture of the surface in the result, making it difficult to identify true edges.

Where a model contains several geometric shapes, it can be challenging if the analysis is required only on one model, therefore in chapter 5, two methods of feature extraction and detection known as the random sample consensus (RANSAC) and linear scattered interpolation (LSI) are discussed. RANSAC uses sample consensus randomly in drawing marginal sets of points from the point cloud, but this process requires specifying the region of interest (ROI). The ROI constraints the search algorithm to avoid over-fitting. LSI is one of the three interpolation methods, and they have the most straightforward application approach when working with scattered data and can be applied to simple geometries. The RANSAC method used the Canny edge detection algorithm for extracting information for specifying the ROI. The LSI method was executed in a shorter time compared to RANSAC; however, the result from the RANSAC method compared well to a conventional CMM result to within 0.02 mm discrepancy, and the accuracy of both methods is dependent on the accuracy evaluation of the AACMM. The execution time can be influenced by the number of

points in a model as the sphere with fewer points and smaller in size was faster. An advantage of RANSAC is the possibility of extraction of models saturated with noise, and the extraction mechanism can be monitored.

When slices were created using the proposed micrometre-level PCD extraction, the slices were classified into two classes of “good” or “damaged”. Slices outside of the specified damaged region were classified as good, and slices in the damaged region were classified as damaged. After classification, further investigation into the properties of the data was performed, and it was observed that the damage on the classified damaged slices tends to be distributed over the whole data. This resulted in the LSTM learning and classifying all slices as good, producing a false classification. This was caused by the wrong classification of the points by the algorithm and the preprocessing methods. Based on this assessment, it was found that the classification results were unsuitable for training an LSTM neural network.

This resulted in the use of wavelet transform to produced radius data for training the LSTM, but the data was randomised due to the randomly distributed nature of the point cloud. They produced data that was not regularised, making it challenging for detecting small-sized damage. The radius data is interpolated and normalised to produce a regularised data set for training the LSTM where the damaged region can be clearly seen in most cases. This process had a limitation of directly being affected by the scanner noise for simulated data. Based on this analysis, it was observed that the self-learning algorithm is only suitable for instruments with noise levels lower than the threshold of the slice resolution. Its computational process does not function well with dense measurement data due to duplicated and overlaying points from the scanning process. To overcome this challenge, the data set is decimated and interpolated to produce points with no duplicates.

A direct verification using the machine learning classification data on the model indicates that the algorithm can approximately identify the region of the damage on the Z-axis of a model it has been trained on, but with misclassification of three or more slices. This misclassification occurs at the transition region between both classes, and there is the possibility that the change in slice structure is challenging for the network. Furthermore, when using the morse wavelet, the energy coefficient of each slice produces information that was analysed to extract damage location. Manufacturing organisations can use this method to verify models before batch manufacturing or analyse for defects or RE after manufacturing for the quality

control processes. Information required for generating NC codes for machining can also be extracted as the relative values of the location of the damage can be obtained.

An investigative algorithm for inspecting the profiles of a model was proposed to generate more data for training the LSTM. This algorithm used a rotational matrix to provide a visual interpretation of the boundary structure of a model. The extracted information was used to inspect a model to determine what is a damage or an intended design, but such information can be quite inconclusive when damage is not obvious. This method was used in performing a visual inspection of a part, and as the part is rotated about the Z-axis, deviations were observed in the boundary profile of the part. However, to ascertain if these deviations are intended design or damage, prior knowledge or physical inspection of the part is required, and this led to the use of machine learning in learning the properties of the profile of a model in making classification decisions.

8.2 Conclusions

The following conclusions can be drawn from the work undertaken in this thesis.

8.2.1 Edge Detection

From the analysis performed in section 4.1, it was observed that most methods of edge detection, excluding the Canny algorithm, have the tendency of losing edges because they return more than a single point response when performing edge detection and edges are not localised. With the Canny algorithm, it was observed that it produced definite edges with single point response when images are converted to grayscale and that way, it also detected weak edges. Based on this evaluation, the Canny edge detection algorithm performs better with grayscale images of models with distinctive boundaries.

Conclusion 1: Edge detection is suitable for isolating primitive geometries from more complex geometric scenes. From the type of data analysed, the Canny algorithm is recommended for this application. However, care must be taken on objects where the boundaries are ill-defined or where thresholding means that the grayscale values are not sufficiently distinct.

8.2.2 Feature Extraction

Based on the evaluation of section 5.3, the execution time for the linear scattered interpolation (LSI) was observed to be faster to within 10 seconds for a feature with fewer points (sphere) and to within 100 seconds for the largest model compared to Random Sample

Consensus (RANSAC). From the investigation of the computational process for each method, it was observed that RANSAC required more computation which might have contributed to the slower process experienced. However, the LSI produced a larger error of 0.09 mm compared to RANSAC 0.02 mm for the smaller part and 0.12 mm for LSI compared to 0.07 mm for RANSAC for the larger part. For larger parts, where more data points will be required, the improvement in computation time can be expected to be higher. Similarly, it is likely that the accuracy will also reduce. However, for the applications presented in this thesis, the accuracy of extracting PCD is of importance and RANSAC would be recommended as there is more control of the parameters for performing computation.

Conclusion 2: For parts where the accuracy of feature extraction is most important, RANSAC should be used. For parts where the tolerance is more relaxed, but computation time is critical, LSI provides the better result.

8.2.3 Classified Slice Data for Training of Machine Learning

The approach of generating slices using wavelet transform has been shown to be a better approach than developing slices using the slice resolution over the model's height, with more preprocessing of the data making it more suitable for training the LSTM. This approach regularised the data making it possible for distinct identification of damage region even for models with small-sized damage. The prediction results compared well to the simulated region of damage as shown in Table 6.4 of section 6.2.6 with an error of up to 2.2 mm except for networks where detection did not occur. Since the achievable resolution with the data used was 0.05 mm, this represents an error of five "slices." With more dense data it is likely that further improvements could be found.

Conclusion 3: Training an LSTM neural network using micrometre-level information of a model can produce predictions that have been shown to accurately detect and localise damage on a model to within the specified slice resolution.

Conclusion 4: Accurate data preprocessing after slice generation is a prerequisite to ensure the neural network is being trained on appropriate data to perform accurate prediction. However, misclassification of slices may occur, so the resolution of slicing, which is limited by the point cloud density, is a critical factor.

Conclusion 5: The computational process showed the direct effect of the data size when generating training data. Therefore, it is recommended to use a data reduction technique to reduce the absolute number of points in the matrix. However, this conflicts with Conclusion 4, which asks for improved point cloud density to retain important features of the model. The effect of the computational process depends on the processing ability of the computer, and it is expected to improve on computers with more processing power. Alternatively, an iterative approach could be developed where a low number of points is used at the macro level to identify possible damage-boundaries, which can then be refined using denser data points in those regions only. Some adjustments are required for different types of parametric functions to accommodate different geometries which can be a limiting factor, however, this proves the principle that RNN can be applied in this case, but a lot of research is required for it to be extensible.

8.2.4 Profile Analysis

The method of sectioning used for populating the machine learning algorithm in this thesis has limitations in dimensionality and is only appropriate for primitive geometries. To solve this, a more advanced method of extracting meaningful 2D information was developed. The profile analysis identified regions of nominally stable profile, and the damaged region appeared as deviations from the nominal profile. However, intentional features, such as engraved text on the model also appears as deviations, which could be misclassified as a damaged profile. Human intervention, through physical visual inspection of the part, is therefore required to distinguish between intentional and non-intentional features on the surface. Such manual intervention makes the evaluation process less objective when applied to parts with complex geometries, as discussed in section 6.3. The result of this work shows clear deviation from the profile where defects and intentional features are highlighted on the series of plots (see Figure 6.34).

Conclusion 6: The method of locating damage on a part's model developed in this thesis relies on conversion from 3D to 2D data streams. The profile generation method created in this thesis has been shown to provide a possible additional solution to the sectioning method already validated.

8.3 Contribution to Knowledge

This thesis made contributions on the aspect of damage detection and localisation at the microns level. This was a step in implementing the proposed surface reconstruction processing using the proposed reverse engineering framework, and these contributions are summarised as follows.

Reverse Engineering Framework for Performing Surface Reconstruction. A framework was developed for performing surface reconstruction (SR) using the concept of reverse engineering (RE). This framework identified approaches for implementing the several methods proposed and identifying alternative approaches. These methods include capturing data, preprocessing, identification and localisation of damage, dimensioning of identified features, intelligent dimensioning to evaluate design intent, generating design drawing, performing reconstruction, and verification of reconstructed artefact. This satisfies the first and second objectives of this thesis.

Contribution 1: Most damage detection is performed on an image or an image data of a physical part, but this thesis has shown that point cloud data using the method of reverse engineering can be used to train a NN for detecting damage.

Profile Evaluation. Following from the process of feature detection, a profile examination algorithm was developed to find the boundary structure of the model. During the monitoring of the profile, it was observed that the damaged section of a model could be identified but by visual inspection and having prior knowledge of the structure of the model either by physical examination or by digitisation. This is not the case for a machine tool, hence, the necessity for developing a system requiring less human influence. The process was further developed to indicate on the model the exact location of the profile as the matrix sweeps over the surface. Hence, without prior knowledge of a model, the dimensionality plot can be analysed and interpreted for detecting discrepancies in the model's profile. This application can be used by metrology establishments when performing inspections and in manufacturing industries as part of the design validation process of a model. This model analysis method can be used by the quality control department of companies, inspection organisations, and research organisations that employ reverse engineering as part of their operations.

Contribution 2: An application of rotation matrix for inspecting the boundary profile of an artefact was developed and this presented a new dimension to performing visual inspection without prior knowledge of an artefact except for intended features.

Micrometre-level PCD Extraction. Further to improving the system to reduce human influence in an objectified manner to accurately identify damage regions, a slicing algorithm was developed, producing section slices of the model in the Z-axis direction, used to localise the damaged region. An algorithm that automatically classifies the slices into “good” or “damaged” was developed. The slices were unwrapped to produce numeric sequence data – time-series data that can be visualised as a graph. These data are used as training and test data for the LSTM. The LSTM result was analysed in terms of the accuracy of the training and classification, and using a confusion matrix chart, a comparison between the automatic classification and the LSTM was deduced. A linear fitting method was used to estimate the nominal values. This is like the circfit function used to fit a circle to a simulated set of points using the initial guess as to the determining factor when performing optimisation of the system. The proposed method was able to perform detection using point cloud as measurement data but required more analysis considering the integrity of the data and how well the extracted information compares with the measured information. Research institutes, organisations performing RE can use this algorithm as well as CNC machinists in a manufacturing setup to identify regions on a model with potential damage, satisfying the fourth and fifth objectives of this thesis.

Contribution 3: Micrometre-level information can be extracted from the PCD of an artefact and with accurate preprocessing, this data can be used in detecting damage on a model a recurrent neural network prediction.

8.4 Recommendation for Further Work

Several researchers have developed different ways of detecting damage on the surface of a model, and a few pieces of research have been done on the aspect of boundary profile and using B-spline curves. This research used a rotational matrix on the model's profile, but the identification process is by visual inspection. More work is required when this process is used for damage detection as a reconstruction process to assess that the identified variation to the normal B-spline of the model is damaged and not a design intent. This will aid the reverse

engineering process without visual inspection or having prior knowledge. Due to the timescale of the project and limitations, further work is required in the areas stated below.

- This thesis proposes and validates a method that can assist with automatic part repair and reconstruction. Significant further work is required to take this forward to a point where the location and shape of damage is automatically detected, and the requisite remedial action is automatically generated. This includes both determining how much additive material is required in a specific location and generating a tool path during subtractive operations to perform the reconstruction. The proposed methods performed well on simple geometry, as in this case a cylinder, but its performance on complex geometries such as freeform and models with combined geometries (such as cylinder on a cone) is inconclusive because it has not been tested on such models. However, application on other model types will require adjustments or improvement to the computational process to accommodate different geometries.
- The proposed framework was predominantly on models with material removal as damage such as blowhole, crack, and scratch. For performing reconstruction, additive manufacturing is required to build the surface up, before machining back to "nominal". In the case of convex surface imperfections such as warts, blister, buckle, deposit, and crater, according to ISO 8785:1999, a subtractive milling operation is required. Validation of the techniques on such features was not completed during this thesis. Since surfaces could also have a combination of both concave and convex damage, further work is needed to understand whether this information can assist with defining the nominal dimensions. The overall framework of detecting damage cannot indicate the type of damage described in ISO 8785:1999 on surface imperfections. However, certain definite imperfections can be seen on the point cloud model, imperfections such as scratches can be challenging because it contains points produced by light scattering on the surface in a random manner due to the irregularity of the surface. To achieve machine learning classification of such surface imperfections, more training is required on several surface imperfection models having different point distributions.
- At this stage of the research, the estimation of nominal values cannot be performed on coated surfaces since the coating adds an extra layer to the nominal surface of the

model. More research is required to determine what extra values, in micrometres, are added to the surface depending upon the material used for coating.

- As part of the sequence data generation, there was misclassification of some slices during the machine learning process, and this predominantly occurred at the transition of good regions to damaged regions. However, this misclassification did not affect the detection process as accuracy of 99% was achieved during training. It had to do with the difficulty faced by the neural network in the understanding of the slices as they change in geometry and structure of points when transiting between regions, requiring further investigation. However, most machine learning applications are on images; the unwrapped numeric sequence data of each slice can be converted into a spectrogram and can be used for training a convolutional neural network. With the right manipulation of parameters of the spectrogram, this function could distinguish variations in the signal and produce an image with a colourmap to help interpret such variations. This could possibly add supplementary information to the process developed in this thesis.
- Probably the most challenging extension of this work would be to translate it to freeform surfaces. Such shapes are difficult to describe even from the forward engineering approach; reverse engineering would require several more steps beyond this thesis. One of the most difficult aspects would be overcoming the uncertain, ill-defined “boundary” between one freeform surface and another, especially since such surfaces can appear to transition between each other seamlessly. This complex nature leads to questions of what is damage and what is an intended freeform feature? It is intended that the framework in this thesis could be extended to models with complex geometries only through more advanced use of machine learning of specific types of damage. While scratches and dents may be able to be classified, manufacturing defects such as poor release from a mould are likely to be more challenging to learn.
- A graphic user interface (GUI) can be developed to perform LSTM training using an already developed neural network structure. The interface can be designed to accept models with complex geometries, provide robustness in using the network's hyperparameters, and allow for numerous training operations and analysis. Each training process should produce a network that can be stored in a folder and employed when needed. Over time, the accuracy of stored networks can be visualised to monitor

how different networks performed, and respective parameters can be investigated. A brief illustration of GUI is in K.

- One important aspect for consideration in the future is the fusion of spatial features from the CNN and the detailed features from LSTM to deduce more meaningful information as regards the detection of damage on the surface of a model. During spatial scanning, strips of scene are used in producing slit spectra and the output image are evaluated per line. The two spectral imaging application of multispectral and hyperspectral does produce spectra bands of different sizes where multispectral is shorter, whereas hyperspectral looks at the collection of a complete spectrum at every pixel in an image. This system does produce wavelengths that are continuously varying, hence, there is the possibility of extracting response signal using wavelet transform as presented in this thesis to generate data suitable for training an LSTM. Also, in finding the location of damage in an image or on a surface, a spatial imaging data from CNN could be fused with a detailed feature from an LSTM to identify deviations that could possibly indicate the presence of damage or an intended feature with further investigation such as inspection.

References

[1] T. Coward, B. Scott, R. M. Watson, and R. Richards, "A comparison of prosthetic ear models created from data captured by computerized tomography, magnetic resonance imaging, and laser scanning," *The International journal of prosthodontics,* vol. 20 3, pp. 275-85, 2007.

[2] M. Trajanovi, M. TUFEGDZIC, S. ARSIC, and D. ILIC, "Morphometric Analysis of the Hip Bone as the Basis for Reverse Engineering," 2013.

[3] U. Buck *et al.*, "Application of 3D documentation and geometric reconstruction methods in traffic accident analysis: With high resolution surface scanning, radiological MSCT/MRI scanning and real data based animation," *Forensic Science International,* vol. 170, no. 1, pp. 20-28, 2007/07/20/ 2007.

[4] U. Buck, S. Naether, B. Räss, C. Jackowski, and M. J. Thali, "Accident or homicide – Virtual crime scene reconstruction using 3D methods," *Forensic Science International,* vol. 225, no. 1, pp. 75-84, 2013/02/10/ 2013.

[5] E. Bagci, "Reverse engineering applications for recovery of broken or worn parts and re-manufacturing: Three case studies," *Advances in Engineering Software,* vol. 40, no. 6, pp. 407-418, 2009, doi: 10.1016/j.advengsoft.2008.07.003.

[6] F. Li, A. P. Longstaff, S. Fletcher, and A. Myers, "Multiple-sensor integration for efficient reverse engineering of geometry," *Proceedings of the 11th International Conference on Manufacturing Research,* pp. 319-324, 2013.

[7] P. L. Seng and H. Habibollah, "Surface Reconstruction Techniques - a review," *Artificial Intelligence Review,* vol. 42, no. 59-78, pp. 59–78, June, 2014 2014, doi: 10.1007/s10462-012-9329-z.

[8] T. K. Dey, *Curve and Surface Reconstruction: Algorithms with Mathematical Analysis (Cambridge Monographs on Applied and Computational Mathematics)*. Cambridge University Press, 2006.

[9] J. Zhang and Z. Yu, "Overview of 3D printing technologies for reverse engineering product design," (in English), *Automatic control and computer sciences,* vol. 50, no. 2, pp. 91-97, 2016, doi: 10.3103/S0146411616020073.

[10] M. Javaid, A. Haleem, R. Pratap Singh, and R. Suman, "Industrial perspectives of 3D scanning: Features, roles and it's analytical applications," *Sensors International,* vol. 2, p. 100114, 2021/01/01 2021.

[11] R. J. Hocken and P. H. Pereira, *Coordinate Measuring Machines and Systems, Second Edition*, 2nd ed. (no. Book, Whole). Hoboken: Taylor and Francis, 2011.

[12] C.-H. She and C.-C. Chang, "Study of applying reverse engineering to turbine blade manufacture," *Journal of Mechanical Science and Technology,* journal article vol. 21, no. 10, p. 1580, 2007, doi: 10.1007/bf03177378.

[13] J. Gao, X. Chen, O. Yilmaz, and N. Gindy, "An integrated adaptive repair solution for complex aerospace components through geometry reconstruction," *International journal of advanced manufacturing technology,* vol. 36, no. 11-12, pp. 1170-1179, 2007, doi: 10.1007/s00170-006-0923-6.

[14] O. Yilmaz, N. Gindy, and J. Gao, "A repair and overhaul methodology for aeroengine components," *Robotics and computer-integrated manufacturing,* vol. 26, no. 2, pp. 190-201, 2010, doi: 10.1016/j.rcim.2009.07.001.

[15] J. M. Wilson, C. Piya, Y. C. Shin, F. Zhao, and K. Ramani, "Remanufacturing of turbine blades by laser direct deposition with its energy and environmental impact analysis," *Journal of cleaner production,* vol. 80, pp. 170-178, 2014, doi: 10.1016/j.jclepro.2014.05.084.

[16] X. Zexiao, W. Jianguo, and Z. Qiumei, "Complete 3D measurement in reverse engineering using a multi-probe system," *International journal of machine tools & manufacture,* vol. 45, no. 12, pp. 1474-1486, 2005, doi: 10.1016/j.ijmachtools.2005.01.028.

[17] F. Li, A. P. Longstaff, S. Fletcher, and A. Myers, "Rapid and accurate reverse engineering of geometry based on a multi-sensor system," *The International Journal of Advanced Manufacturing Technology,* vol. 74, no. 1-4, pp. 369-382, 2014, doi: 10.1007/s00170-014-5997-y.

[18] F. Li, A. P. Longstaff, S. Fletcher, and A. Myers, "Integrated Tactile-Optical Coordinate System for the Reverse Engineering of Complex Geometry," S. Hinduja and L. Li Eds.: Springer, 2013.

[19] Y.-C. Tsai, C.-Y. Huang, K.-Y. Lin, J.-Y. Lai, and W.-D. Ueng, "Development of automatic surface reconstruction technique in reverse engineering," *The International Journal of Advanced Manufacturing Technology,* vol. 42, no. 1-2, pp. 152-167, 2009, doi: 10.1007/s00170-008-1586-2.

[20] S. Singh and R. Singh, "Comparison of various edge detection techniques," in *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, 11-13 March 2015 2015, pp. 393-396.

[21] L. Zhang, Z. Li, A. Li, and F. Liu, "Large-scale urban point cloud labeling and reconstruction," (in English), *ISPRS journal of photogrammetry and remote sensing,* vol. 138, pp. 86-100, 2018, doi: 10.1016/j.isprsjprs.2018.02.008.

[22] M. Berger *et al.*, "State of the Art in Surface Reconstruction from Point Clouds," in *Eurographics 2014 - State of the Art Reports*, Strasbourg, France, 2014-04-07 2014, vol. 1, no. 1, pp. 161-185, doi: 10.2312/egst.20141040.

[23] S. P. Lim and H. Haron, "Surface reconstruction techniques: a review," *Artificial Intelligence Review,* journal article vol. 42, no. 1, pp. 59-78, June 01 2014, doi: 10.1007/s10462-012-9329-z.

[24] J. Liu, J. Zhao, X. Yang, J. Liu, X. Qu, and X. Wang, "A Reconstruction Algorithm for Blade Surface Based on Less Measured Points," *International Journal of Aerospace Engineering,* vol. 2015, pp. 1-11, 2015, doi: 10.1155/2015/431824.

[25] S. Yuwen, G. Dongming, J. Zhenyuan, and L. Weijun, "B-spline surface reconstruction and direct slicing from point clouds," *The International Journal of Advanced Manufacturing Technology,* vol. 27, no. 9, pp. 918-924, 2006/02/01 2006, doi: 10.1007/s00170-004-2281-6.

[26] K.-Y. Lin, C.-Y. Huang, J.-Y. Lai, Y.-C. Tsai, and W.-D. Ueng, "Automatic reconstruction of B-spline surfaces with constrained boundaries," *Computers & Industrial Engineering,* vol. 62, no. 1, pp. 226-244, 2012/02/01/ 2012.

[27] M. B. Rak, A. Wozniak, and J. R. R. Mayer, "The use of low density high accuracy (LDHA) data for correction of high density low accuracy (HDLA) point cloud," *Optics and Lasers in Engineering,* vol. 81, pp. 140-150, 2016/06/01/ 2016.

[28] M. d. Berg, O. Cheong, M. V. Kreveld, and M. Overmars, *Computational geometry: algorithms and applications*, 3rd ed. (no. Book, Whole). Berlin: Springer, 2010.

[29] N. Amenta and M. Bern, "Surface Reconstruction by Voronoi Filtering," *Discrete & Computational Geometry,* journal article vol. 22, no. 4, pp. 481-504, 1999, doi: 10.1007/pl00009475.

[30] N. Amenta, S. Choi, T. K. Dey, and N. Leekha, "A simple algorithm for homeomorphic surface reconstruction," presented at the Proceedings of the sixteenth annual symposium on Computational geometry, Clear Water Bay, Kowloon, Hong Kong, 2000.

[31] M. Eck and H. Hoppe, "Automatic reconstruction of B-spline surfaces of arbitrary topological type," presented at the Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, 1996.

[32] H. Ying and Q. Hong, "Surface Reconstruction with triangular B-splines," pp. 47-58, 2004.

[33] W. Ma and N. Zhao, "Catmull–Clark surface fitting for reverse engineering," *Geometric Modeling and Processing 2000. Theory and Applications. Proceedings,* 2000, doi: 10.1109/GMAP.2000.838259.

[34] T. S. Lau, S. H. Lo, and C. K. Lee, "Generation of quadrilateral mesh over analytical curved surfaces," *Finite Elements in Analysis and Design,* vol. 27, no. 3, pp. 251-272, 1997/11/01 1997.

[35] L. Kobbelt, "Interpolatory Subdivision on Open Quadrilateral Nets with Arbitrary Topology," *Computer Graphics Forum,* vol. 15, no. 3, pp. 409-420, 1996, doi: doi:10.1111/1467-8659.1530409.

[36] H. Borouchaki and P. J. Frey, "Adaptive triangular–quadrilateral mesh generation," *International Journal for Numerical Methods in Engineering,* vol. 41, no. 5, pp. 915-934, 1998, doi: 10.1002/(SICI)1097-0207(19980315)41:5<915::AID-NME318>3.0.CO;2-Y.

[37] H. Hoppe, T. DeRose, T. Duchampt, J. McDonald, and W. Stuetzle, "Surface Reconstruction from Unorganized Points," 1992.

[38] J.-D. Boissonnat, "Geometric structures for three-dimensional shape representation," *ACM Trans. Graph.,* vol. 3, no. 4, pp. 266-286, 1984, doi: 10.1145/357346.357349.

[39] S.-W. Cheng, J. Jin, and M.-K. Lau, "A fast and simple surface reconstruction algorithm," presented at the Proceedings of the twenty-eighth annual symposium on Computational geometry, Chapel Hill, North Carolina, USA, 2012.

[40] D. Dumitriu, S. Funke, M. Kutz, and N. Milosavljevic, "On the Locality of Extracting a 2-Manifold in," presented at the Proceedings of the 11th Scandinavian workshop on Algorithm Theory, Gothenburg, Sweden, 2008.

[41] T. K. Dey, S. Funke, and E. A. Ramos, "Surface Reconstruction in Almost Linear Time under Locally Uniform Sampling," *European Workshop on Computational Geometry,* 2001.

[42] T. K. Dey and S. Goswami, "Tight cocone: a water-tight surface reconstructor," presented at the Proceedings of the eighth ACM symposium on Solid modeling and applications, Seattle, Washington, USA, 2003.

[43] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen, "Decimation of triangle meshes," *SIGGRAPH Comput. Graph.,* vol. 26, no. 2, pp. 65-70, 1992, doi: 10.1145/142920.134010.

[44] N. Amenta, S. Choi, and R. K. Kolluri, "The power crust," presented at the Proceedings of the sixth ACM symposium on Solid modeling and applications, Ann Arbor, Michigan, USA, 2001.

[45] L. J. Guibas and S. Y. Oudot, "Reconstruction Using Witness Complexes," *Discrete & Computational Geometry,* journal article vol. 40, no. 3, pp. 325-356, 2008, doi: 10.1007/s00454-008-9094-6.

[46] V. D. Silva and G. Carlsson, "Topological estimation using witness complexes," presented at the Proceedings of the First Eurographics conference on Point-Based Graphics, Switzerland, 2004.

[47] J. Cohen *et al.*, "Simplification envelopes," presented at the Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, 1996.

[48] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," presented at the Proceedings of the 24th annual conference on Computer graphics and interactive techniques, 1997.

[49] H. Hoppe, "Progressive meshes," presented at the Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, 1996.

[50] K. Zhao, Y. Xu, and R. Wang, "A preprocessing method of 3D point clouds registration in urban environments," (in English), *Guangdian Gongcheng/Opto-Electronic Engineering,* vol. 45, no. 12, 2018, doi: 10.12086/oee.2018.180266.

[51] X. Guo, J. Xiao, and Y. Wang, "A survey on algorithms of hole filling in 3D surface reconstruction," *Visual Computer,* pp. 1-11, 2016, doi: 10.1007/s00371-016-1316-y.

[52] "BS EN ISO 8785:1999: Geometrical product specification (GPS). Surface imperfections. Terms, definitions and parameters," ed: British Standards Institute, 1999.

[53] "BS EN ISO 8402:1995: Quality management and quality assurance. Vocabulary," ed: British Standards Institute, 1995.

[54] S. Chen, B. Lin, X. Han, and X. Liang, "Automated inspection of engineering ceramic grinding surface damage based on image recognition," *The International Journal of Advanced Manufacturing Technology,* vol. 66, no. 1, pp. 431-443, 2013, doi: 10.1007/s00170-012-4338-2.

[55] F. C. Campbell, *Inspection of metals: understanding the basics* (no. Book, Whole). Materials Park, Ohio: ASM International, 2013.

[56] X. Huang, Z. Liu, X. Zhang, J. Kang, M. Zhang, and Y. Guo, "Surface damage detection for steel wire ropes using deep learning and computer vision techniques," *Measurement,* vol. 161, p. 107843, 2020/09/01/ 2020, doi: https://doi.org/10.1016/j.measurement.2020.107843.

[57] A. Shihavuddin *et al.*, "Wind Turbine Surface Damage Detection by Deep Learning Aided Drone Inspection Analysis," *Energies,* vol. 12, no. 4, p. 676, 2019. [Online]. Available: https://www.mdpi.com/1996-1073/12/4/676.

[58] K. Song and Y. Yan. "NEU surface defect database." http://faculty.neu.edu.cn/yunhyan/NEU_surface_defect_database.html (accessed 15/10/2019, 2019).

[59] K. Song and Y. Yan, "A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects," *Applied Surface Science,* vol. 285, pp. 858-864, 2013, doi: 10.1016/j.apsusc.2013.09.002.

[60] U. Galan, P. Orta, T. Kurfess, and H. Ahuett-Garza, "Surface defect identification and measurement for metal castings by vision system," *Manufacturing Letters,* vol. 15, pp. 5-8, 2018, doi: 10.1016/j.mfglet.2017.12.001.

[61] G. Rosati, G. Boschetti, A. Biondi, and A. Rossi, "Real-time defect detection on highly reflective curved surfaces," *Optics and Lasers in Engineering,* vol. 47, no. 3, pp. 379-384, 2009, doi: 10.1016/j.optlaseng.2008.03.010.

[62] B. Liu, Y. Yang, S. Wang, Y. Bai, Y. Yang, and J. Zhang, "An automatic system for bearing surface tiny defect detection based on multi-angle illuminations," *Optik,* vol. 208, p. 164517, 2020/04/01/ 2020, doi: https://doi.org/10.1016/j.ijleo.2020.164517.

[63] S. Satorres Martínez, C. Ortega Vázquez, J. Gámez García, and J. Gómez Ortega, "Quality inspection of machined metal parts using an image fusion technique," *Measurement,* vol. 111, pp. 374-383, 2017/12/01/ 2017, doi: https://doi.org/10.1016/j.measurement.2017.08.002.

[64] J. Lin, D. Wang, H. Tian, and Z. Liu, "Surface defect detection of machined parts based on machining texture direction," (in English), *Measurement science & technology,* vol. 32, no. 2, 2020, doi: 10.1088/1361-6501/abb485.

[65] Alaknanda, R. S. Anand, and P. Kumar, "Flaw detection in radiographic weld images using morphological approach," (in English), *NDT & E international : independent nondestructive testing and evaluation,* vol. 39, no. 1, pp. 29-33, 2006, doi: 10.1016/j.ndteint.2005.05.005.

[66] Y. Tang, X. Zhang, X. Li, and X. Guan, "Application of a new image segmentation method to detection of defects in castings," *The International Journal of Advanced Manufacturing Technology,* vol. 43, no. 5, pp. 431-439, 2009/07/01 2009, doi: 10.1007/s00170-008-1720-1.

[67] D. Huang, S. Du, G. Li, C. Zhao, and Y. Deng, "Detection and monitoring of defects on three-dimensional curved surfaces based on high-density point cloud data," *Precision Engineering,* vol. 53, pp. 79-95, 2018, doi: 10.1016/j.precisioneng.2018.03.001.

[68] E. Wiedenmann, M. Afrough, S. Albert, R. Schott, J. Tusch, and A. Wolf, "Long wave infrared 3D scanner," K. G. Harding and T. Yoshizawa, Eds., 2015, vol. 9489, no. Conference Proceedings: SPIE, pp. 94890G-94890G-11, doi: 10.1117/12.2076111.

[69] A. Khatamian and H. R. Arabnia, "Survey on 3D Surface Reconstruction," *Journal of Information Processing Systems,* Invitd Paper vol. 12, no. 3, pp. 338-357, 2016.

[70] Q.-X. Huang, B. Adams, and M. Wand, "Bayesian surface reconstruction via iterative scan alignment to an optimized prototype," presented at the Proceedings of the fifth Eurographics symposium on Geometry processing, Barcelona, Spain, 2007.

[71] J.-H. Jeon, B.-Y. Choi, C.-M. Kim, J.-H. Kim, H.-Y. Kim, and W.-C. Kim, "Three-dimensional evaluation of the repeatability of scanned conventional impressions of prepared teeth generated with white- and blue-light scanners," *The Journal of Prosthetic Dentistry,* vol. 114, no. 4, pp. 549-553, 2015/10/01/ 2015.

[72] J.-H. Jeon, D.-Y. Kim, J.-J. Lee, J.-H. Kim, and W.-C. Kim, "Repeatability and reproducibility of individual abutment impression, assessed with a blue light scanner," (in eng), *J Adv Prosthodont,* vol. 8, no. 3, pp. 214-218, 2016, doi: 10.4047/jap.2016.8.3.214.

[73] K. Panjvani, A. V. Dinh, and K. A. Wahid, "LiDARPheno – A low-cost LiDAR-based 3D scanning system for leaf morphological trait extraction," (in English), *Frontiers in plant science,* vol. 10, pp. 147-147, 2019, doi: 10.3389/fpls.2019.00147.

[74] V. Apte, "White Light vs Blue Light Scanning," vol. 2021, ed, 2020.

[75] J. Rodríguez-Quiñonez, O. Sergiyenko, D. Hernandez-Balbuena, M. Lopez, W. Flores-Fuentes, and L. Basaca-Preciado, "Improve 3D laser scanner measurements accuracy using a FFBP neural network with Widrow-Hoff weight/bias learning function," *Opto-Electronics Review,* vol. 22, 12/01 2014, doi: 10.2478/s11772-014-0203-1.

[76] D. McGahan. "FabScan open source 3D scanner." https://www.ponoko.com/blog/digital-manufacturing/fabscan-open-source-3d-scanner/ (accessed 15/08/2021, 2021).

[77] U. Mutilba, G. Kortaberria, A. Olarra, A. Gutiérrez, E. Gomez-Acedo, and M. Zubieta, "Performance Calibration of Articulated Arm Coordinate Measuring Machine," (in English), *Procedia Engineering,* vol. 63, pp. 720-727, 2013, doi: 10.1016/j.proeng.2013.08.264.

[78] ASME, "ASME B89.4.22 Methods for Performance Evaluation of Articulated Arm Coordinate Measuring Machines," ed: American Society of Mechanical Engineers, 2004, p. 52.

[79] X. H. Li, B. Chen, and Z. R. Qiu, "The calibration and error compensation techniques for an Articulated Arm CMM with two parallel rotational axes," (in English), *Measurement : journal of the International Measurement Confederation,* vol. 46, no. 1, pp. 603-609, 2013, doi: 10.1016/j.measurement.2012.08.020.

[80] D. Zheng, G. Zhao, Z. Xiao, Z. Luo, T. Zhou, and J. Zhang, "Error Compensation on Probe Parameters of Articulated Arm CMM," Les Ulis, H. L. Yuan, R. K. Agarwal, P. Tandon, and E. X. Wang, Eds., 2017, vol. 95, no. Conference Proceedings: EDP Sciences, p. 13006, doi: 10.1051/matecconf/20179513006. [Online]. Available: http://hud.summon.serialssolutions.com/2.0.0/link/0/eLvHCXMwrV1Lb9QwEB613QsICShFDZQqlx6zdRw72RwQWvoQINLm0ApxsuzYhgvdks1K_fmd8Sb76IETUg55OLHsmczD4_kGIONjljyRCYZnQjJm0B5Bh0Jbl3nkbuf1hNBXPKUSn5dXVfl9Km5-7MD1kBrTk3uQkkF0L_GfaSc3iuVTO2toDR2deEn1s2RRfLr_m1BVKYq-9iU2dmHEUc-LPRjVX6v653oVBtUxCxVYOKeNiTwVQ5qPTE_RZnTUkcevI-NKCv3kWwos4PxvG6eUTbKYbyipy5fQrVdmdKiDOP69sH2y2Db6438c9Cs4WGcNxvVKH76GHXe3D883AA_fwMeLtp21MQkhdJ8DU8R41JSTFNeaNooR2mc88_G0DbAgODcWz__EZ1V1ALeXFzdnX5K-gEPSUHg3CTHTkonUOtOUZtJkhW5yy5jzTWFNXhYus9JZbVNPOPHGFXiVW2ly77Vn2Vt4oWmj_10XEgLtIcRozxpu0apLrRQueIyFNrrgJnMSX4xgPJBH3S8BO1QItMtUreipNugZwWci4qox4W2HG7P2l-p_X5U2FieZs1xLIZhLSyFL4Q1ad8ZrYdMIjgZyqV4IzNWaOhGcLNli1QtXc66YmkhkwaxAn1eo7qGL4PBJO7SeJMfn7N2_e3gPz2hMy8WhI9jr2oX7ALvIZscwOr-eVt-Oe95_BGBaF5U

[81] D. Zheng, S. Yin, Z. Luo, J. Zhang, and T. Zhou, "Measurement accuracy of articulated arm CMMs with circular grating eccentricity errors," (in English), *Measurement science & technology,* vol. 27, no. 11, p. 115011, 2016, doi: 10.1088/0957-0233/27/11/115011.

