A Semi-Autonomous Robotic-Based Repair System for Revolving Mechanical Components via Laser Metal Deposition Process

by

Hamdan Mohammed Saqqaf Al-Musaibeli

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Mechanical Engineering

University of Alberta

Abstract

Natural resources are finite, and the demand for such resources is increasing. Thus, an urgent need for systems to support the reusability and lifespan extendibility of products, such as those with a high demand and manufacturing cost, is essential. Furthermore, cylindrical components are widely used across many industries, such as mining, oil and gas, and agriculture, due to their mechanical and geometrical unique characterizations. Practically, the repair, restoration, and remanufacturing processes of such mechanical components are currently being achieved manually, which require professional manpower and long processing time. Therefore, this research aims to develop a semi-autonomous repair and restoration robotic-based system for revolving mechanical components using laser metal deposition technology to eliminate the aforementioned limitations, thus supporting a sustainable society.

In this project, a framework is developed for a semi-autonomous repair process of end-of-life mechanical components using additive manufacturing technology. The designed system consists of three main stages: an intelligent autonomous recognition method for detecting defective areas on a component, an automatic localization and spatial measurement process of the recognized faulty areas, and an automatic generation of the robot paths for the laser metal deposition repair process.

In the first stage, a vision-based pipeline was adopted to recognize the damaged sections on the component's surface. A preliminary investigation was conducted to study the behavior and performance of the state-of-the-art model on the object detection task. This study leverages the transfer learning concept to build upon the knowledge gained on previous detection tasks to boost the learning process time and model convergence to the new detection task. Furthermore, the state-of-the-art was also trained on public metal defects

datasets for further study of the model on different metal defects datasets. After the deep study, one dataset is selected to be used as a base for detecting defects on metal surfaces.

The second part of this framework is the localization and topography data acquisition step. A novel localization method is developed based on image-spatial mapping of cylindrical components. The method uses a vision sensor to locate bounding points of the damage in 3D space for the robot to acquire the point cloud scan using a proximity sensor. This method covers process planning, image mapping, and spatial mapping.

In the last stage, robot paths for the repair process of the damaged surface are then generated automatically to the nominal dimension of the nearby areas. Further, the build target is adjustable in the developed method to support the manufacturability of new features. Moreover, this method is capable of repairing the internal and external surfaces of a component. With this, a virtual repair process using the generated tool-paths is then produced to verify the repair process using an online/offline platform. Lastly, an actual repair process is carried out using the laser metal deposition process.

Preface

This thesis follows a publication-based format provided by the FGSR at the University of Alberta. Thus, it consists of three journal papers. One is published and the other two are submitted. The thesis outline follows the following papers' order.

1. Al-Musaibeli, H., Ahmad, R. A defect detection method based on YOLOv7 for automated remanufacturing via laser cladding process. Journal of remanufacturing. **(Submitted)**
2. Al-Musaibeli, H., Ahmad, R. Towards an Autonomous Robot-Based Laser Cladding Repair System: an Automatic Defect Detection, Localization, and 3D Scanning Method using a Camera and Proximity Sensor. Robotics and Computer-Integrated Manufacturing. **(Submitted)**
3. Al-Musaibeli, H., Ahmad, R. A path planning method for surface damage repair using a robot-assisted laser cladding process. Int J Adv Manuf Technol 122, 1259–1279 (2022). https://doi.org/10.1007/s00170-022-09933-3

"إِنَّ اللَّهَ عَزَّ وَجَلَّ كَتَبَ الإِحْسَانَ عَلَى كُلِّ شَيْءٍ"

"Allah has prescribed proficiency in all things"

Prophet Mohammed (PBUH)

Acknowledgments

I would like to take this opportunity to express my sincere gratitude and thanks to my supervisor Dr. Rafiq Ahmad for his endless guidance, support, and oversight. Thank you for this opportunity to work on this exciting project with a fantastic team.

In addition, I would like to thank our industrial partner, Group Six Technologies Inc., for their continuous support and collaboration, to name Matt, Steven, and Craig. Their welcome to use their experimental setup is much appreciated.

I also would like to thank all my colleagues in the SMART LAB (formerly LIMDA) and the office for their endless support throughout my graduate studies.

Finally, thank my mother, brothers, and sisters for your love, prayers, and encouraging words. You have always been there for me.

Table of Contents

List of Tables

List of Figures

List of Algorithms

Abbreviations

CE	Circular Economy
LE	Linear Economy
EoL	End-of-Life
AM	Additive Manufacturing
LMD	Laser Metal Deposition
LC	Laser Cladding
DED	Direct Energy Deposition
DLD	Direct Laser Deposition
DMD	Direct Metal Deposition
AI	Artificial Intelligence
3D	Three dimensional space
PCD	Point cloud data
CNNs	Convolutional neural networks
CASAE	Cascaded autoencoder
FCNN	Fully connected neural network
Mask RCNN	Mask region-based CNN
YOLO	You Only Look Once
mAP	mean Average Precision
NRSD	No-service rail surface defect
KSDD	Kolektor Surface-Defect Dataset
NEU-DET	Northeastern University Defect Dataset
BSD	Ball Screw Drive Surface Defect Dataset
RSDD	Rail surface discrete defects

IoU	Intersection over union
ICP	Iterative closest point
ToF	Time-of-flight
DoF	Degree of freedom
TCP	Tool center point
FoV	Field of view
FPS	Frame per second
ELAN	Extended Efficient Layer Aggregation Network
BoF	Bag of freebies
k-d tree	k-dimensional tree
LMR	Linear mapping ratio
CAD	Computer Aided Design

Chapter 1
Introduction

1.1 Background

Circular economy (CE) is an economic model focusing on minimizing waste by optimizing the value of products, materials, and resources to tackle environmental degradation and resource shortage problems [1]. CE is adopted to overcome the existing challenges with the traditional linear economy (LE) in terms of a sustainable model for the cycle life of a product. Furthermore, the LE is based on "take-make-use-dispose", where the life cycle is open, and products are discarded at the end of their usable life as in **Fig. *1.1***.. Unlike LE, CE closes this chain by means of the 3R's pillars, namely reduce, reuse, recycle of the end-of-life (EoL) products [2]. Therefore, the CE surpasses the LE in various aspects, including product life extension, thus a decrease in the consumption of natural resources that goes towards sustainability [3].

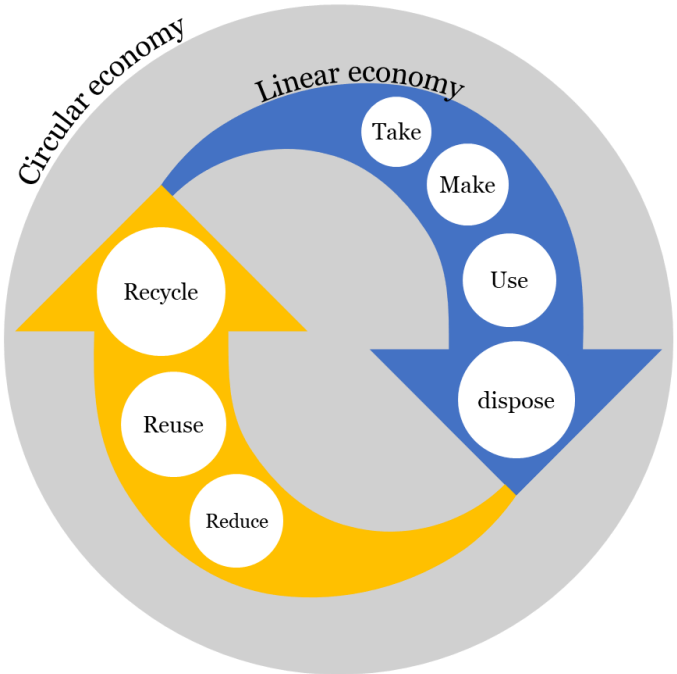


Fig. 1.1: Illustration of circular economy and linear economy models

All three principles of the CE focus on reducing waste and increasing resource efficiency. The first principle, reducing, targets the reduction of a production process input in terms of both energy and raw material, which is achieved by process efficiency enhancement. Moreover, reusing suggests expanding the life of used products through repair and remanufacturing techniques. In addition, recycling refers to the replacement of the new raw material with the used once collected from the EoL products [4]. The reusability of EoL products is considered one of the most important practices of the CE. This is due to its significant and direct impact on overcoming LE challenges by utilizing lower energy and material consumption and cost reduction [5].

Remanufacturing is the process of restoring the EoL products to their original usable or even better products. It is the most value-added and efficient technique among the three cores of the CE. It involves several steps to get to the remanufactured products, including disassembly, cleaning, examination, repair, reassembly, and final testing [3]. Furthermore, repair, where defects are identified and fixed, is the most crucial step in remanufacturing [6]. This is due to its complexity and requirements for professional labor and advanced technologies [7]. Although remanufacturing and repair terms refer to different concepts, they both are very similar and usually used interchangeably among the scientific community. In this thesis, the author assumes no distinction between the two terms, and they both refer to the repair or restoration process of EoL products.

Additive manufacturing (AM) processes have enabled the creation of advanced components with very complex shapes and features. Thus, many industrial sectors have embraced such technologies in their business model [8]. Furthermore, defective regions in a product come in various types depending on its serving environment. Therefore, irregular and complex defective regions need to be repaired, which can be processed easily using an additive manufacturing process [9]. This research focuses on metal additive manufacturing processes, which are defined as a process of building an object from a metal solidification by using a heat source, such as a laser or electron beam, to melt metal in the form of powder or wire [9]. For instance, Laser Metal Deposition (LMD) process uses a powder as raw material with a laser as a heat source. This process is also known as Laser Cladding (LC), Direct Energy Deposition (DED), Direct Laser Deposition (DLD), and Direct Metal Deposition (DMD). Furthermore, LMD provides many advantages over other processes, including multi-material printing capability, higher printing volume, highly controlled heat source, deposition over

manufactured surfaces functionality, and instantaneous change of the deposition material during the process [10].

Robot manipulators are the key component of a highly advanced repair system. This is due to its capability to perform sophisticated tasks exceeding human ability in speed and accuracy. Unlike gantry systems, robot-based systems allow for more flexibility and reachability [11]. Therefore, building overhang and curved features without support, as shown in **Fig. *1.2***, is enabled with such a configuration. However, current robot path planning algorithms are not optimized and integrated well for the repair process using this system [12].

Fig. 1.2: Additively manufactured components using an LMD robot-based system [13].

1.2 Motivation

The human population is growing rapidly with continuous demand for natural resources, which led to a surge in energy and material consumption. An increase of %50 in the global population is predicted by 2100. Furthermore, studies anticipated a doubled demand for materials globally by 2030 compared to 2010 [14]. Therefore, a movement towards sustainable resources is of serious interest to governances and scientists. Recently, the CE paradigm gained increasing attention for its advantage in conserving natural resources [15].

One of the CE cores is the reusability of the EoL products, which is accomplished by means of remanufacturing and repair. This is proven to have a direct impact on society and the economy in many aspects. Furthermore, repair and remanufacturing of EoL products are found to reduce 70% of the material, 60% of energy, 50% of the cost, and 80% of emission

compared to newly manufactured products shown in **Fig. *1.3*** [16]. The overall advantage of such a process is to eliminate and reduce waste of different types, thus supporting a sustainable global economy. Many efforts have been taken to adopt and increase manufacturability by many governments across the globe.

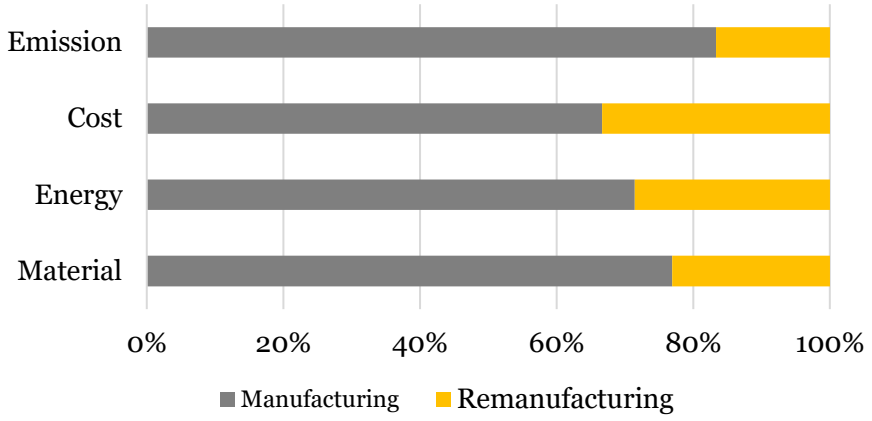


Fig. 1.3: Resources consumption of manufacturing products versus remanufacturing

Rotational mechanical components are extremely important parts of any mechanical system. Moreover, rotational mechanisms are usually exposed to an extended level of friction, thus creating a shorter product life cycle [17]. Therefore, revolving mechanical components contribute the most to EoL components in terms of geometry. In addition, the literature is mostly focused on remanufacturing revolving components such as engine turbine blades, mining sprockets, and crankshafts [7]. Furthermore, the integration of additive manufacturing processes with robot manipulators enabled repairing complex shaped components without the need for support [12].

The repair process has been achieved manually for a long time. It heavenly depends on a skilled worker and advanced tools; thus, it is time-consuming, missing repeatability, and prone to human errors [2]. Therefore, automation of the repair process is crucial to overcome those problems. This area is recently attracting more attention from the scientific community. In general, a repair process of EoL components involves four stages: pre-machining of the defective region, material deposition, surface finishing, and testing. The most critical stage among those is the material deposition process, which is highly dependable on a professional worker; thus, it is of consideration for automation.

A common automatic repair process follows six steps to achieve the renewed part. This approach is a 3D-based approach; the part is first scanned to acquire spatial measurements, which are used to obtain the nominal model, a fixed or original version of the component. Then, the reconstructed model is aligned with the damaged model to extract the defective volumes to be repaired. This is to generate robot tool paths via a slicing algorithm. This method is widely adopted in repairing rotary components such as impellers; however, it requires high processing time and cost since it depends on scanning the entire damaged part. However, with the rapid development of Artificial Intelligence (AI), a new trend to recognize defective regions on parts is evolving to reduce scanning and processing times. Furthermore, a current AI-based automatic repair process consists of four steps, as shown in **Fig. *1.4*** [18]. It follows a similar approach as the traditional repair process except for the ability to recognize damaged areas on cylindrical components using a vision-based deep learning model for a localized repair process. This process is reported to reduce the processing time by about 63% compared to the traditional repair process.

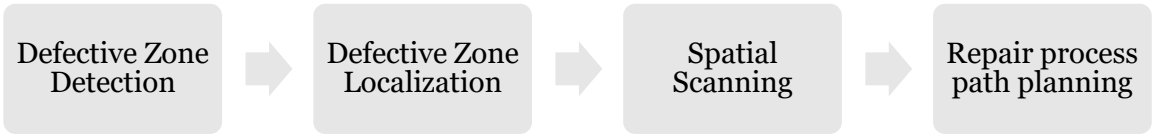


Fig. 1.4: AI-based repair process flowchart

Although the current approach is achieving promising progress in the automation of the repair process, it is in the very first steps of development, which requires more investigation and research. One of the limitations of this method is the constrains on the deep learning model to detection pre-defined regions on the part (i.e., pads) that might contain defects. Moreover, the localization step is based on edge detection that localizes only quadrilateral regions. Furthermore, the generated repairing tool paths are restricted to scanning points and paths, which limits the ability to produce multidirectional paths. To address the aforementioned limitations, this research is aimed to improve, enhance and build upon this approach to achieve autonomy in the repair process. Therefore, a generalized semi-autonomous repair system for revolving mechanical components is investigated in this research.

1.3 Thesis Objectives

The focus of this research is to develop and enhance a recently emerging semi-autonomous repair framework to automatically repair damaged cylindrical parts effectively and efficiently with minimal human dependency. This research is aimed to achieve three objectives as stated below:

OBJ1. Investigate the state-of-the-art object detection architecture in detecting metal defects of different types.

OBJ2. Develop an effective localization and quantification method for defective regions on cylindrical parts for a robotic cell using vision and proximity sensors.

OBJ3. Develop an automatic toolpath planning method to repair defective surfaces of cylindrical parts using a laser metal deposition process.

1.4 Methodology

The goal of this research is to automate the repair process of damaged cylindrical metal components thus, a framework is proposed to integrate intelligent decision-making, efficient defective regions localization, and automatic toolpath generation methods. The first step, shown in **Fig. *1.5***, is to adopt a deep learning model to achieve the automation of defective regions recognition using a vision sensor. A preliminary investigation was conducted on the state-of-the-art model to study its detection performance on different metal defect datasets. Based on this study, a surface defect detection pipeline was developed to recognize areas of interest in an image of the damaged cylindrical components.

This pipeline is used as a base for the overall repair system framework. With the component loaded in the robotic cell, the system begins collecting video scans of the cylindrical component to further compose a one-image view of the entire surface of the component. This image is generated using a stitching algorithm of the video frames. The image is then passed to the defect detection pipeline to automatically recognize any defective regions on the component. Furthermore, the detected regions of interest boundary pixel points are then processed to obtain the three-dimensional spatial points with a specified reference frame on the robotic cell. With this, a detection and localization stage is conducted and its output is passed to the next stage for acquiring point clouds of the defective surface.

Further, the defective region size is then computed to determine if it's to be processed and within the system limitation. In addition, optimal scanning paths are generated based on the bounding points obtained in the localization process. Based on the generated scanning paths, topography information of the defective surface region is acquired using a laser proximity sensor integrated into the robotic cell. The obtained point cloud data is then processed automatically to generate the repair process paths for the LC robotic system. The paths are then validated in a virtual environment before sending them to the actual robot system. Finally, defective patches are processed and repaired using a laser metal deposition process.

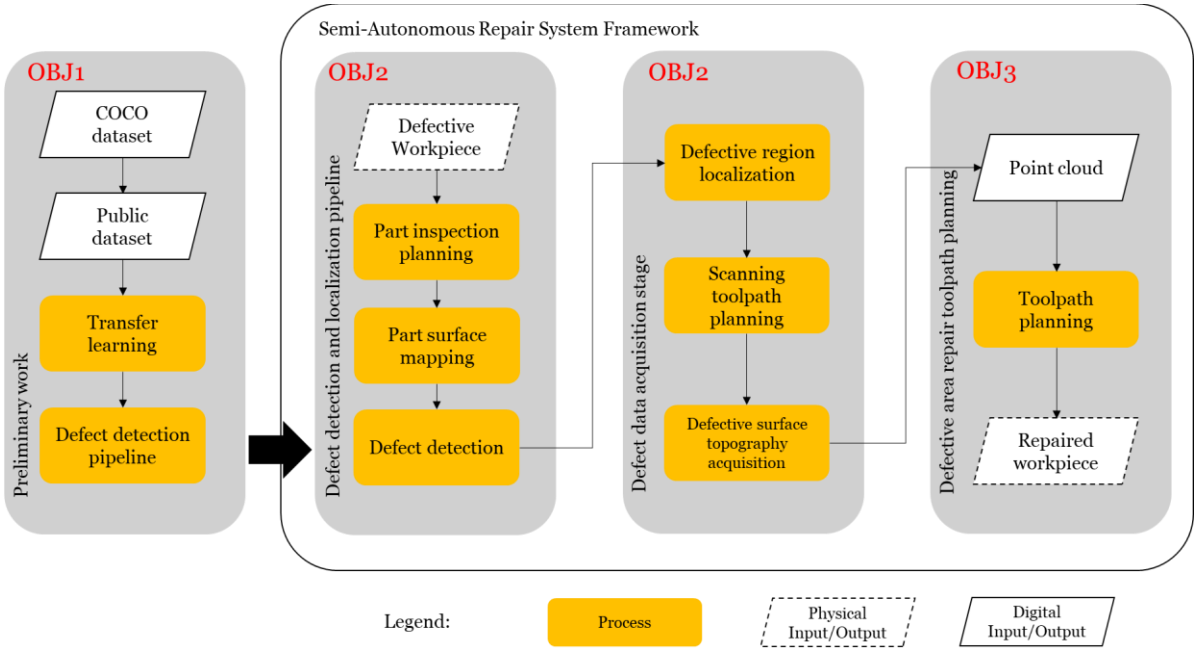


Fig. 1.5: Automatic repair framework for cylindrical components

1.5 Thesis Outline

The outline of this thesis begins with a background on the repair process and its impact on the economy and society. Following a motivation section for conducting and developing this research is presented. Further, a description of the targeted objectives of this thesis is stated in the paragraph of the objectives. Moreover, the research methodology is presented under the methodology section, which includes a brief description of the overall semi-autonomous repair framework.

In Chapter 2 (OBJ1), an investigation of the performance of the state-of-the-art deep learning model on metal defect detection tasks was conducted and presented in this chapter. Based on this study, an optimal variant of the model was selected and adopted into the overall framework to achieve independent defect recognition. Using various defect public datasets, knowledge was transferred to the collected custom image dataset via the transfer learning concept. Finally, a full detection pipeline is then presented and validated.

Chapter 3 (OBJ2) proposes and describes a defective region localization method for cylindrical components, which is based on a vision sensor. First, a component video scanning and stitching algorithm are introduced to obtain a 2D image map of the surface's component on a single image. Further, the stitched image is then processed using the defect detection established in Chapter 2 to acquire boundary pixel points of the defective region. Then, a spatial localization method is discussed based on the eye-in-hand calibration process. In addition, the size of the defective region is calculated following the spatial localization of the boundary points. Then, the point cloud acquisition automatic path planning method is discussed. Furthermore, a virtual robotic cell with vision and proximity sensors was introduced in this section. Finally, two virtual case studies were introduced for validation of the proposed method on the actual robotic cell.

Chapter 4 (OBJ3) presents an automatic toolpath planning method for repairing the defective surface region on a robotic work cell. The configuration of the cell is discussed to establish a clear understanding of the overall experimental system and help introduce the developed method. Further, the proposed method consists of three main stages. First, a defective surface of the component is reconstructed based on the acquired point cloud. The novel reconstruction process mainly uses slicing and projection techniques, which were not found in the literature. Then, laser processing paths, with their normal vectors, are generated from the reconstructed surface. This is followed by an interpolation process to obtain practical paths to be processed by the robot. Moreover, the proposed method was tested on three practical case studies to explore its effectiveness. Finally, the generated toolpaths of three case studies were validated in a virtual environment of the actual robot cell.

The final chapter, Chapter 5, summarizes the achieved contributions of this research and discusses the limitation of the proposed framework. Therefore, a future work section is addressed in this chapter to further enhance the autonomy of this system.

Chapter 2
Defect detection on metal surfaces

2.1 Introduction

Defect detection on metal surfaces [19] is an important area of research that is gaining more attention from the scientific community when moving more into a sustainable environment. Defects on metal surfaces occur in several types, including cracks, scratches, inclusions, corrosion, spots, patches, etc. [20]. Such defects can affect the quality and performance of the product. Thus, there have been many efforts to develop and adopt different models for efficient and accurate detection of such defects.

Some researchers have focused on developing novel deep learning architecture targeting a specific environment and dataset [21–25]. Others preferred to adopt and fine-tune some of the existing models that are trained and developed for a general object detection domain [26–32]. A recent approach is to improve a well-performed existing model utilizing hybrid methods [33–36]. Therefore, a behavioral study of the state-of-the-art object detection model on detecting defects on metal surfaces is important for further decision support in the selection repairing process. Thus, this paper focuses on investigating the state-of-the-art object detection model for the defect detection task.

Many novel approaches that have been developed are based on convolutional neural networks (CNNs). CNNs are robust networks for extracting embedded features for an image, such as corners, edges, etc., which makes them widely adopted in the object detection task [37]. Tao et al. [21] developed a detection pipeline based on cascaded autoencoder (CASAE) and CNN networks. This architecture uses CASAE to localize and segment the defect, whereas CNN is used to classify the defect type. Moreover, Parvez et al. [22] and Han et al. [23] designed similar networks that used CNN for feature extraction and a fully connected neural network (FCNN) for the classification step. Both focused their work on detecting defects in additively manufactured parts that include cracks, porosity, and lack of fusion. In addition, Xu et al. [24] developed a novel self-supervised efficient defect detector (SEDD) that focuses on eliminating the annotation step of the data by using a homographic enhancement method.

They also designed a custom detector based on depthwise convolution layers and an attention module to enhance performance and accuracy.

Another approach is to utilize state-of-the-art object detection detectors for defect detection applications. This approach is known as transfer learning, where the model is trained on a new dataset to perform a related task using initial weights [38]. Zheng et al. [26,28] adopted the mask region-based CNN (Mask RCNN) architecture for detecting and segmenting the damaged area on metal parts for further repairing applications. In addition, Zheng et al. [29] investigated the performance of Faster R-CNN, YOLOv3, and RetinaNet architectures on rail crack detection employing knowledge transfer of the COCO dataset [39]. Furthermore, Konovalenko et al. [30] and Litvintseva et al. [31] based their detection model on the U-Net architecture for defect detection. In [27,32], Zahir et al. leverage one of the most known two-stage models, Faster R-CNN using the transfer learning concept using the COCO dataset to detect and localize steel parts' wear for a remanufacturing task.

A recent work by Li et al. [33] proposed a new method that leverages and enhances the CSPDarknet53 architecture by integrating a multi-head self-attention block and using it as the backbone of their detector. They also adopted some simple yet effective techniques to enhance the model's performance, such as augmentation and grayscale filtering. Moreover, Pan et al. [34] integrate a dual attention module with DeepLabv3+ architecture to detect and segment metal defects. Furthermore, Wang et al. [35] developed a new module that follows a similar pipeline of ResNet based on transformers and CNNs to retain both local and global information. Moreover, Gao et al. [36] adopted the swin transformer with a variant shift window called Cas-VSwin Transformer as the backbone network for better performance on the defect detection task. In addition, they used Feature Pyramid Network (FPN) as the neck, whereas the Cascade Mask network is the head of the architecture.

One feature of the YOLO detectors is real-time inference. They gained a lot of attention on the object detection task. With the speed requirement for the defect inspection process in production lines, many scientists have investigated and adopted YOLO for the defect detection problem. In addition, the YOLO series provides the lowest processing time compared to all other state-of-the-art detectors. However, higher detection accuracy is achieved with enhanced versions of the YOLO series by integrating it with other networks, such as self-attention mechanisms. Li et al. [40] developed an improved version of the YOLO

network using only CNNs layers. This method performed well; however, no comparison with the standard YOLO network is conducted.

Furthermore, Kou et al. [41] proposed a new detector based on YOLOv3 architecture with an anchor-free feature selection method and custom dense convolution blocks. This improves the training process and inference accuracy by about 26.7% on the NEU-DET dataset. In addition, Xu et al. [42] modified the YOLOv3 network to improve accuracy by focusing on extracting more features of small defects using a new scale feature layer. With this, an improvement of 4.6% in precision is gained on the Tianchi dataset. On the other hand, Guo et al. [43] introduced an improved architecture of YOLOv5 with a transformer encoder as the backbone of the network to integrate more global information into the model. This model was tested with the NEU-DET dataset and achieved %75.2 mean average precision (mAP) on inference.

YOLOv7 is the current state-of-the-art for real-time object detection tasks in terms of both speed and accuracy [44]. However, this benchmark is based on the Microsoft COCO dataset. A benchmark of the YOLOv7 on the defect datasets is essential to observe and investigate its performance on metal defect detection tasks. This paper focuses on adopting the state-of-the-art model, YOLOv7, for detecting defects and damages on metal surfaces and investigating the model's behavior on some public datasets of metal defects. This can also be used as a reference for future comparisons of the enhanced models based on YOLOv7 with the standard. Furthermore, a comparison of the existing defects detection models with the YOLOv7 is conducted and studied.

2.2 YOLOv7 Architecture

YOLOv7 is the latest improved model of the YOLO series. YOLO models, including YOLOv7, are a type of single-stage framework that contains three main components named backbone, neck, and head [45–51], as shown in **Fig. 2.1**. The backbone extracts feature maps of an image and transfers them to the neck layers. Those maps are combined, fused, and passed to the next layers. Then, the head network predicts the objects' bounding boxes and their classes. Unlike the two-stage models, the YOLO series, a single-stage model, reconsidered the object detection task as a regression problem instead of a classification problem, which is the main feature of real-time detection algorithms [52].