[82] FARO, "Technology White Paper: Laser Line Scanning - QualityDigest," 10/06/2009 ed, 2009.

[83] B. I. d. P. e. M. BIPM. GUM: Guide to the Expression of Uncertainty in Measurement - Evaluation of measurement data [Online] Available: https://www.bipm.org/en/committees/jc/jcgm/publications

[84] S. Woodward, S. Brown, and M. McCarthy, "Good practice guide for generating high-density 3D point-cloud data sets of complex freeform machined surfaces using optical scanning techniques positioned directly on the machine tool," European Metrology Research Programme. Accessed: 07/04/2020.

[85] W. Cuypers, N. Van Gestel, A. Voet, J. P. Kruth, J. Mingneau, and P. Bleys, "Optical measurement techniques for mobile and large-scale dimensional metrology," *Optics and lasers in engineering,* vol. 47, no. 3, pp. 292-300, 2009, doi: 10.1016/j.optlaseng.2008.03.013.

[86] K. H. Strobl *et al.*, "The self-referenced DLR 3D-modeler," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louls, USA, 10/2009 2009, no. Conference Proceedings: IEEE, pp. 21-28, doi: 10.1109/IROS.2009.5354708.

[87] K. H. Strobl, E. Mair, and G. Hirzinger, "Image-based pose estimation for 3-D modeling in rapid, hand-held motion," in *2011 IEEE International Conference on Robotics and Automation*, Shanghai, China, 05/2011 2011, no. Conference Proceedings: IEEE, pp. 2593-2600, doi: 10.1109/ICRA.2011.5979944.

[88] D. Sangeetha and P. Deepa, "FPGA implementation of cost-effective robust Canny edge detection algorithm," *Journal of Real-Time Image Processing,* journal article March 31 2016, doi: 10.1007/s11554-016-0582-2.

[89] R. Maini and H. Aggarwal, "Study and Comparison of Various Image Edge Detection Techniques," *International Journal of Image Processing,* vol. Volume 3, no. Issue 1, pp. 1-11, 2009.

[90] Computerphile. Canny Edge Detector - Computerphile. (11 November 2015). Accessed 5 October 2021. [Online Video]. Available: https://www.youtube.com/watch?v=sRFM5IEqR2w&t=333s.

[91] Priyam, D. Dey, Shreya, and D. Polley, "Edge Detection by Using Canny and Prewitt," *International Journal of Scientific & Engineering Research,* vol. 7, no. 4, pp. 251-254, 2016.

[92] G. Papari and N. Petkov, "Edge and line oriented contour detection: State of the art," *Image and Vision Computing,* vol. 29, no. 2, pp. 79-103, 2011, doi: 10.1016/j.imavis.2010.08.009.

[93] W. Qiuping, W. Tiepeng, and Z. Ke, "Image edge detection based on the grey prediction model and discrete wavelet transform," *Kybernetes,* vol. 41, no. 5/6, pp. 643-654, 2012, doi: doi:10.1108/03684921211243301.

[94] T. Kuang, Q.-X. Zhu, and Y. Sun, "Edge detection for highly distorted images suffering Gaussian noise based on improve Canny algorithm," *Kybernetes,* vol. 40, no. 5/6, pp. 883-893, 2011, doi: doi:10.1108/03684921111142430.

[95] S. Gupta and S. G. Mazumdar, "Sobel edge detection algorithm," *International journal of computer science and management Research,* vol. 2, no. 2, pp. 1578-1583, 2013.

[96] J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-8, no. 6, pp. 679-698, 1986, doi: 10.1109/TPAMI.1986.4767851.

[97] W. Rong, Z. Li, W. Zhang, and L. Sun, "An improved Canny edge detection algorithm," in *2014 IEEE International Conference on Mechatronics and Automation*, 3-6 Aug. 2014 2014, pp. 577-582, doi: 10.1109/ICMA.2014.6885761.

[98] M. Roushdy, "Comparative study of edge detection algorithms applying on the Grayscale noisy image using morphological filter," *Graphics, Vision and Image Processing Journal,* Journal vol. Volume 6, no. Issue 4, pp. 17-23, 2006.

[99] A. C. Bovik, T. S. Huang, and D. C. Munson, "Nonparametric tests for edge detection in noise," *Pattern Recognition,* vol. 19, no. 3, pp. 209-219, 1986/01/01/ 1986.

[100] Y. Yakimovsky, "Boundary and Object Detection in Real World Images," *Journal of the ACM (JACM),* vol. 23, no. 4, pp. 599-618, 1976, doi: 10.1145/321978.321981.

[101] Y.-H. Yu and C.-C. Chang, "A new edge detection approach based on image context analysis," *Image and Vision Computing,* vol. 24, no. 10, pp. 1090-1102, 2006/10/01/ 2006.

[102] K. Ahmad, J. Khan, and M. S. U. D. Iqbal, "A comparative study of Different Denoising Techniques in Digital Image Processing," presented at the 2019 8th International Conference on Modeling Simulation and Applied Optimization (ICMSAO), 2019, 2019.

[103] W. Wu, "Paralleled Laplacian of Gaussian (LoG) edge detection algorithm by using GPU," in *Eighth International Conference on Digital Image Processing (ICDIP 2016)*, 2016: SPIE, doi: 10.1117/12.2244599.
[104] R. Biswas and J. Sil, "An Improved Canny Edge Detection Algorithm Based on Type-2 Fuzzy Sets," *Procedia Technology,* vol. 4, pp. 820-824, 2012/01/01 2012.
[105] G. Kumar and T. Jipeng, "Different Edge Detection Algorithms Comparison and Analysis on Handwritten Chinese Character Recognition," *International Journal of Computer Applications,* vol. 47, pp. 20-23, 2012.
[106] W. Xiao and X. Hui, "An improved canny edge detection algorithm based on predisposal method for image corrupted by gaussian noise," in *2010 World Automation Congress*, 19-23 Sept. 2010 2010, pp. 113-116.
[107] L. Er-sen, Z. Shu-long, Z. Bao-shan, Z. Yong, X. Chao-gui, and S. Li-hua, "An Adaptive Edge-Detection Method Based on the Canny Operator," in *2009 International Conference on Environmental Science and Information Application Technology*, 2009 2009, vol. 1, no. Conference Proceedings: IEEE, pp. 465-469, doi: 10.1109/ESIAT.2009.49.
[108] C. Gentsos, C.-L. Sotiropoulou, S. Nikolaidis, and N. Vassiliadis, "Real-time canny edge detection parallel implementation for FPGAs," in *2010 17th IEEE International Conference on Electronics, Circuits and Systems*, 2010, no. Conference Proceedings: IEEE, pp. 499-502, doi: 10.1109/ICECS.2010.5724558.
[109] M. Hagara and P. Kubinec, "About edge detection in digital images," *Radioengineering,* vol. 27, no. 4, pp. 919-929, 2018, doi: 10.13164/re.2018.0919.
[110] P. L. Rosin, "Unimodal thresholding," *Pattern Recognition,* vol. 34, no. 11, pp. 2083-2096, 2001, doi: 10.1016/S0031-3203(00)00136-9.
[111] P. Martin, "Some Aspects of Integrated Product and Manufacturing Process," in *Advances in Integrated Design and Manufacturing in Mechanical Engineering*, 2005, pp. 215-226.
[112] R. Geelink, O. Salomons, F. Slooten, and H. J. J. Kals, "Unified Feature Definition For Feature Based Design And Feature Based Manufacturing," 11/02 1995.
[113] E. S. A. Nasr, A. A. Khan, A. M. Alahmari, and H. M. A. Hussein, "A Feature Recognition System Using Geometric Reasoning," *Procedia CIRP,* vol. 18, pp. 238-243, 2014, doi: 10.1016/j.procir.2014.06.138.
[114] S. Subrahmanyam and M. Wozny, "An overview of automatic feature recognition techniques for computer-aided process planning," *Computers in Industry,* vol. 26, no. 1, pp. 1-21, 1995, doi: 10.1016/0166-3615(95)80003-4.
[115] B. Babic, N. Nesic, and Z. Miljković, "Automatic feature recognition using artificial neural networks to integrate design and manufacturing: Review of automatic feature recognition systems," *AI EDAM,* vol. 25, pp. 289-304, 08/01 2011, doi: 10.1017/S0890060410000545.
[116] B. Babic, N. Nesic, and Z. Miljkovic, "A review of automated feature recognition with rule-based pattern recognition," *Computers in Industry,* vol. 59, no. 4, pp. 321-337, 2008/04/01/ 2008.
[117] Mathworks. "What is Machine Learning." Mathworks. https://uk.mathworks.com/discovery/machine-learning.html?s_tid=srchtitle#how-it-works (accessed 24/03/2021, 2021).
[118] C. Chun and S.-K. Ryu, "Road Surface Damage Detection Using Fully Convolutional Neural Networks and Semi-Supervised Learning," (in eng), *Sensors (Basel),* vol. 19, no. 24, p. 5501, 2019, doi: 10.3390/s19245501.
[119] J. Brownlee. "A Gentle Introduction to Transfer Learning for Deep Learning." https://machinelearningmastery.com/transfer-learning-for-deep-learning/ (accessed 24/03/2021, 2021).
[120] Mathworks. "Transfer Learning for Training Deep Learning Models." Mathworks. https://uk.mathworks.com/discovery/transfer-learning.html?s_tid=srchtitle (accessed 24/03/2021, 2021).

[121] B. Guldur and J. Hajjar, *Automated Classification of Detected Surface Damage from Point Clouds with Supervised Learning*. 2016.

[122] M. A. H. Ali and A. K. Lun, "A cascading fuzzy logic with image processing algorithm--based defect detection for automatic visual inspection of industrial cylindrical object's surface," *The International Journal of Advanced Manufacturing Technology,* vol. 102, no. 1-4, p. 81, 2019, doi: 10.1007/s00170-018-3171-7.

[123] Y. J. Cha, W. Choi, and O. Büyüköztürk, "Deep Learning-Based Crack Damage Detection Using Convolutional Neural Networks," *Computer-Aided Civil and Infrastructure Engineering,* vol. 32, no. 5, pp. 361-378, 2017, doi: 10.1111/mice.12263.

[124] Z. Zhao, B. Li, R. Dong, and P. Zhao, "A Surface Defect Detection Method Based on Positive Samples," Cham, 2018: Springer International Publishing, in PRICAI 2018: Trends in Artificial Intelligence, pp. 473-481.

[125] D. Soukup and R. Huber-Mörk, "Convolutional Neural Networks for Steel Surface Defect Detection from Photometric Stereo Images," Cham, 2014: Springer International Publishing, in Advances in Visual Computing, pp. 668-677.

[126] X. J. Jiang and D. J. Whitehouse, "Technological shifts in surface metrology," *CIRP Annals,* vol. 61, no. 2, pp. 815-836, 2012/01/01/ 2012.

[127] J. D. Williams, W. H. Woodall, and J. B. Birch, "Statistical monitoring of nonlinear product and process quality profiles," *Quality and Reliability Engineering International,* vol. 23, no. 8, pp. 925-941, 2007, doi: doi:10.1002/qre.858.

[128] E. Chicken, J. J. Pignatiello, and J. R. Simpson, "Statistical Process Monitoring of Nonlinear Profiles Using Wavelets," *Journal of Quality Technology,* vol. 41, no. 2, pp. 198-212, 2009/04/01 2009, doi: 10.1080/00224065.2009.11917773.

[129] L. Kai and Q. Kemao, "Dynamic 3D profiling with fringe projection using least squares method and windowed Fourier filtering," *Optics and Lasers in Engineering,* vol. 51, no. 1, pp. 1-7, 2013/01/01/ 2013.

[130] H. F. Durrant-Whyte, "Sensor Models and Multisensor Integration," *The International Journal of Robotics Research,* vol. 7, no. 6, pp. 97-113, 1988, doi: doi:10.1177/027836498800700608.

[131] S. Orts-Escolano, V. Morell, J. García-Rodríguez, and M. Cazorla, "Point cloud data filtering and downsampling using growing neural gas," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*, 4-9 Aug. 2013 2013, pp. 1-8, doi: 10.1109/IJCNN.2013.6706719.

[132] X.-F. Han, J. S. Jin, M.-J. Wang, W. Jiang, L. Gao, and L. Xiao, "A review of algorithms for filtering the 3D point cloud," *Signal Processing: Image Communication,* vol. 57, pp. 103-112, 2017/09/01/ 2017.

[133] T. M. Awwad, Q. Zhu, Z. Du, and Y. Zhang, "An improved segmentation approach for planar surfaces from unstructured 3D point clouds," (in English), *Photogrammetric record,* vol. 25, no. 129, pp. 5-23, 2010, doi: 10.1111/j.1477-9730.2009.00564.x.

[134] Y. Z. Cheong and W. J. Chew, "The Application of Image Processing to Solve Occlusion Issue in Object Tracking," 2018, vol. 152, no. Conference Proceedings: EDP Sciences, p. 3001, doi: 10.1051/matecconf/201815203001.

[135] A. Youssef, "Analysis and comparison of various image downsampling and upsampling methods," in *Data Compression Conference, 1998. DCC '98. Proceedings*, March 30 1998-April 1 1998 1998, p. 583, doi: 10.1109/DCC.1998.672325.

[136] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, 4-7 Jan 1998 1998, pp. 839-846, doi: 10.1109/ICCV.1998.710815.

[137] A. Levin, "Filling N-sided holes using combined subdivision schemes," 10/20 2001.

[138] C. K. Chui and M.-J. Lai, "Filling polygonal holes using C1 cubic triangular spline patches," *Computer Aided Geometric Design,* vol. 17, no. 4, pp. 297-307, 2000, doi: 10.1016/S0167-8396(00)00005-4.

[139] Y. Jun, "A piecewise hole filling algorithm in reverse engineering," *Computer-Aided Design,* vol. 37, no. 2, pp. 263-270, 2005, doi: 10.1016/j.cad.2004.06.012.
[140] O. Marques, *Practical image and video processing using MATLAB*, 1. Aufl.;1; ed. (no. Book, Whole). Hoboken, New Jersey: Wiley-IEEE Press, 2011.
[141] B. M. Colosimo, Q. Semeraro, and M. Pacella, "Statistical Process Control for Geometric Specifications: On the Monitoring of Roundness Profiles," *Journal of Quality Technology,* vol. 40, no. 1, pp. 1-18, 2008/01/01 2008, doi: 10.1080/00224065.2008.11917709.
[142] M. M. Gardner *et al.*, "Equipment fault detection using spatial signatures," *IEEE Transactions on Components, Packaging, and Manufacturing Technology: Part C,* vol. 20, no. 4, pp. 295-304, 1997, doi: 10.1109/3476.650961.
[143] M. A. Bezerra, Q. O. dos Santos, A. G. Santos, C. G. Novaes, S. L. C. Ferreira, and V. S. de Souza, "Simplex optimization: A tutorial approach and recent applications in analytical chemistry," *Microchemical Journal,* vol. 124, pp. 45-54, 2016/01/01/ 2016.
[144] M. H. Asghari and B. Jalali, "Edge Detection in Digital Images Using Dispersive Phase Stretch Transform," *International Journal of Biomedical Imaging,* vol. 2015, p. 687819, 2015/03/23 2015, doi: 10.1155/2015/687819.
[145] V. B. Sunil and S. S. Pande, "Automatic recognition of features from freeform surface CAD models," *Computer-Aided Design,* vol. 40, no. 4, pp. 502-517, 2008, doi: 10.1016/j.cad.2008.01.006.
[146] M. Attene, B. Falcidieno, and M. Spagnuolo, "Hierarchical mesh segmentation based on fitting primitives," *The Visual Computer,* vol. 22, no. 3, pp. 181-193, 2006, doi: 10.1007/s00371-006-0375-x.
[147] A. Agathos, I. Pratikakis, S. Perantonis, N. Sapidis, and P. Azariadis, "3D Mesh Segmentation Methodologies for CAD applications," *Computer-Aided Design and Applications,* vol. 4, no. 6, pp. 827-841, 2007, doi: 10.1080/16864360.2007.10738515.
[148] R. Liu and H. Zhang, "Segmentation of 3D meshes through spectral clustering," 2004, no. Conference Proceedings: IEEE, pp. 298-305, doi: 10.1109/PCCGA.2004.1348360.
[149] M. Cadoni, A. Lagorio, and E. Grosso, "Large scale face identification by combined iconic features and 3D joint invariant signatures," *Image and Vision Computing,* vol. 52, pp. 42-55, 2016, doi: 10.1016/j.imavis.2016.05.008.
[150] L. Villalobos and F. L. Merat, "Manufacturing primitive-based object identification using recognition-by-components," 1997 1997, vol. 2, no. Conference Proceedings: IEEE, pp. 1638-1644 vol.2, doi: 10.1109/ROBOT.1997.614378.
[151] P. H. S. Torr and A. Zisserman, "MLESAC: A New Robust Estimator with Application to Estimating Image Geometry," *Computer Vision and Image Understanding,* vol. 78, no. 1, pp. 138-156, 2000/04/01/ 2000.
[152] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," vol. 24, no. 6, pp. 381-395doi: 10.1145/358669.358692.
[153] B. J. Tordoff and D. W. Murray, "Guided-MLESAC: faster image transform estimation by using matching priors," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 27, no. 10, pp. 1523-1535, 2005, doi: 10.1109/TPAMI.2005.199.
[154] I. Amidror, "Scattered data interpolation methods for electronic imaging systems: a survey," *Journal of Electronic Imaging,* vol. 11, no. 2, pp. 157-176, 2002, doi: 10.1117/1.1455013.
[155] J. Santolaria, A. C. Majarena, D. Samper, A. Brau, and J. Velázquez, "Articulated arm coordinate measuring machine calibration by laser tracker multilateration," *The Scientific World Journal,* vol. 2014, pp. 681853-11, 2014, doi: 10.1155/2014/681853.
[156] "BS EN ISO 25178-2:2022: Geometrical product specifications (GPS). Surface texture: Areal. Terms, definitions and surface texture parameters," ed: British Standards Institute, 2022.

[157] W. Kaibo and T. Fugee, "Using Profile Monitoring Techniques for a Data-rich Environment with Huge Sample Size," *Quality and Reliability Engineering International,* vol. 21, no. 7, pp. 677-688, 2005, doi: doi:10.1002/qre.711.

[158] B. E. I. 10360-7:2011, "BS EN ISO 10360-7:2011: Geometrical product specifications (GPS). Acceptance and reverification tests for coordinate measuring machines (CMM). CMMs equipped with imaging probing systems," ed: British Standards Institute, 2011.

[159] E. B. Li, X. Peng, J. Xi, J. F. Chicharo, J. Q. Yao, and D. W. Zhang, "Multi-frequency and multiple phase-shift sinusoidal fringe projection for 3D profilometry," *Opt. Express,* vol. 13, no. 5, pp. 1561-1569, 2005/03/07 2005, doi: 10.1364/OPEX.13.001561.

[160] G. A. Ayubi, J. M. Di Martino, J. R. Alonso, A. Fernández, C. D. Perciante, and J. A. Ferrari, "Three-dimensional profiling with binary fringes using phase-shifting interferometry algorithms," *Applied Optics,* vol. 50, no. 2, pp. 147-154, 2011, doi: 10.1364/AO.50.000147.

[161] Mathworks. "pcfitcylinder - Fit cylinder to 3-D point cloud." https://www.mathworks.com/help/vision/ref/pcfitcylinder.html (accessed 15/06/2018, 2018).

[162] Mathworks. "Zeros - Create array of all zeros." https://www.mathworks.com/help/matlab/ref/zeros.html?searchHighlight=zeros&s_tid=srchtitle (accessed 10/09/2017, 2017).

[163] I. Guskov, W. Sweldens, and P. Schroder, "Multiresolution signal processing for meshes," presented at the Proceedings of the 26th annual conference on Computer graphics and interactive techniques, 1999.

Appendix A Circularity Test for Both the Arm Calibration Sphere and a Cylindrical Part used in Chapter 5 section 5.2

ZEISS Calypso

Measurement Plan Measurement Plan 20	Date January 30, 2020	
Drawing No. * drawingno *	Time 1:28:00 pm	Order * order *
Operator Master	CMM C32Bit	Incremental Part Number 4

	Actual	Nominal	Upper Tol.	Lower Tol.	Deviation
Overall Result					
All Characteristics:		2			
...in Tolerance:		1			
...Out of tolerance:		1			
...Over Warning Limit:		0			
...Not Calculated:		0			
Total Coord. systems:		0			
...Not Calculated:		0			
Total Text elements:		0			
DiameterCircle3					|-
	29.5095	29.5000	0.1000	-0.1000	0.0095
Roundness3					0.0030
	0.0030	0.0000	0.0000		0.0030

Figure A.0.1: Parametric information for cylinder 1

ZEISS Calypso

Measurement Plan Measurement Plan 20	Date January 30, 2020	
Drawing No. * drawingno *	Time 1:16:24 pm	Order * order *
Operator Master	CMM C32Bit	Incremental Part Number 3

	Actual	Nominal	Upper Tol.	Lower Tol.	Deviation
Overall Result					
All Characteristics:		4			
...in Tolerance:		2			
...Out of tolerance:		2			
...Over Warning Limit:		0			
...Not Calculated:		0			
Total Coord. systems:		0			
...Not Calculated:		0			
Total Text elements:		0			
DiameterCircle1					-|
	101.7999	101.8286	0.1500	-0.1500	-0.0288
Roundness1					0.1264
	0.1264	0.0000	0.0000		0.1264
DiameterCircle2					--|
	101.7806	101.8286	0.1500	-0.1500	-0.0481
Roundness2					0.1067
	0.1067	0.0000	0.0000		0.1067

Figure A.0.2: Parametric information of the cylinder 2

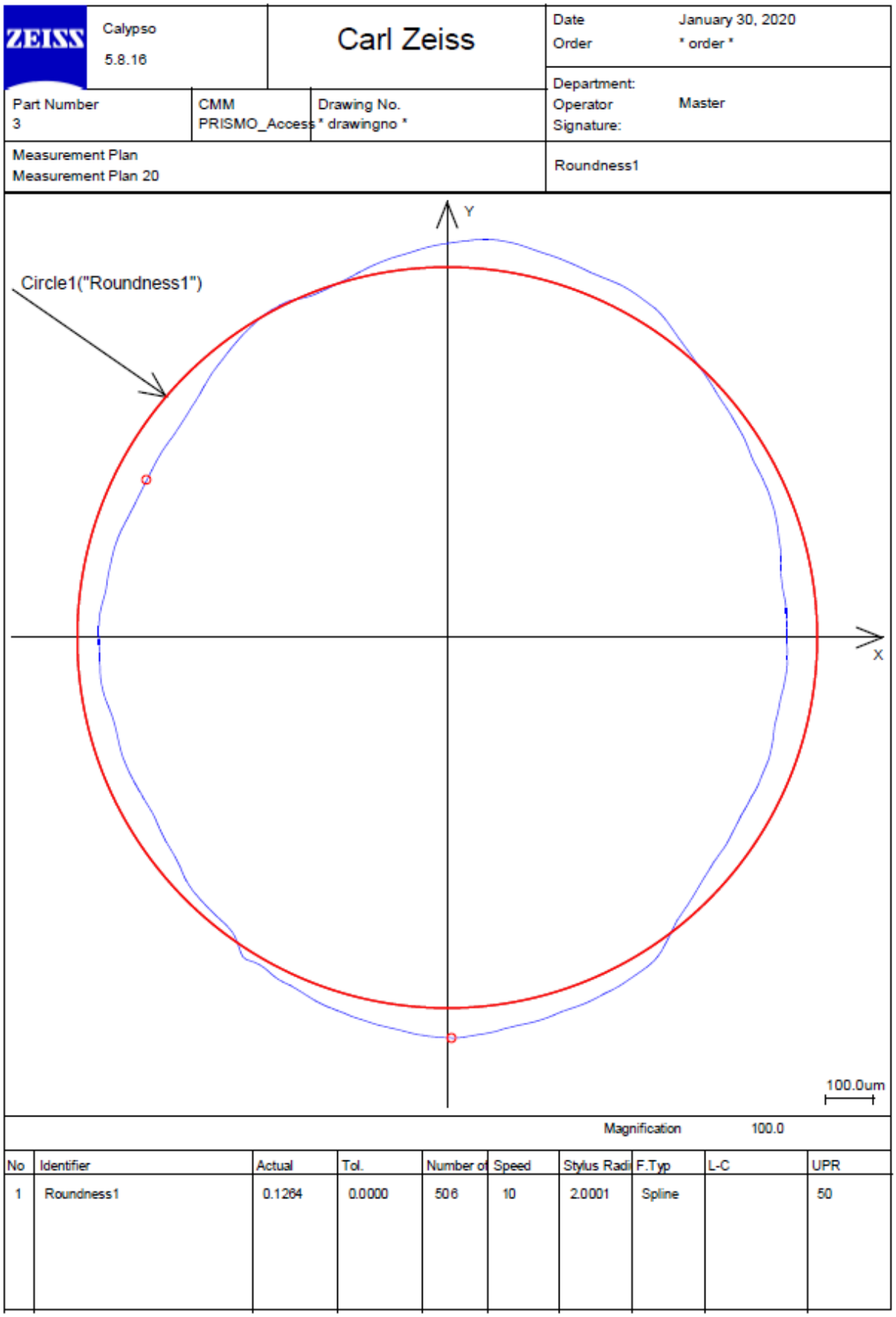


Figure A.0.3: First roundness test for cylinder 1

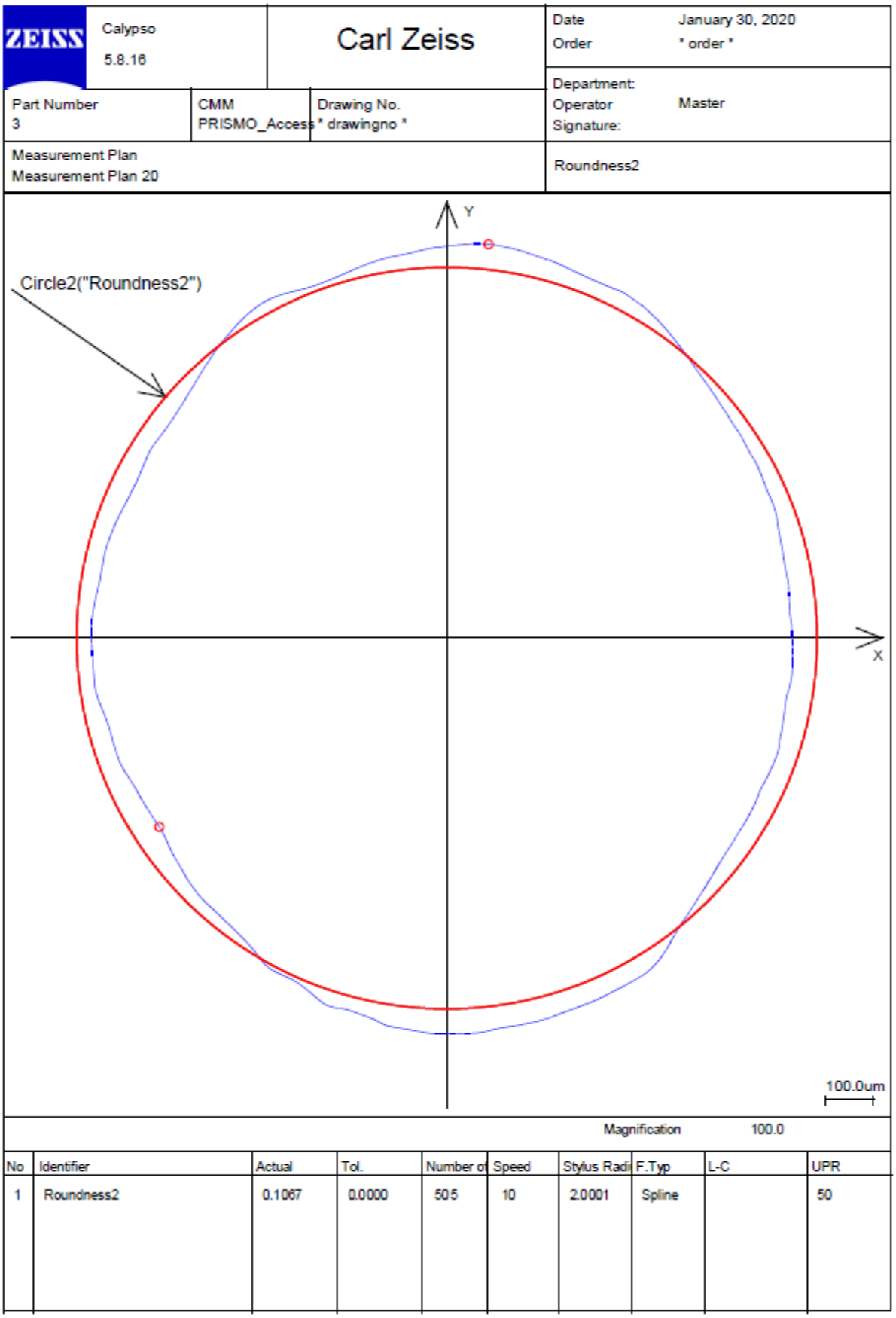


Figure A.0.4: Second roundness test for cylinder 1

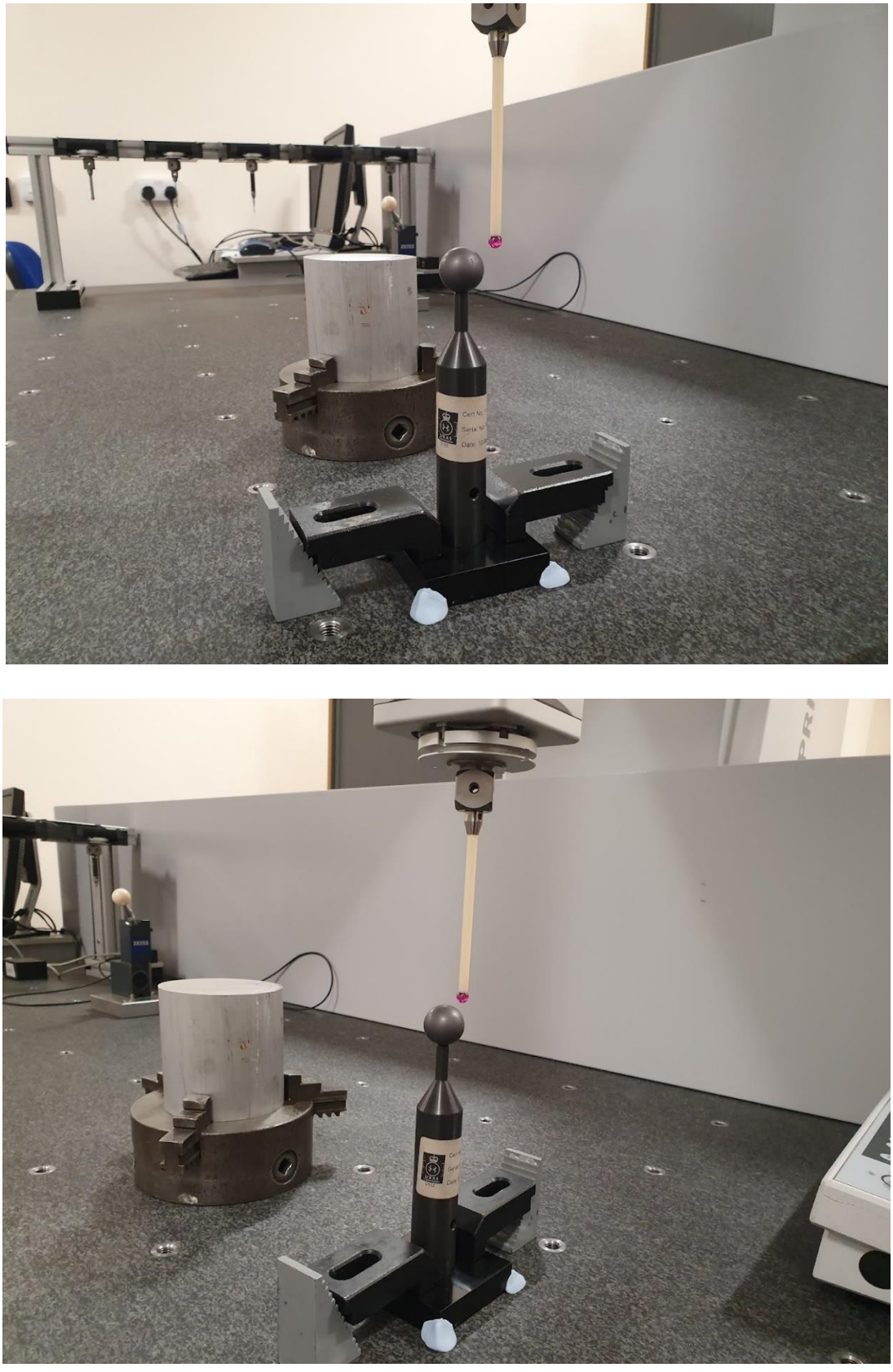


Figure A.0.5: Measurement using Co-ordinate Measuring Machine

Appendix B Mean Square Error (MSE) of the Various Noises Associated with Image Processing with R, G, and B Values

Table B.0.1: Mean Square Error (MSE) of the various noises associated with image processing with R, G, and B values [102]

Noise Type	Linear Filtered		FIR Filtered		Median Filtered		Adaptive Filtered	
Salt and pepper noise	R	22.31	R	10.83	R	**2.32**	R	6.81
	G	22.76	G	10.65	G	**2.48**	G	6.75
	B	22.31	B	7.89	B	**2.56**	B	6.25
	Avg.	22.46	Avg.	9.79	Avg.	**2.45**	Avg.	6.60
Gaussian noise	R	61.25	R	23.07	R	25.97	R	**18.03**
	G	56.42	G	26.74	G	24.97	G	**23.08**
	B	60.31	B	21.33	B	25.65	B	**16.12**
	Avg.	59.32	Avg.	23.71	Avg.	25.53	Avg.	**19.07**
Speckle noise	R	57.36	R	11.03	R	13.53	R	**9.87**
	G	56.02	G	10.44	G	12.55	G	**9.47**
	B	48.45	B	8.64	B	10.06	B	**7.44**
	Avg.	53.94	Avg.	10.03	Avg.	12.04	Avg.	**8.92**
Poisson noise	R	49.40	R	23.79	R	22.10	R	**13.24**
	G	50.04	G	22.29	G	20.40	G	**13.35**
	B	52.31	B	16.64	B	14.85	B	**13.09**
	Avg.	50.58	Avg.	20.90	Avg.	19.11	Avg.	**13.22**

Table B.0.2: Colour peak-signal-to-noise ratio (CPSNR) showing R, G, and B values calculations for different filter performance (db) [102]

Noise Type	Linear Filtered		FIR Filtered		Median Filtered		Adaptive Filtered	
Salt and pepper noise	R	34.68	R	37.82	R	**44.50**	R	39.83
	G	34.59	G	37.89	G	**44.22**	G	39.86
	B	34.68	B	39.19	B	**44.09**	B	40.20
	Avg.	34.65	Avg.	38.30	Avg.	**44.27**	Avg.	39.96
Gaussian noise	R	30.29	R	34.53	R	34.01	R	**35.60**
	G	30.65	G	33.89	G	34.19	G	**34.5**
	B	30.36	B	34.87	B	34.07	B	**36.09**
	Avg.	30.43	Avg.	34.43	Avg.	34.09	Avg.	**35.40**
Speckle noise	R	30.57	R	37.73	R	36.85	R	**38.22**
	G	30.68	G	37.97	G	37.17	G	**38.39**
	B	31.31	B	38.79	B	38.13	B	**39.44**
	Avg.	30.85	Avg.	38.10	Avg.	37.38	Avg.	**38.68**
Poisson noise	R	31.22	R	34.40	R	34.72	R	**36.94**
	G	31.17	G	34.68	G	35.06	G	**36.91**
	B	30.97	B	35.95	B	36.44	B	**36.99**
	Avg.	31.12	Avg.	35.01	Avg.	35.41	Avg.	**36.95**

Appendix C Reverse Engineering Tools

Figure C.0.1: Tools for performing reverse engineering

Figure C.0.2: Laser Scanner

Table C.0.1: Specification of the RS4 laser scanner

	RS4
Accuracy	0.028 mm (2σ)
Point Acquisition Rate	752 000 points/s
Points per Line	Max. 7520
Line Rate	Max. 100 Hz
Line Width (mid)	115 mm
Standoff	165 ± 50 mm
Minimum Point Spacing	0.011 mm
System Scanning Certification	Yes
Laser Class	2M
Operating Temperature	5-40°C
Weight	8.4 kg

Appendix D Steps for Performing Feature Fitting and Extraction in MATLAB

Feature Recognition

This is a MATLAB function to show how a feature can be fitted to point cloud data. Point cloud data could be measured or simulated data. The function can fit a cylinder or a sphere to a set of points by manipulating specific parameters – the most important is specifying the region of interest (roi) to constraint the search and the amount of time required to run the function.