The efficiency of the CNNs in the backbone network is important to enhance the inference process. Thus, the author of YOLOv7 proposed an enhanced method based on Efficient Layer Aggregation (ELAN) network named Extended-ELAN (E-ELAN). This method improves the ELAN architecture to boost the learning ability of a scaled network without disturbing or changing the original gradient propagation path. E-ELAN has a stronger learning ability for various features.

Furthermore, YOLOv7 comes with a new method of scaling for concatenation-based models. The proposed method, named corresponding compound model scaling, addresses the issue of a larger width output of the computational block when depth scaling is performed on the architecture. With the proposed method, the depth of the concatenation-based model is scaled directly; however, the width of transition layers is scaled with a corresponding factor calculated from the change of the output width of the block.

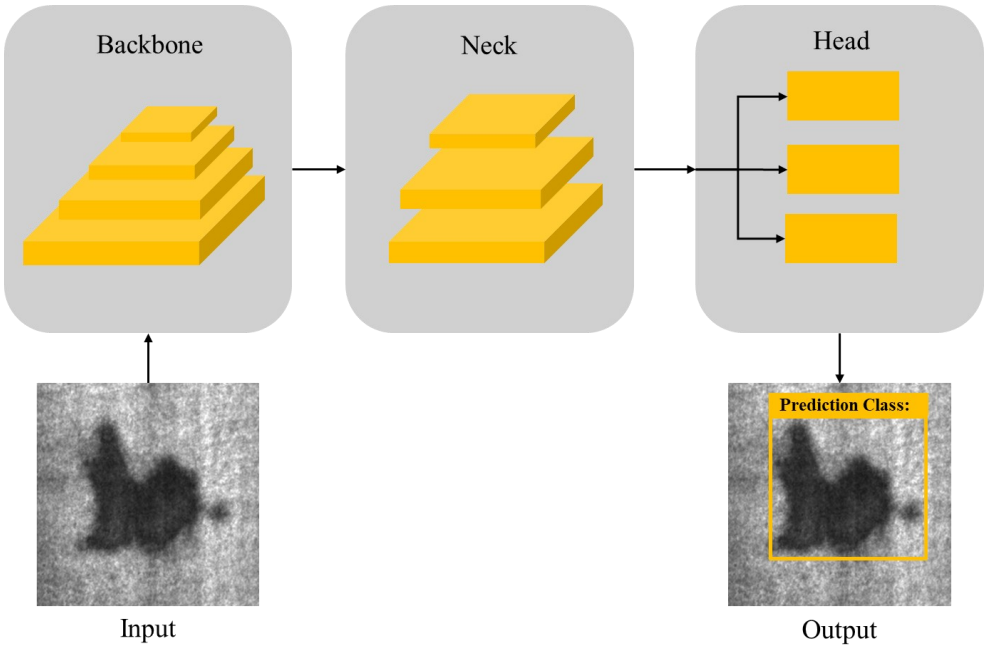


Fig. 2.1: YOLOv7 architecture design

Moreover, serval techniques have been introduced in the YOLOv7 to improve the model inference accuracy with maintaining a low training cost. Those strategies are called bags of freebies (BoF), including planned re-parameterization and dynamic label assignment. After a deep investigation of the re-parametrized convolution behavior when combined with various networks, the author showed an increase in the model's accuracy when using the

RepConv without identity connection (RepConvN). Further, in supervised deep learning techniques, the head that represents the final prediction of the model is called the lead head, whereas the auxiliary head is the head that is used to assist the lead head. Previously, both heads are independent of each other, and their prediction and the ground truth are used as a soft label for label assignment. However, YOLOv7 proposed a new method for lead-dependent label assignment. Two types of label assigners were developed with the YOLOv7. One is lead head guided, where the soft label is mainly generated from the leader's head and ground truth. Another is coarse-to-fine lead head guided, where two different soft labels are produced, including fine and coarse labels. The fine label is the same as the one generated in the lead head guided assigner; however, the coarse label is generated with relaxed rules on the positive sample assignment process.

Other BoFs techniques are also adopted and used in YOLOv7, such as batch normalization, implicit knowledge, and the exponential moving average (EMA) model. Normalization of the training batch, by integrating the mean and variance of the data to the bias and weight of the convolutional layer, is proved to directly affect the training process by utilizing a higher training rate and faster convergence [53]. Another technique is the implicit knowledge, adopted from the YOLOR [51], computed as a vector in the inference stage of YOLOv7, improving the prediction accuracy in previous versions. Lastly, adopting the EMA model as the final inference model in YOLOv7 has improved the inference accuracy.

YOLOv7 outperforms all existing models in the object detection task in terms of both speed and accuracy. According to the author, the focus of this version is to optimize the training process for better detection accuracy and speed, as well as for the inference process. This is achieved by several factors including model training parameters reduction and enhancing the learning process.

2.3 Training and Results

2.3.1 Datasets

Since the YOLOv7 was trained and tested on the Microsoft COCO dataset, the model's behavior on the metal defect datasets has not yet been studied by researchers. Therefore, this paper investigates the behavior of the YOLOv7 model on metal defect detection tasks. An optimal approach for this study is to use public datasets for standard benchmarks and replication of the same results in the future. In addition, a comparison of the YOLOv7

performance with previously reported results is conducted. A list of the available public datasets is mentioned in **Table 2.1**.

Table 2.1: Public Metal Defects Datasets

Dataset	Set Size	Raw Image Size (pixels)	# of Classes
Severstal Defect Dataset [54]	18,074	1600 x 256	4
No-service rail surface defect **(NRSD)** [55]	4,101	600 x 600	1
Kolektor Surface-Defect Dataset 2 **(KSDD2)** [56]	3,335	230 x 630	1
DAGM 2007 [57]	2,300	512 x 512	10
GC10 Defect Dataset **(GC10-DET)** [58]	2,294	415 x 416	10
Northeastern University Defect Dataset **(NEU-DET)** [59]	1,800	200 x 200	6
Ball Screw Drive Surface Defect Dataset **(BSData)** [60]	1,104	1130 x 460	1
Kolektor Surface-Defect Dataset **(KSDD)** [61]	399	500 x (1240 – 1270)	1
Rail surface discrete defects **(RSDD)** [62]	167	various	1

There are many public datasets available for the metal defect detection task; however, investigating the YOLOv7 model performance on all of them is time and resource-consuming and adds a minimal contribution to the study. Thus, some of the datasets listed in **Table 2.1**, are selected to perform this study following specific criteria. This study selected datasets that are widely used, recently collected and practically sized. As a result, three datasets were selected to perform this investigation and obtain a general performance of the YOLOv7 model on the defect detection task. Those are NEU-DET, NRSD, and KolektorSDD2 datasets. Samples of the annotated datasets are shown in **Fig. 2.2**, **Fig. 2.3**, and **Fig. 2.4**.

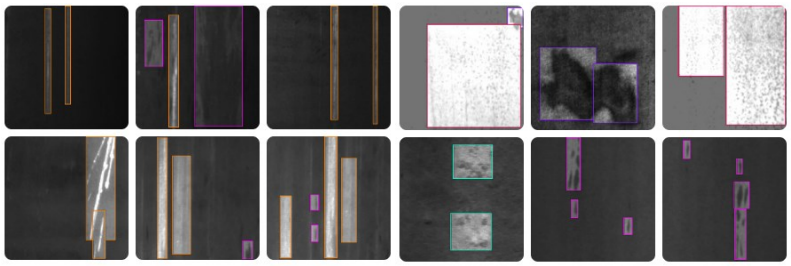


Fig. 2.2: Annotated samples of the NEU-DET dataset.

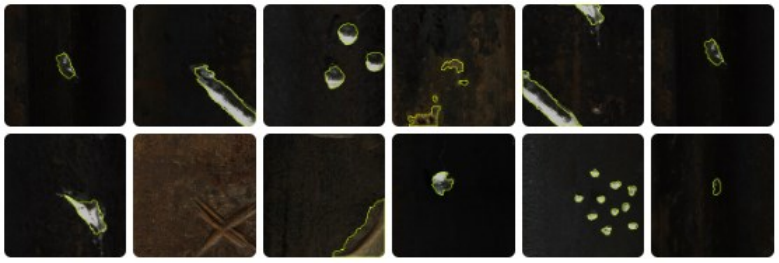


Fig. 2.3: Annotated samples of the NRSD dataset.

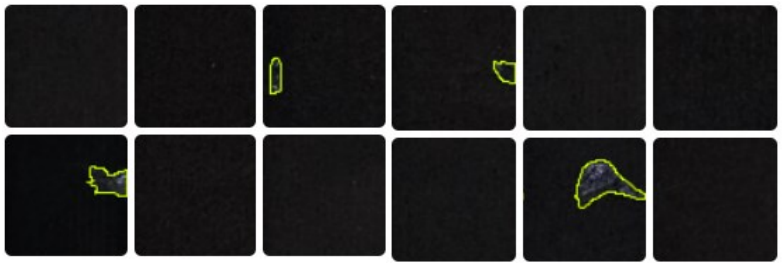


Fig. 2.4: Annotated samples of the Kolektor 2 dataset.

The NEU-Det dataset contains grayscale images collected from real-life samples. In addition, the NEU-Det dataset is annotated for an object detection task meaning only bounding boxes of the defect exist, whereas NRSD and KSDD2 are annotated for both object detection and segmentation tasks containing bounding boxes and masks. Furthermore, NSRD and KSDD2 consist of colored (i.e., RBG) images of planned and unplanned defects. Furthermore, it is important to note that the NRSD dataset contains some synthetic segmentation of the collected images generated by the MCnet network. Therefore, annotation is not accurate and might cause difficulty for the model to perform.

2.3.2 Training Environment

The model was trained and tested locally on a machine with the specifications listed in **Table 2.2**. Those specifications are considered at the high end of the time performing this study.

Table 2.2: Training Machine Specifications

Property	Value
CPU	AMD Ryzen Threadripper 3970X 32-Core
GPU	NVIDIA GeForce RTX 3090 / 24 GB
CUDA cores/version	10496 / 11.8
Operating System	Windows Server 2019
RAM	128 GB
PyTorch	1.10.1

2.3.3 Training Parameters

To maintain a standard benchmark with the original results of the YOLOv7 models, this work uses similar training parameters as in the original paper with an increased training iteration for the larger variants of the YOLOv7 model, i.e., yolov7-d6, yolov7-e6, and yolov7-e6e. There are three different hyper-parameter settings depending on the model size, including small, medium, and larger. Furthermore, each dataset has a different set size and image resolution; as a result, training hyper-parameters are slightly different for each dataset. A general range of those parameters is listed in **Table 2.3**. In addition, each dataset is split into three sets: 70% training, 20% validation, and 10% testing. Each training parameter will be mentioned in the results section in detail.

Table 2.3: Training parameters

Parameter	Value	Parameter	Value
Learning Rate	0.001	**Batch Size**	8 - 32
Momentum	0.937	**Image Size**	Depends on dataset
Weight Decay	0.0005	**Epochs**	100 - 300

2.3.4 Evaluation Metrics

In this work, standard evaluation metrics are adopted, the same as in YOLOv7, to study and compare the performance of the different variants of the YOLOv7 models on different metal defect datasets. Traditionally, the mean average precision (mAP), which is the area under the precision and recall curve calculated using (2.1), is calculated at a 0.5 threshold of the intersection over union (IoU) (mAP_0.5). However, a recent trend in the research field is to compute mAP over multiple IoU values following the COCO interpretation from 0.5 to 0.95 with a step of 0.05 (mAP_0.5:0.95). This has proven to affect the model with a better localization reward. However, some of the recent studies in the defect detection field still use the mAP_0.5; thus, both metrics will be reported in this study. In addition, inference speed is also observed for each dataset.

$$AP = \int_0^1 (precision \times recall)\, d(recall)$$
$$Precision = \frac{TP}{TP + FP}$$
$$Recall = \frac{TP}{TP + FN} \qquad (2.1)$$
$$IoU = \frac{|A \cap B|}{|A \cup B|}$$

Where TP, FP, and FN are the true positive, false positive, and false negative of the bounding box predictions, respectively. For the supervised learning model, one useful matric is the confusion matrix, shown in **Fig. *2.5*** which a class-wise distribution that maps the predicted classes to the actual classes. It is used to measure the global performance of the deep learning model. This matrix shows the strength and weakness of the model indicating the areas of improvements.

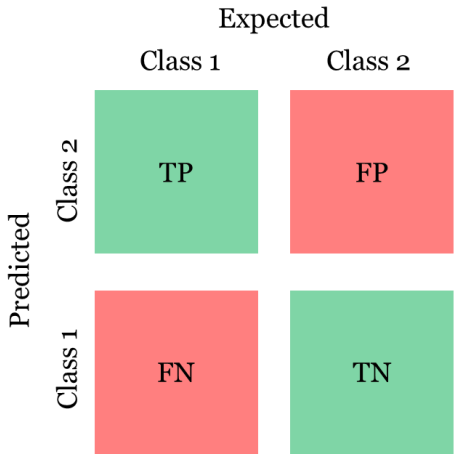


Fig. 2.5: Confusion matrix for binary class

2.3.5 Training Results

The YOLOv7 model was trained on each of the selected datasets in section 2.3.1 to measure its performance in detecting defects on metal surfaces. Since the NEU-Det dataset was annotated for an object detection task meaning, the first section of this study will focus on investigating the behavior of each variant of the model on this dataset. The remaining investigation of this study is to analyze and report the performance of the base variant of the YOLOv7 model on the metal defect detection and segmentation task. Moreover, this work used the initial weights of the trained YOLOv7 model on the Microsoft COCO dataset. This is to leverage and transfer the knowledge from the previous object detection task and facilitate the training process for faster convergence.

- NEU-DET

In this study, each model variant was trained and tested on this dataset to observe a wider behavior of the model. The validation and inference results are shown in **Table 2.4**. There are three main evaluation metrics presented in the table. AP^{test}, AP^{val}, and AP_{50}^{test} are mAP_0.5:0.95 for the testing set, mAP_0.5:0.95 for the validating set, and mAP_0.5 for the testing set, respectively. Those parameters are selected for better visualization of the model's performance. As shown in the table, the main difference between each variant is the model size, increasing as going down the table. The larger the model size, the more training parameters it has, which increases the learning knowledge and training time.

Furthermore, the smallest variant of the YOLOv7 model took about 84 minutes to complete 150 epochs of training, whereas the largest version took four times the smallest one. The table shows that the YOLOv7 achieved about %73.9 mAP with a 0.5 threshold on the testing set. However, as expected, a lower accuracy was observed with an interval threshold of (0.5-0.95). Although each variant was trained in slightly different parameters, the reported results are all for 150 training epochs. In **Fig. 2.6**, the training process curve of each variant is plotted to represent the mAP_0.5. As shown in this figure, the small-sized models tend to converge faster compared to the larger ones due to the number of parameters that require optimization. It is also worth noticing that smaller models have a smoother learning curve, which might be due to the loss of some information in the images when resizing, especially for the small objects. In addition, a smaller batch size might reduce the sharp fluctuation in the learning curve. **Fig. 2.7** shows the mAP_0.5 of the testing set with its trend. The performance trend of the YOLOv7 model variants represents a parabolic trend where the performance decreases in the middle. This could be due to the increase in the image size and the model, which led to more knowledge gained.

Table 2.4: Performance of each variant of YOLOv7 on NEU-DET metal defect datasets

YOLOv7 variant	#Params	Image Size	FPS RTX 3090	AP *test* / AP *val*	AP_{50}^{test}
Tiny	6M	224	103	37.0% / 35.6%	73.9%
Base	37M	224	78	37.1% / 35.4%	73.9%
X	70.8M	224	63	30.2% / 30.3%	65.8%
W6	81M	448	61	31.7% / 30.5%	69.3%
E6	110M	448	45	31.1% / 29.8%	67.2%
D6	153M	448	40	33.8% / 32.6%	70.9%
E6E	164M	704	30	36.0% / 31.7%	73.3%

This dataset is public and has been used in serval studies. Furthermore, comparing the YOLOv7 results with other previously reported results is valuable. Lv et al. [58] reported a 72.2% in mAP_0.5 on their proposed method based on EDDN. Their model performed slightly better in the pitted surface and scratches classes but worse in detecting cracks. Furthermore, Guo et al. [43] reported better results with 75.2% of mAP_0.5 of their proposed model based on an improved YOLOv5 that, an increase of about 1.3%. Moreover, a recent model proposed by [36] based on transformers has achieved higher results in the NEU-DET

dataset with 80.5% mAP_0.5. It is worth noting that integrating YOLO models with transformer networks enhances the model performance, as shown; thus, it is worth exploring this approach to further increase detection accuracy. Lastly, the base variant of the YOLOv7 model provides the right balance between accuracy and speed for real-time defect detection.

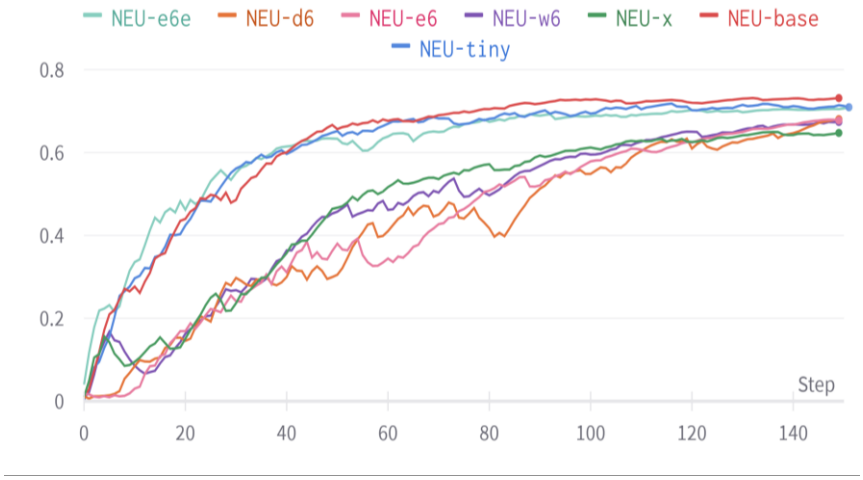


Fig. 2.6: YOLOv7 varients' training process on the NEU-DET dataset (mAP_0.5)

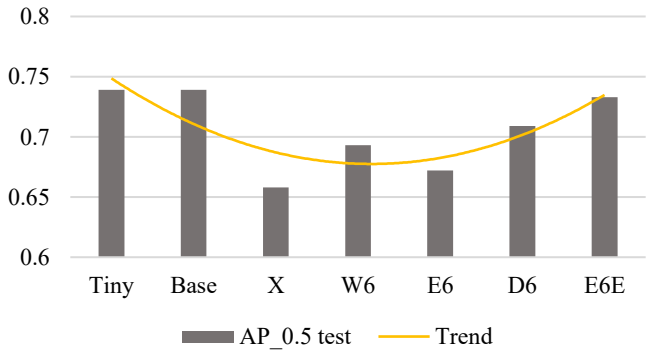


Fig. 2.7: YOLOv7 varients' testing performance on the NEU-DET dataset

Looking further into the base variant results, the model struggles to detect cracks and rolled-in scale defects with about 19% of mAP, as reported in **Table 2.5**. This might be due to the nature of those types of defects existing with minimal features and are less distinguishable with the metal surface compared to other types of defects. In addition, the ratio of defect to background pixels in those types is usually small compared to other types. In contrast, patch defect provides unique features with a high defect/background ratio leading to better results of about 54% mAP.

Table 2.5: YOLOv7 base variant model performance per class

Class	AP_{50}	mAP
Crack	54.5%	19.0%
Inclusion	81.9%	40.8%
Patch	92.4%	54.2%
Pitted Surface	79.7%	39.8%
Rolled-In Scale	55.1%	18.5%
Scratch	80.5%	40.2%

From the precision-recall curves of each class plotted in

Fig. 2.8, it can be observed that the model achieved above the average mAP for most of the defect types, especially those with a larger defect-to-background ratio. The area under this curve shows %74 mAP for all classes for a 0.5 threshold. Since this dataset contains gray scale images, a preprocessing step to reduce background intensity might boost the model's performance on detecting smaller objects e.g. cracks and rolled-in_scale.

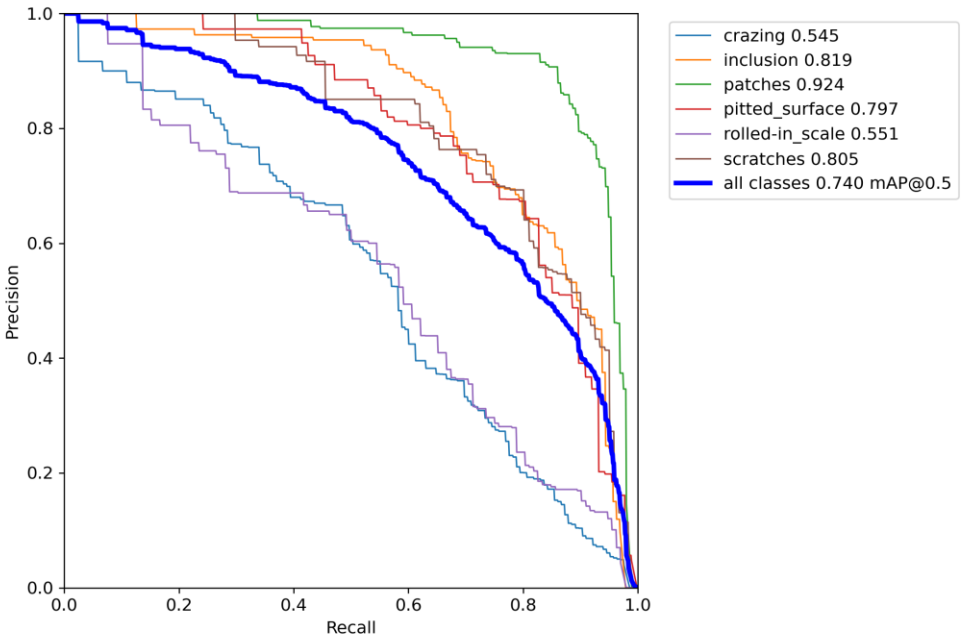


Fig. 2.8: Precision-recall curves for NEU-DET dataset

Another metric for evaluating the model's performance is the confusion matrix shown in **Fig. 2.9**, this matrix is very helpful to understand the behavior of the inference of this

model. It provides more details on the model's detection accuracy and indicates areas of improvement. For example, the model was not detecting around half the times when it is tested on cracking defect type, as indicated by 0.42. However, the model was confusing around a quarter of the detected rolled-in_scale defect type with the background, as shown in **Fig. 2.10**. This indicates that more cracking and rolled-in_scale samples will increase the model's knowledge and performance for both classes.

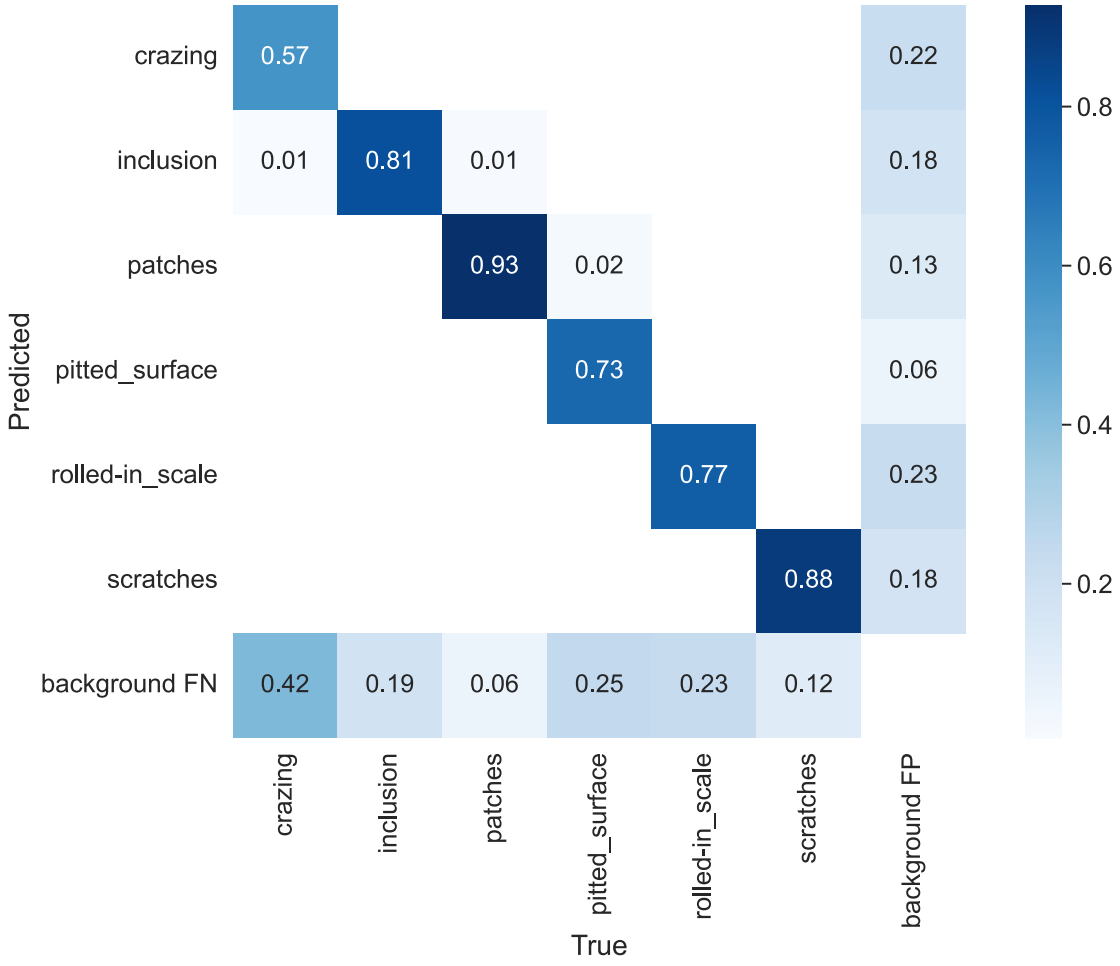


Fig. 2.9: YOLOv7-base confusion matrix for NEU-DET dataset

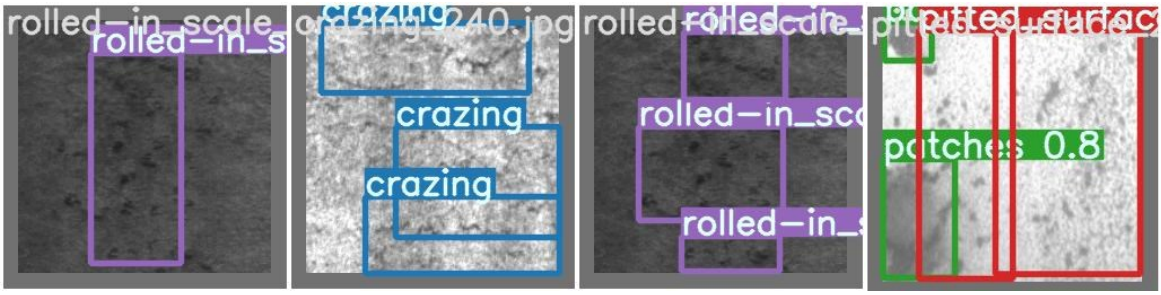
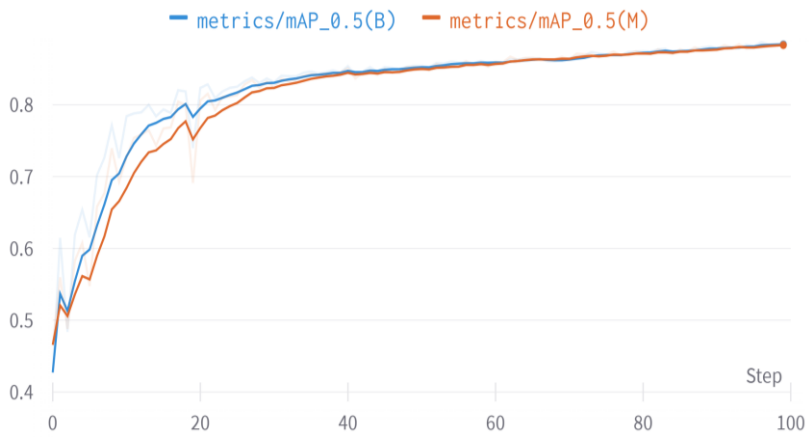