Fitting a Cylinder to Point Cloud Data

STEP 1: Firstly, we will need to have both the data and the function in the same path, as shown below in Figure D.1.

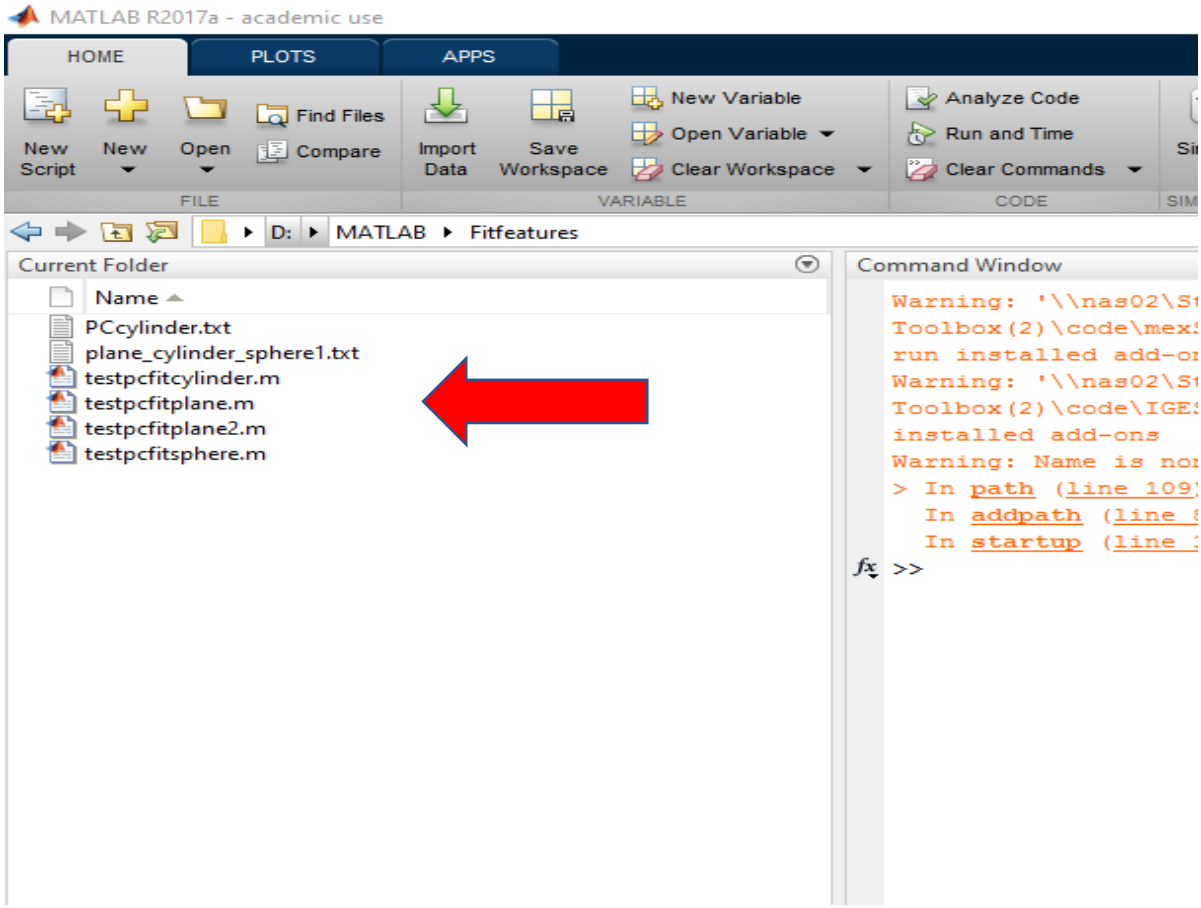


Figure D.1

STEP 2: Load the m-file called testpcfitcylinder.m, run the section shown below to plot the point cloud of the raw data. This can be done by highlighting the section, Right-click, then click Evaluate section (F9) to run the section.

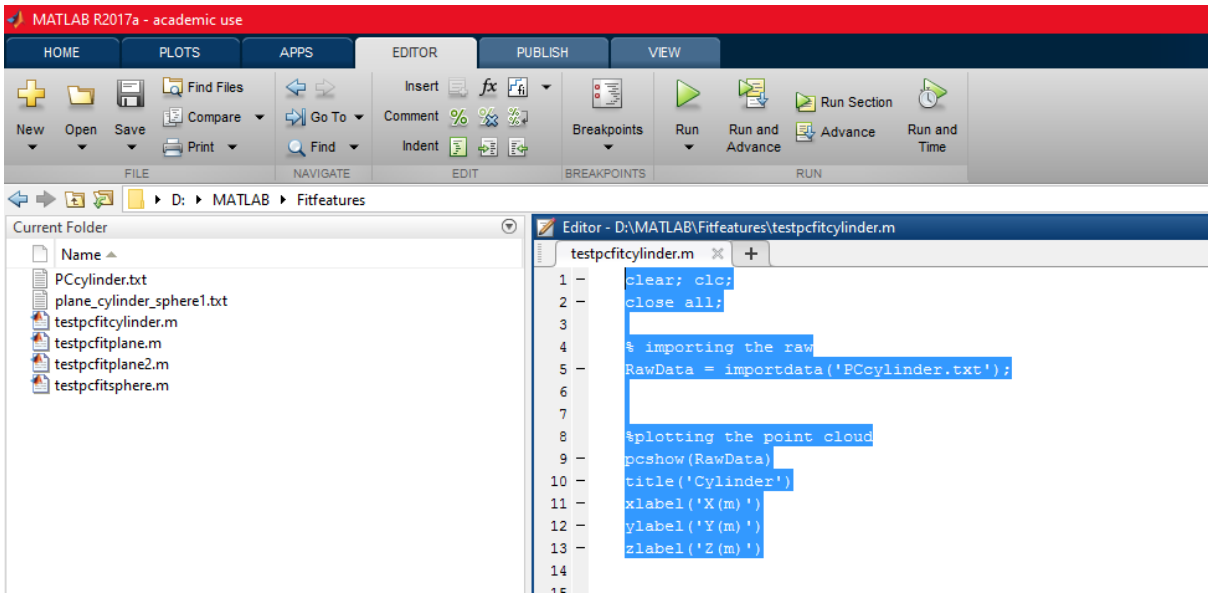


Figure D.2

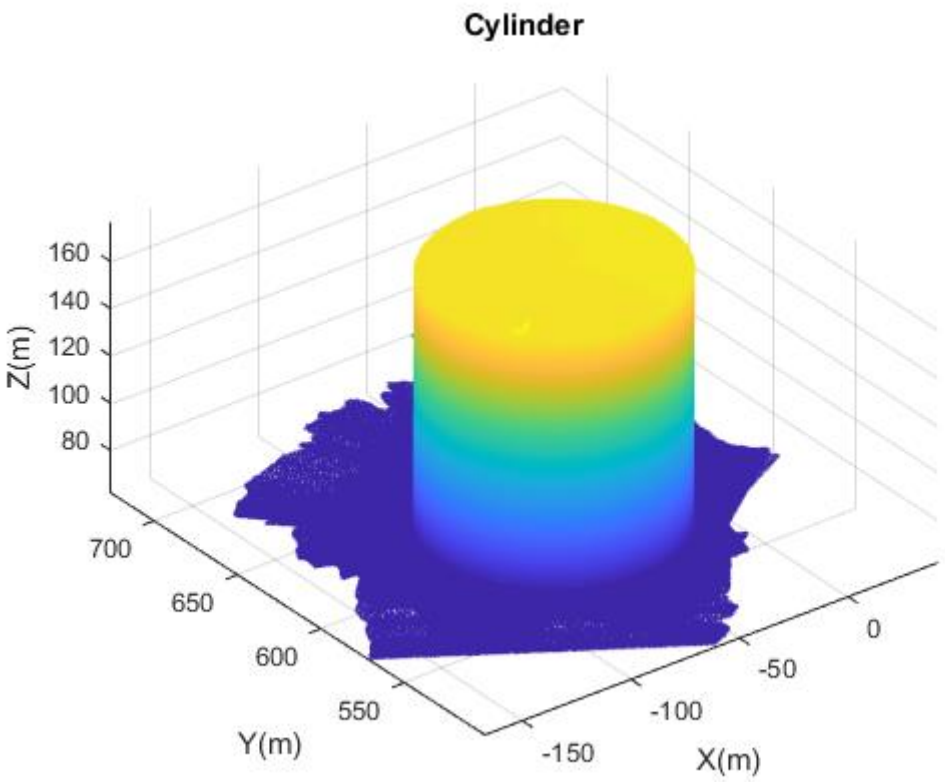


Figure D.3

STEP 3: We could set the roi, but for this plot, we have got one model, so we can set the x, y, and z coordinates from –inf to inf as shown below. This fits a cylinder to the whole model.

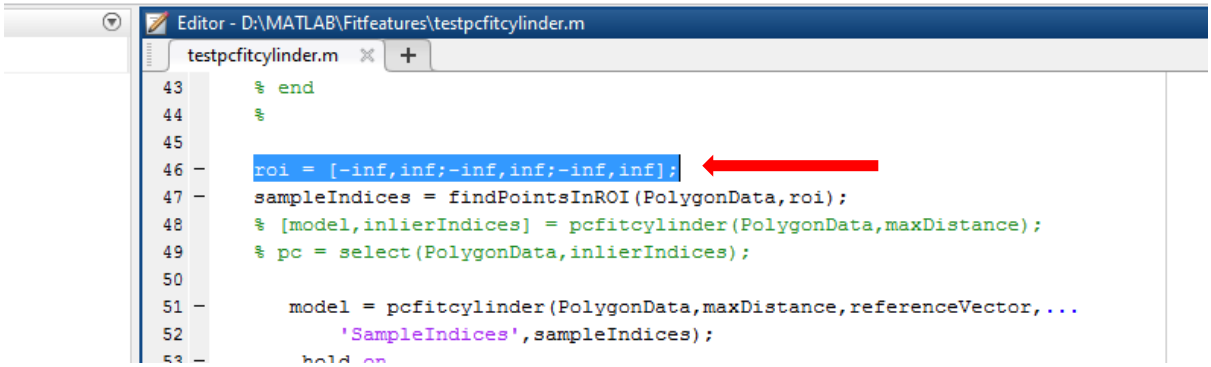


Figure D.4

STEP 4: Click on RUN under the EDITOR tab to run the programme. You can see how the function fits a cylinder to the model. You can zoom in to have a closer view of the plot.

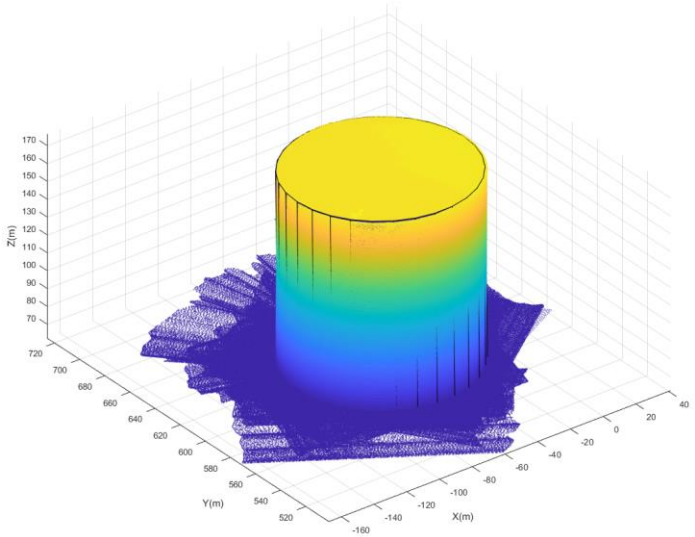


Figure D.5

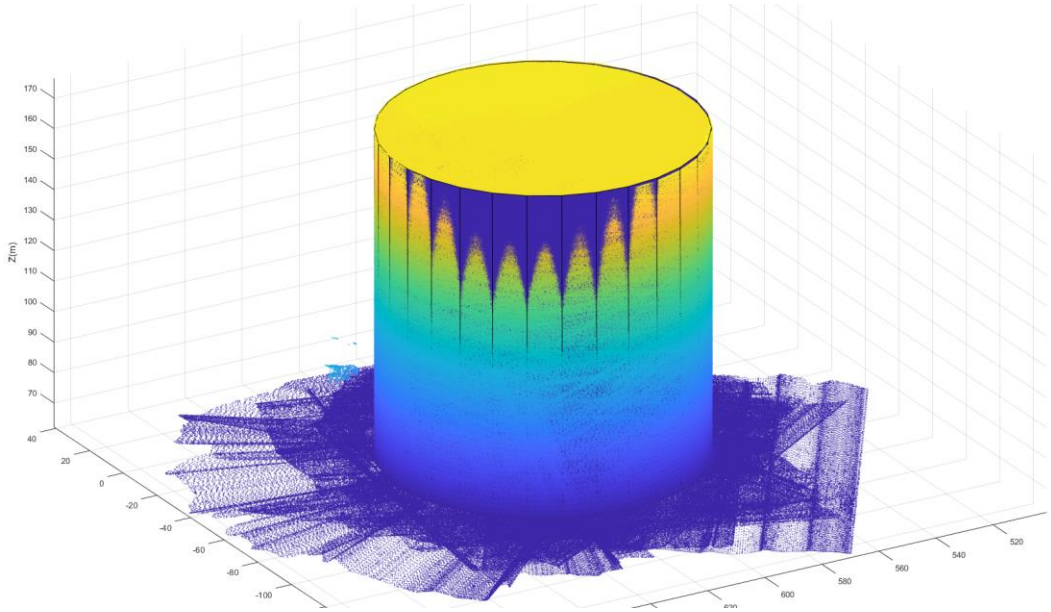


Figure D.6

STEP 5: Now, let us look at a model having different geometric shapes. On the left-hand side of MATLAB, where you have got the Current Folder as shown in the Figure below, copy the file name plane_cylinder-sphere1.txt and replace the PCcylinder.txt with plane_cylinder-sphere1.txt in the programme, do not forget to have it within the apostrophes as shown in Figure 7.

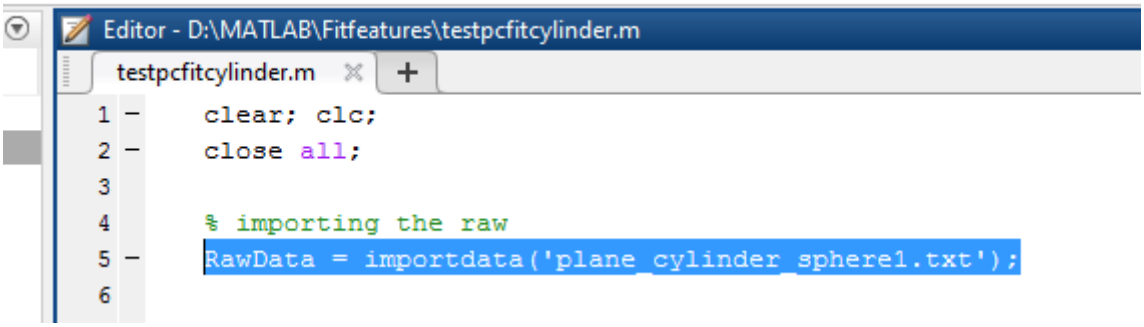


Figure D.7

STEP 6: Highlight the section as shown in Figure 8 to plot the point cloud of the raw data, Right-click, then click Evaluate section (F9) to run the section.

Editor - D:\MATLAB\Fitfeatures\testpcfitcylinder.m

testpcfitcylinder.m

```
1 -  clear; clc;
2 -  close all;
3
4    % importing the raw
5 -  RawData = importdata('plane_cylinder_sphere1.txt');
6
7
8    %plotting the point cloud
9 -  pcshow(RawData)
10 - title('Cylinder')
11 - xlabel('X(m)')
12 - ylabel('Y(m)')
13 - zlabel('Z(m)')
14
```

Figure D.8

STEP 7: To fit a cylinder, sphere or plane to the plot shown in Figure 9, we will need to specify the region of interest (roi) to constraint the search.

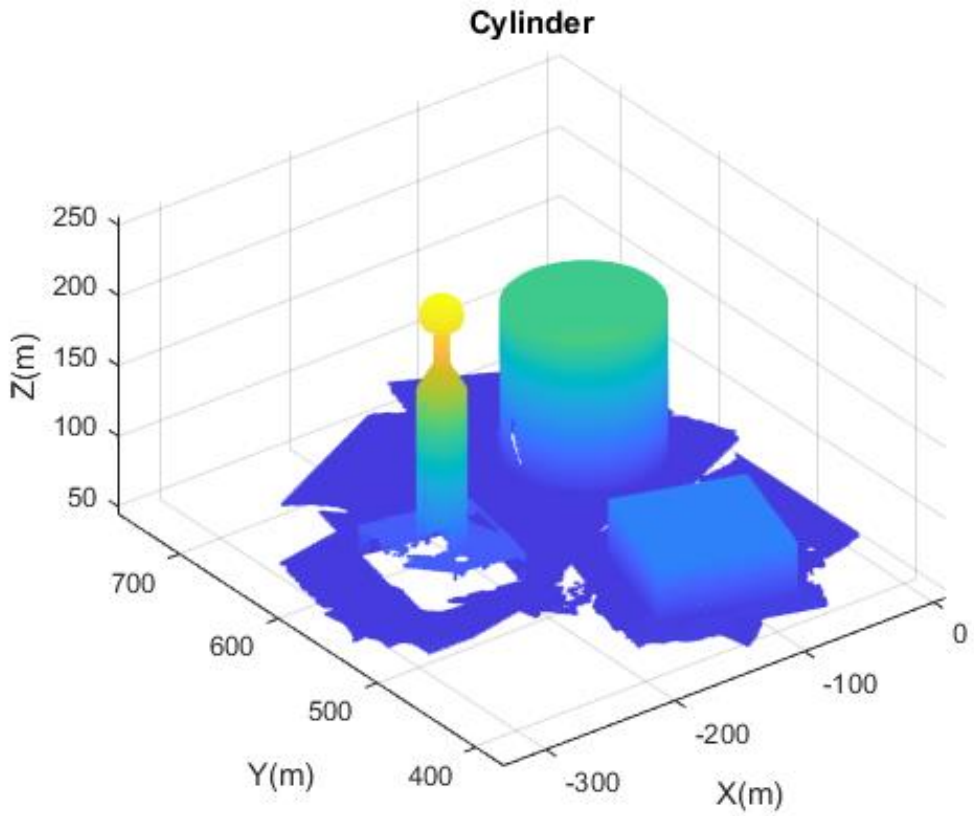


Figure D.9

STEP 8: To fit a cylinder to the biggest cylinder in the point cloud, we specify the x, y and z coordinates as follows:

X – axis = -150, 0; Y – axis = 550, 700; Z – axis = 50, inf

The function tried to fit a cylinder to the point cloud within our specified region of interest.

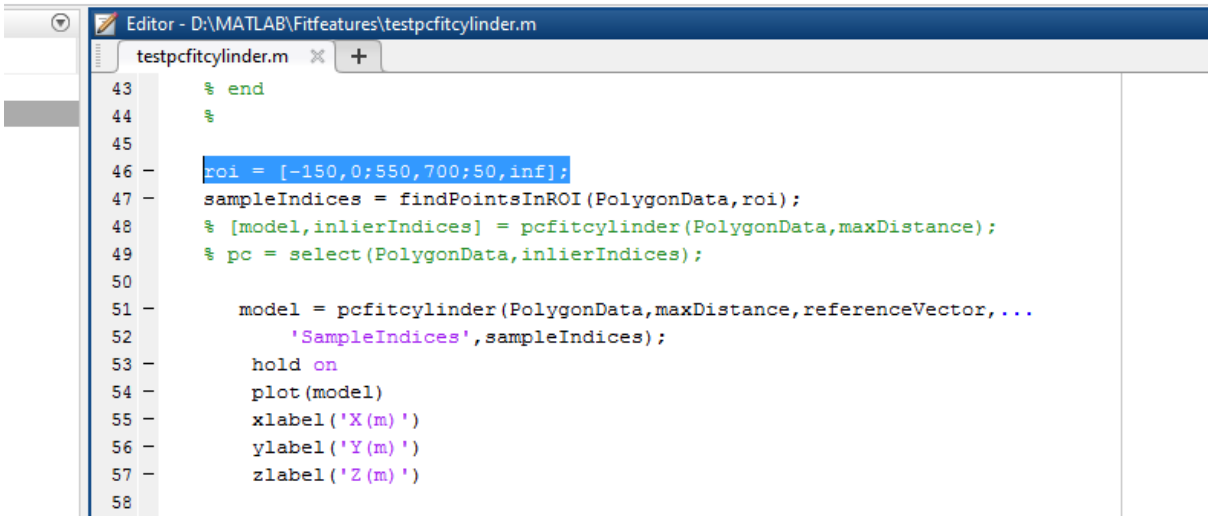


Figure D.10

STEP 9: Click on RUN under the EDITOR tab to run the programme. You can see how the function fits a cylinder to the model. You can zoom in to have a closer view of the plot.

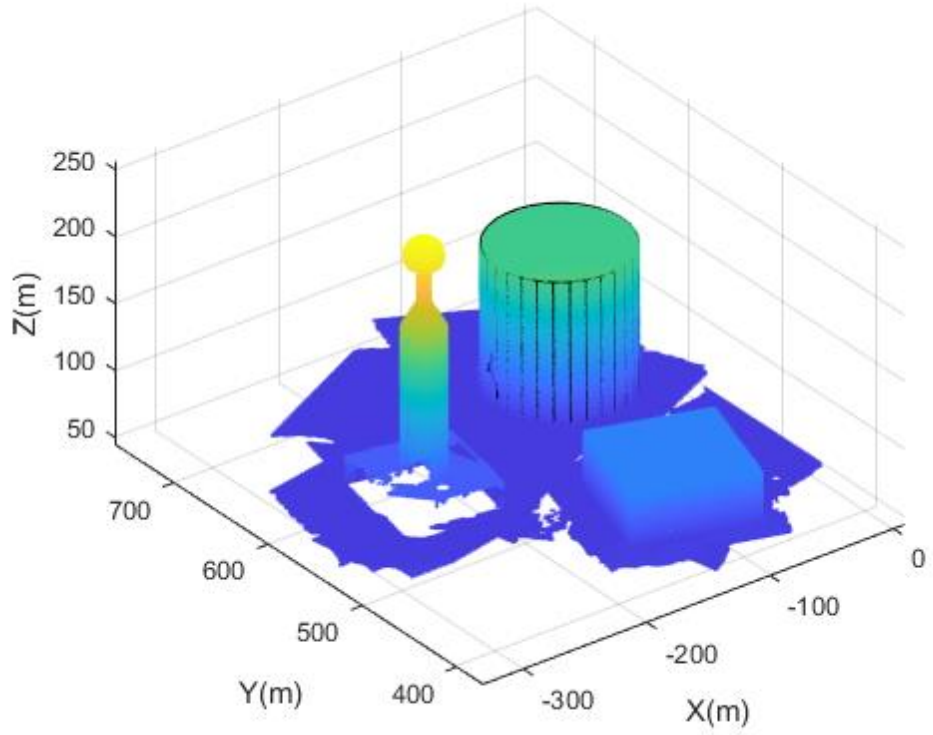


Figure D.11

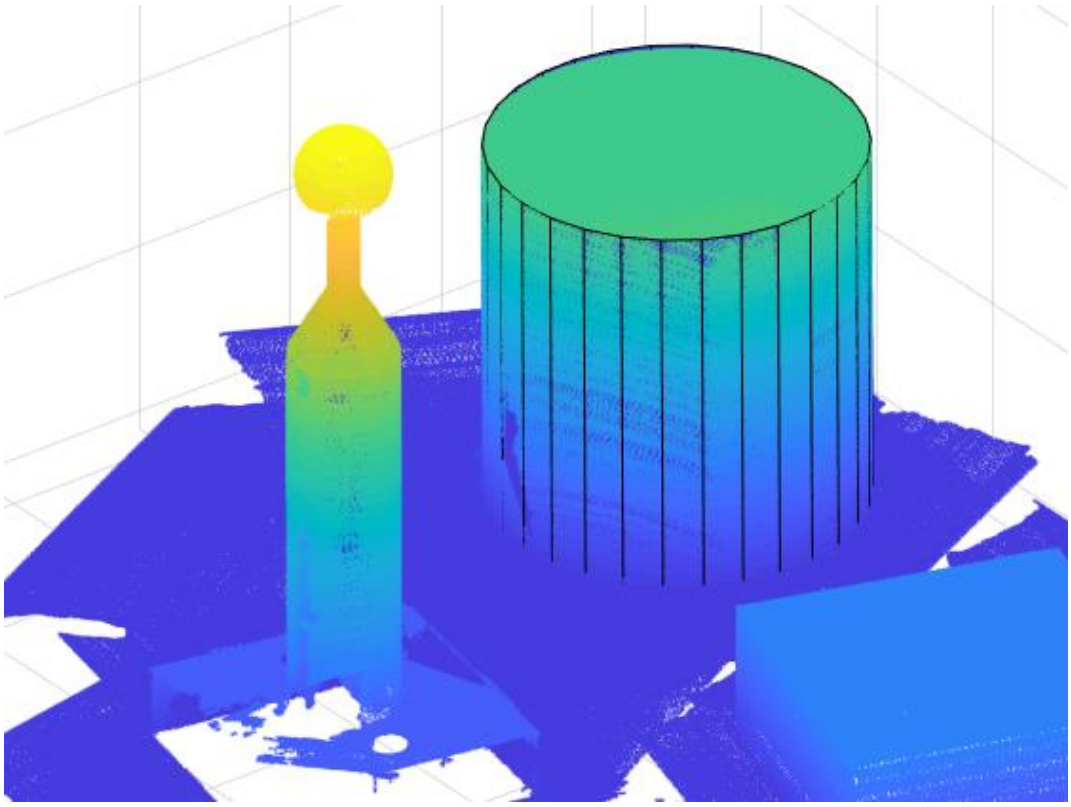


Figure D.12

STEP 10: To fit a cylinder into the smaller cylinder point cloud, we will need to specify the region of interest. To do this, we look at Figure D.9 and determine the x, y, and z coordinates. From Figure D.9, the coordinates can be specified as follows:

X – axis = -300, -200; Y- axis = 500, 600; Z – axis = 75, 200; from Figure 13 below. Figure 14 shows the roi specification.

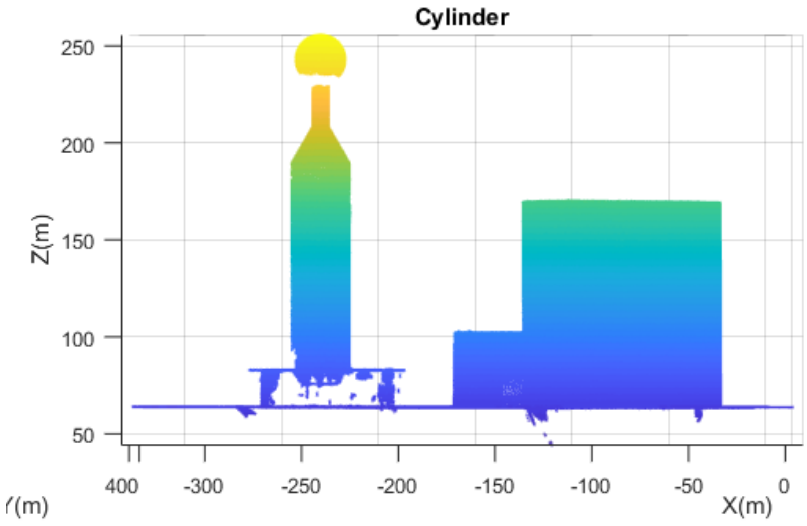


Figure D.13

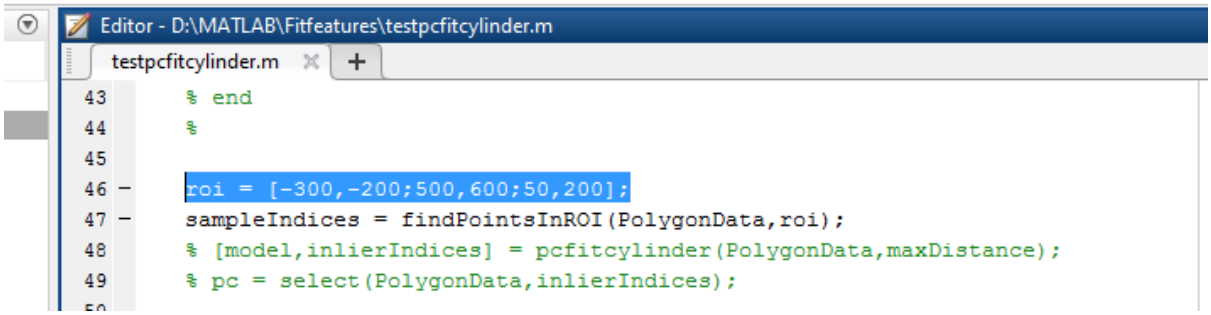


Figure D.14

STEP 11: Click on RUN under the EDITOR tab to run the programme. You can see how the function fits a cylinder to the model. You can zoom in to have a closer view of the plot.

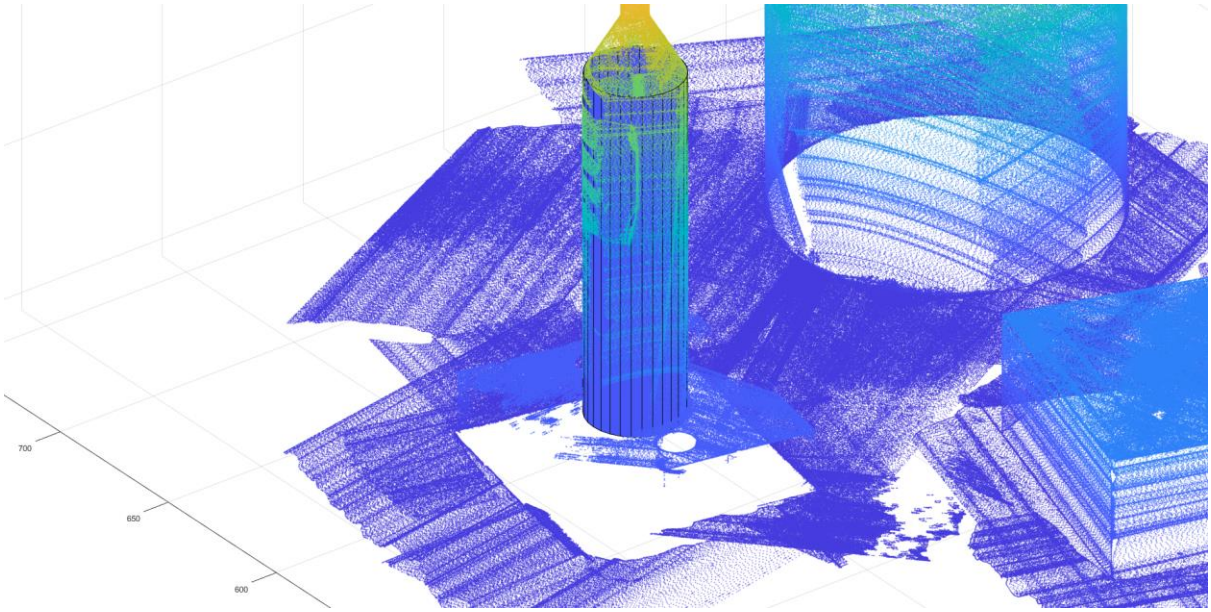


Figure D.15

Fitting a Sphere to Point Cloud Data

STEP 1: From the left-hand panel in MATLAB, as shown in Figure D.1, load the m-file called testpcfitsphere.m, the raw data is the same as the one in the previous function. Comment the code line 21-23 as shown below by highlighting and pressing Ctrl+R on the keyboard; you will notice it turns green with the % sign. Do not forget; the roi has already been specified for you from the plot in Figures D. 9 and D.13.

```
Editor - D:\MATLAB\Fitfeatures\testpcfitsphere.m
testpcfitsphere.m
 9 -     ylabel('Y(m)')
10 -     zlabel('Z(m)')
11 -     title('Original Point Cloud')
12
13 -     maxDistance = 0.01;
14 -     referenceVector = [0,0,1];
15 -     maxAngularDistance = 5;
16       % RawData = rot_mat(RawData,'x',90);
17 -     PolygonData = pointCloud(RawData);
18
19 -     roi = [-300,-200;500,600;200,inf];
20 -     sampleIndices = findPointsInROI(PolygonData,roi);
21 -     [model,inlierIndices] = pcfitsphere(PolygonData,maxDistance,...
22                     'SampleIndices',sampleIndices);
23 -     calibrator = select(PolygonData,inlierIndices);
24
25 -        model = pcfitsphere(PolygonData,maxDistance,...
26             'SampleIndices',sampleIndices);
27 -        hold on
28 -        plot(model)
```

Figure D.16

STEP 2: Click on RUN under the EDITOR tab to run the programme. You can see how the function fits a sphere to the model. You can zoom in to have a closer view of the plot.

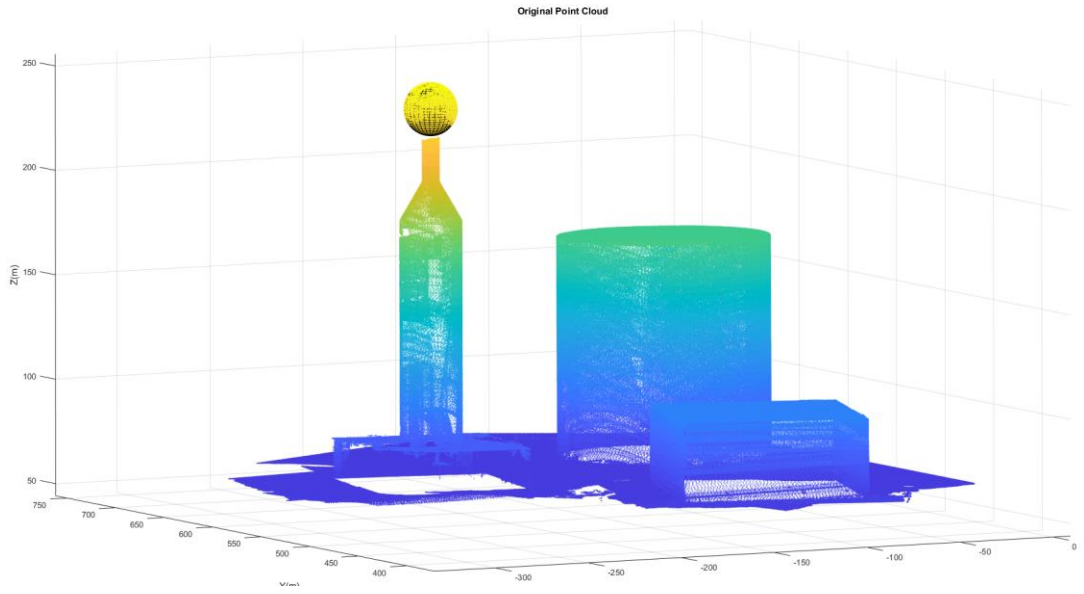


Figure D.17

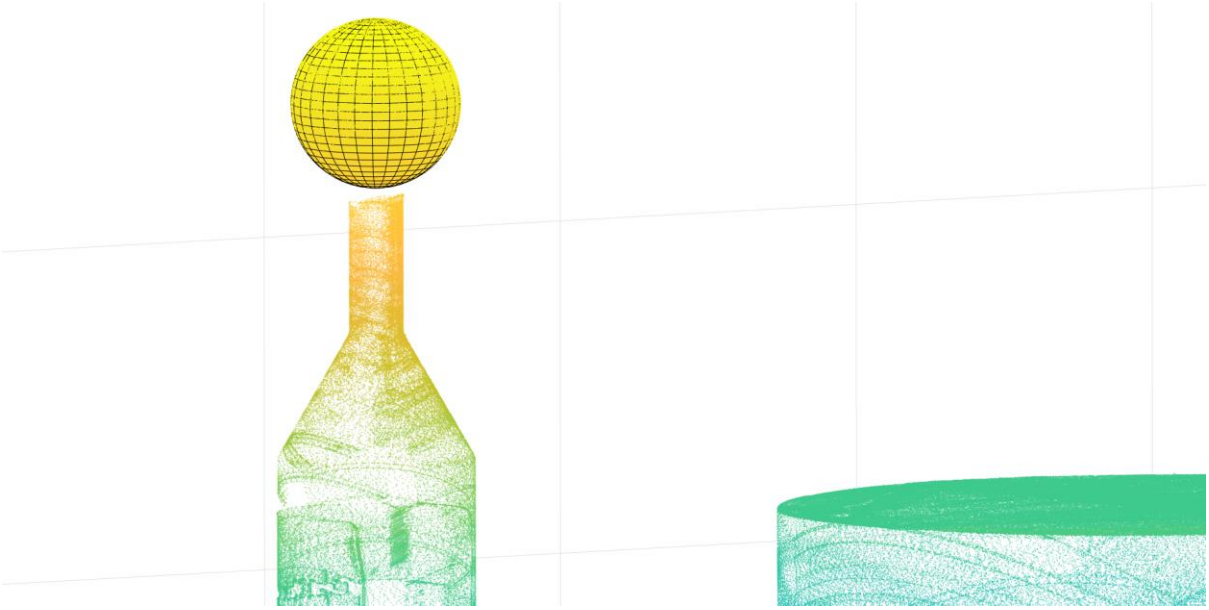


Figure D.18

STEP 3: The function can also extract a sphere from the point cloud. In the programme, uncomment the code line 21-23 which we commented on initially, do this by highlighting the code and on the keyboard Ctrl+T to uncomment, then comment code line 25 – 28 by highlighting the lines and Ctrl+R.