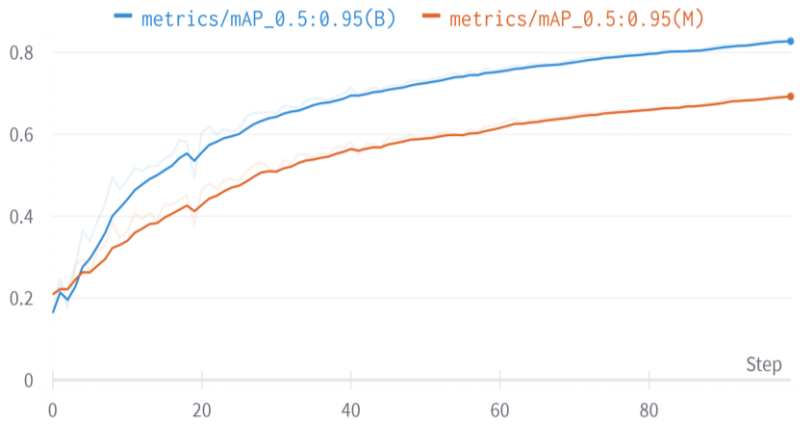
Fig. 2.10: Samples of the NEU-DET dataset inference results

- NRSD

For this dataset, the base variant of the YOLOv7 model was selected to investigate its instance segmentation performance. Although this dataset contains multiple types of defects, including scratches, wear, welding spots, etc., it was annotated with one class which is a defect. Furthermore, this work reports both object box and mask results. The dataset images were all resized to 640x640 for the training process. The model was trained for 100 epochs and 24 batch sizes, and the same parameters mentioned in **Table 2.3** were used. As shown in **Fig. 2.11**, the YOLOv7 model performed better with the NRSD dataset than the NEU-DET for many reasons, including RGB information, higher quality images, large objects, higher defect/background ratio, and larger dataset size. The model has a similar performance for detecting the bounding box of the defects with about 88.5% for both metrics (mAP_0.5 and mAP_0.5:0.95). However, the segmentation task has a slightly worse accuracy, where it achieved 69.6% mAP_0.5:0.95 in detecting masks. The similarity of the defects to the background might affect the segmentation results, as shown in the testing results in **Fig. 2.12**. Furthermore, the segmentation training process converges slower than the object detection process due to the larger knowledge that needs to be learned. Thus, increasing training epochs might increase the segmentation accuracy. In addition, the inference process is observed to take about 18ms per image on RTX 3090 GPU that about 55 fps. In contrast with previously reported results, Li et al. [33] proposed a detection model that accomplished 81.09% of mAP_0.5, which is lower by about 6.91% compared to the YOLOv7 model.



(a) mAP_0.5



(b) mAP_0.5:0.95

Fig. 2.11: YOLOv7 base variant model training processing performance on the NRSD dataset (B: box, M: mask)

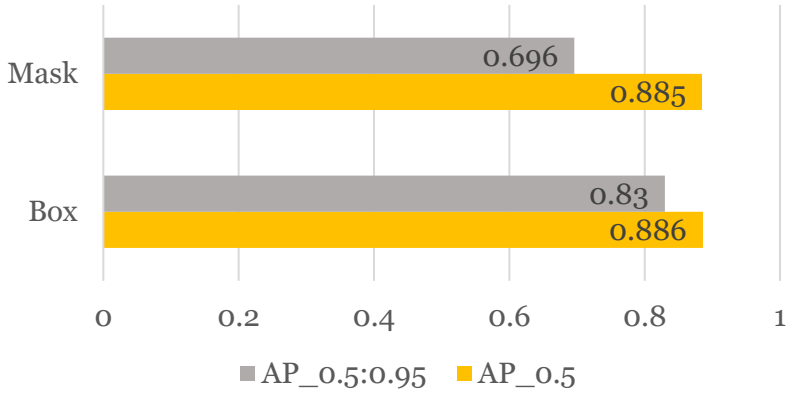


Fig. 2.12: YOLOv7 base variant model inference results

Further, the performance of the YOLOv7 in detecting defects of the NRSD dataset was excellent, as shown by the larger area under the PR curve in **Fig. *2.13***. This is for both defects bounding box and segmentation mask. Moreover, the confusion matrix was plotted unnormalized, in **Fig. *2.14*** to visualize the distribution of the actual detection since it is only one class with the background. In this confusion matrix, the left column shows the total actual annotated defects in the testing dataset, which is 1,147. This also shows the stability and reliability of the YOLOv7 model in detecting defects on the NRSD dataset. In addition, only 53 detections were false detections compared to 1013 true detections. On the other hand, the model missed 134 defects in the testing dataset, which indicates no overfitting behaviour. Samples of the inference detection are shown in **Fig. 2.15**.

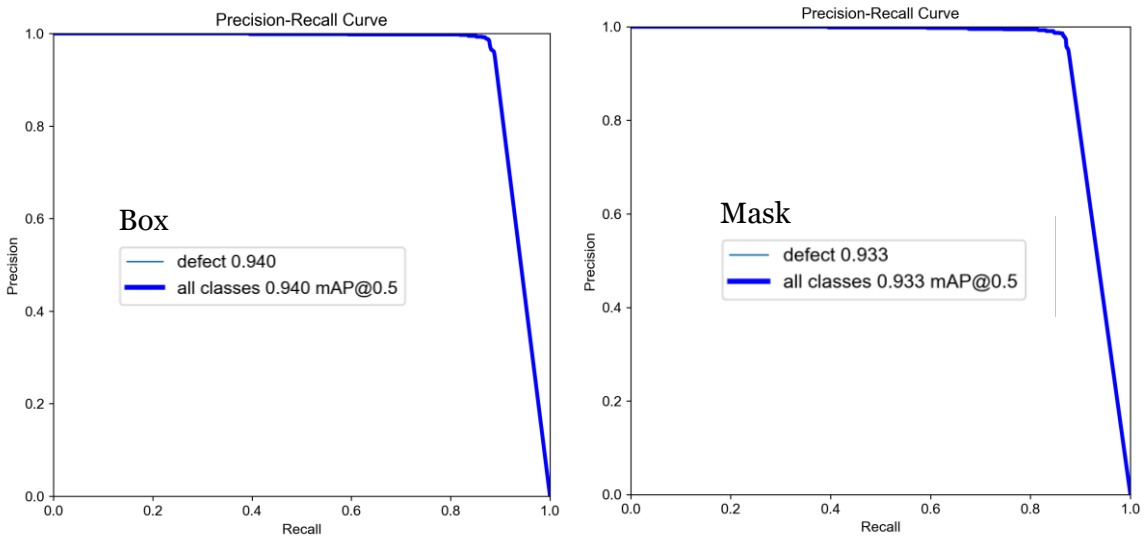


Fig. 2.13: PR-curves for bounding box and mask of the NRSD dataset

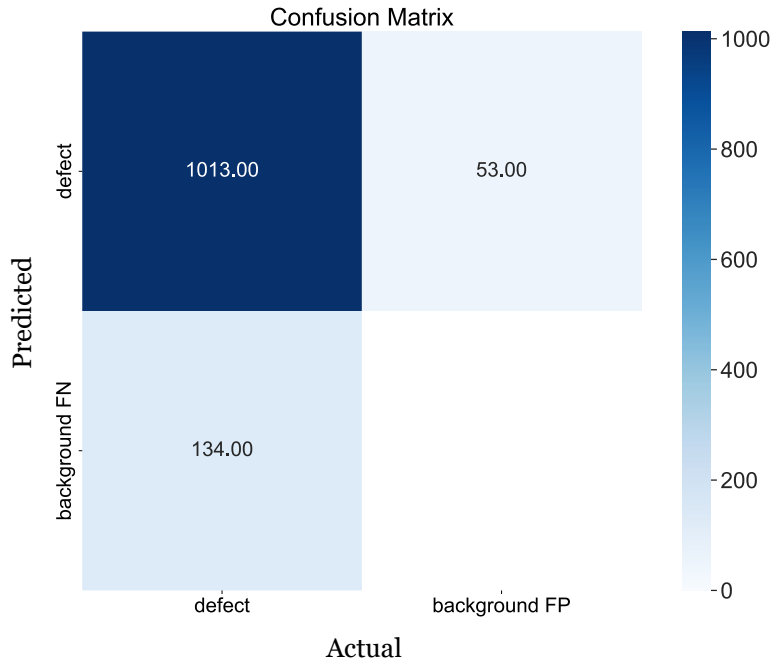


Fig. 2.14: Confusion matrix for NRSD dataset

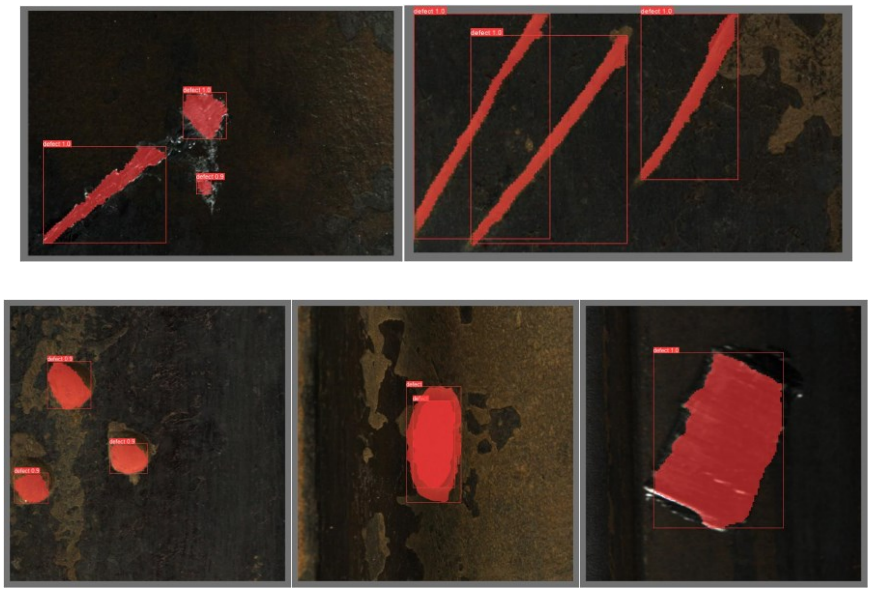
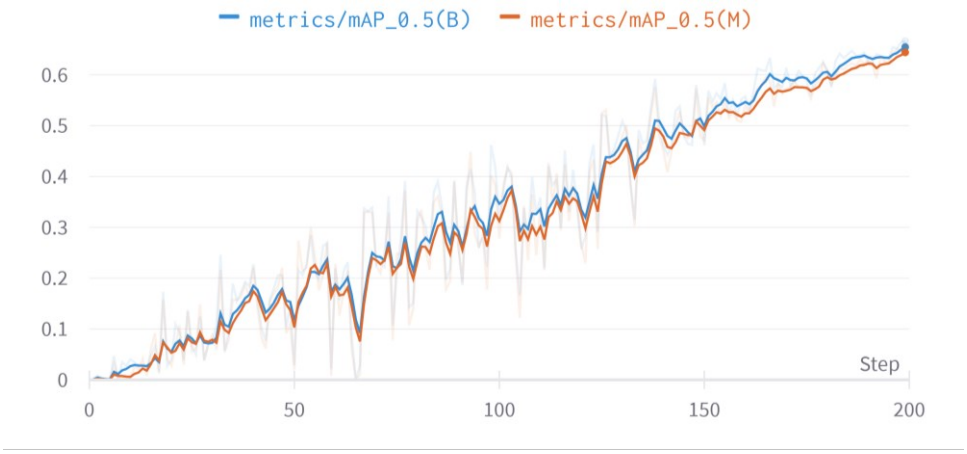


Fig. 2.15: Sample results of the model prediction for the NRSD dataset

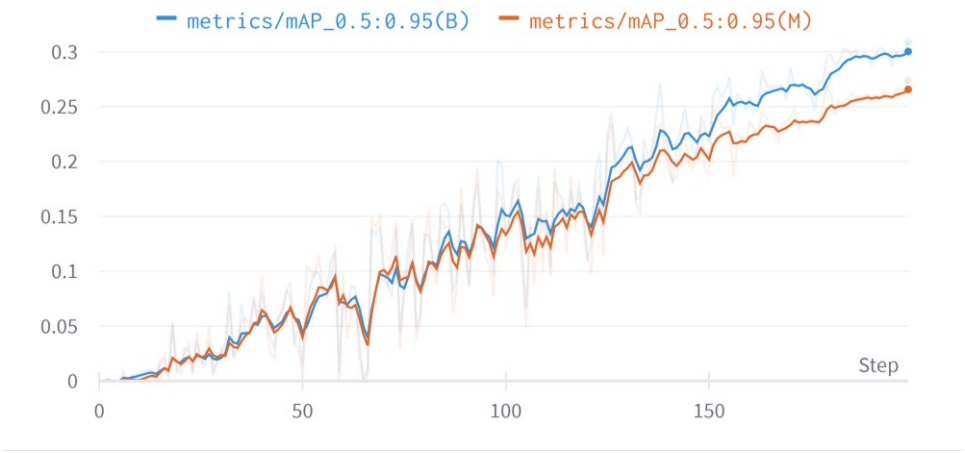
- KSDD2

In this challenging dataset, where samples containing defects are only about 10% of the total samples, training results are shown in **Fig. 2.16**. As the NRSD dataset, the base model variant was selected and trained with similar parameters as with NEU-DET mentioned in **Table 2.3**. In addition, images are resized to 640x640 to reduce training time. Furthermore, the model was trained for 200 epochs since this dataset has fewer true positive samples, which is expected to slow the learning process, as shown in **Fig. 2.16**. The training process results plotted in **Fig. 2.16** show a difficult learning curve dues to serval reasons, including a small number of defects samples and a similar texture of defect with the background. For the mAP with a 0.5 IoU threshold, the model performance with about 65% for both the bounding box and mask. However, the accuracy is reduced dramatically for the mAP over the range (0.5-0.95) IoU threshold with about 30% for the bounding box and 26% for detecting masks. This is due to smaller portions of the defective areas being detected that are ignored when the threshold is larger. For this dataset, Jakob et al. [56] introduce a weakly supervised model that achieved a 73.3% mAP_0.5 which is about 8.3% better than the YOLOv7 model in segmentation.

Furthermore, the difficulty in the training process was also reflected in the inference process, where the area under the PR curves, in **Fig. *2.17***, are estimated to be 0.72 and 0.70 for both box and mask, respectively. In addition, the confusion matrix in **Fig. *2.18***, shows the struggle of the YOLOv7 model to detect a defect of KSDD2 type indicated by the larger value of the false negative, meaning the model is skipping or missing more than half of the detection. Moreover, empty images meaning non-defective images, are not included in the confusion matrix because there is no class associated with them; as a result balancing the training and testing samples might improve the model's performance. Lastly, some of the inference results are shown in **Fig. 2.19**.



(a) mAP_0.5



(b) mAP_0.5:0.95

Fig. 2.16: YOLOv7 base variant model training processing performance on the NRSD dataset (B: box, M: mask)

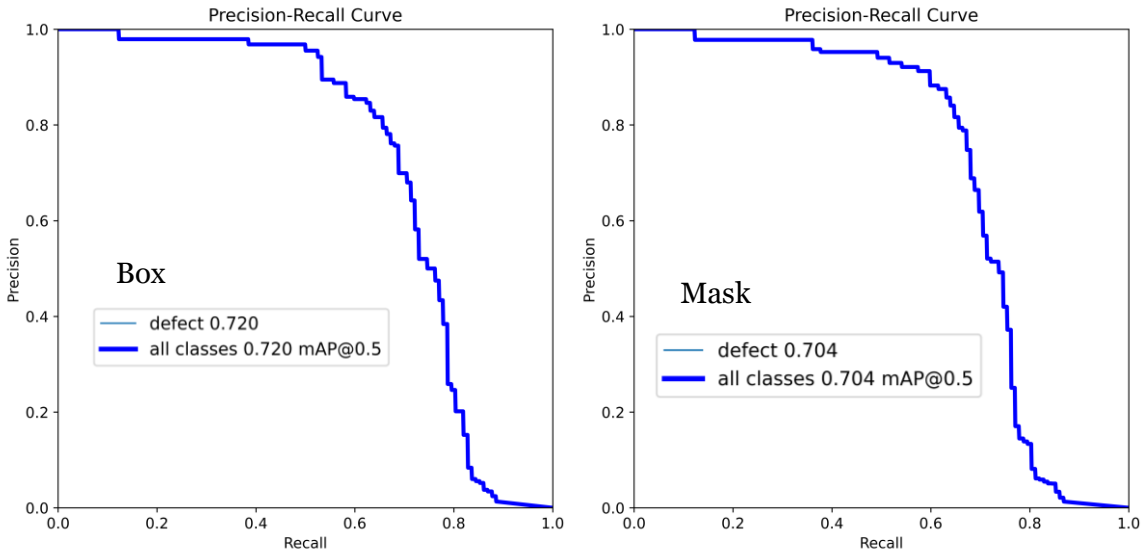


Fig. 2.17: PR-curves for bounding box and mask of the KSDD2 dataset

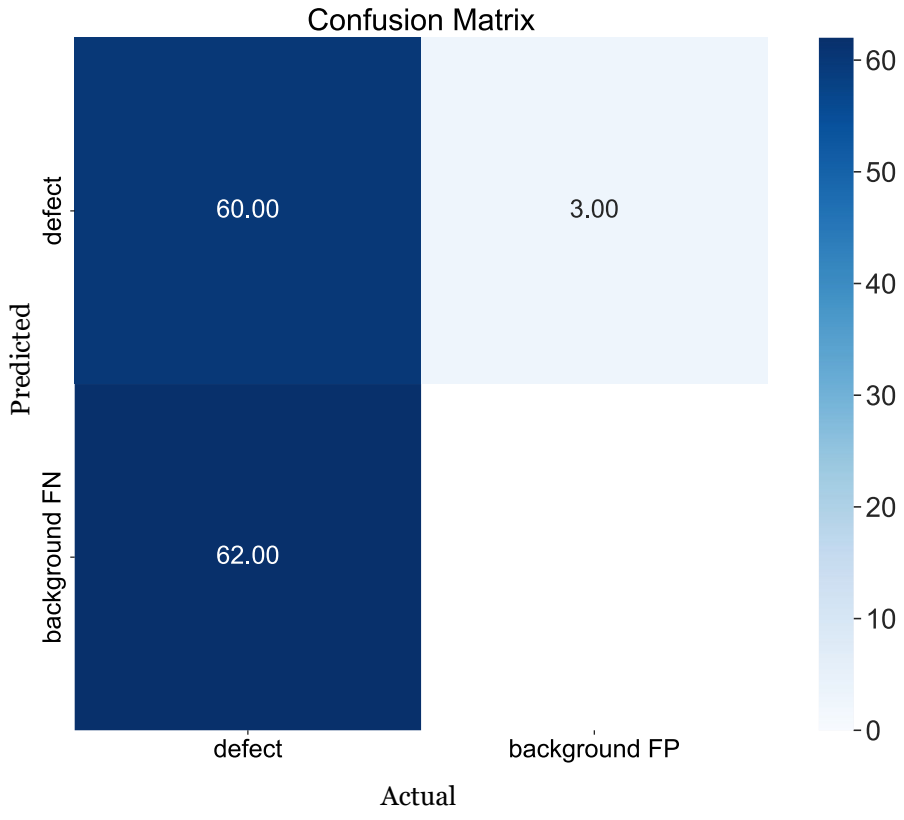


Fig. 2.18: Confusion matrix for the KSDD2 dataset

Fig. 2.19: Sample results of the model prediction for the KSDD2 dataset

2.3.6 Model Generalization and Deployment

After training the YOLOv7 model on different datasets and studying its behavior for metal defect detection, it is crucial for the model to be general enough to performance as expected during the training and inference processes. One advantage of adopting deep learning model, YOLOv7, is its performance stability for new different data that is close to the training dataset. As result, increasing the training dataset containing different types of defects is a good approach to expose the model for different defects. Further, increasing the training dataset can combine multiple datasets together to produce a single dataset. In addition, there are several techniques to increase the training datasets, including augmentation, synthetic datasets generated by deep learning models, and collecting new datasets. Before deploying the deep learning application, a fine-tuning is step is important to ensure the reliable detection of different defects in the real world. This step is accomplished by using a real-life dataset with stable ambient light to train and test the model before deployment. In addition, one technique when deploying is to add constrains to the model to remove noise (false detection). Finally, continuous improvement of the detector is very important by collecting the processed images for a larger training dataset in the future.

Chapter 3
Autonomous localization and quantification of metal defects on cylindrical components

3.1 Introduction

Natural resources on earth are limited and need further consideration in consumption. This includes materials, energy, and workforce. One approach to overcome this problem is to adopt the circular economy [63]. Unlike the linear economy, the circular economy focuses on reducing waste by different means of recovery through reduce, recycle, remanufacture, and repair [64]. Remanufacturing and repair processes were found to have the most impact on resource-saving compared to other processes. For instance, repairing a damaged turbine blade reduces carbon emission by about 45% and saves 36% in energy consumption compared to manufacturing a new blade [65]. Those processes have attracted the attention of academia, industries, and policymakers for their direct impact on sustainability and economic growth [66].

Remanufacturing is the process of restoring the end-of-life (EOL) product to its original state or better. It involves several steps, including product disassembly, cleaning, inspection, restoration, reassembly, and testing. Furthermore, the recovery process is among the most time-consuming processes of this flow. It involves rework, refurbishment, and replacement of damaged components. Refurbishing defective components can be achieved by several means, including subtractive (grinding, machining, etc.) and additive processes (welding, spraying, etc.) [67–70]. One of those processes is the laser cladding (LC) process, which is a type of direct energy deposition (DED) process. This process works by continuously feeding a metallic material in the form of powder or wire into a molten metallic pool using a heat source generated by a high-power laser in a coaxial or preplaced powder system configuration [71]. This process has been adopted in many industries, including aerospace, oil and gas, and agriculture, for its unique characteristics and advantages over other additive processes in repair and surface modification. Among these advantages is the ability to

generate a highly focused and controlled heat source with minimal distortion to the mechanical and microstructure properties of the substrate. In addition, it also provides a high dimensional accuracy, a high depositing rate, and excellent metallurgical bonding to the substrate [72]. Thus, this research focuses on the automation of laser cladding-based repair systems.

The repairing step is one of the most crucial steps in restoring components. In this step, a damaged region on the EOL component is recognized and located for a repair task. The defective region is usually pre-processed utilizing subtractive processes to simplify the defective surface topology followed by an additive process, i.e., LC process [73]. Moreover, this process depends highly on a skilled worker for decision-making and planning. Thus, the larger component to repair, the more time and effort it requires. In addition, automation of such a process eliminates serval drawbacks, including higher processing times, production inconsistency, and a high operation cost [74].

The first and most important step in any repair process is to recognize defects and damages on the EOL components that need a rebuild. This step has been accomplished in two approaches: (1) nominal-damaged models' comparison-based method and (2) visual inspection-based method. Both methods attracted attention from academia and industry for their unique advantages in inspecting damaged components. Furthermore, the current progress on each approach is reviewed in the following paragraphs.

3.1.1 3D-based defect detection

In general, the current laser cladding-based repair process consists of five main steps to restore a damaged component to its nominal geometry, as shown in **Fig. 3.1**. This includes surface data acquisition of the entire damaged component, nominal model reconstruction from the acquired data, defective-to-nominal models registration followed by a repair patch extraction, and lastly a path planning generation of the obtained volumes [75–77]. The first step in this process is to obtain a digital model of the damaged component employing reverse engineering techniques represented by a collection of three-dimensional (3D) points. Optical laser-based range scanners and structured light scanners are widely adopted for this purpose [78]. Usually, the scanning process is performed manually by a skilled worker with consideration to the part size and shape complexity. This step raises challenges for large components, including significant scanning time and a large size of the acquired point cloud.

This can also lead to an inefficient and inaccurate reconstruction process of the damaged model due to the limited computational resources [79]. Next, the reconstructed nominal model, obtained in the previous step or provided with the damaged part, is aligned with the defective model using a point-to-point alignment algorithm [80].

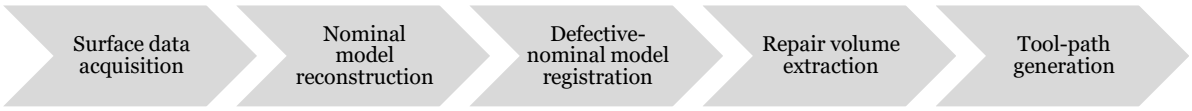


Fig. 3.1: Traditional repair process flowchart

Moreover, the speed of this registration process heavily depends on the availability of computational resources. Thus, a limitation is observed. With a successful registration process, a volume of the defective patch is extracted based on a boolean operation comparing the defective and nominal models. The last step in this approach is to generate tool-paths for the additive (i.e., laser cladding) process. Those generated tool-paths are highly dependable on previous steps. With the aforementioned challenges, this approach is still relying on the manual intervention of a skilled worker to achieve the required results.

In the point cloud-based defect detection approach, the digital model of the damaged component provides only spatial measurements (local) information with the lack of any high-level (global) information, such as geometric features [81]. This raises challenges for the automatic detection of defects on point clouds; thus, researchers have investigated this area since then. The current progress on detecting defects on point clouds has two directions: surface segmentation method and points comparison method. Segmentation is the process of grouping point clouds with similar properties. In general, point cloud segmentation methods are divided into five groups named edge-based method, region-based method, model fitting method, hybrid method, machine learning method, and graph-based method [82–84]. In the edge-based method, edges are defined when the change in a local property of point clouds, such as normal, gradients, and curvatures, surpasses a selected threshold [85]. While this approach has low computational complexity, it is sensitive to noise and may give inaccurate results in dense point clouds [86]. Region growing method groups neighboring points of similar characteristics into a group to obtain distinct regions of unique features [87,88]. Unlike the edge-based method, the region-based method is more robust with a noisy point cloud due to the integration of global information [89]. The third approach focuses on

grouping point clouds using a primitive surfaces fitting methods including Hough Transform and Random Sample Consensus [90,91]. This approach is suitable for simple shapes but might fail with complex point clouds. Some researchers leverage the advantages of different methods and combine them in a called hybrid method [92]. In the machine learning methods, a neural network is trained on predefined features of the point clouds for later extraction of the targeted features.

Scientists have developed two approaches in this category, including an indirect method (i.e., multi-view and voxel grid) and a direct method (i.e., multi-scale, feature fusion, etc.), each with its advantages [93]. The graph-based method generates a graph model of points and edges representing the relation between different points [94]. After the segmentation stage, the classification step is performed on the grouped regions of the point cloud to recognize defective and non-defective regions [82]. In the repair domain, many applications have been emerging that are based on the point cloud segmentation approach. Hitchcox et al. [95] proposed a method for surface repair in the aerospace industry that is based on the graph-based method. In addition, Chen et al. [96] proposed a framework for surface monitoring of the additive manufacturing process using the RANSAC method on point clouds. Recently, a framework by Wang et al. [97] was proposed to detect defects on point clouds of complex geometries. This approach uses machine learning models integrating support vector machine, spreading algorithm, and covariance matrix.