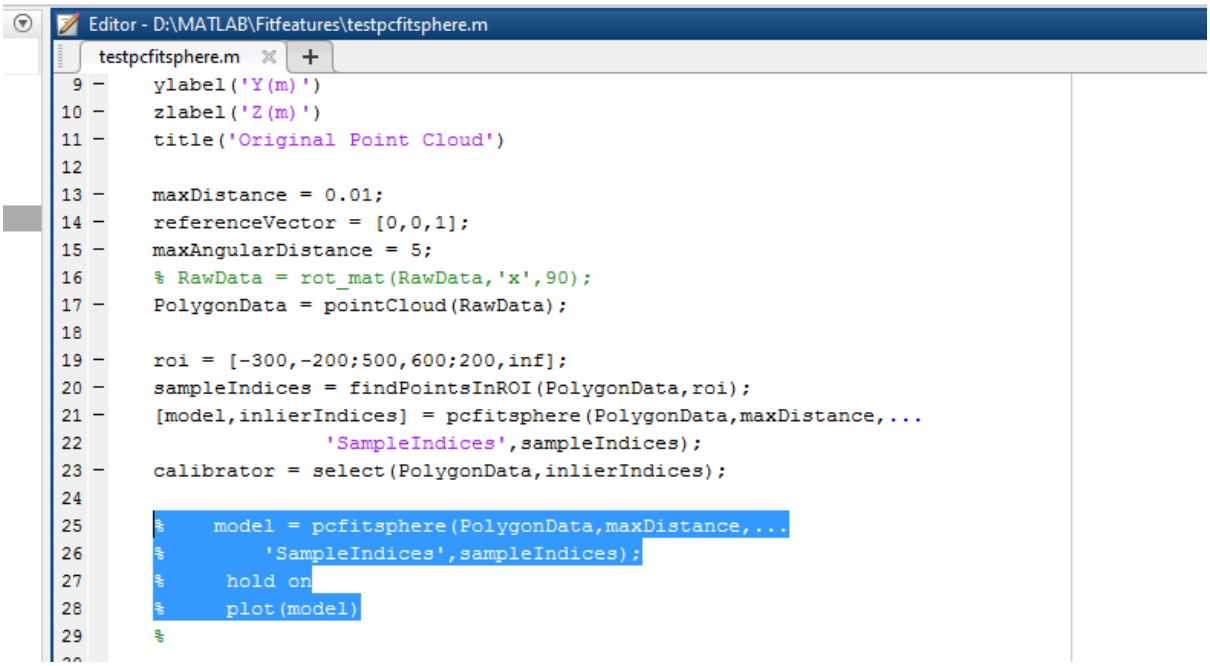


Figure D.19

STEP 4: Click on RUN under the EDITOR tab to run the programme. You can see how the function fits a sphere to the model. You can zoom in to have a closer view of the plot.

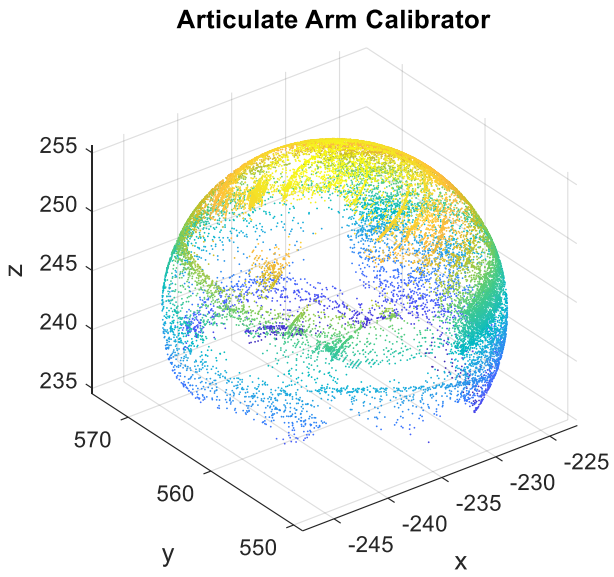


Figure D.20

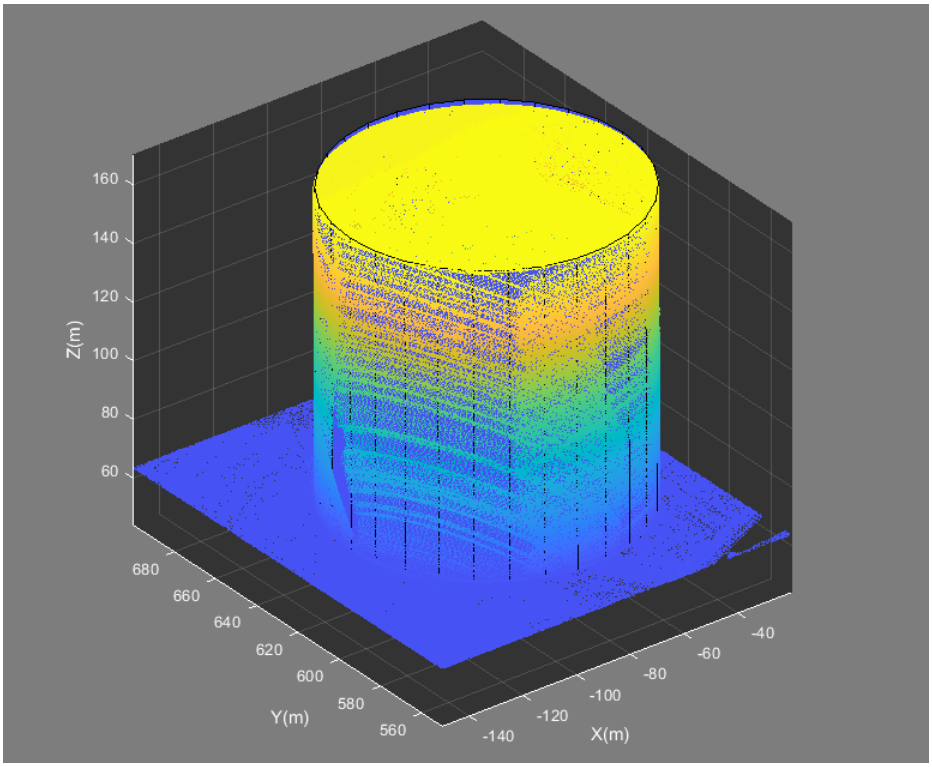


Figure D.21: Fitting a cylinder to the point cloud

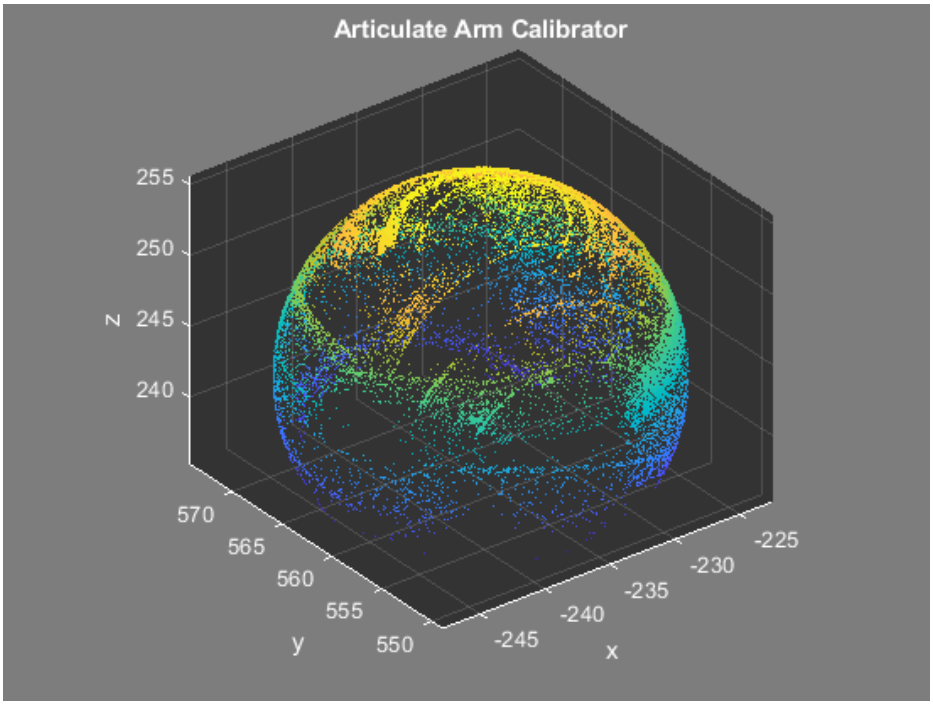


Figure D.22: Extraction of the sphere

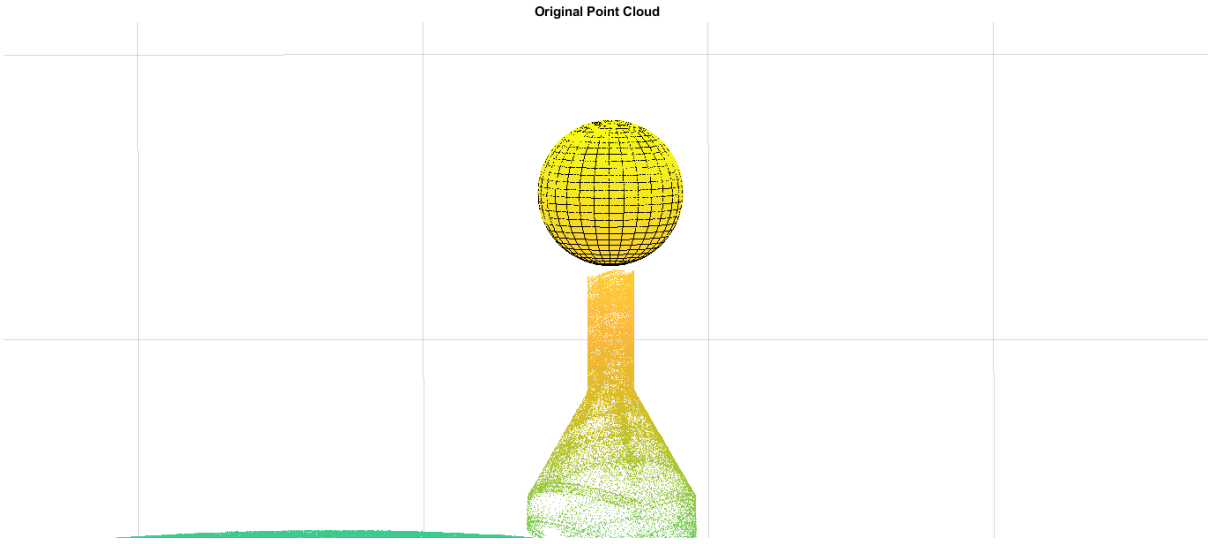


Figure D.23: Figures showing the fitting of the sphere to the point cloud

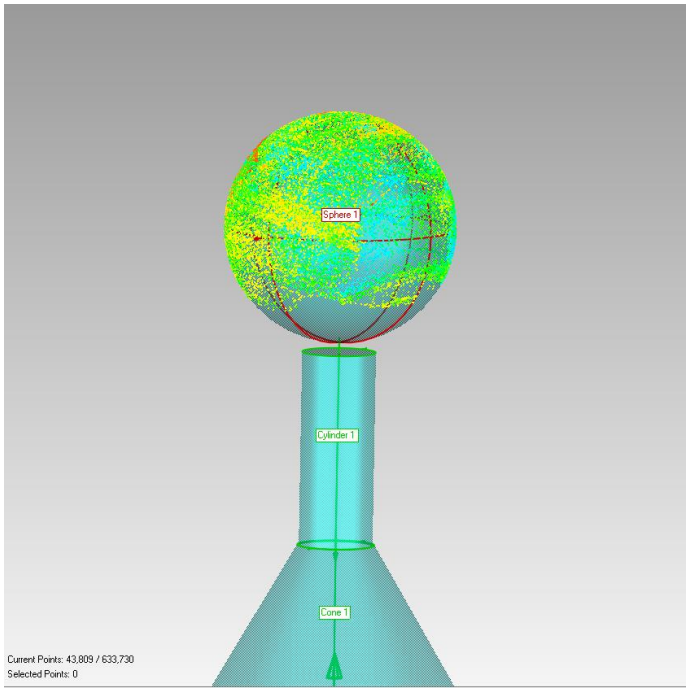


Figure D.24: Colour map of the sphere

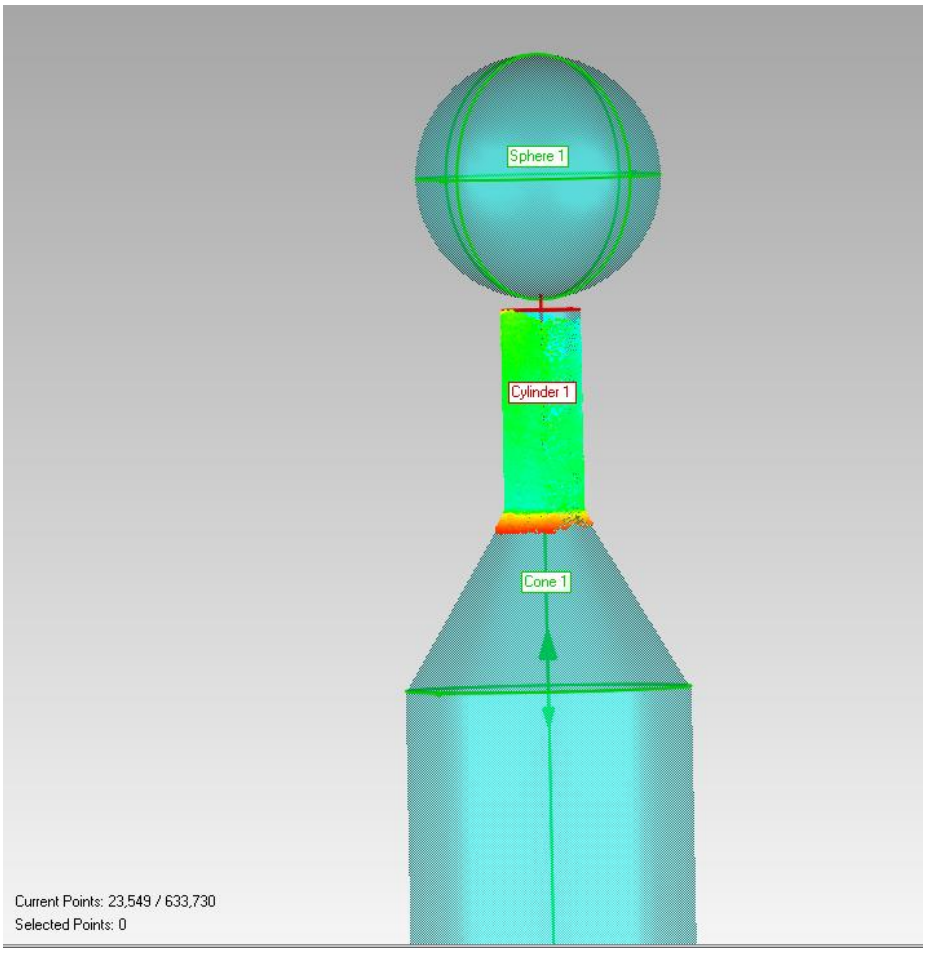


Figure D.25: Colour map of cylinder 1

Figure D.26: Colour map of cylinder 2

Table D.0.1: Comparison of the Arm Calibration sphere

	Methods for Measurement			
Shapes	Micrometre	MATLAB	Geomagic	UKAS Certificate
Sphere	Ø 25.398 mm	Ø25.4012 mm	Ø25.4031 mm	Ø25.39995 mm
Cylinder 1	Ø8.033 mm	Ø8.0388 mm	Ø8.01153 mm	
Cylinder 2	Ø29.525 mm	Ø29.326 mm	Ø29.5915 mm	

MATLAB Code for Feature Fitting Adapted from Mathworks [161]

```
% importing measurement data
RawData = importdata('plane_cylinder_sphere1.txt');

%plotting the point cloud
pcshow(p)
title('Calibrator sphere')
xlabel('X(m)')
ylabel('Y(m)')
zlabel('Z(m)')

maxDistance = 0.005;
referenceVector = [0,0,1];
maxAngularDistance = 5;
% RawData = rot_mat(RawData,'y',90);
% RawData = rot_mat(RawData,'x',90);
PolygonData = pointCloud(p);
figure
pcshow(PolygonData)

% specifying the region of interest to constrain the search -
based upon example from Mathworks [161]
roi = [-250,-200;520,600;50,inf];
sampleIndices = findPointsInROI(PolygonData,roi);
%                     [model,inlierIndices]                    =
pcfitcylinder(PolygonData,maxDistance);
% pc = select(PolygonData,inlierIndices);
%
   model                                                       =
pcfitcylinder(PolygonData,maxDistance,referenceVector,...
       'SampleIndices',sampleIndices);
    hold on
    plot(model)
```

```
% importing measurement data for the algorithm
RawData = importdata('plane_cylinder_sphere1.txt');
Figure;
pcshow(RawData)
title('Original Point Cloud')
maxDistance = 0.01;
% referenceVector = [0,0,1];
% maxAngularDistance = 5;
% RawData = rot_mat(RawData,'x',90);
PolygonData = pointCloud(RawData);
roi = [-300,-200;500,600;200,inf];
sampleIndices        =        findPointsInROI(PolygonData,roi);
model=pcfitsphere(PolygonData,maxDistance,referenceVector,...
       'SampleIndices',sampleIndices);
    hold on
    plot(model)
```

pcfitcylinder and pcfitplane adapted from Mathworks [161]

```
% importing measurement data for the algorithm
RawData = importdata('plane_cylinder_sphere1.txt');
 % plotting the point cloud of the measurement data
figure
pcshow(RawData)
title('Original Point Cloud')
% setting initial parameters based upon example from Mathworks
[161]
maxDistance = 0.02;
referenceVector = [0,0,1];
maxAngularDistance = 5;
% RawData = rot_mat(RawData,'x',90);
PolygonData = pointCloud(RawData);

 [model1,inlierIndices] = pcfitplane(PolygonData,...
    maxDistance,referenceVector,maxAngularDistance);
plane1 = select(PolygonData,inlierIndices);
remainPtCloud = select(PolygonData,outlierIndices);
% specifying region of interest to constraint the search
roi = [-inf,inf;-inf,inf;-inf,inf];
sampleIndices = findPointsInROI(remainPtCloud,roi);

[model2,inlierIndices,outlierIndices]                          =
pcfitplane(remainPtCloud,...
            maxDistance,'SampleIndices',sampleIndices);
plane2 = select(PolygonData,inlierIndices);
remainPtCloud = select(PolygonData,outlierIndices);
 % plotting the first plane
figure; pcshow(plane1);figure; pcshow(plane2);
```

Appendix E MATLAB Code for Boundary Profiling using Point Cloud Data Adapted from MATLAB

MATLAB Code for profile monitoring

```
% Damage Detection
% Importing measurement data and decimating the original point
cloud to
% Remove noise and outliers using downsample
Rawdata = importdata('mymodel.txt');
a2 = downsample(Rawdata, 10);
a2 = a2(a2(:,3)>50,:);
figure;whitebg([0.2 0.2 0.2]);
sp1 = subplot(211);
%Centralizing
a2(:,1) = a2(:,1)-mean(a2(:,1));
a2(:,2) = a2(:,2)-mean(a2(:,2));
pcshow(a2);hold on;

%Rotation Matrix about Z-Axis
theta = 0.1;
M = [cosd(theta) -sind(theta) 0; sind(theta) cosd(theta) 0; 0 0
1];
theta1 = 0.1;
M1 = [cosd(theta1) -sind(theta1) 0; sind(theta1) cosd(theta1)
0; 0 0 1];
% figure, defining the profile line as it moves over the model
profileline = a2(a2(:,1)>0 & floor(a2(:,2)) == 0, :);
h1                                                              =
plot3(profileline(:,1),profileline(:,2),profileline(:,3),'.','
Color','r');
hold off;
xlabel('x');ylabel('y');zlabel('z');grid on;
sp2 = subplot(212);
h2 = plot(profileline(:,1),profileline(:,2),'Color','y');
xlabel('x');ylabel('y');zlabel('z');grid on;
for i = 0 : 0.1 : 720
    %Rotation
    a2 = a2*M;
    %Define the line point set and expressing it with the
rotational
    % matrix about z-axis
    profileline = a2(a2(:,1)>=0 & abs(a2(:,2)) < 1e-2, :);
    theta1 = -i;
    M1   =   [cosd(theta1)   -sind(theta1)   0;   sind(theta1)
cosd(theta1) 0; 0 0 1];
    profileline = profileline*M1;
    profileline = sortrows(profileline,1);
    tempradius                                                  =
sqrt((profileline(:,1)).^2+(profileline(:,2)).^2);
```

```
set(h1,'xData',profileline(:,1),'yData',profileline(:,2),'zDat
a',profileline(:,3));
    set(h2,'xData',tempradius,'yData',profileline(:,3));
    xlim(sp1,[-120 120]);
    ylim(sp1,[-120 120]);
    zlim(sp1,[-10 120]);
    xlim(sp2,[00 120]);
    ylim(sp2,[0 150]);
    drawnow;
    pause(0.5);
end
```

Appendix F MATLAB Code for Simplex Optimisation Adapted from MATLAB

MATLAB code for optimisation of circfit

```
function p = optimisation6

% Initial adjustable set points for the nominal radius of the
system, radius of the dent and noise level
% (tolerance for the Articulate Arm is 50microns)
NoiseLevel=0.065;
NomRadius=1;
DentRadius=0.65;
DentNoiseLevel=NoiseLevel;

% The NomialVariables define the x, y, and radius
NominalVariables=[2.4,2,NomRadius];

% Parametric function for defining a circle
% And generating random points through which a circle is being
fitted to these points.

x1          =          NominalVariables(3)*(sind(12:0.1:348))'+
NoiseLevel*rand(3361,1)+NominalVariables(1);
x2          =          NominalVariables(3)*(cosd(12:0.1:348))'+
NoiseLevel*rand(3361,1)+NominalVariables(2);

% Adding a measurement dent with a smaller radius
if DentRadius>0
x11                                                              =
DentRadius*(sind(90:270))'+DentNoiseLevel*rand(181,1)+NominalV
ariables(1);
x21                                                              =
DentRadius*(cosd(90:270))'+DentNoiseLevel*rand(181,1)+NominalV
ariables(2)+NominalVariables(3);
% combining both sets of data
x1 = [x1; x11];
x2 = [x2; x21];
end

pg_initial = [1.09 1.21 1.24]; % Initial guess for implementing
fminsearch
ps = [x1 x2];
options = optimset('PlotFcns',@optimplotfval);
x = fminsearch(@(pg) myfun(pg,ps), pg_initial, options);
figure;
% Add set of points generated to the circle
whitebg([0.2 0.2 0.2]); % background colour
plot(x1,x2,'.','Color','r');hold on;
```

```
% Plot of the circle
t = 0:360;
xe = x(3)*sind(t)+x(1);
ye = x(3)*cosd(t)+x(2);
plot(xe,ye);hold on;
plot(x(1),x(2),'o');hold off;
title('circfit of sampled points')
axis equal;

% trying to compare system interaction and difference in
response with changes
% in the initial guess of the system
pg_initial = [0.9 0.21 1.24];  % Initial guess for implementing
fminsearch
y = fminsearch(@(pg) myfun(pg,ps),pg_initial, options);
figure;
% add set of points generated to the circle
whitebg([0.2 0.2 0.2]);
plot(x1,x2,'.','Color','r');hold on;
% Plot of the circle
t = 0:360;
xe = x(3)*sind(t)+y(1);
ye = x(3)*cosd(t)+y(2);
plot(xe,ye);hold on;
plot(y(1),y(2),'o');hold off;
xlabel('X')
ylabel('Y')
axis equal;

P = [NominalVariables ; x ;y]

Residual   =   [NominalVariables   ;   x-NominalVariables   ;y-
NominalVariables]

Difference = y-x

function F1 = myfun(pg,ps)
    F1 = 0;
    for i = 1 : size(ps,1)
        E = pg(3)^2-((ps(i,1)-pg(1))^2+(ps(i,2)-pg(2))^2);
        F1 = abs(E) + F1;
    end
end
testtilefigs([2 3],5)
end
```

Table F.0.1: Table showing the residual computation of the optimisation system

	Coordinates		
Primitive circle	X coordinates	Y coordinates	Radius
Perfect Circle	0	0	0
Circle with noise	0.0345	0.0328	-0.0008
Circle with missing data	0.0332	0.0338	-0.0001
Circle with measured dent	0.0318	0.0309	-0.0009
Mean	0.024875	0.024375	-0.00045

The system is robust because the interaction between the first and second functions remains the same with alteration of the initial guess as to the factor. No matter how high or low the step length of the initial guess, the system is optimised to a current function value within 33.6924 mm to 36.3253 mm. Increasing the initial guess values increases the number of iterations. However, the effect of the step length is minimal and the number of iterations changes with each run, which means this is a dynamic system. At a very high initial guess value, the function value decreases while the iteration increases, but the number of iterations differs with each run. At a lower value of initial guess, function value increases while iteration changes with each run, but the current function value of the optimisation function remains constant. The function value increase with a further decrease in the initial guess. At a much lower value for the initial guess (1.9/1.21), the function value increases drastically while the changes in the iteration are small in section 4.3.1.

Appendix G MATLAB Code for profile monitoring using rotational matrix

```
% Damage Detection
a = importdata('mymodel.txt');
a1 = downsample (a, 10);
a2 = a1(a1(:,3)>50,:);
figure;
pcshow(a2)

%Centralizing
a2(:,1) = a2(:,1)-mean(a2(:,1));
a2(:,3) = a2(:,3)-mean(a2(:,3));

%Central Point
x0 = 0;
y0 = 0;

%Rotation Matrix with Z-Axis
theta = 0.1;
M = [cosd(theta) -sind(theta) 0; sind(theta) cosd(theta) 0; 0
0 1];

%Line of each degree based upon example from Mathworks [162]
x = zeros(1,1);
y = zeros(1,1);
z = zeros(1,1);

tempdata = a2(a2(:,1)>0 & fix(a2(:,2)) == 0, :);
h = plot(tempdata(:,1),tempdata(:,3),'.');
title('Boundary Profile');

for i = 0: 0.1: 360
    %Define the line point set
    tempdata = a2(a2(:,1)>0 & floor(a2(:,3)) == 0, :);
    %Rotation
    a2 = a2*M;
    set(h, 'xData',tempdata(:,1),'yData',tempdata(:,3));
    xlim([0 90])
    ylim([90 120])
    drawnow;
    F(i) = getframe(gcf);
end
%Function is VideoWriter is based on example from Mathworks
[90]
video = VideoWriter('Boundary_Profile1.avi','Uncompressed
AVI');
open(video)
writeVideo(video,F);
close(video)
```

MATLAB code for generating simulation data _ ordered points

```
function [data] = pointcloudcylinder(varargin)
% defining parameters
N =1000;
R = 5;
theta =2*pi/N;

counter = 0;
x1 = zeros(1,1);
y1 = zeros(1,1);
z1 = zeros(1,1);

for i = 0 : 0.1 : 10
   for j = 0:theta:2*pi
       counter = 1 + counter;
    x1(counter,1) = R*cos(j);
    y1(counter,1) = R*sin(j);
    z1(counter,1) = i;
   end
end
for R = 0.1 : 0.1 : 5
   for j = 0:theta:2*pi
       counter = 1 + counter;
    x1(counter,1) = R*cos(j);
    y1(counter,1) = R*sin(j);
    z1(counter,1) = 0;
   end
end
for R = 0.1 : 0.1 : 5
   for j = 0:theta:2*pi
       counter = 1 + counter;
    x1(counter,1) = R*cos(j);
    y1(counter,1) = R*sin(j);
    z1(counter,1) = 10;
   end
end
plot3(x1,y1,z1, '.');
PCRaw = [x1,y1,z1];
PCNoisy = PCRaw;

% To create a damage on the model
%PCNoisy(PCNoisy(:,2)>3 & PCNoisy(:,3)>4 & PCNoisy(:,3)<6, 2) = 3;
PCNoisy(PCNoisy(:,2)>3.4 & PCNoisy(:,3)>4.6 & PCNoisy(:,3)<5.6, 2) =
3.4;
PCNoisy(PCNoisy(:,2)>4.8 & PCNoisy(:,3)>9 & PCNoisy(:,3)<9.78,
3)=4.8;
```

```
 data = PCNoisy;
pcshow(PCNoisy)
axis equal;
grid on;
title('Ordered Point Cloud Data')
xlabel('X(mm)');
ylabel('Y(mm)');
zlabel('Z(mm)');
end
```

MATLAB code for generating simulation data _ randomly distributed points

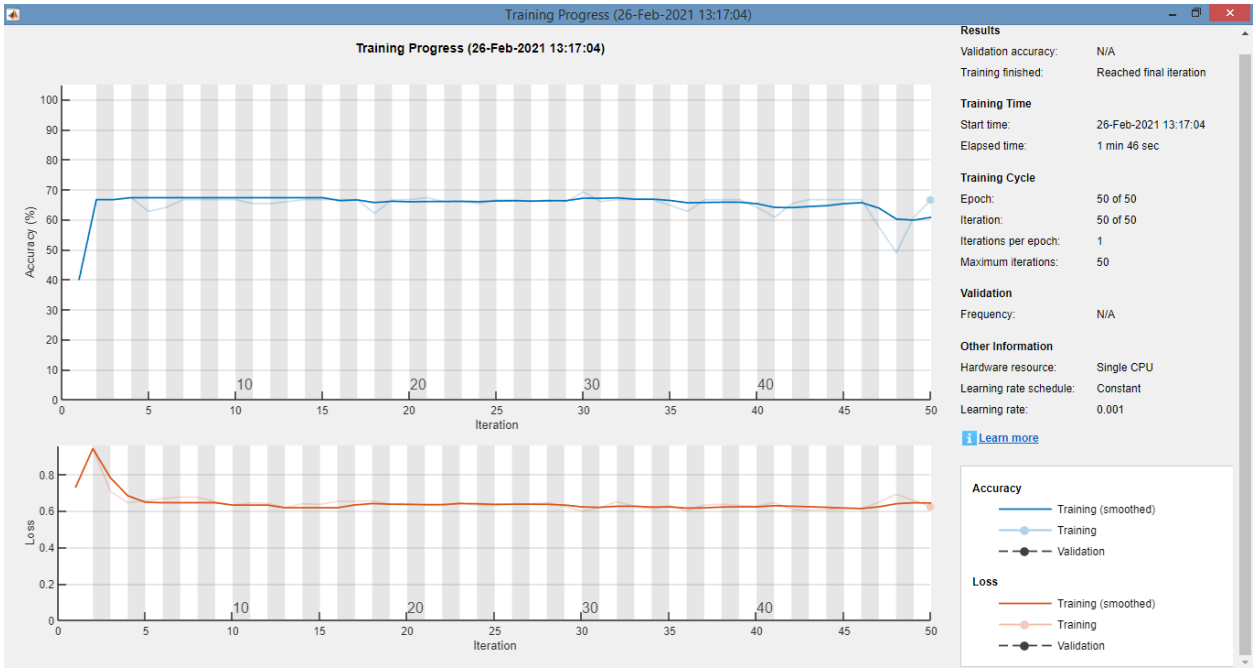
```
% Example: Varargin parameters
%   data = damaged_cylinder_pc([3,5],0.1,0.00002,1,[0,90, 1.5],0.5);
%   pcshow(data);view(0,90)

function varargout = damaged_cylinder_pc(varargin)
    % Para1: dimension <height, radius>
    % Para2: chaos level
    % Para3: point density float value (0 - 1)
    % Para4: bool - random sampling or even
    % Para5: damaged_location [theta1, theta2] (specified in 0 - 360
degs)
    obj_dimension = varargin[7];
    obj_chaos_level = varargin[163];
    obj_density = varargin[154];
    obj_dloc = varargin[158];
    obj_dlevel = varargin[163];

    if varargin[163]
        disp('Random distribution');
        % Bottom and Top
        pc_size = floor(1/obj_density);

        % Side Wall Unwrapped PC
        % pc_side[radians, z_height]
        pc_side_rz = [rand(pc_size,1)*2*pi*obj_dimension(2),...
            rand(pc_size, 1)*obj_dimension(1)];
        pc_degs = (pc_side_rz(:,1)/(2*pi*obj_dimension(2)))*360;

        [r,~] = size(pc_side_rz);
        d1 = reset_value(obj_dloc(1), obj_dloc(2));
        d2 = obj_dimension(2)/5;
        d3 = reset_value(obj_dloc(3)-d2, obj_dloc(3)+d2);
        pc_side1 = zeros(r,3);
        for i = 1 : r
            if pc_degs(i,1)>=obj_dloc(1) &&
pc_degs(i,1)<=obj_dloc(2)...
                    && pc_side_rz(i,2)>=obj_dloc(3)-d2...
```

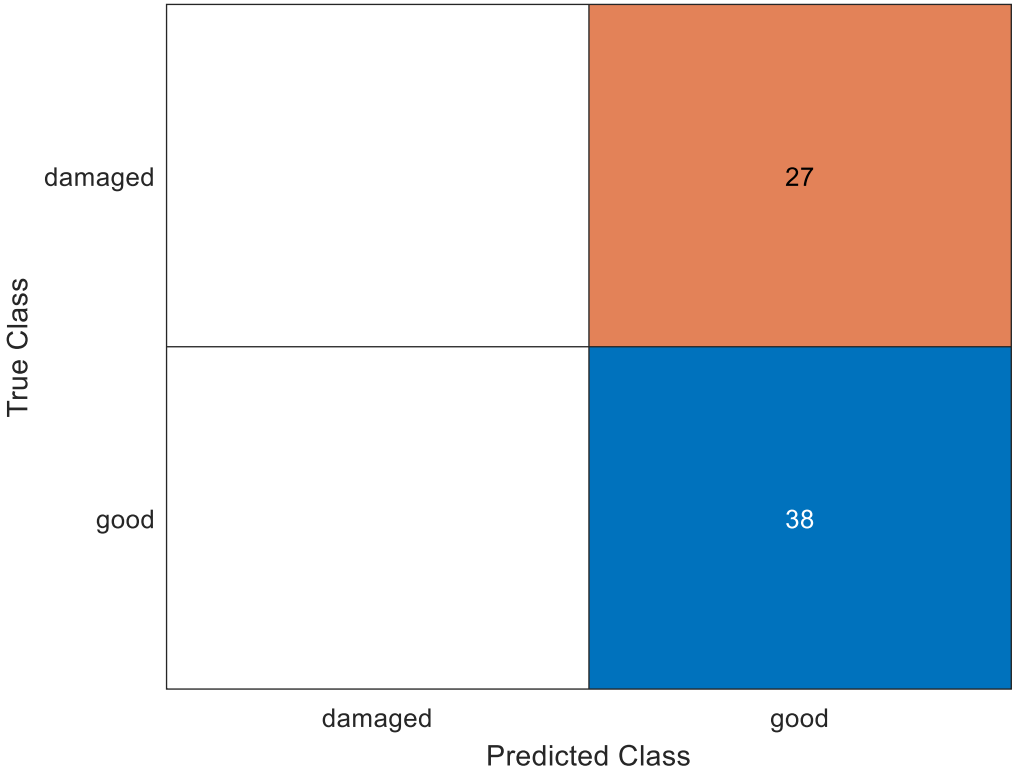
```
                    && pc_side_rz(i,2)<=obj_dloc(3)+d2
                n1=(obj_dimension(2)+obj_dlevel*...
                    (cosd(pc_degs(i,1)*d1(1)+d1(2))+...
                    rand*cosd(pc_side_rz(i,2)*d3(1)+d3(2))))...
                    *cosd(pc_degs(i,1));
                n2=(obj_dimension(2)+obj_dlevel*...
                    (cosd(pc_degs(i,1)*d1(1)+d1(2))+...
                    rand*cosd(pc_side_rz(i,2)*d3(1)+d3(2))))...
                    *sind(pc_degs(i,1));

                or1=obj_dimension(2)*cosd(pc_degs(i,1));
                or2=obj_dimension(2)*sind(pc_degs(i,1));
                if sqrt((n1)^2+(n2)^2) >= sqrt(or1^2+or2^2)
                    pc_side1(i,1) = or1;
                    pc_side1(i,2) = or2;
                else
                    pc_side1(i,1) = n1;
                    pc_side1(i,2) = n2;
                end
            else
                pc_side1(i,1)=obj_dimension(2)*cosd(pc_degs(i,1));
                pc_side1(i,2)=obj_dimension(2)*sind(pc_degs(i,1));
            end
            pc_side1(i,3)=pc_side_rz(i,2);
        end
        varargout[158]
 = pc_side1;
    else
        disp('Even distribution');
    end
end

function varargout = cylinder_inspection(varargin)
    pc_data = varargin[158];
    radius = zeros(1,1);
    for i = 1 : length(pc_data)
        radius(i,1) = sqrt((pc_data(i,1))^2+(pc_data(i,2))^2);
    end
    temp_1 = max(radius);
    d_range_x = 5;
    d_range_y = 50;
    subplot(221);
    h1 = plot3(pc_data(:,1),pc_data(:,2),pc_data(:,3),'.');hold on;
    h1_1 = plot([0,0],[0,0],'r');
    h2 = plot3(1,1,1,'o','Color','g');hold off;
    view(0,90);
    axis equal
    title('Original Point Cloud');
```

```
    subplot(222);
    h3 = plot(1,1);
    xlim([0 3]);
    ylim([3 6]);
    title('Projection Line of Selected Points');

    subplot(223);
    h4 = mesh([0,0],[0,0],zeros(2,2));
    view(0,90);
    xlim([0 250]);
    title('CWT Analysis of Projection Line');

    subplot(224);
    h5 = animatedline; hold on
    h6 = plot(0,0,'*','Color','r');hold off;
    title('Wavelet Coefficient');

    sum_v = zeros(1,1);
    Five_Days = zeros(10,1);

    % Two cycles
    for i = 0 : 360
        temp_data = pc_data(pc_data(:,1)>temp_1-temp_1/d_range_x...
            & pc_data(:,2)>-temp_1/d_range_y...
            & pc_data(:,2)<temp_1/d_range_y,:);
        pc_data = transform_pc(pc_data, 1, [0,0,1]);
        temp_data = sortrows(temp_data,3);

        % Calculate wavelet power spectrum based on example from
Mathworks
        [wt,f,c] = cwt(temp_data(:,1),'morse',length(temp_data));

        % Compute coefficient surface geometry variation with
wavelet spectrum
        a1 = sum(abs(wt(1:20,:)), 'All');

        % Use finance Ten-Days average line for calculating the
features
        Five_Days(1:end-1,1) = Five_Days(2:end,1);
        Five_Days(end,1) = a1;
        sum_v(i+1,1) = sum(abs(Five_Days))/10;

        % Animation updates

set(h1,'xdata',pc_data(:,1),'ydata',pc_data(:,2),'zdata',pc_data(:,3
));

set(h2,'xdata',temp_data(:,1),'ydata',temp_data(:,2),'zdata',temp_da
ta(:,3));
```

```
        set(h3,'xdata',(temp_data(:,3)),'ydata',temp_data(:,1));
        set(h4,'xdata',f,'ydata',c,'zdata',(abs(wt))');
        %set(h4,'xdata',f,'ydata',t,'zdata',(p)');
        if i > 10
            addpoints(h5,i, sum_v(i+1,1));
        end
        drawnow();
    end
%     sum_v(:,2)=(0:360)';
    disp(length(sum_v))
    [pks, locs] = findpeaks(sum_v(:,1),'MinPeakDistance', 359);
    set(h6, 'xdata', locs, 'ydata', pks);
    set(h1_1, 'xdata', [0, temp_1*cosd(locs(1)-5)],...
        'ydata', [0, temp_1*sind(locs(1)-5)]);
    drawnow();
    varargout[158] = temp_data;
    varargout[163] = sum_v;
end

% code for calling both the damaged_cylinder_pc function and the
cylinder_inspection function
% figure ([255 255 255]);
% Cylinder:
% Height: 3
% Radius: 5
% Resolution: 1e-5
% Damage start angle: 0
% Damage end angle: 90
% Damage z-height: 1.5
% Damage level: 0.5
a = damaged_cylinder_pc([5,5],0.1,0.00001,1,[70, 90, 1.5], 0.2);
pcshow(a);

% Subplot four status images
% Image 1: Original Point cloud and Selected Points
% Image 2: The projected line of selected points
% Image 3: The wavelet
[b,c]=cylinder_inspection(a);
```

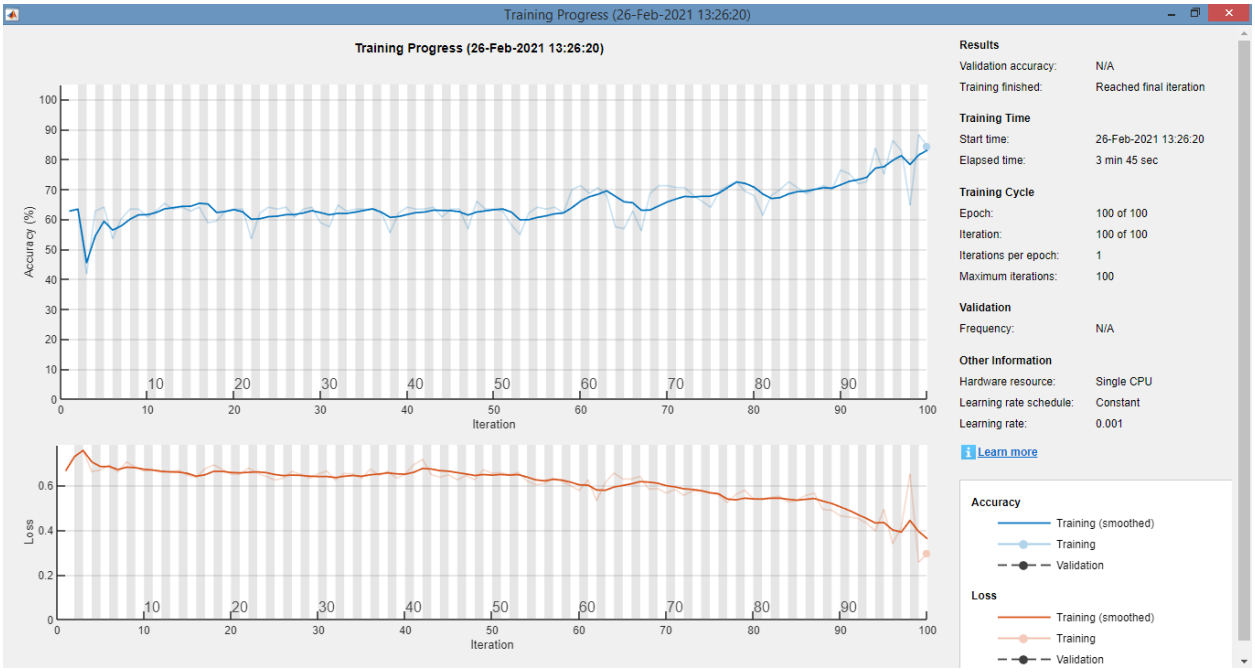
Appendix H LSTM Training of First Sets of Data Producing False Classification Due to Preprocessing Methods of Generating and Classifying the Slices