An alternative approach is a comparison of defective point clouds to the nominal point cloud for volume repair patch recognition and extraction. This approach is widely adopted in the repair and remanufacturing sectors. Wilson et al. [65] introduced a semi-automated method to repair turbine blades by extracting parametrized defective volume by computing the Boolean difference between defective and reconstructed models. In recent work, Li et al. [77] proposed an automatic method for extracting defective volumes in worn components using a modified iterative closest point (ICP) algorithm for the registration step and Boolean operation for the extraction step. Similarly, Xiao et al. [98] developed a method to repair engine blades by analyzing geometric information, including normal and spatial distance from the acquired point cloud to the nominal model. He et al. [99] recently worked on a two-step method for surface defect detection based on the octree method. The method initially registers the defective and nominal point clouds, then performs coarse and detailed defect detection.

3.1.2 2D-based defect detection

A recent approach for repairing damaged components based on image processing applications is emerging, shown in **Fig. 3.2**. This approach focuses on detecting and localizing damaged regions on the EOL components based on images instead of point clouds, which eliminates several steps in the traditional method [26,27,32]. Furthermore, this framework is based on a computer vision application that adopts deep learning networks to detect worn regions that require rebuilding. With the spatial location of the defective region, a localized 3D point cloud is obtained using standard scanning technology. Although this approach showed promising results in moving the repair process towards autonomy and reducing processing time, it still requires further investigation and improvement regarding accuracy and generalization to different types of defects.

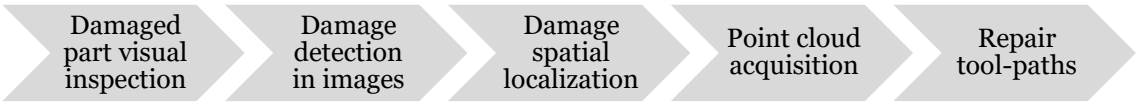


Fig. 3.2: Vision-base repair process flowchart

With the rapid development in the computer vision sector, the defect detection problem can be solved using two different approaches named traditional methods and deep learning methods. While traditional approaches are methods that are based on mathematical analysis of captured images for the detection process, prediction of defects using deep learning methods results from a gained knowledge of sample dataset. The typical defect detection methods are divided into four categories based on the feature extraction type, including color, shape, and texture [100]. The Color feature-based method extracts global color distributions on images for further analysis and detection of defects such as color histograms, moments, and coherence maps [101,102]. In shape feature-based method, the contour and region of the defect are transformed from a two-dimensional problem (2D shape) to a one-dimensional (description of feature), including Hough and Fourier transforms [103,104]. For the texture feature-based method, texture feature extraction methods are typically categorized into four methods: statistical, spectral, structural, and model-based [105,106]. In the statistical methods, the distribution of pixel intensities is computed and analyzed to detect defective pixels using different statistical methods such as pixel-threshold comparison, pixel clustering, edge detection, gray-level distribution, gray-

level spatial correlation, local binary pattern, and histogram statistics [107,108]. Unlike the direct processing method (in the pixel domain), indirect methods (in the transform domain) are also proposed, where they are generalized for intensity variations and less sensitive to noise. In general, spectral methods work on finding an optimal transform domain to separate defect pixels from the background for easy and accurate detection [105]. For the model-based methods, a projection of the texture distribution is obtained in the lower dimension by a special structure model such as the MRF model and FD model [109]. In practice, feature-based methods are combined for enhanced detection results.

With the popularity of artificial intelligence technology, machine learning-based methods gained a lot of attention from both academia and industry for their ability to generalize for diverse types of defects [19]. The focus of this method is to analyze and learn from sample data for further prediction of defects based on the gained knowledge. Machine learning defect detectors are usually classified into three groups: supervised learning models, unsupervised learning models, and reinforcement learning models according to the learning mode [100]. Furthermore, supervised learning models are deep learning neural networks that are trained on labeled image samples of the defects resulting in a reliable effect in detection. In contrast, unsupervised learning models use unlabeled image samples where the knowledge is gained by clustering similar image pixels using the extracted texture unrevealed information; however, those models are sensitive to noise. In addition, semi-supervised reinforcement learning models are simply the combination of both approaches, where a small number of labeled images are required. In recent years, convolution neural networks (CNNs), a type of supervised learning model, have gained a lot of attention for their robust performance on object detection and classification tasks, including defects detection [19].

Moreover, some researchers developed simple yet optimized defect detection pipelines based on CNNs as in [110–112]. Others have adopted well-known advanced models for detecting defects on two-dimensional images supported by transfer learning techniques gained in previous object detection similar tasks. Zhang et al. and Wang et al. [113,114] proposed defect detection and segmentation pipeline on turbine blades and rails using an improved two-stage model, namely Mask-RCNN, while [40,43] adopted the YOLO models series, a single-stage network, to achieve a real-time defect detection on steel surfaces of different types of defects.

After a deep review of the current progress in repair process methods, recognizing defective regions in components is a crucial step in the repair process. According to the literature, there are two different approaches for detecting and localizing defects on damaged components: the three-dimensional method (point cloud-based) and the two-dimensional (image-based) method. In the 3D approach, defective volumes are extracted using segmentation techniques providing accurate results based on acquired point clouds. As observed from the literature, surface segmentation still suffers from the misclassification of different defects. In addition, the 3D approach is not suitable for real-time applications due to the high computational complexity, especially for dense point clouds. In contrast, 2D defect detection methods showed the potential of real-time detection using 2D images. Moreover, deep learning models provide optimal surface damage detection results for diverse types and senses. The recent hybrid approach of combining both approaches using a vision sensor to detect defects on images followed by a 3D quantification of the defective region showed promising results in automating the repair process. However, a further investigation of the proposed framework is required for better accuracy and generalization to repair damaged components.

This thesis proposes an improved automatic defect detection, localization, and quantification method by integrating a camera and proximity sensors. This method uses a 2D camera to inspect the damaged component for detecting and localizing the defects, followed by a 3D scan of the defects using the laser proximity sensor. Furthermore, an overview of the current method for defect inspection is provided in this study. In addition, a robotic cell configuration used in this work is presented below, followed by a detailed explanation of the proposed method. Moreover, a virtual robotic cell is developed to verify the proposed method in CoppeliaSim software with two case studies. Finally, the results are analyzed and discussed, and future work is proposed.

3.2 Automatic Damage Detection, Localization, and Quantification Method

3.2.1 Overall proposed framework

This thesis proposes a method for the automatic detection, localization, and quantification of defects on damaged cylindrical components for an autonomous repair process. This method has four sequential steps shown in **Fig. *3.3***. A visual inspection of the damaged workpiece is

performed by capturing a series of images of the defective part using a 2D camera (frame camera) in the form of videos or images. The inspection process is achieved following the generated tool-paths based on the workpiece data, including part length and extreme diameter. In addition, the angular position of the workpiece is also recorded during the inspection stage for further accurate mapping of the workpiece's surface. Next, a single image of the part's surface is generated using pixels stitching algorithm. In addition to the single image, a pixel-spatial map of the unfolded image is computed based on the pin-hole camera model. Using the unfolded image, defects are detected using a deep learning pre-trained network resulting in a bounding box of the defective region. With this, a pixel location of the defective area on an image is obtained. Furthermore, a cylindrical spatial position can be acquired from the pixel-spatial map obtained in the previous step. In the last step of this method, a time-of-flight (ToF) sensor is used to scan the defective region on the workpiece based on the automatically generated tool-paths.

This method focuses on improving the time efficiency of the current repair process through the automation of defect localization on damaged components. Compared to the traditional method, the hybrid approach enables a real-time repair process by eliminating high computational complexity steps, i.e., point cloud processing, using 2D image processing techniques. In addition, this approach scans only defective regions on a component instead of a full scan or a manual inspection, significantly decreasing non-productive time. Moreover, the hybrid approach surpasses the traditional repair process in speed, cost, and efficiency with reliable and accurate results as the traditional approach.

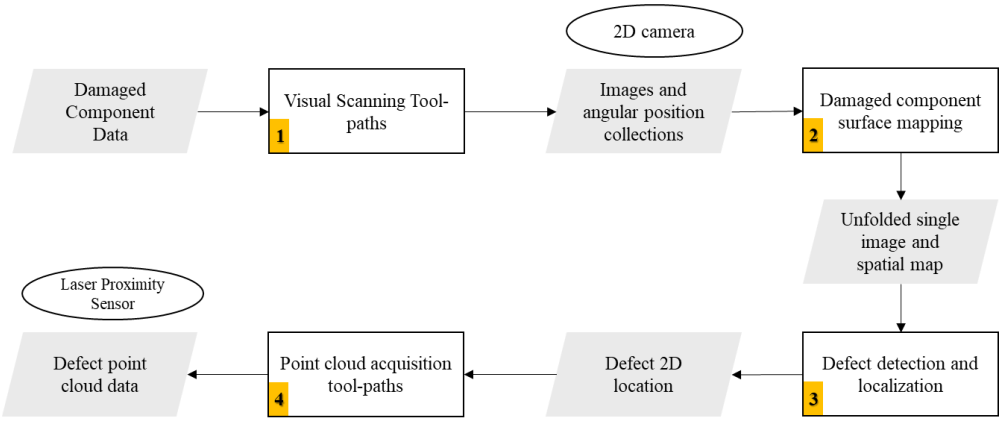


Fig. 3.3: Proposed framework for detecting, localizing, and scanning defects on damaged components.

The system configuration of the semi-autonomous repair cell is shown in **Fig. 3.4**. This system consists of a two degrees of freedom (DOFs) positioner and a 6-DOFs robot arm. While the positioner is configured as the leader group, the follower group is the robot arm with the laser head system attached to it as an end-effector. Both sensors (vision and ToF) are fixed to the laser head frame resulting in an eye-in-hand configuration. It is important to position the ToF sensor as close as possible to the tool center point (TCP) to avoid collision of the laser head and workpiece when planning. Similar to the vision sensor, placing it close to the TOF sensor is recommended to minimize the reproject errors in the localization process. Maintaining a similar orientation of sensors' frames with the TCP will enable simple transformation matrices. This configuration is developed to mainly process revolved components since those types of parts are widely used across many industries, including oil and gas and agriculture. Thus, this study will focus on automating the localization step in the repair process of damaged cylindrical components.

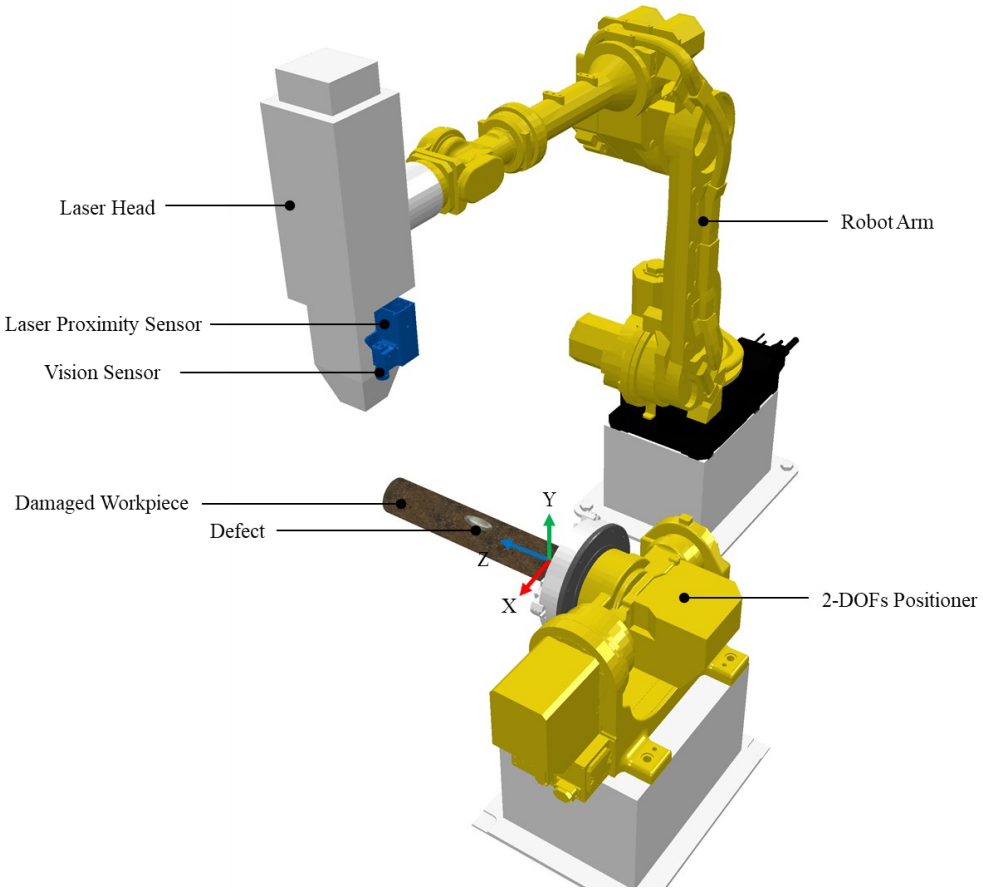


Fig. 3.4: Autonomous laser cladding repair system configuration

The following sections explain each step in the proposed method shown in **Fig. 3.3**. Following the same order, inspection tool paths are first discussed, followed by a mapping process and detection and localization step, and closing with the point cloud acquisition tool-paths.

3.2.2 Visual scanning

With the workpiece loaded into the autonomous repair system, the first step of the proposed method is inspecting the EOL component using a vision sensor. This step requires path planning for collecting a series of images for the next step, which is generating a single unfolded image. In practice, repair tasks usually have small distinct (i.e., different sizes and lengths) batches compared to manufacturing tasks that require new paths for inspecting the components. Thus, automatic image-capturing path planning is essential for a fully automated repair process. With the current configuration of the repair system, the visual scanning paths are not trivial to generate since the reference frame of the leader is in a dynamic mode. This means a movement of the workpiece will cause a motion to the laser head. Therefore, a method is developed to generate optimal paths for the eye-in-hand system to ensure an accurate and efficient video recording.