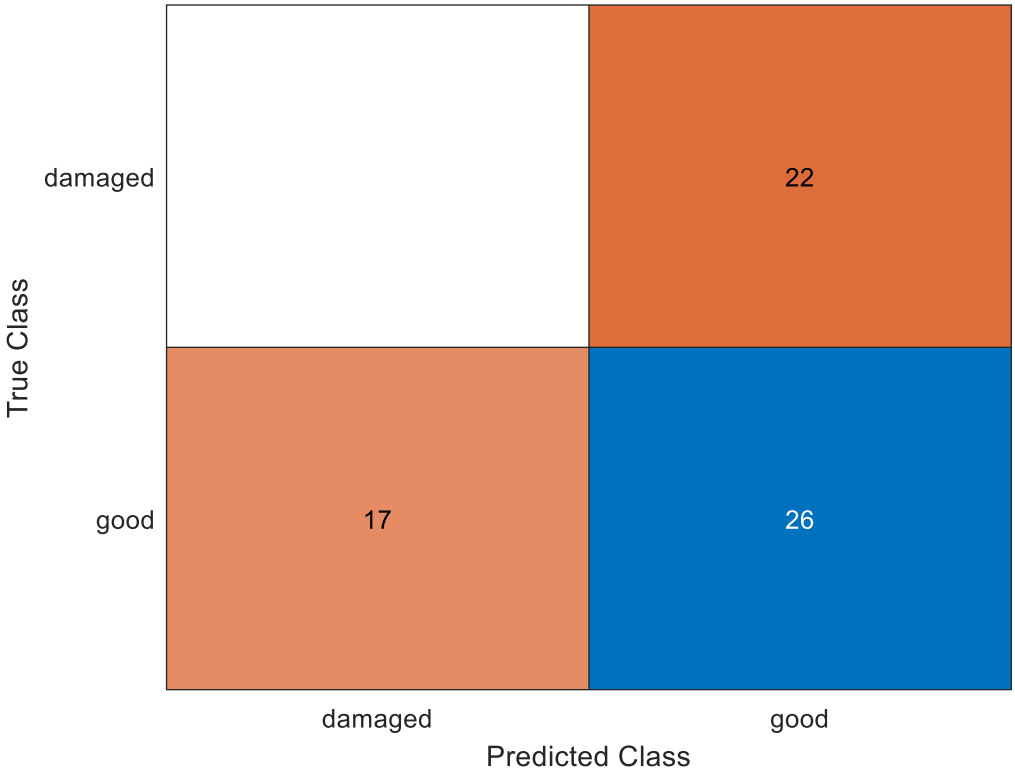




Training LSTM with lstm_data3/lstm_classes3

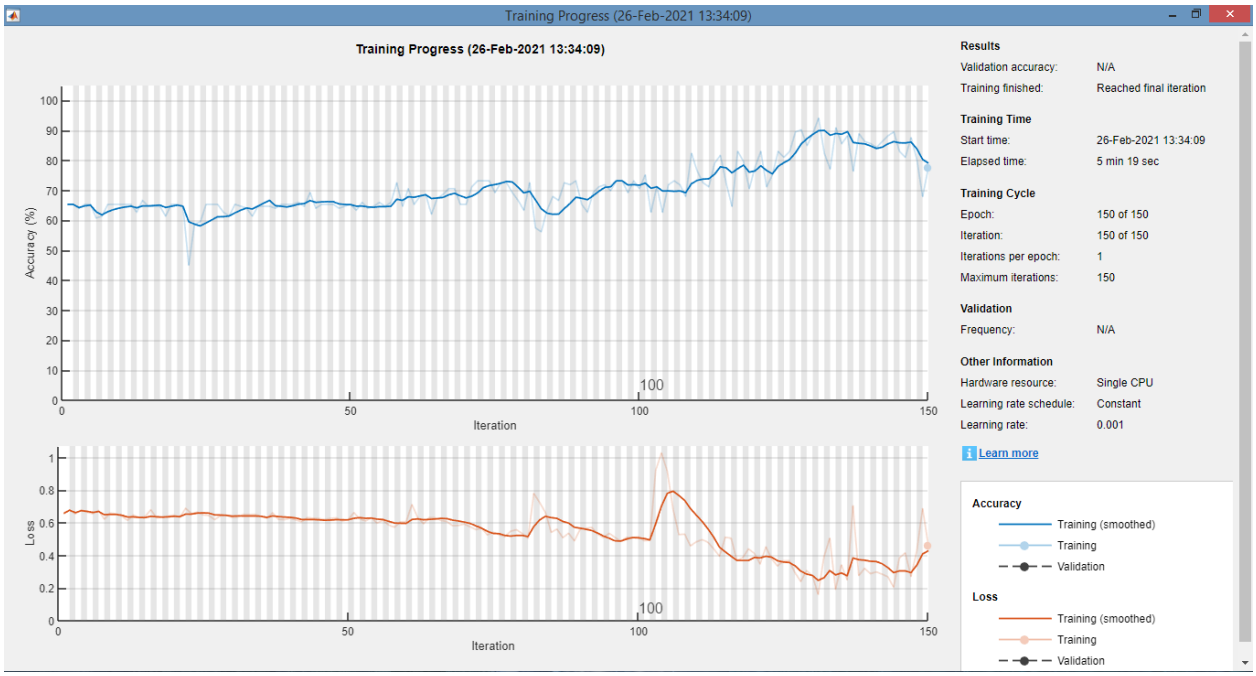
accuracy = 58% Epoch = 50

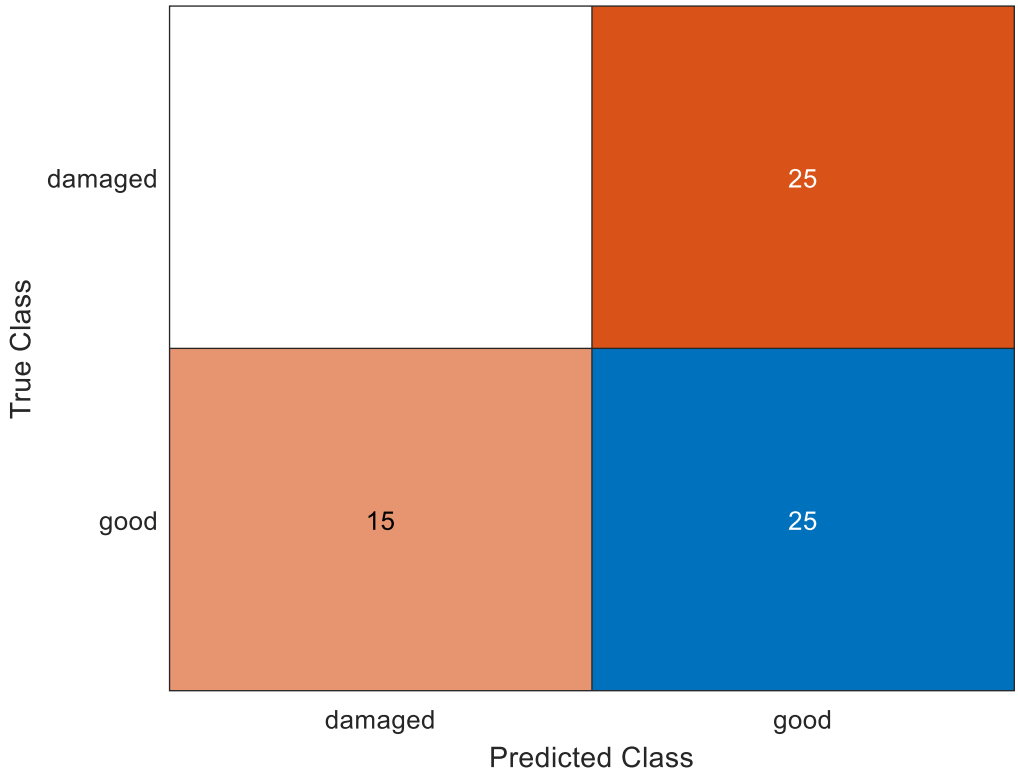




Training LSTM with lstm_data3/lstm_classes3

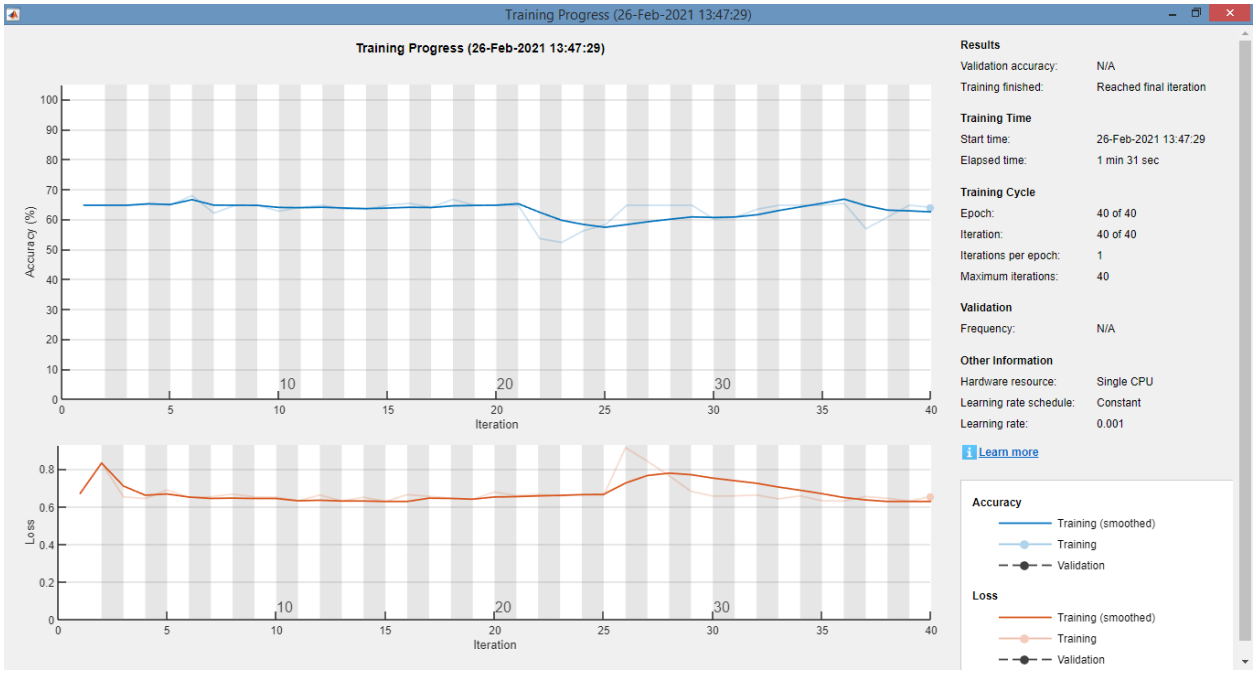
accuracy = 40% Epoch = 100

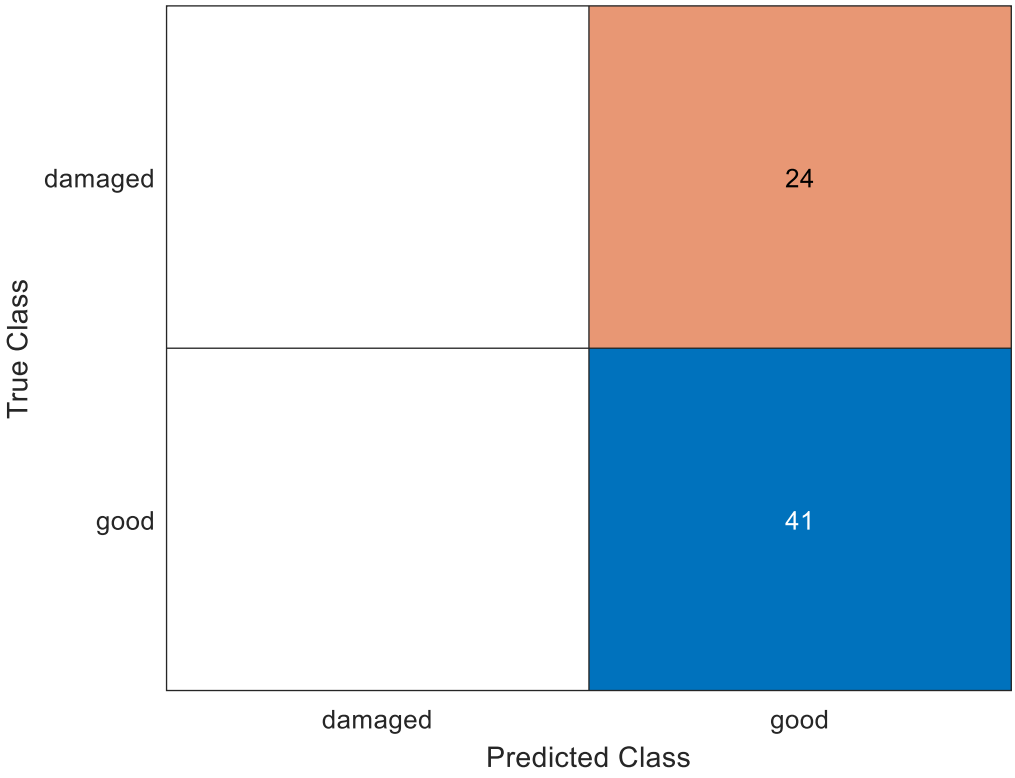




Training LSTM with lstm_data3/lstm_classes3

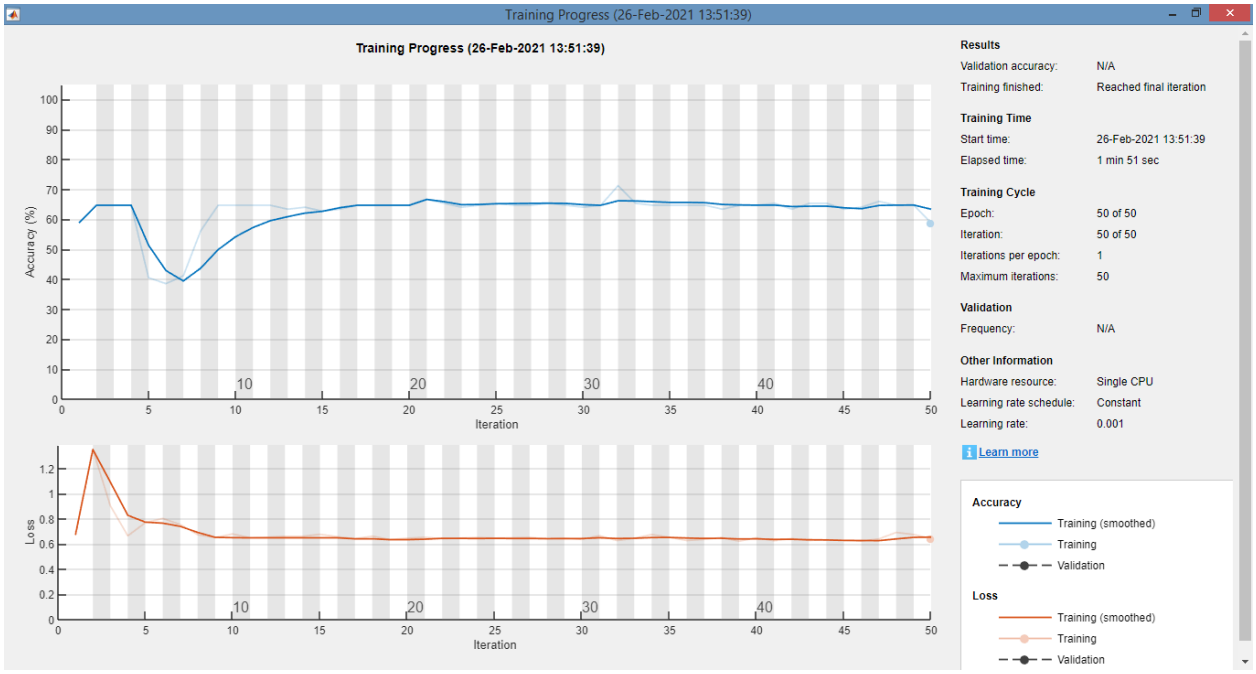
accuracy = 38% Epoch = 150

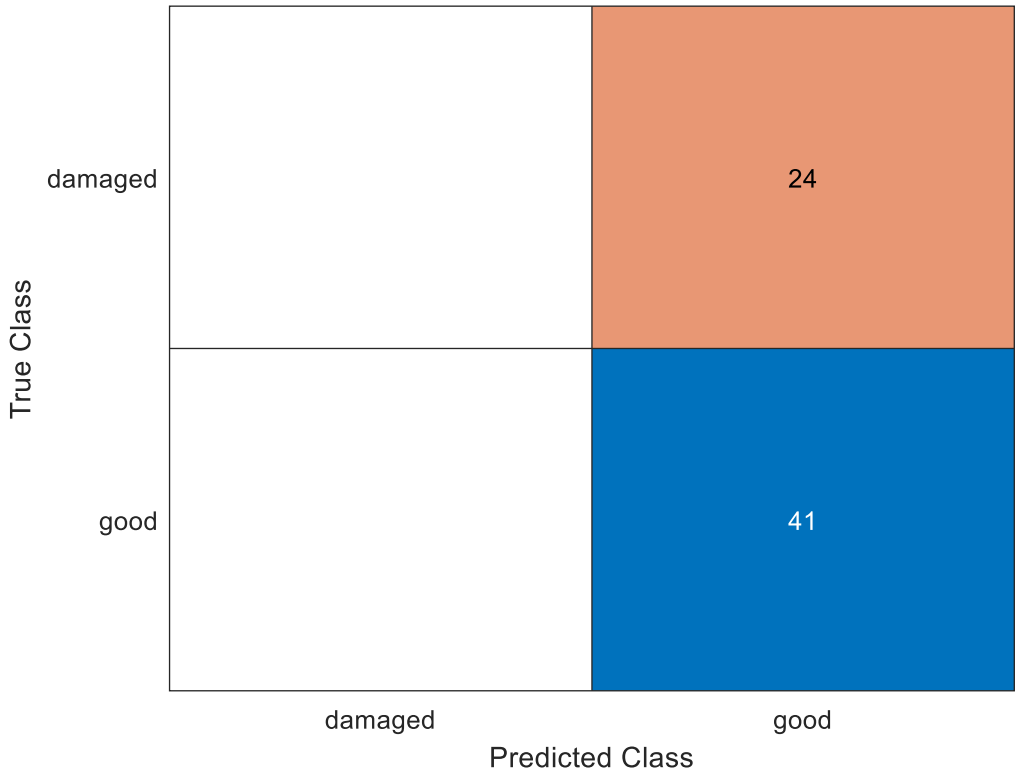




Training LSTM with lstm_data3/lstm_classes3

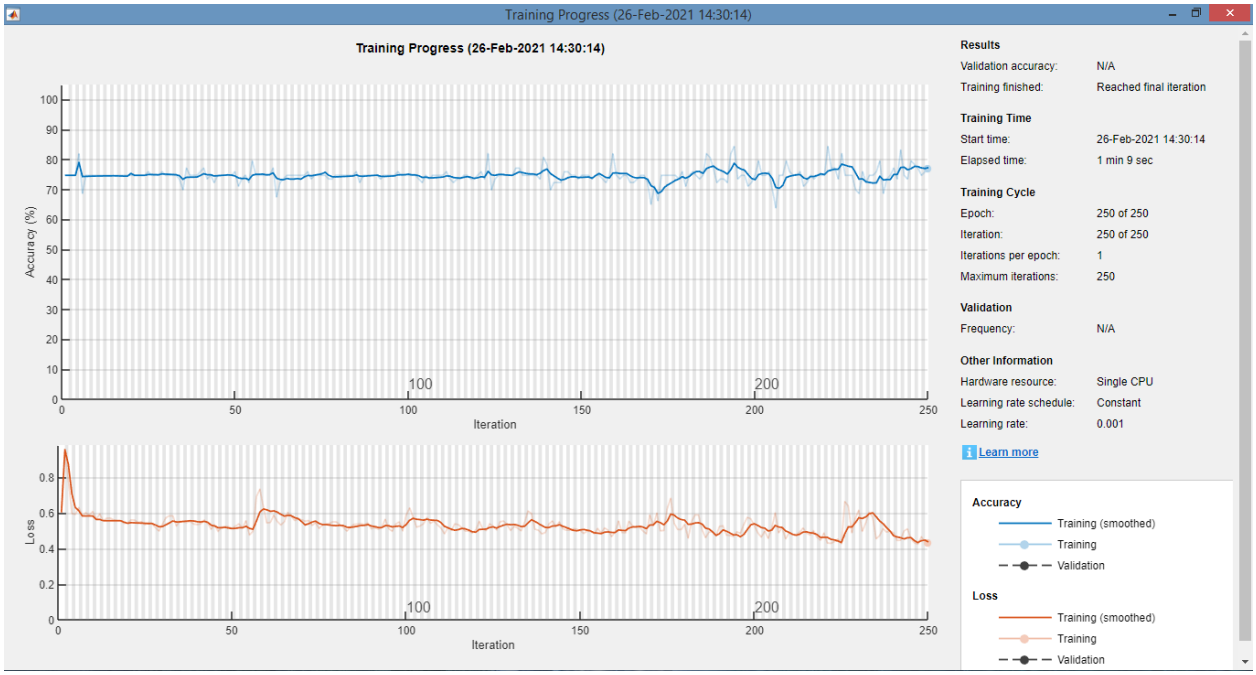
Accuracy = 63% Epoch = 40

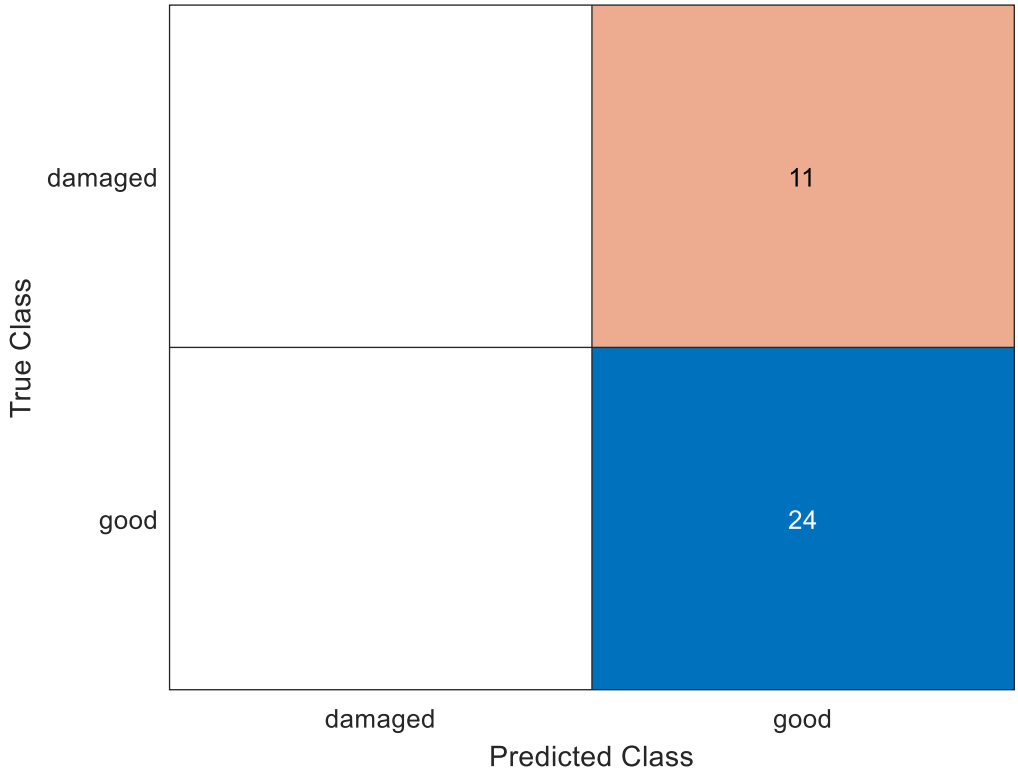




Training LSTM with lstm_data3/lstm_classes3

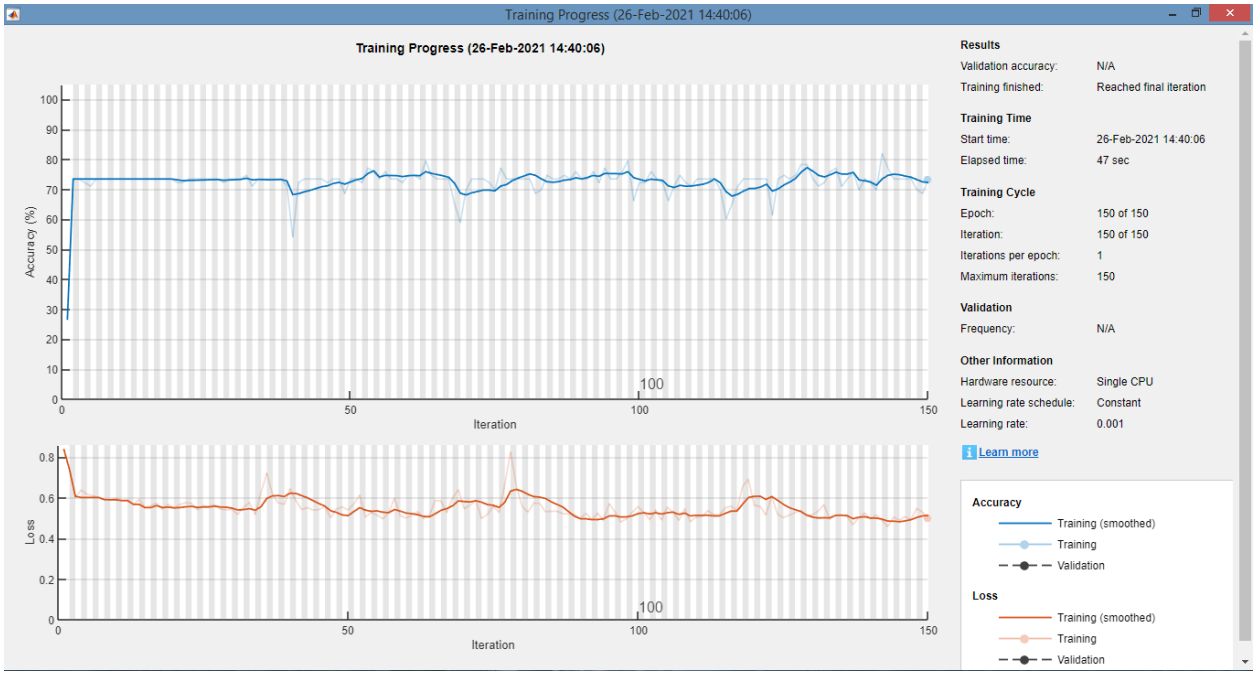
Accuracy = 63% Epoch = 50

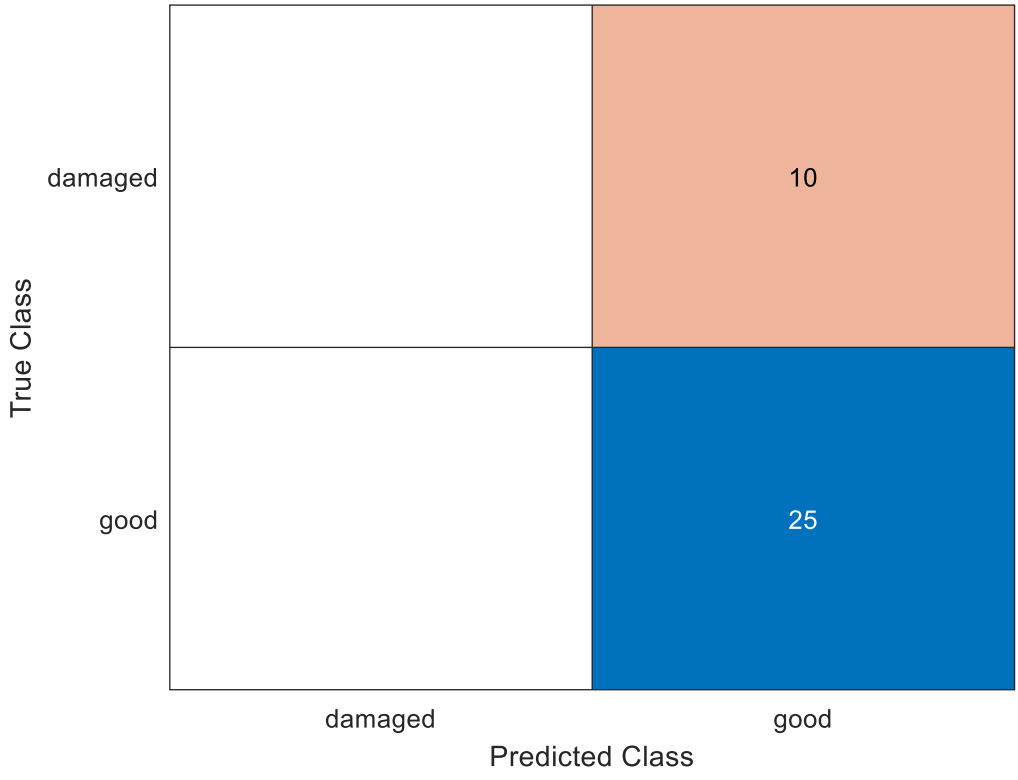




Training LSTM with lstm_data/lstm_classes

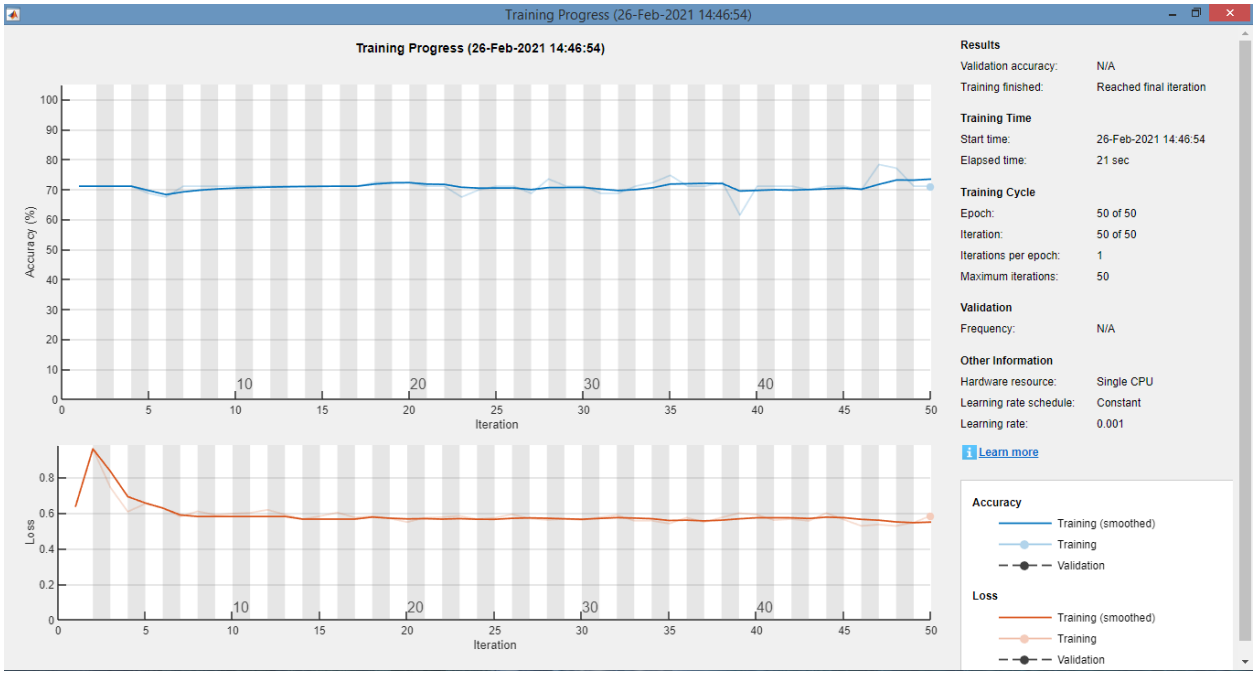
Accuracy = 68% Epoch = 250

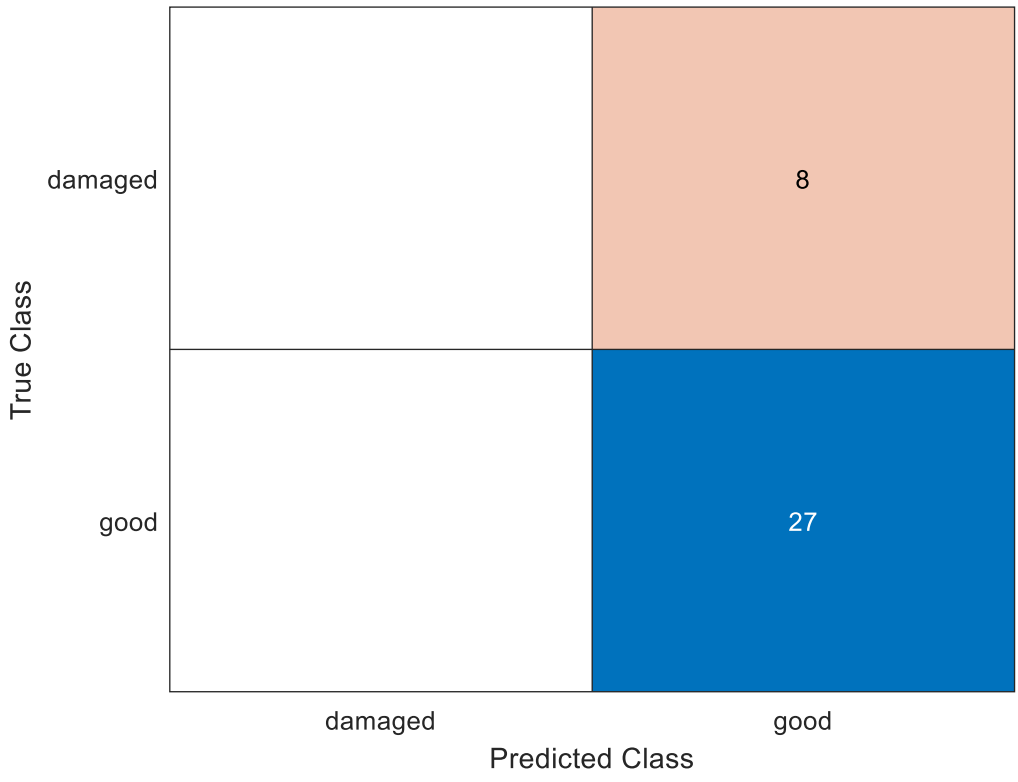




Training LSTM with lstm_data/lstm_classes

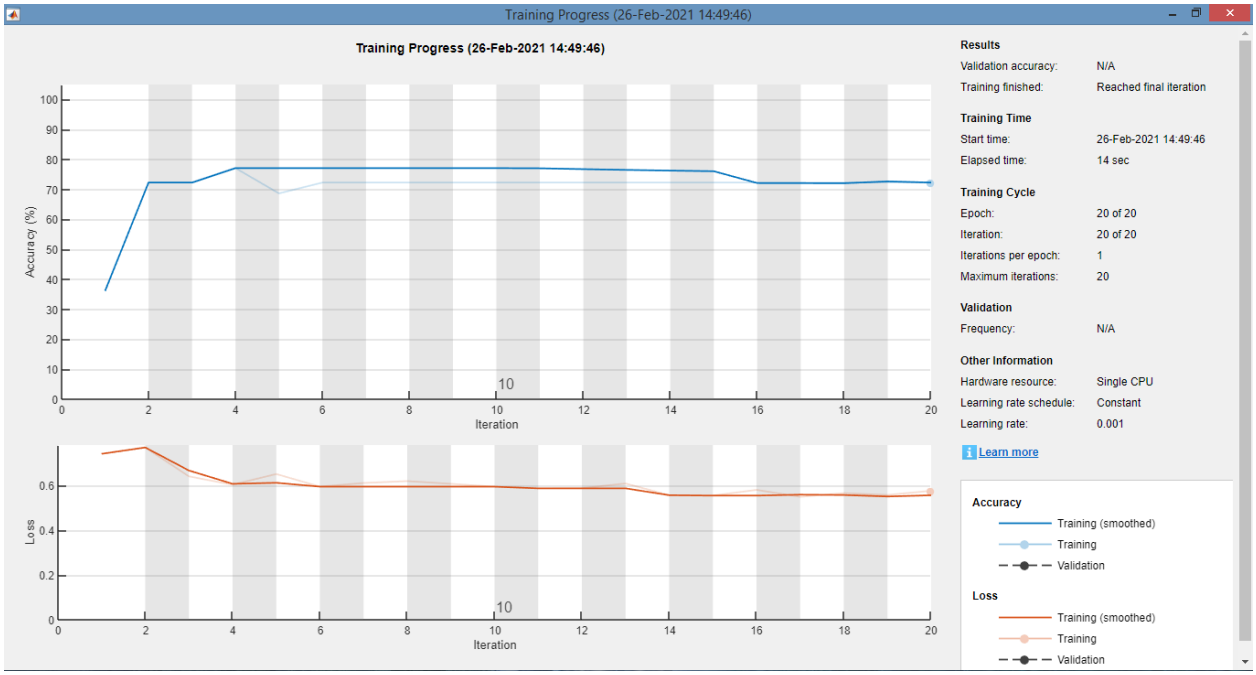
Accuracy = 71% Epoch = 150

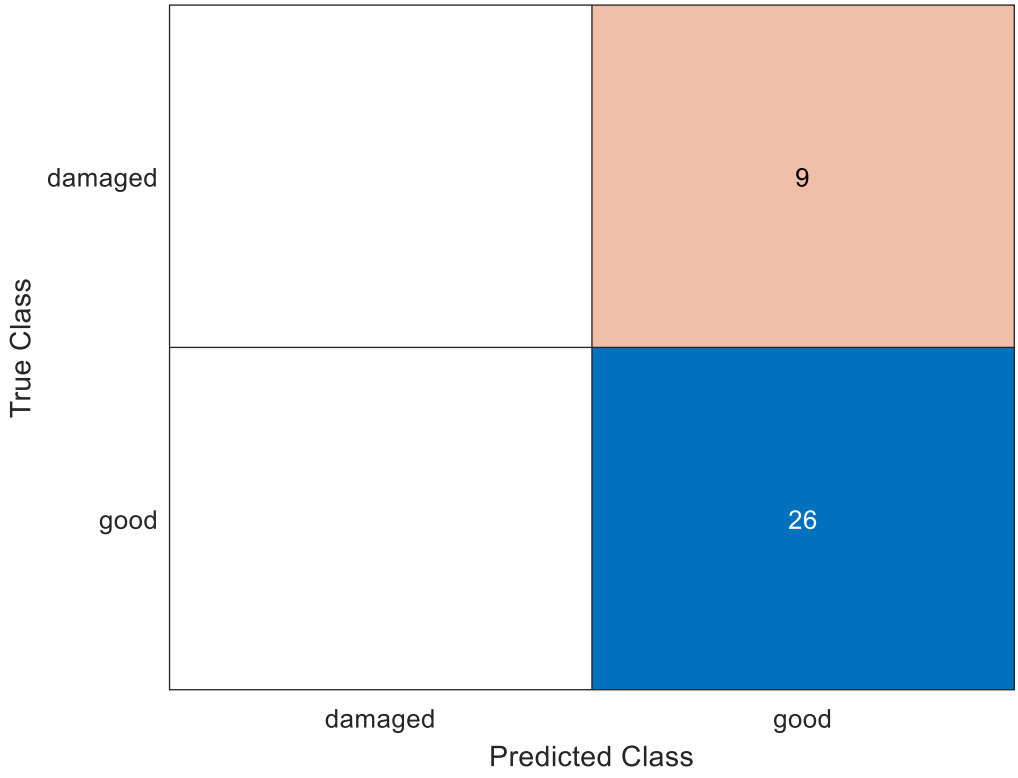




Training LSTM with lstm_data/lstm_classes

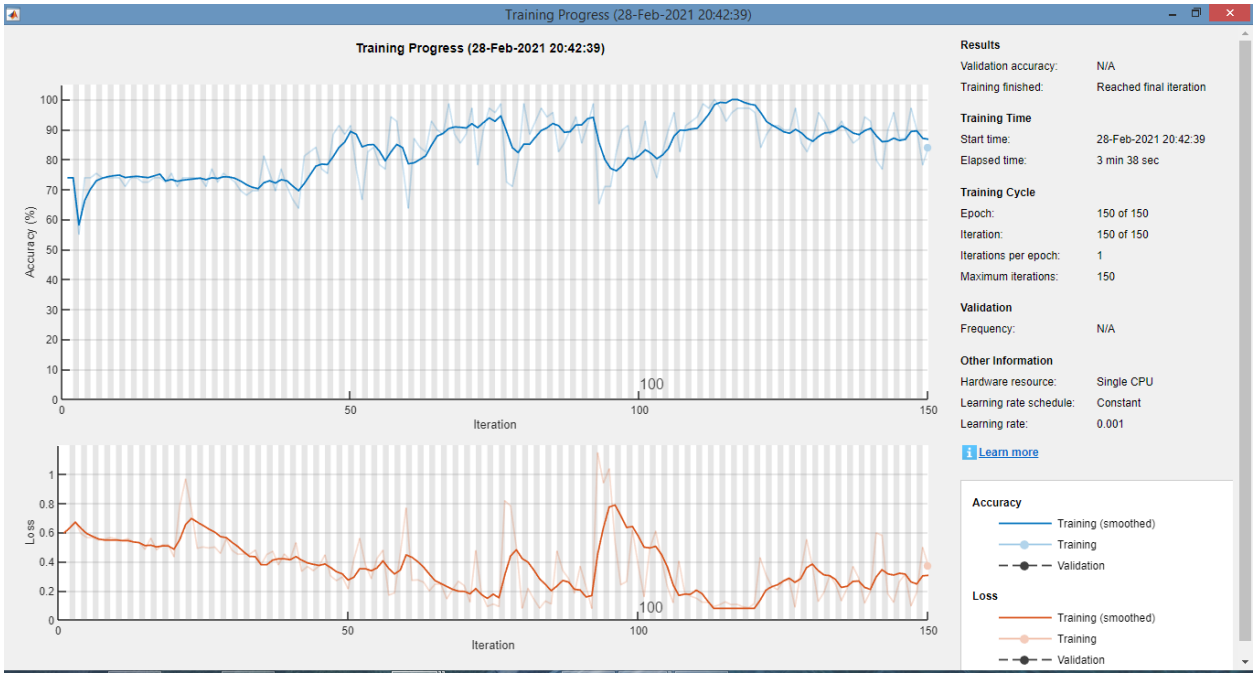
accuracy = 77% Epoch = 50

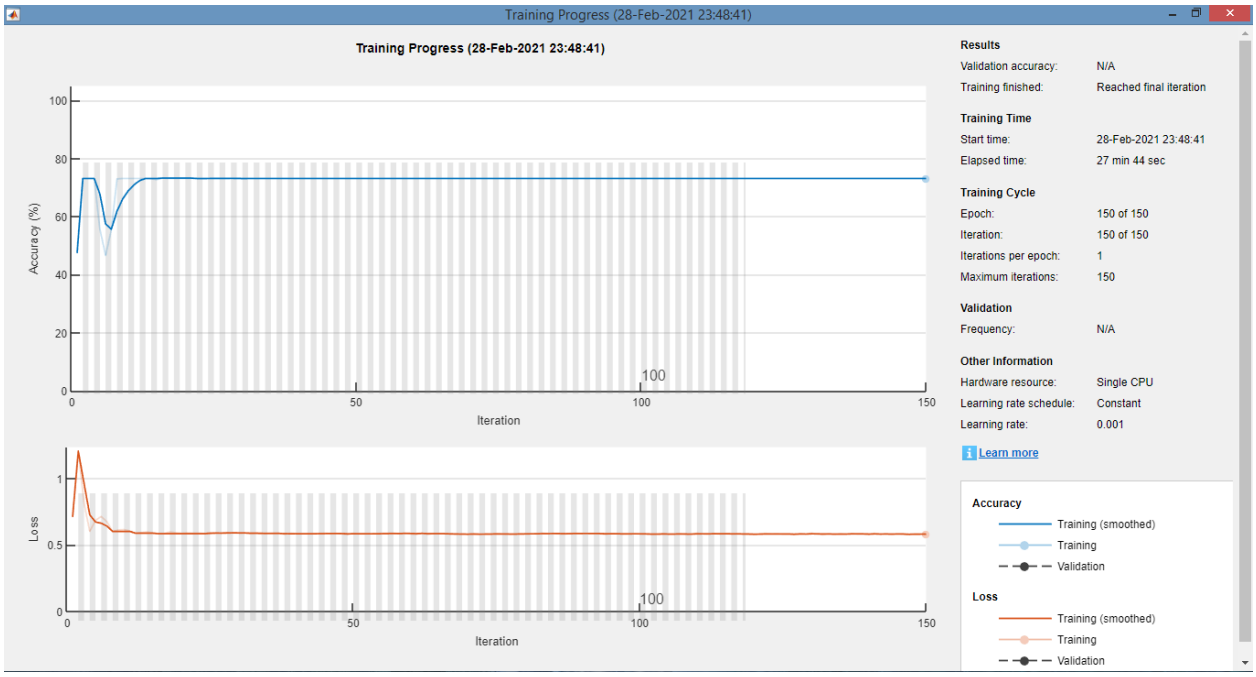


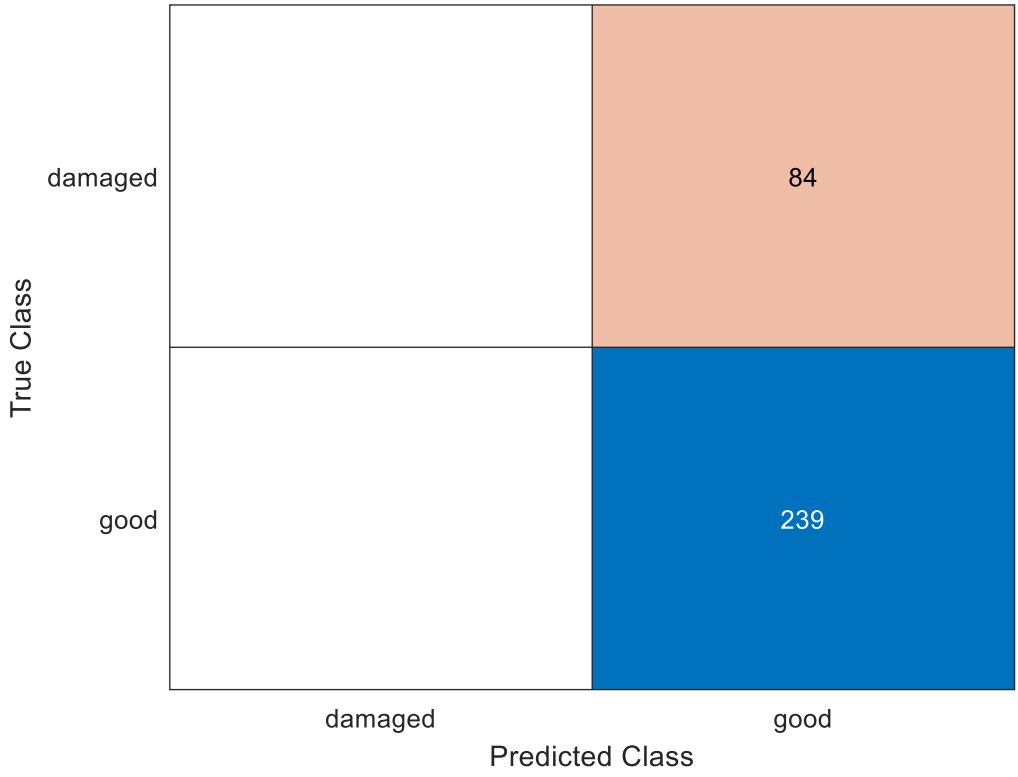


Training LSTM with lstm_data/lstm_classes

accuracy = 74% Epoch = 20

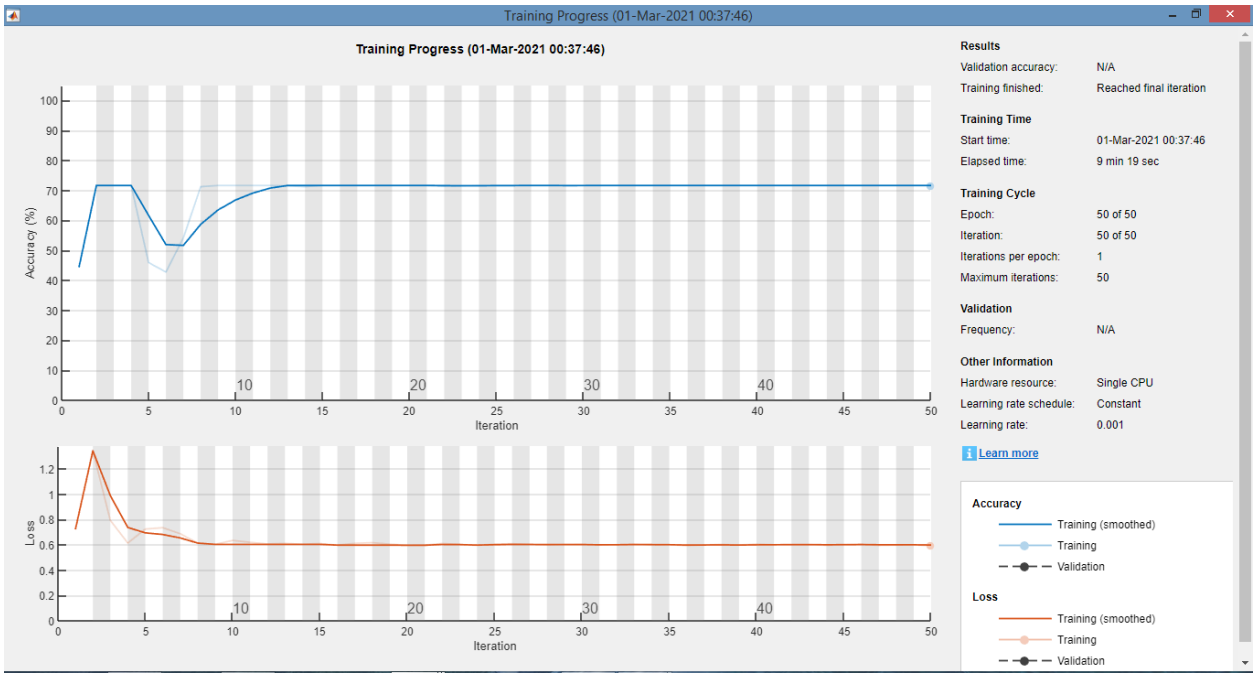

Training Progress (28-Feb-2021 20:42:39)
Training Progress (28-Feb-2021 20:42:39)
Accuracy (%)
Iteration
Loss
Iteration
Results
Validation accuracy: N/A
Training finished: Reached final iteration
Training Time
Start time: 28-Feb-2021 20:42:39
Elapsed time: 3 min 38 sec
Training Cycle
Epoch: 150 of 150
Iteration: 150 of 150
Iterations per epoch: 1
Maximum iterations: 150
Validation
Frequency: N/A
Other Information
Hardware resource: Single CPU
Learning rate schedule: Constant
Learning rate: 0.001
Learn more
Accuracy
Training (smoothed)
Training
Validation
Loss
Training (smoothed)
Training
Validation

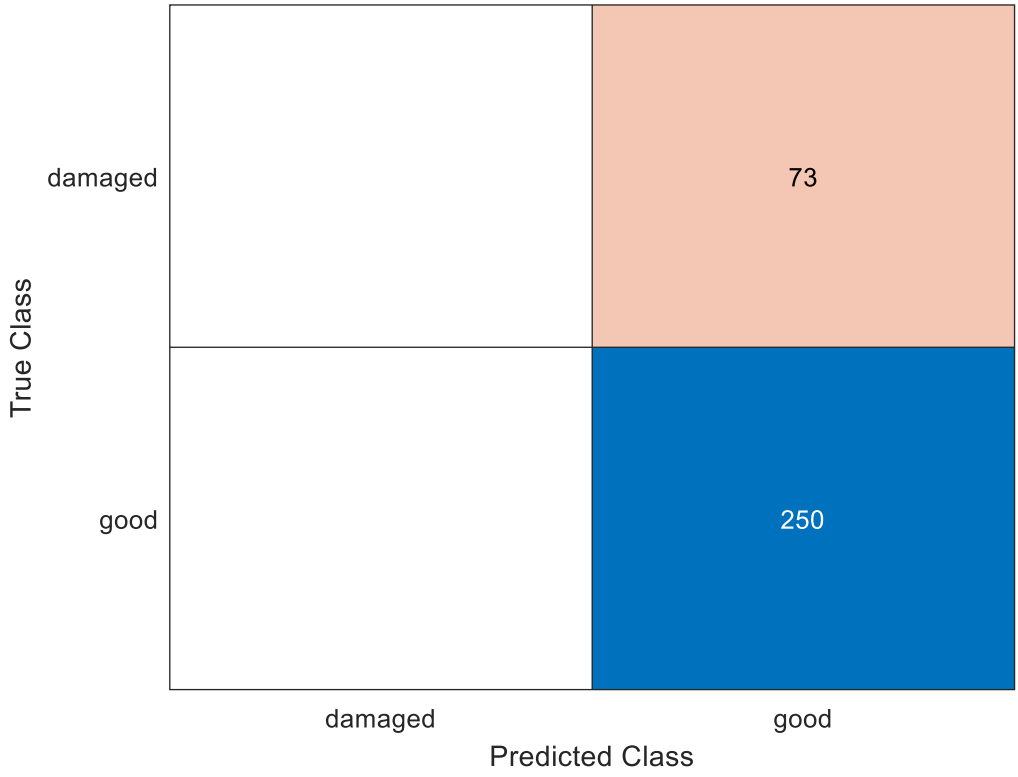

Training Progress (28-Feb-2021 23:48:41)
Training Progress (28-Feb-2021 23:48:41)
Accuracy (%)
Iteration
Loss
Iteration
Results
Validation accuracy: N/A
Training finished: Reached final iteration
Training Time
Start time: 28-Feb-2021 23:48:41
Elapsed time: 27 min 44 sec
Training Cycle
Epoch: 150 of 150
Iteration: 150 of 150
Iterations per epoch: 1
Maximum iterations: 150
Validation
Frequency: N/A
Other Information
Hardware resource: Single CPU
Learning rate schedule: Constant
Learning rate: 0.001
Learn more
Accuracy
Training (smoothed)
Training
Validation
Loss
Training (smoothed)
Training
Validation



Training LSTM with lstm_simulated_data6/lstm_simulated_classes6

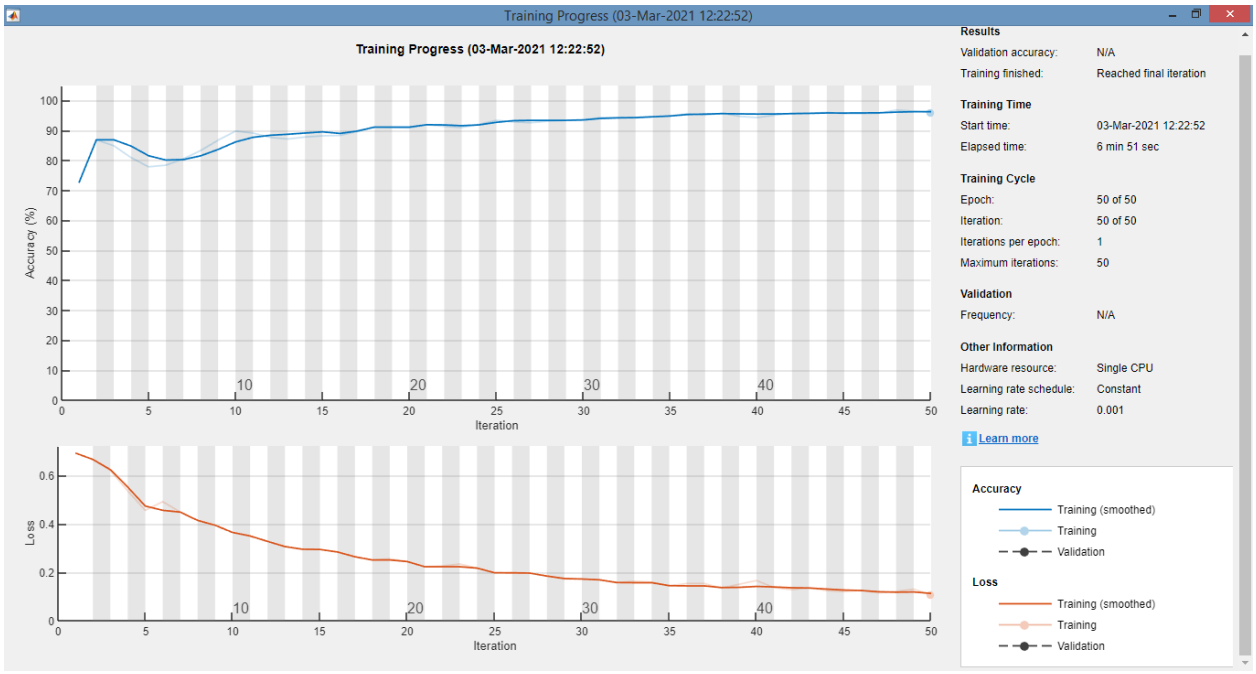
accuracy = 74% Epoch = 150

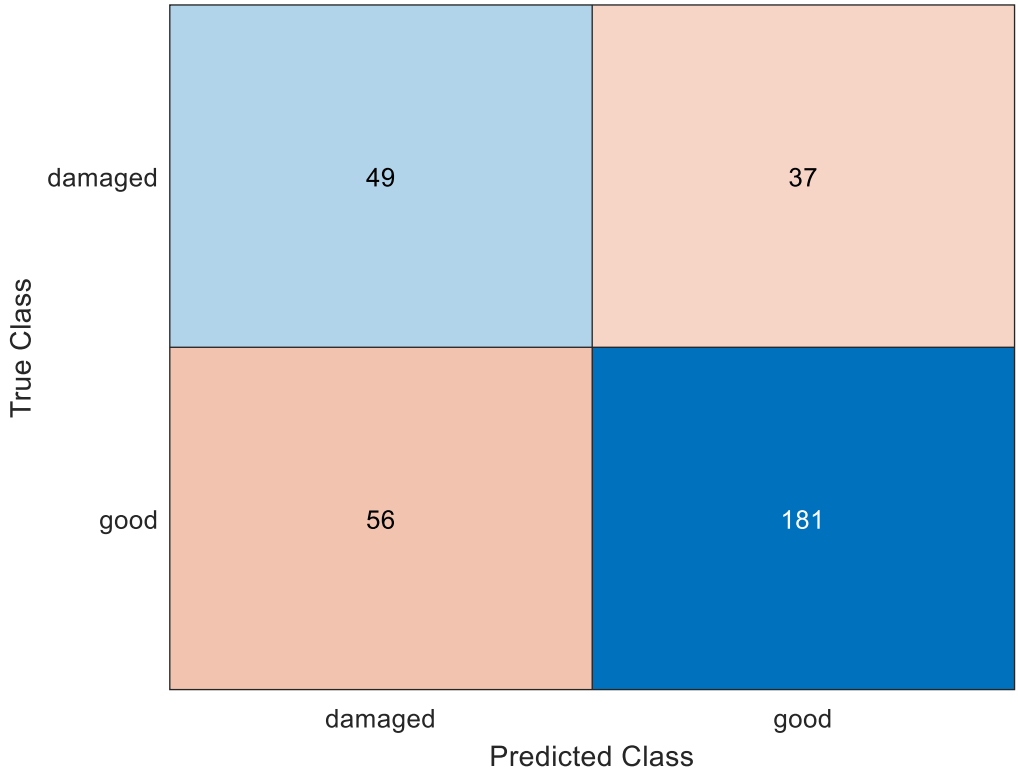




Training LSTM with lstm_simulated_data6/lstm_simulated_classes6

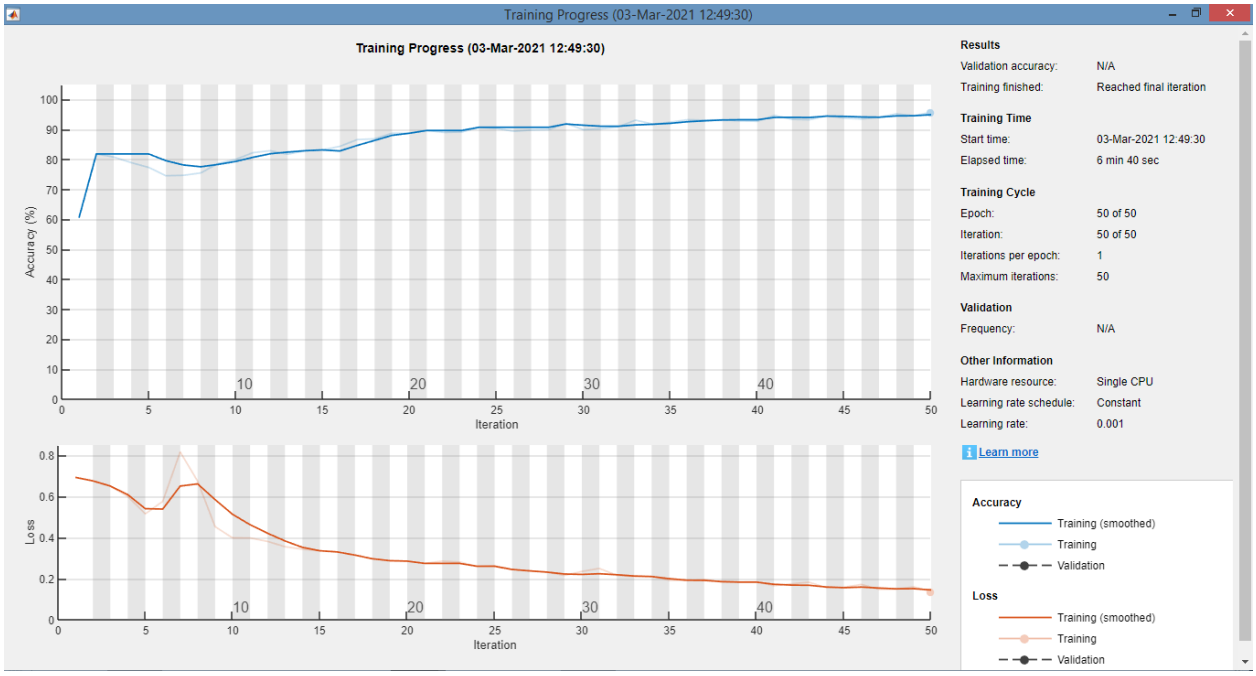
accuracy = 77% Epoch = 50

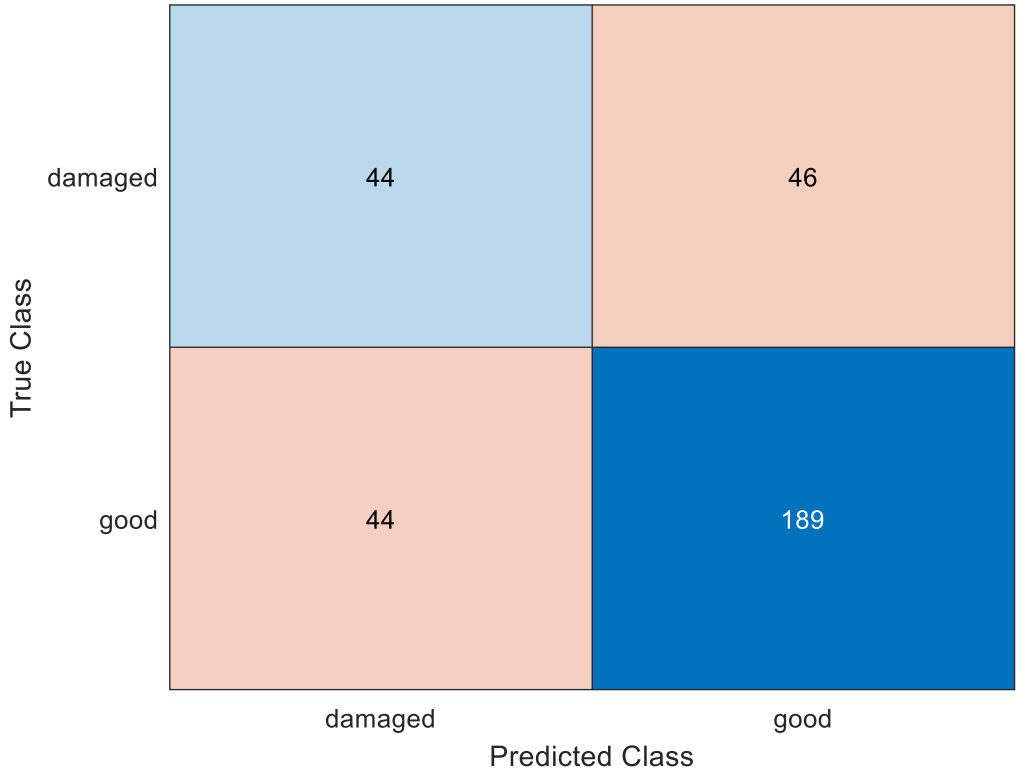




Training LSTM with lstm_simulayed_data6/lstm_simulated_classes6

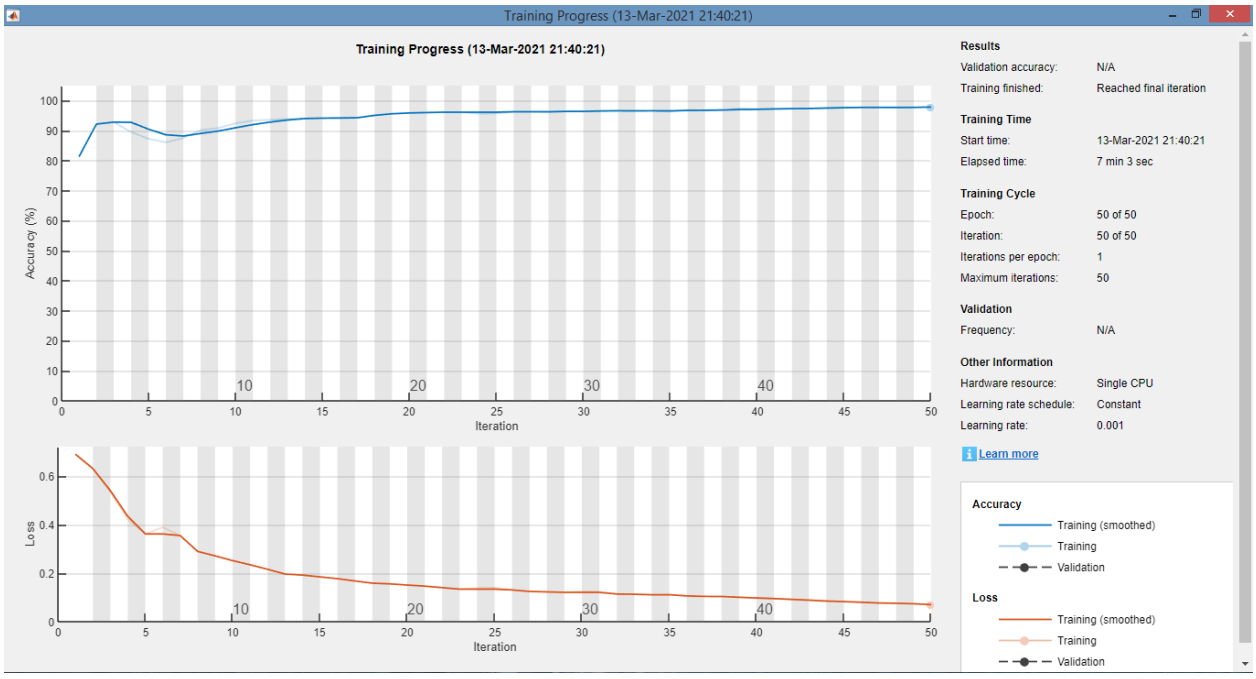
accuracy = 71% Epoch = 50 (but subtracted the radius from each point in lstm_simulated_data6 and squared them – normalisation)

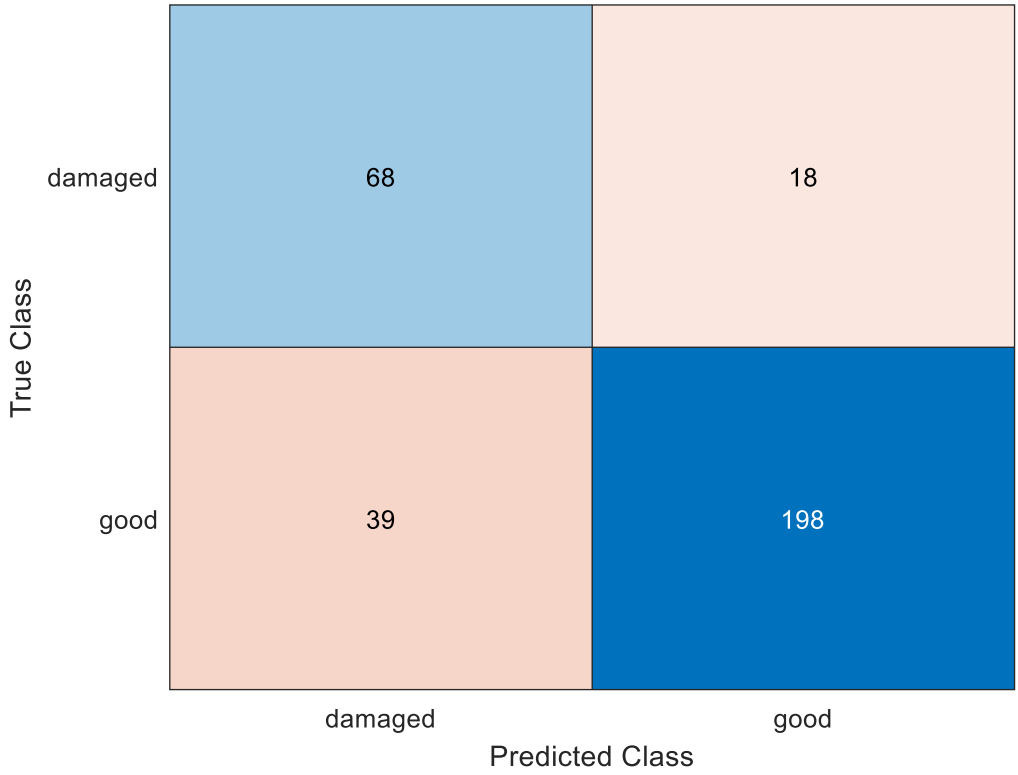




Training LSTM with lstm_simulayed_data6/lstm_simulated_classes6

accuracy = 72% Epoch = 50 (but subtracted the radius from each point in lstm_simulayed_data6 and cubed)



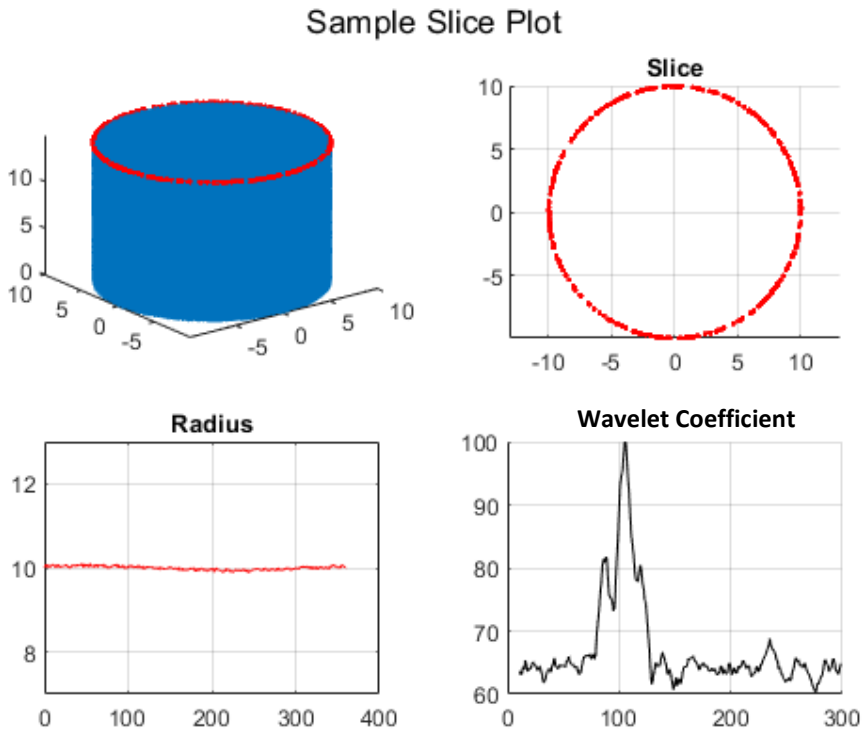


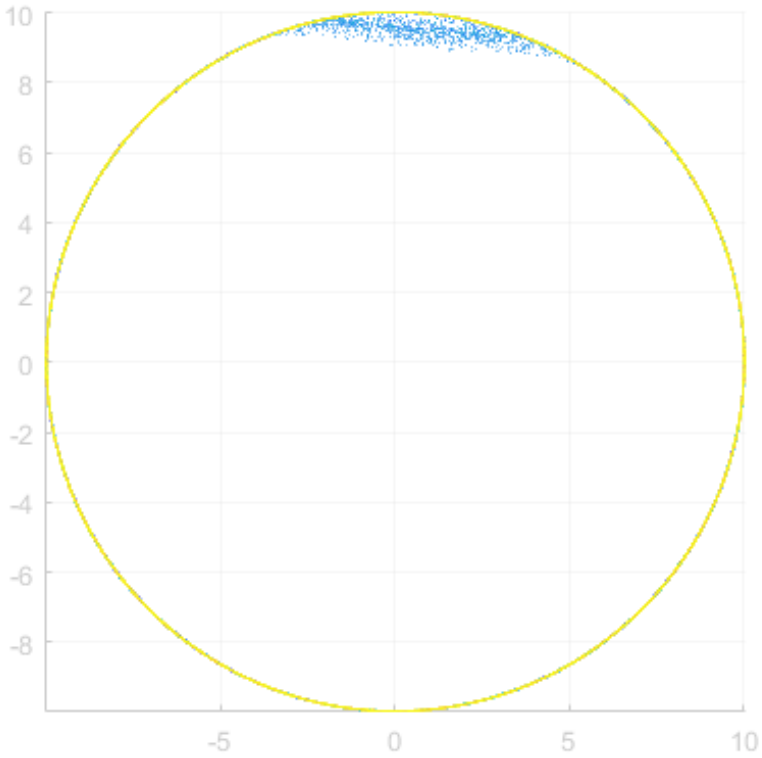
Training LSTM with lstm_simulayed_data6/lstm_simulated_classes6

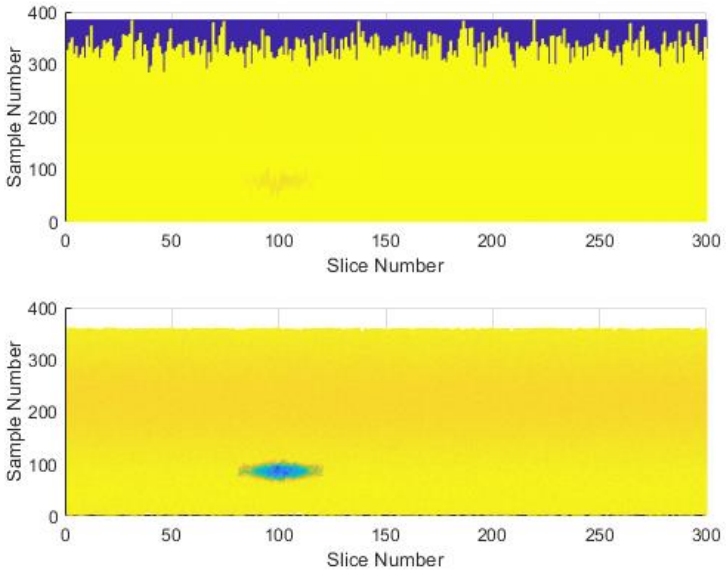
Accuracy = 72% Epoch = 50 (but subtracted the radius from each point in lstm_simulayed_data6)

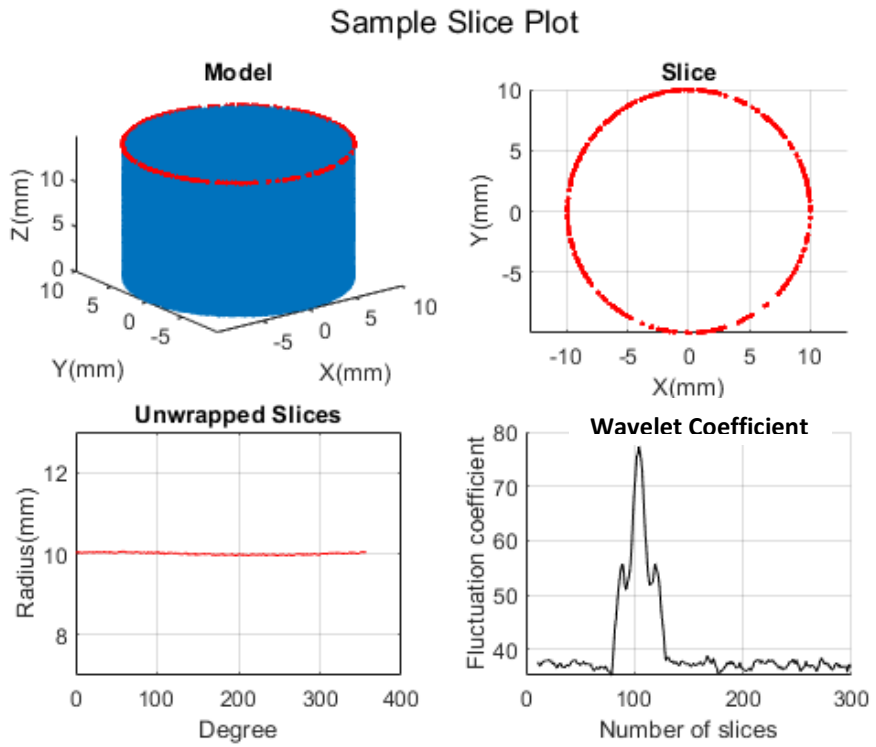
LSTM Training of Data Preprocessed Using Wavelet Transform

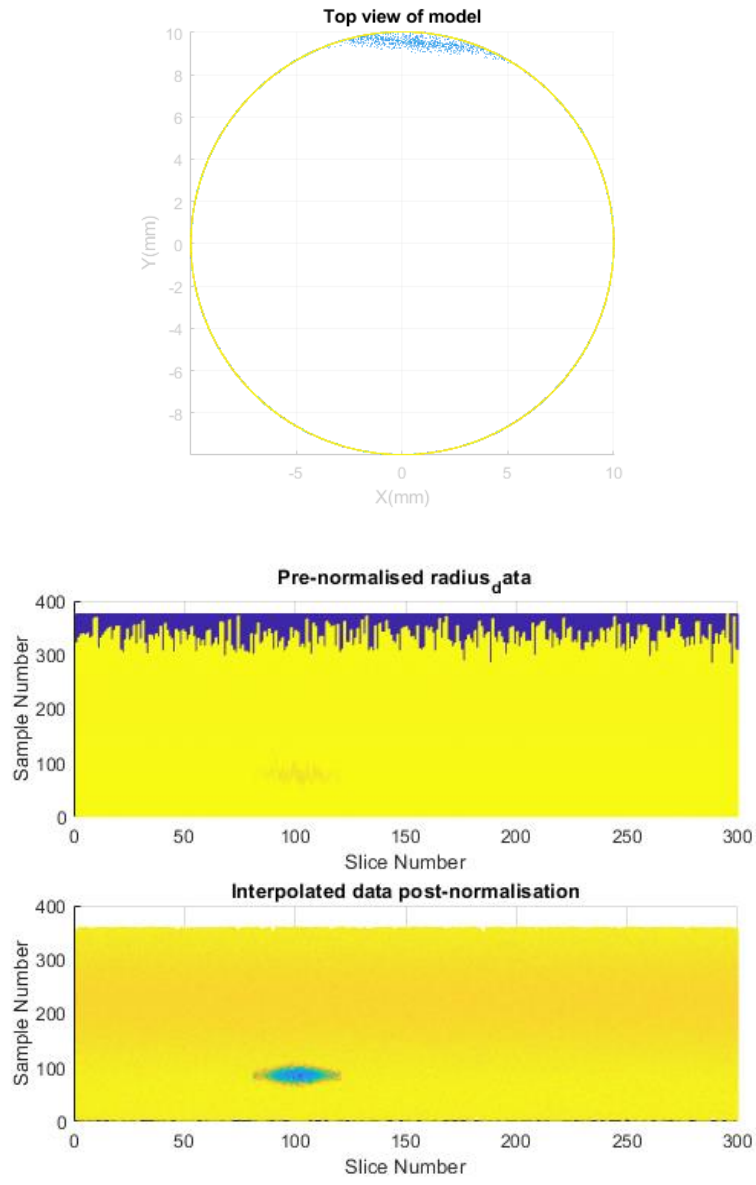
```
% Experiment Parameter Setup
cylinder_radius = 10;
cylinder_height = 15;
PointDistributeChaos = 0.1;
PointDistributeResolution = 1e-5;
dmg_start = 50; % this is the start angle
dmg_height = 5; % height of the centre of the damage
% dmg_start = randi([20,180],testNO,1); % this is the start angle
% dmg_height = randi([0,cylinder_height],testNO,1); % height of the centre
of the damage
scanner_noise = rand(testNO,1)*0.1; % Gaussian noise for the scanner, range
is from 0:0.01(to change the range, change 0.1)
```


Sample Slice Plot
Slice
Radius
Wavelet Coefficient



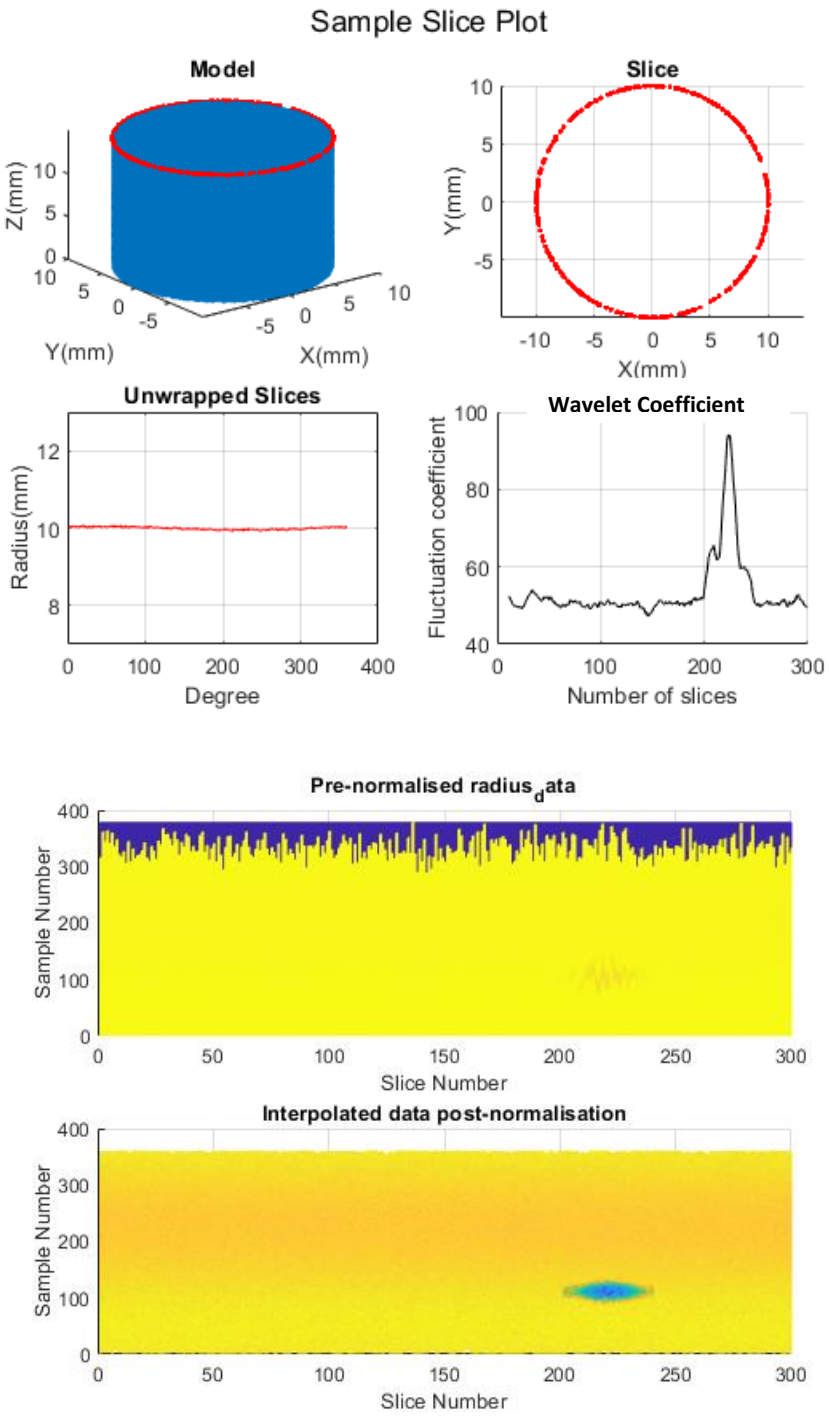

Sample Number
Slice Number
Sample Number
Slice Number

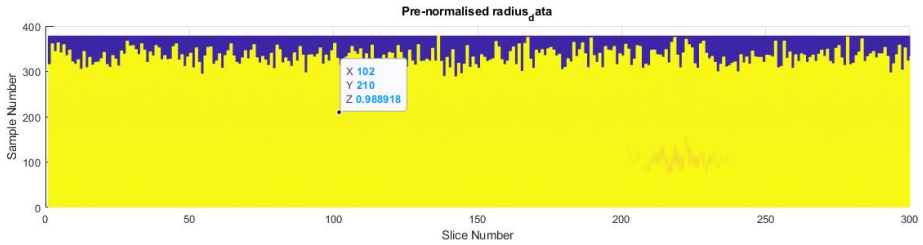

Sample Slice Plot
Model
Z(mm)
Y(mm)
X(mm)
Slice
Y(mm)
X(mm)
Unwrapped Slices
Radius(mm)
Degree
Wavelet Coefficient
Fluctuation coefficient
Number of slices

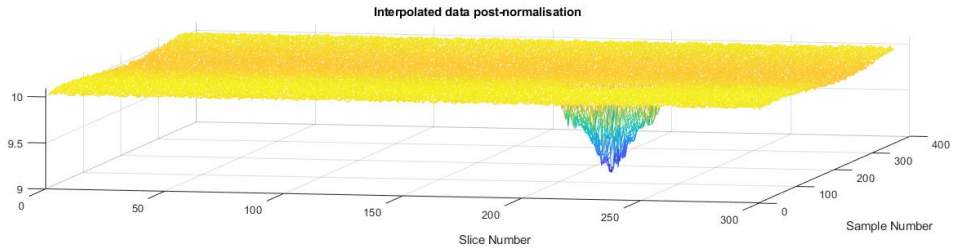


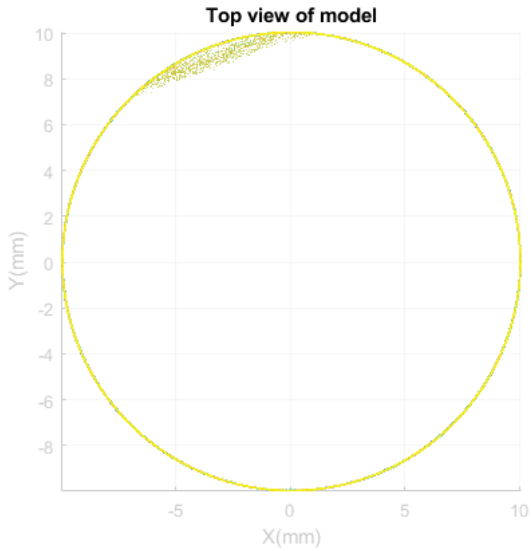
The top is the radius data before performing normalisation and dealing with the unequal values of the slices, while the second plot is the normalised and interpolated data.

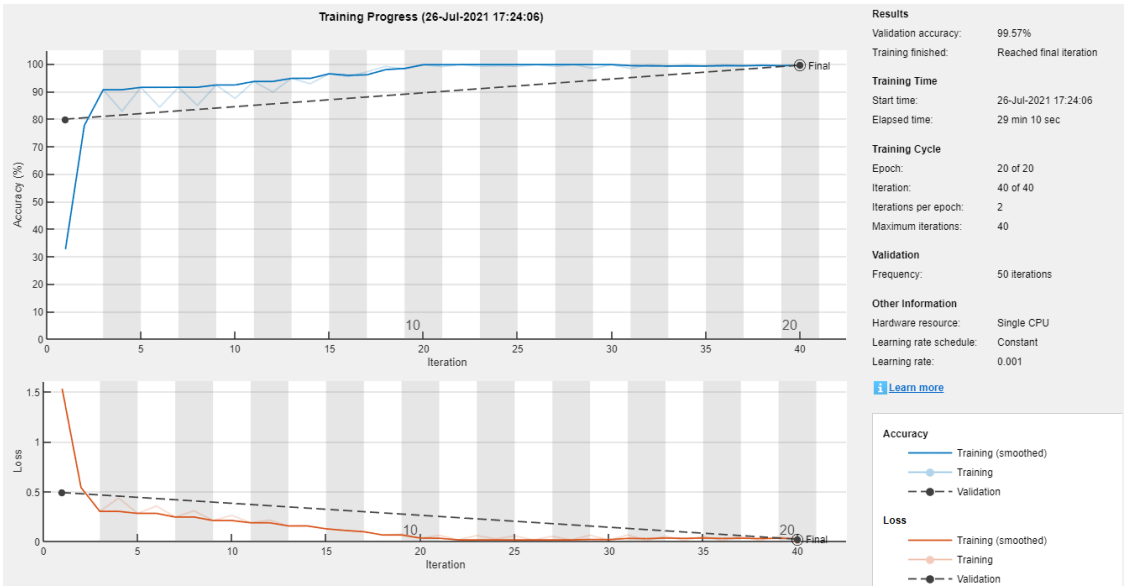
```
% Experiment Parameter Setup
cylinder_radius = 10;
cylinder_height = 15;
PointDistributeChaos = 0.1;
PointDistributeResolution = 1e-5;
dmg_start = 75; % this is the start angle
dmg_height = 11; % height of the centre of the damage
% dmg_start = randi([20,180],testNO,1); % this is the start angle
% dmg_height = randi([0,cylinder_height],testNO,1); % height of the centre
of the damage
scanner_noise = rand(testNO,1)*0.1; % Gaussian noise for the scanner, range
is from 0:0.01(to change the range, change 0.1)
```


Sample Slice Plot
Model
Z(mm)
Y(mm)
X(mm)
Slice
Y(mm)
X(mm)
Unwrapped Slices
Radius(mm)
Degree
Wavelet Coefficient
Fluctuation coefficient
Number of slices
Pre-normalised radius data
Sample Number
Slice Number
Interpolated data post-normalisation
Sample Number
Slice Number



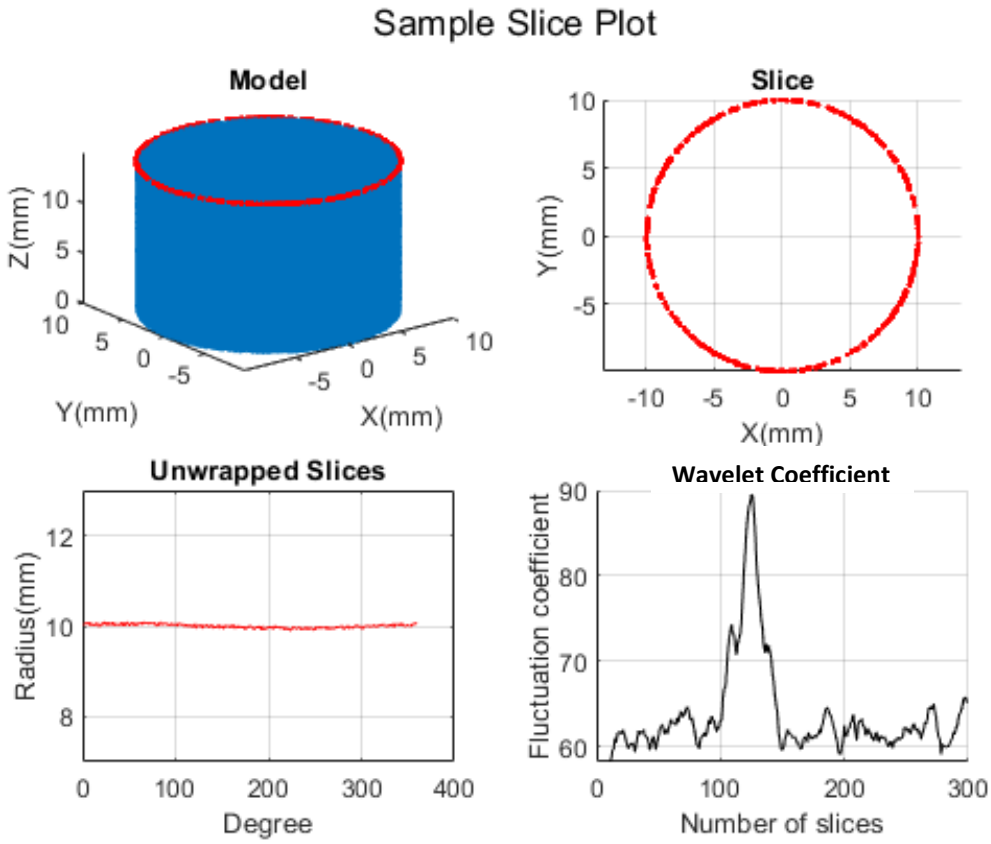


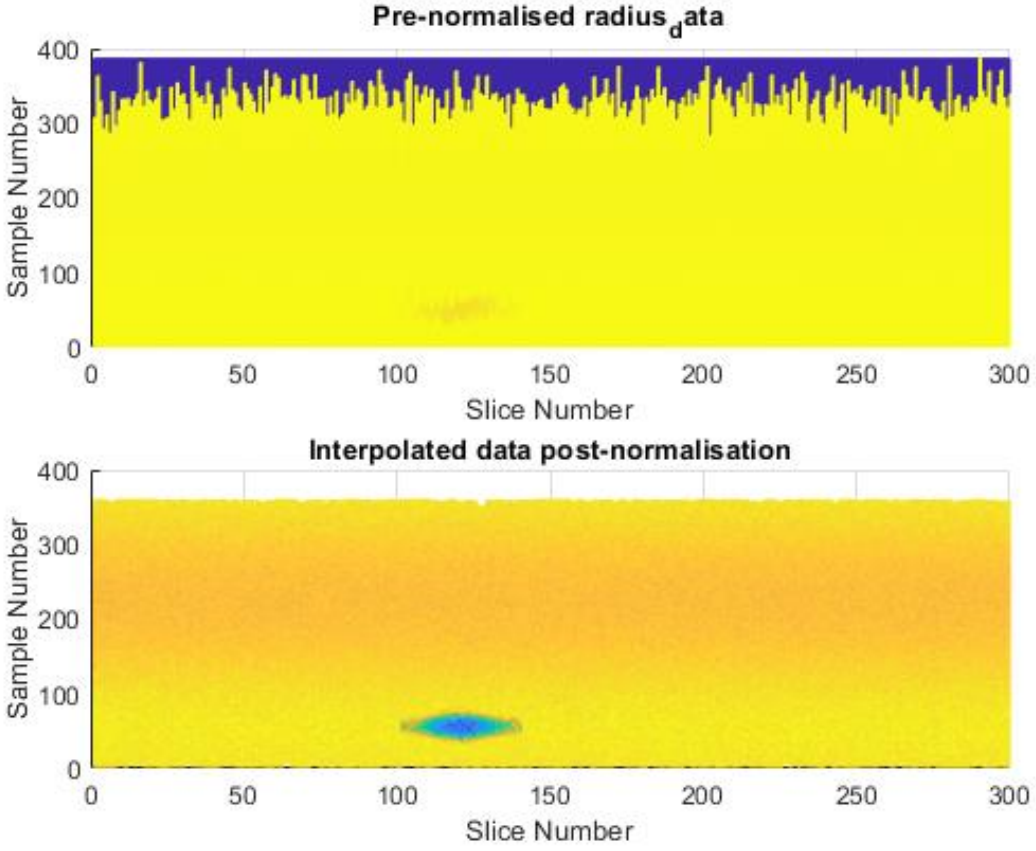


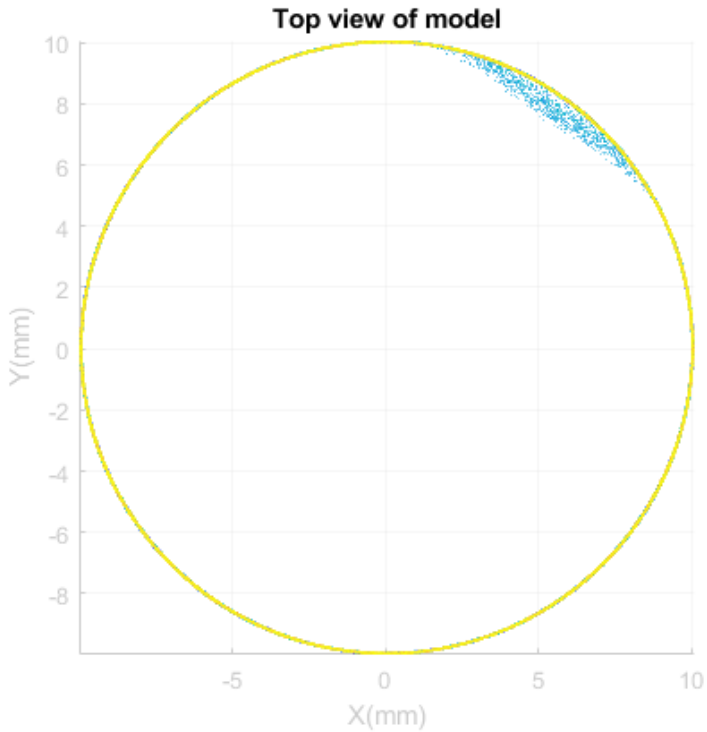


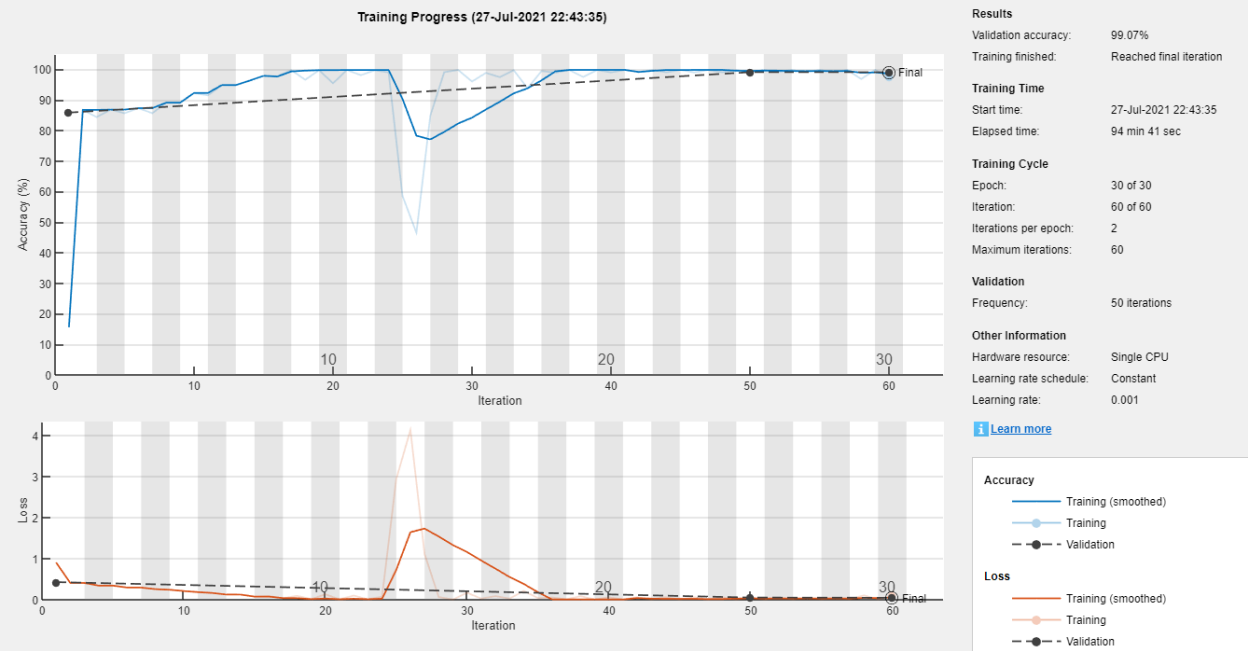

```
% Experiment Parameter Setup
cylinder_radius = 10;
```

```
cylinder_height = 15;
PointDistributeChaos = 0.1;
PointDistributeResolution = 1e-5;
dmg_start = 20; % this is the start angle
dmg_height = 6; % height of the centre of the damage
scanner_noise = rand(testNO,1)*0.1; % Gaussian noise for the scanner, range
                is from 0:0.01(to change the range, change 0.1)
```

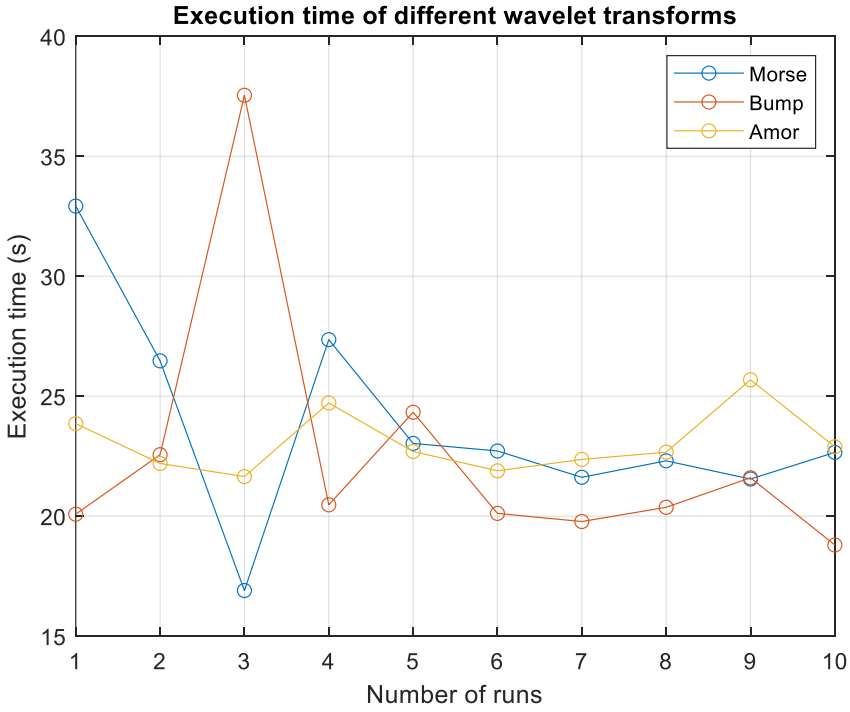


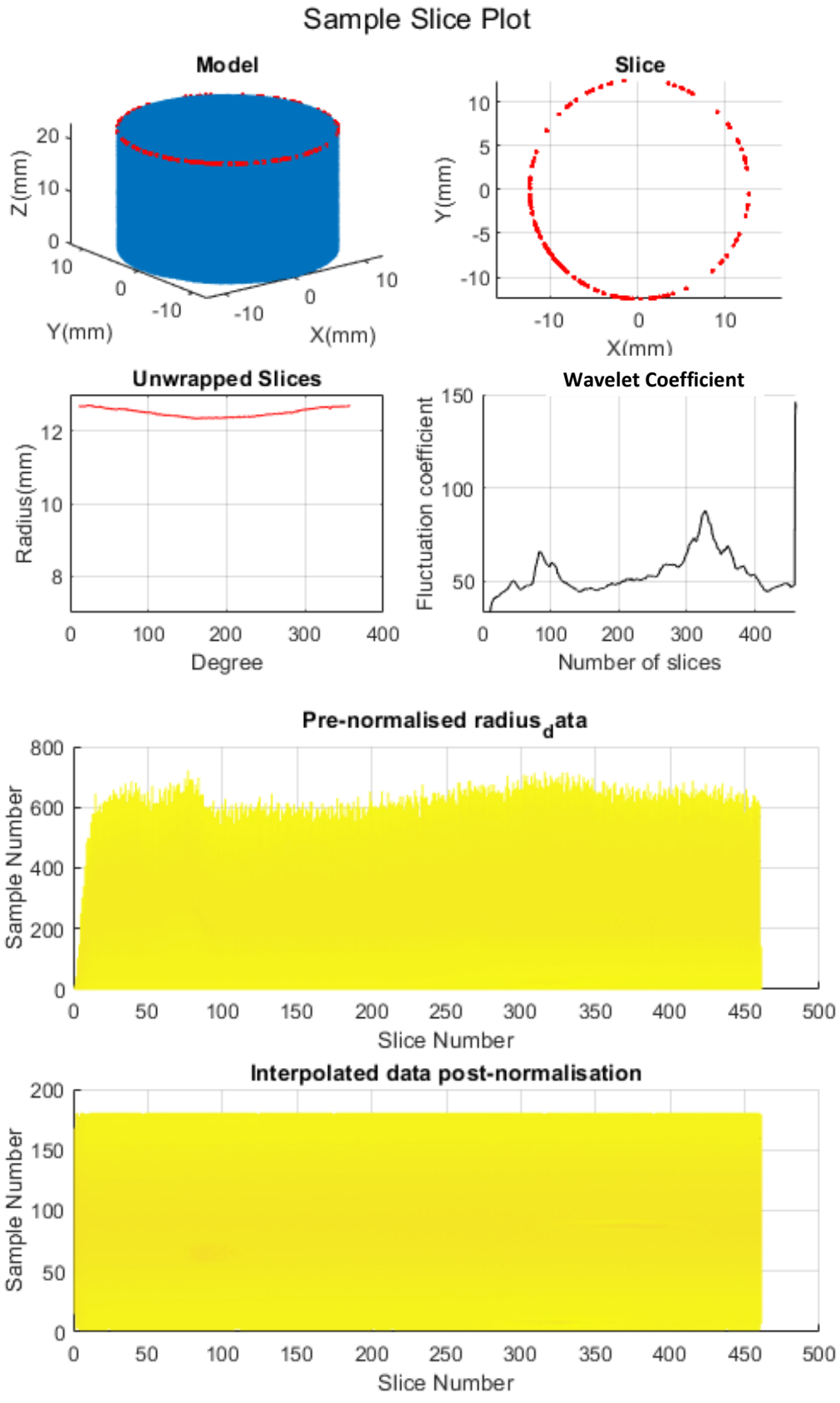

Pre-normalised radius data
Sample Number
Slice Number
0
100
200
300
400
0
50
100
150
200
250
300
Interpolated data post-normalisation
Sample Number
Slice Number

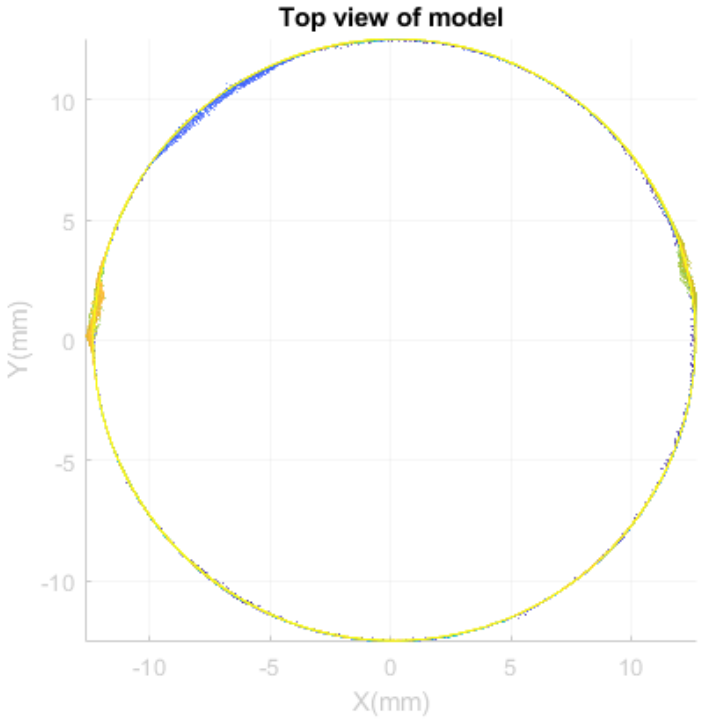

Top view of model
Y(mm)
X(mm)
10
8
6
4
2
0
-2
-4
-6
-8
-5
0
5
10

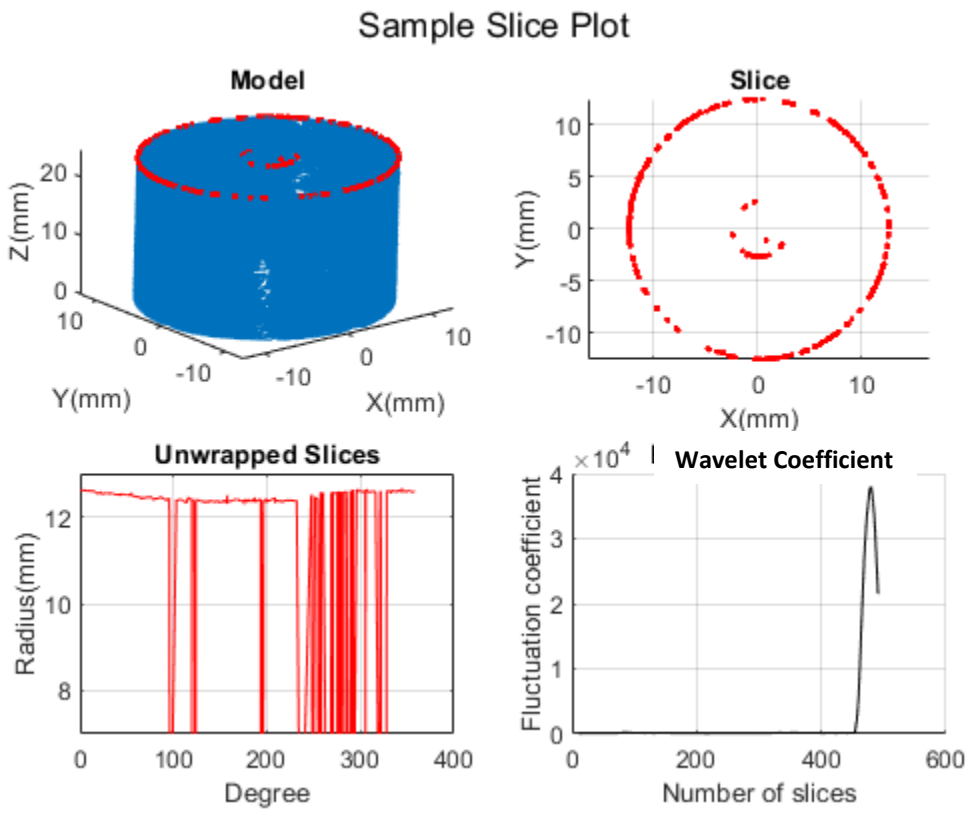


The final coefficeient plot produces the same response for all methods, but the morse ran faster than amor and bump at some point on the same set of data. All methods became comparable to within 5 seconds on the later runs, as shown below. All runs were performed on a randomly generated damage with varying damage sizes and locations.

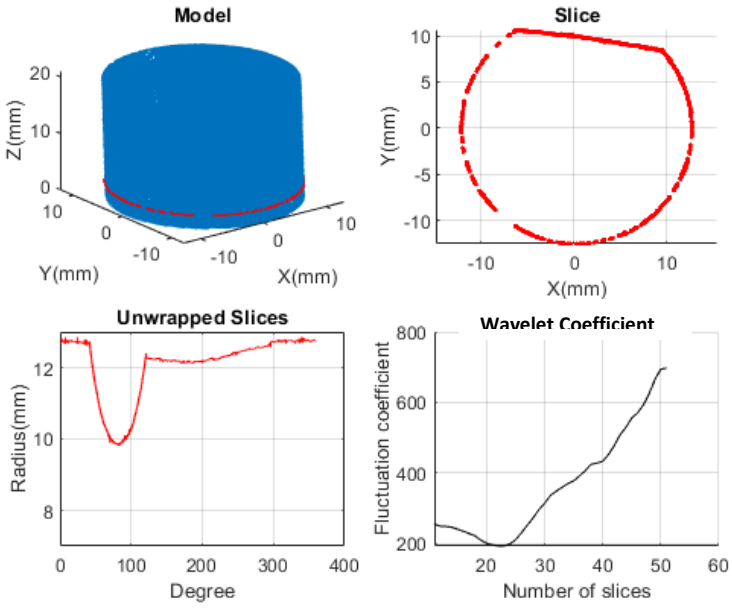


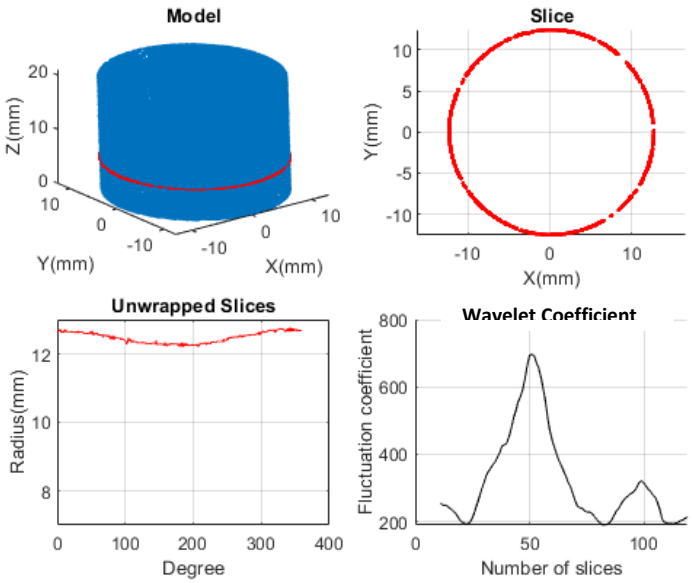

Sample Slice Plot
Model
Z(mm)
Y(mm)
X(mm)
Slice
Y(mm)
X(mm)
Unwrapped Slices
Radius(mm)
Degree
Wavelet Coefficient
Fluctuation coefficient
Number of slices
Pre-normalised radius data
Sample Number
Slice Number
Interpolated data post-normalisation
Sample Number
Slice Number



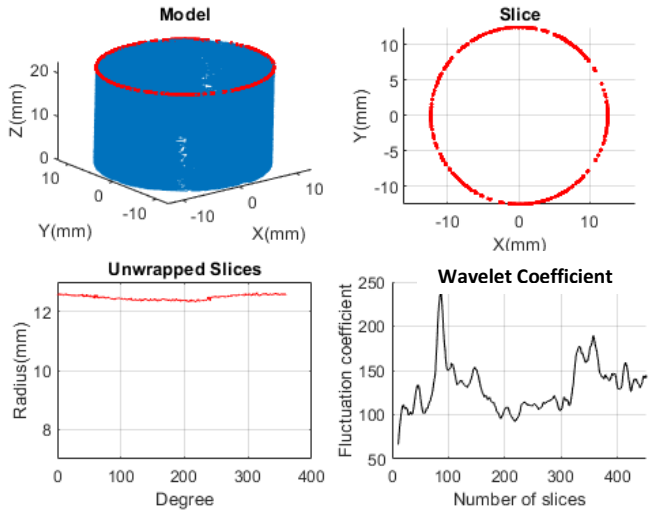


Effect of outlier points in the data



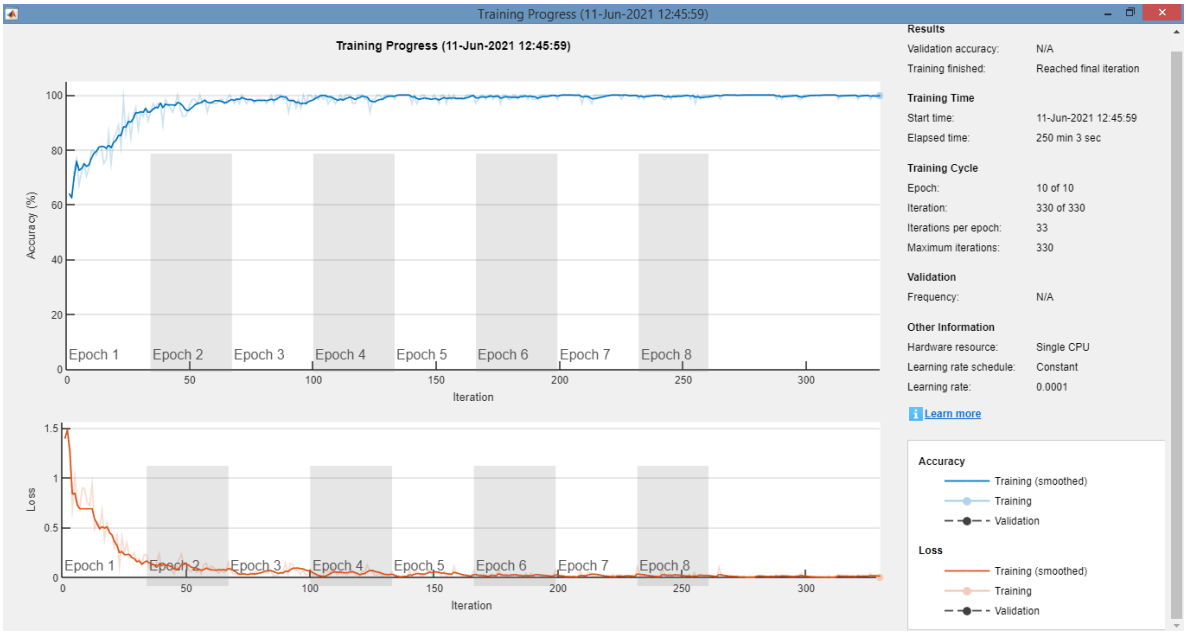


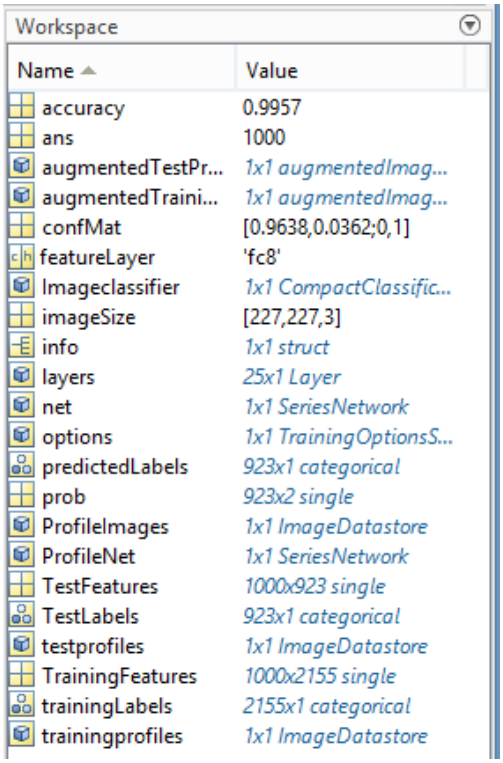
Sample Slice Plot



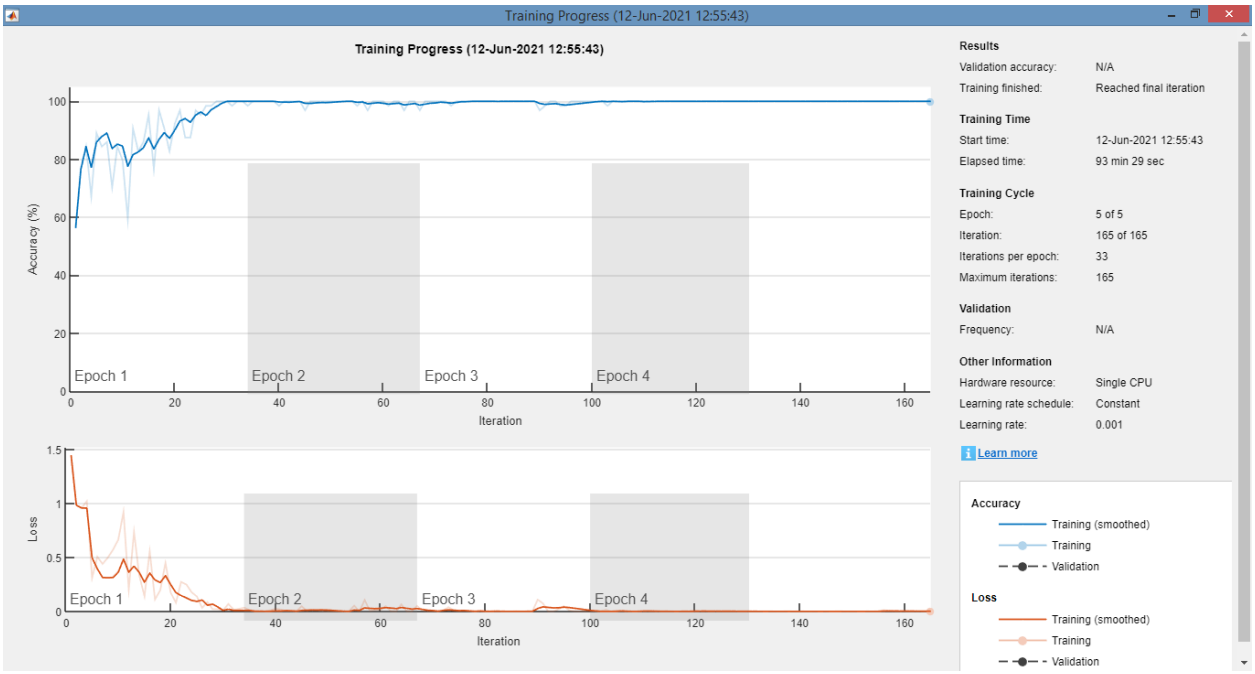
Appendix I CNN Training with the Slices Converted to Images Using a Power Spectrum Algorithm

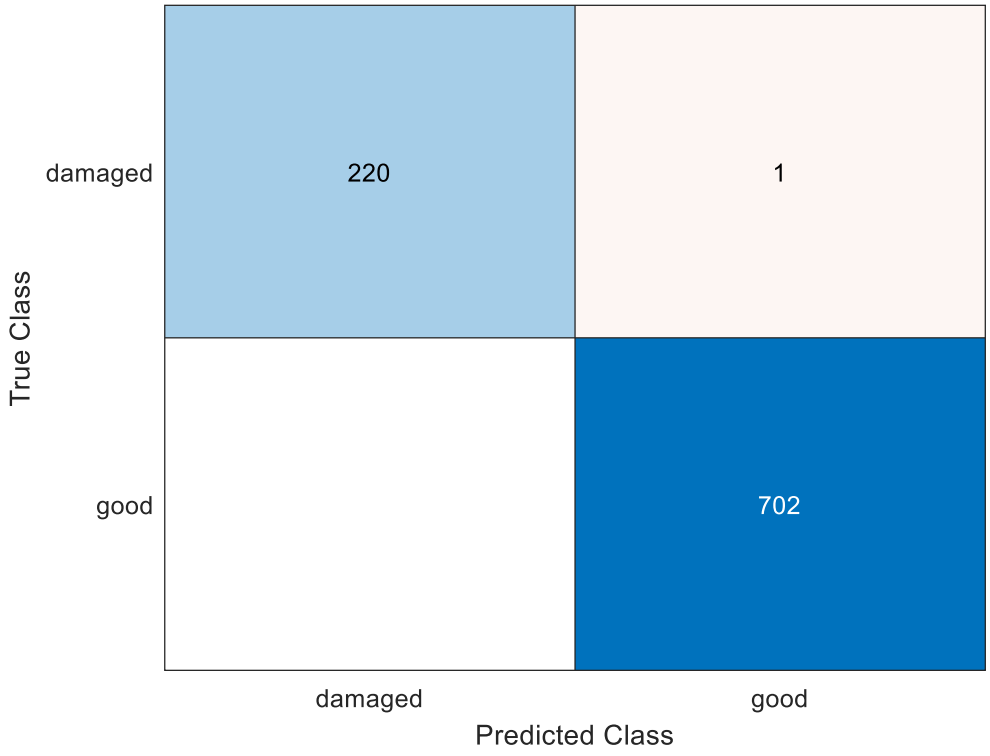
MaxEpochs 10





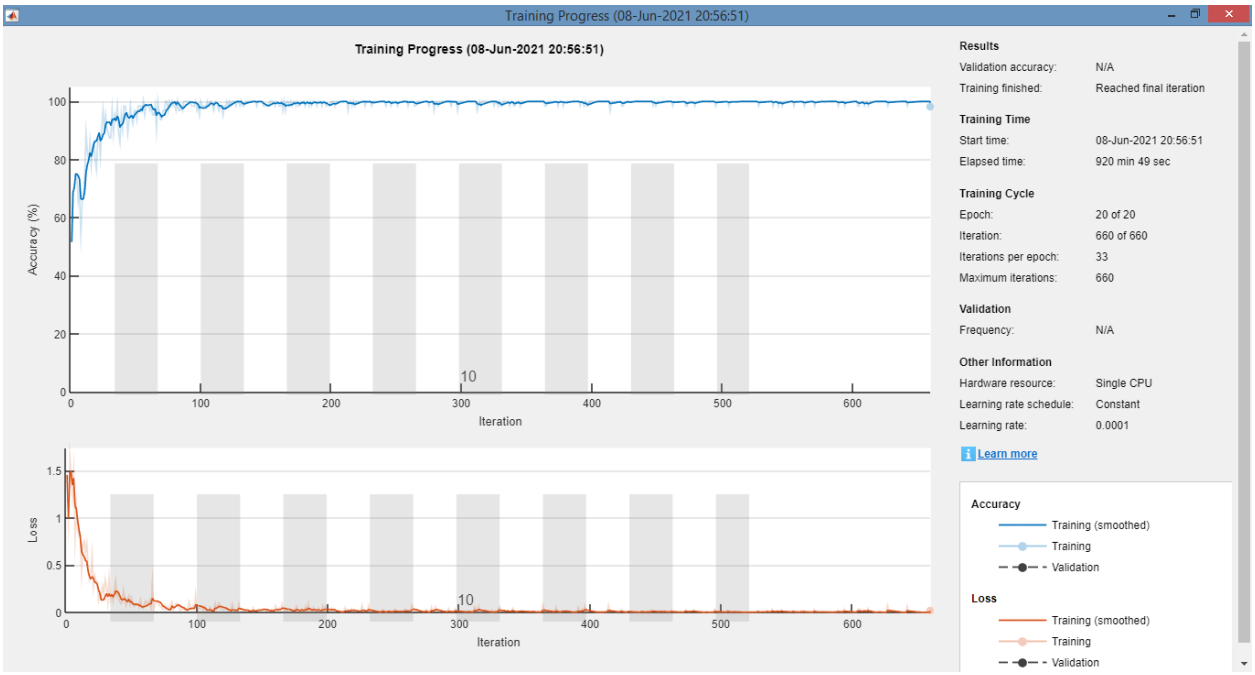
MaxEpoch 5

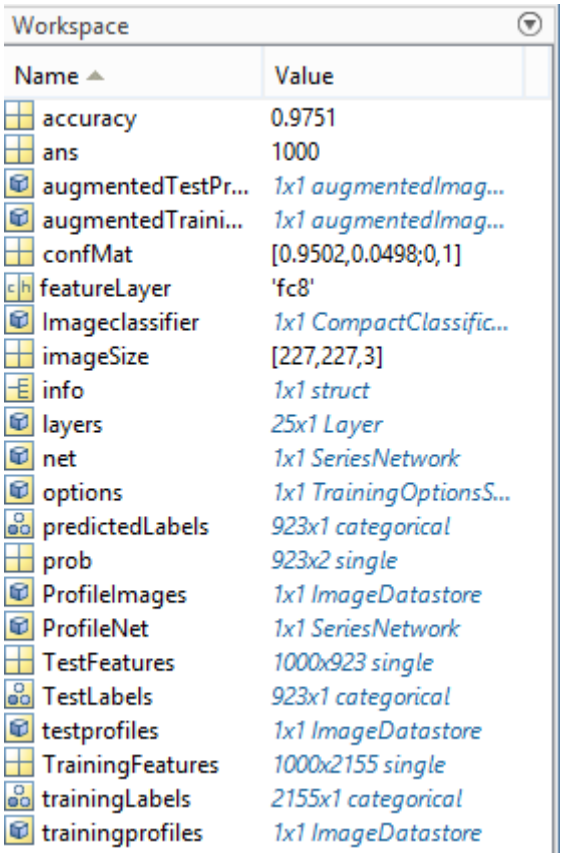




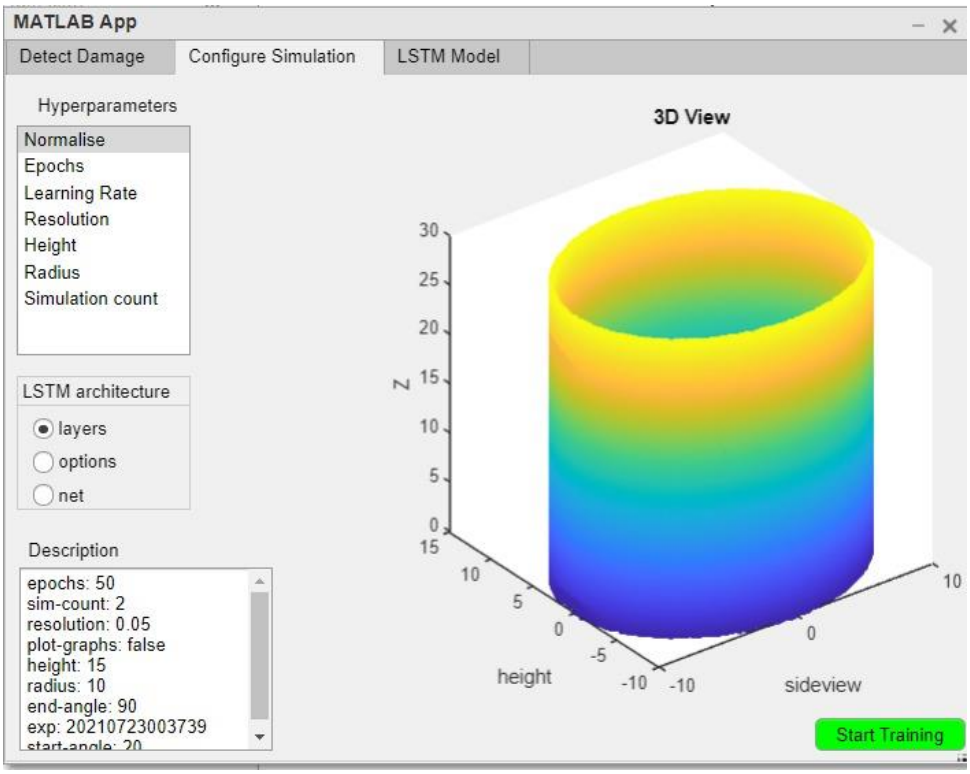
accuracy = 0.9910

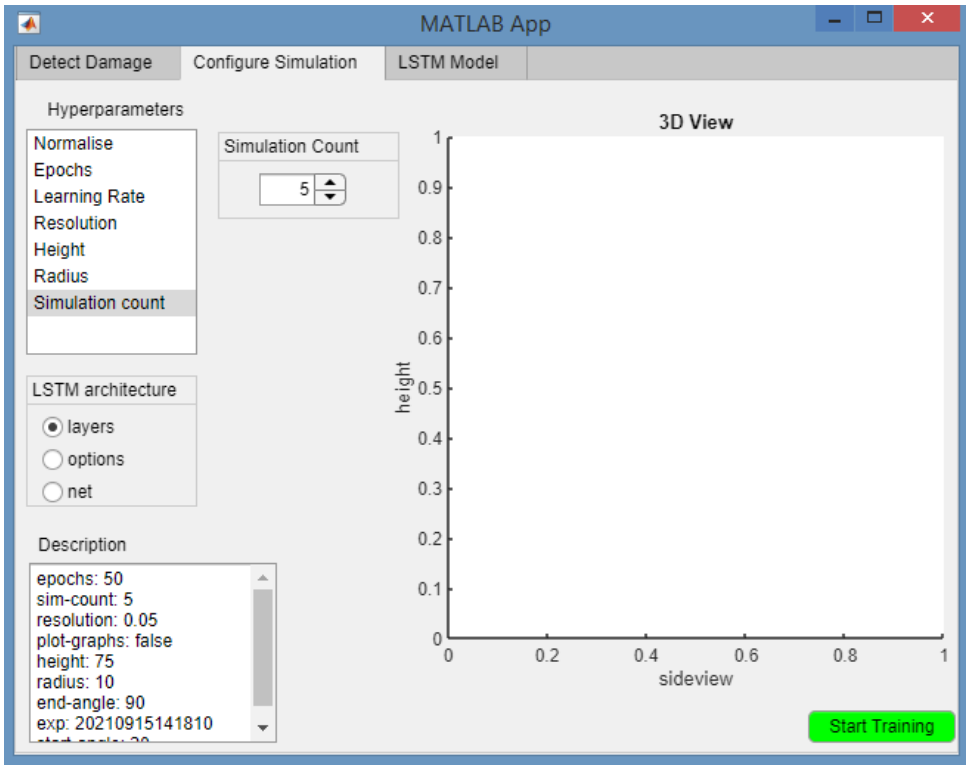
MaxEpochs 20

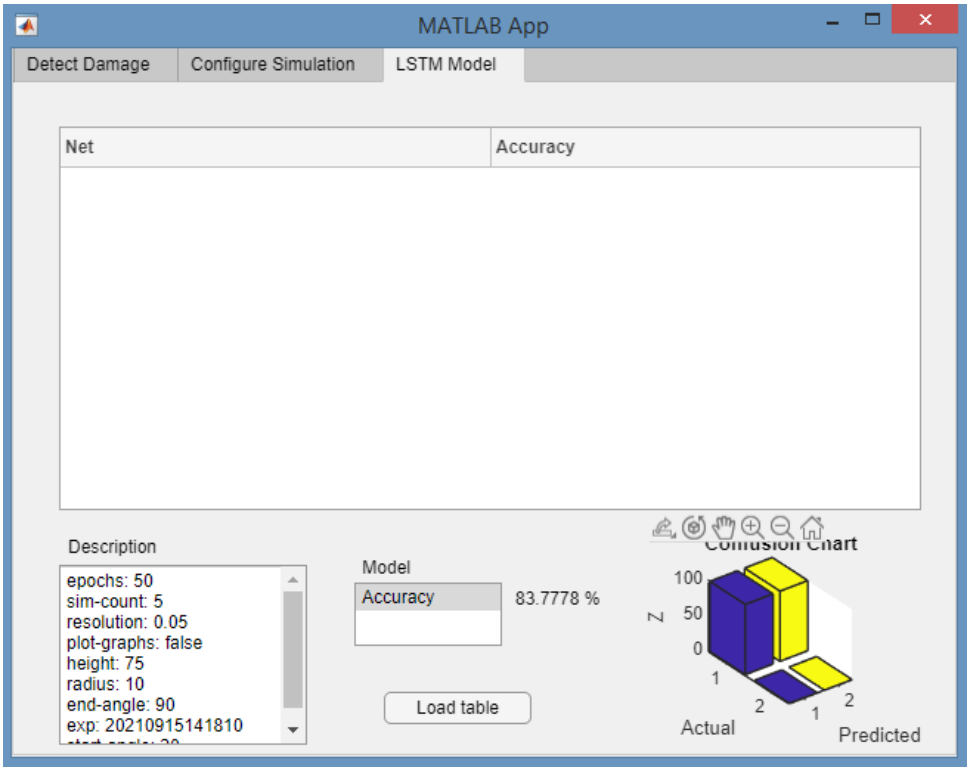


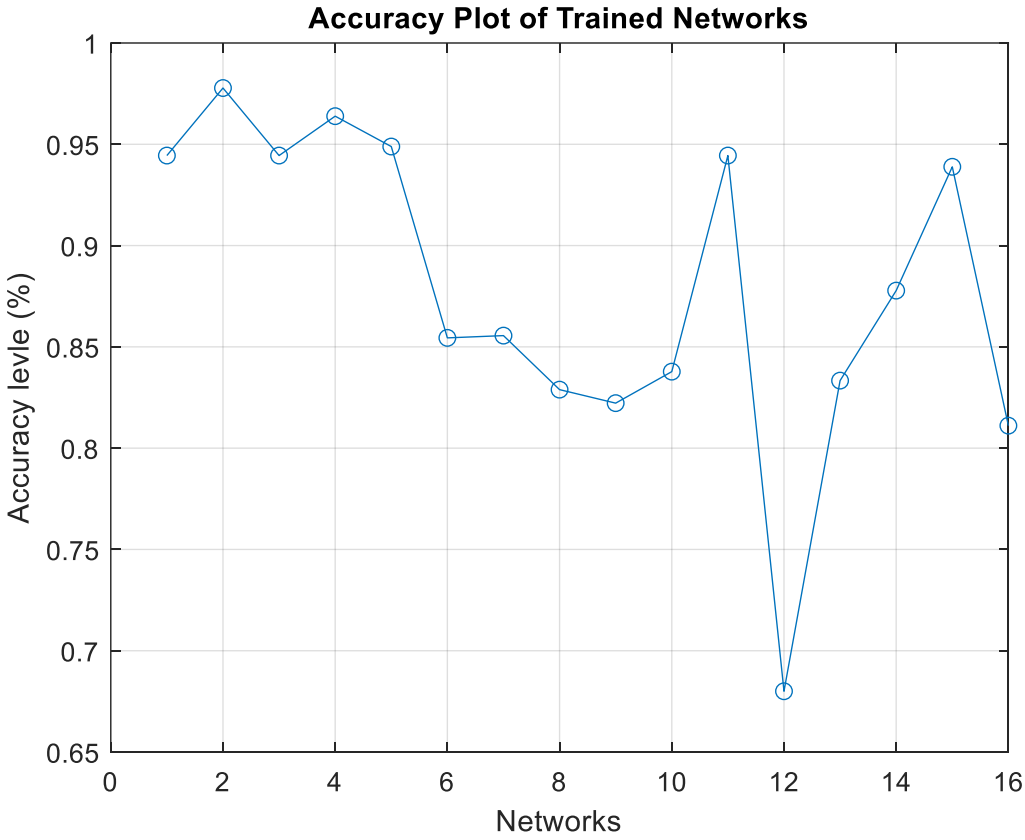


Appendix J Graphical User Interface for Performing LSTM Training









Monitoring accuracy of 10 trained networks