This method plans for a widely adopted configuration of the robot arm and positioner when inspecting, shown in **Fig. 3.4**. In this configuration, the robot arm maintains a vertical pose, whereas the positioner holds a horizontal orientation. This configuration simplifies the localization method and produces predictable paths. Usually, the camera field of view (FoV) is smaller than the damaged workpiece length; thus, the workpiece is scanned in multiple sections to cover its full length. The number of required scans mainly depends on the workpiece's length and the camera's FoV, which is computed using **(3.1)**. In addition, this equation usually results in a fraction that is rounded up to result in an integer number of scans. While the part's length is provided by the user, the camera FoV is obtained from the camera calibration process discussed in section **3.2.3**.

$$Number\ of\ Scans\ (NS)\ =\ Workpiece\ Length\ (L)\ /\ Field\ of\ View\ (FoV) \qquad \textbf{(3.1)}$$

A detailed pseudocode of the developed algorithm is presented in **Algorithm 3.1**. For each scan, the algorithm returns the target poses of the robot arm and the positioner with a recording trigger for the vision device to start and stop video recording. Each target pose consists of the position and orientation of the TCP. For the position of the TCP with the

positioner frame as a reference frame, computation is conducted in cylindrical 3D space, then converted to the cartesian coordinate system before sending it to the robot, as shown in **Table 3.1**. This simplifies the calculation of the pose angles based on the unit circle, with processing components in this system usually revolving workpieces. The scanning process is performed at a fixed standoff distance from the workpiece named scanning radius. Illustrated in **Fig. *3.5***, the scanning radius equals the workpiece radius plus the camera standoff distance minus the fixed distance from TCP to the camera center ($\mathbf{TCP}_{v}^{Vision}$). The reason for this is to maintain accurate localization results that match the calibration process. Furthermore, a full rotation of the workpiece is required to cover the entire surface of the damaged workpiece. Scanning targets are computed at a fixed interval of the angular position of the workpiece. Furthermore, the longitudinal position of the target (i.e., z) is computed using the equation in **Table 3.1**, which means the robot arm's longitudinal position mainly depends on the amount of workpiece length covered per each scan.

Algorithm 3.1 Defective component visual scanning tool-paths generation

- **INPUTs:** Workpiece Length (L), Diameter (D), Field of View (FoV), Camera Standoff Distance (CSOD)
- **OUTPUTs:** Robot path targets and image-capturing trigger

Compute the number of scans **NS**
Compute scanning standoff distance $\mathbf{r_{tcp}}$ using **Table 3.1**
for NS **do**:
  Compute longitudinal position **z** using **Table 3.1**
  Set trigger = ON > Video recording trigger.
  for $\theta = 0, 2\,\pi$ **do**
    Construct target position in cylindrical coordinate $[r_{tcp}, \theta, z]$
    Compute robot arm orientation **W** = $\pi/2$, **P** = 0, **R** = θ
    Compute positioner joints configuration **J1** = $\pi/2$, **J2** = $\pi/2 - \theta$
    Convert target position to cartesian $[x, y, z]$
    target pose = [x, y, z, W, P, R, J1, J2] > with respect to reference frame.
  end for
  Set trigger = OFF
end for

Table 3.1 Construction and conversion of the scanning target position

Cylindrical Coordinate	**Cartesian Coordinate**
$\mathbf{r_{tcp}} = \mathbf{r_{part}} + \textbf{Camera Standoff Distance} - \textbf{TCP}_{v}^{Vision}\ \textbf{Distance}$	$x = r_{tcp}\cos(\theta)$
$\boldsymbol{\theta} = \textbf{Target Angular Position}$	$y = r_{tcp}\sin(\theta)$
$\mathbf{z} = \mathbf{FoV/2} + (\mathbf{FoV} \times \textbf{Scan Cycle Number}) + \textbf{TCP}_{h}^{Vision}\ \textbf{Distance}$	$z = z$

With this, the position of the TCP is defined and computed. Next, the orientation of the TCP is calculated using Euler's convention (roll, pitch, and yaw). This step aims to maintain a vertical pose of the tooling as well as the vision system of the workpiece. Practically, the reference frame of the scanning paths is at the center of the positioner table shown in **Fig. *3.5***, which requires specific handling of the TCP orientation as the reference frame is dynamic. The dynamic reference frame handling method is illustrated in **Fig. *3.6***. Using the target position in the cylindrical coordinate system, TCP orientation angles as well as positioner configuration joints, are computed for each target. For instance, the orientation (roll angle) of the targeted position is calculated as the angle from the positive X-axis to the targeted position, donated as θ_{tcp}. In addition, the required positioner rotation to the targeted position is the angle from the positive Y-axis to the targeted position, donated as θ_p. However, positioner should be rotated counter-clockwise by θ_p for the robot to reach the correct position and orientation.

Furthermore, the positioner tilt angle (rotation around the first axis) is positioned horizontally at 90 degrees with the longitudinal axis of the workpiece, as shown in **Fig. 3.4**. This is implemented by applying a fixed angle to the first joint (J1) of the positioner which equals 90 degrees. The second axis (J2) of the positioner is given the value of θ_p. Furthermore, the pitch angle (p) is selected to equal zero which ensure that the camara frame and the workpiece frame are parallel, as a result reducing the complexity of the localization process. In addition, the yaw angle (rotation around the x-axis) is picked to be parallel to the positioner face, equal to 90 degrees. With this, a relation between the robot TCP and a dynamic reference frame (positioner frame) is established and tracked effectively.

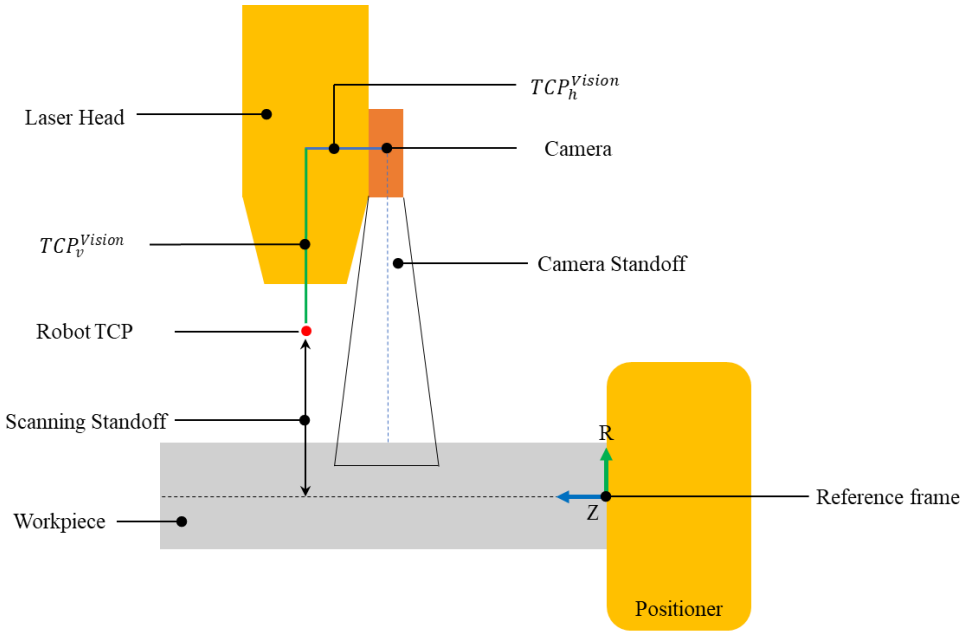


Fig. 3.5: Scanning Standoff distance illustration

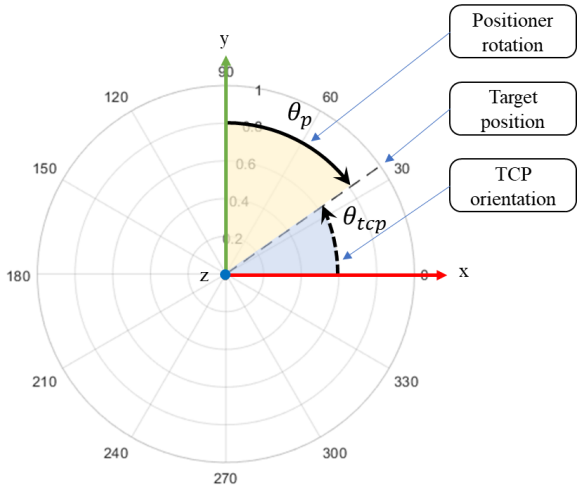


Fig. 3.6: TCP orientation handling illustration

A proper rotational speed of the workpiece for the scanning process is important. A fast rotational speed may result in skipping some areas of the damaged part when scanning. In contrast, slower speeds can result in dense acquired image data and resulting an inefficient scanning time. Thus, an optimal scanning rotational speed needs to be selected. Using equation **(3.2)** and in short form **(3.3)**, the optimal scanning speed (in rad/s) is calculated by the multiplication of the camera capturing speed (fps) and the covered arc length of the

workpiece surface per frame (lpf) divided by averaged workpiece radius (r). In addition, an angular position recording time step is calculated using the inverse of the fps of a camera for further usage in the next step.

$$\omega = 2\pi / t, \quad t = frames / fps, \quad frames = 2\pi r / lpf \tag{3.2}$$

$$\omega \approx \frac{fps \times lpf}{r} \tag{3.3}$$

3.2.3 Surface mapping

After obtaining a series of images of the workpiece's surface, an automatic pixel stitching method is developed to construct an unfolded single image of the workpiece surface from the collected images in the previous step with a pixel-to-spatial map. This map enables localizing spatial points from pixels on an image, which will be used later to localize boundary box corners of the defect in 3D cartesian space using pixels obtained in the defect detection step. Obtaining a single image of the damaged component surface enables the detection of multiple defects in single inference.

In addition, defects might cover a larger area of the component's surface that exceeds the FoV of the vision sensor; as a result, a single image view supports a wide defective area detection. This method integrates the pixel domain and spatial domain based on the camera projection model illustrated in **Fig. *3.7***. This model is based on internal and external camera parameters, where a camera calibration process is necessary to those parameters and construct a true and accurate projection matrix. In this section, a camera model is discussed as well as the calibration process of the area camera. An image-spatial surface mapping method is explored in detail based on the camera model.

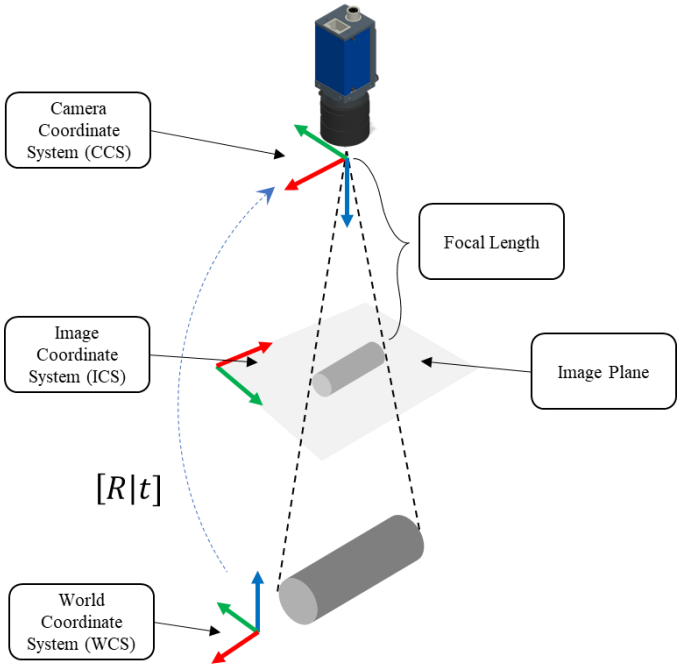


Fig. 3.7: Pinhole Camera Model

3.2.3.1 Camera calibration

The area camera can be modeled using the pinhole camera mathematical model in **(3.4)**, a well-established model for those types of cameras. This model maps a 3D world point P_w to pixels point P_i in the image plane. The pinhole model consists of two matrices, namely the intrinsic matrix (affine transformation) and the extrinsic matrix (rigid transformation). The intrinsic matrix is shown in **(3.5)**, which represents the camera's internal parameters, including the focal length and the optical center position in the image plane. Assuming a non-squared pixel shape, this is described in pixels by the two components f_x and f_y in the intrinsic matrix. Those represents the change of unit on the two axes of the image plane. In addition, an offset in the actual optical center position of the image center with the camera center is represented by o_x and o_y, which equals to center of image plane in an ideal alignment condition. Furthermore, a skewness in the image plane is included in the model using the S parameter that is when the angle between the two axes of the image plane is not exactly perpendicular.

$$P_i = \overbrace{K}^{\text{Intrinsic Matrix}} \quad \overbrace{[R \mid t]}^{\text{Extrinsic Matrix}} \quad P_w \qquad \textbf{(3.4)}$$

$$K = \begin{bmatrix} f_x & S & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \textbf{(3.5)}$$

$$f_x = F/p_x \quad , \quad f_y = F/p_y \quad , \quad s = f_x \tan\alpha$$

Where F is the focal length in the unit world, p_x , p_y are the pixel size in the unit world, and α is the angle of skewness. In addition, this model assumed an ideal camera without any distortion; however, in practice, distortions in the image exist from the lenses, which are not accounted for in the pinhole camera model. This is due to manufacturing defects and handling conditions. There are two types of distortions, including tangential and radial distortions. Tangential distortion exists in an image when the lens is not exactly parallel to the image sensor. The other distortion is radial distortion, which occurs due to manufacturing errors in the lens's shape. Those distortions are represented in **(3.6)**.

$$\begin{aligned} x_{rd} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ y_{rd} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ x_{td} &= x + [2p_1 xy + p_2 (r^2 + 2x^2)] \\ y_{td} &= y + [2p_2 xy + p_1 (r^2 + 2y^2)] \end{aligned} \qquad \textbf{(3.6)}$$

Where k_1, k_2, k_3 are the radial distortion coefficients, and p_1, p_2 are the transnational distortion coefficients that are obtained using the camera calibration process. Obtaining those parameters is achieved using a calibration process of the camera. There are several open-source methods for calibration area cameras, including the OpenCV library [115]. This process uses a known calibration pattern (e.g., square, circular, or special patterns) to obtain those parameters based on minimizing the errors of reprojection of specific points on the calibration pattern. With this, an accurate intrinsic matrix is obtained for the localization step.

3.2.3.2 Eye-in-hand calibration

The extrinsic matrix represents a rigid transformation matrix of a 3D point between the camera frame and the word frame using rotation and translation shown **(3.7)** following the XYZ (roll-pitch-yaw) sequence. This transformation represents the pose of the camera center in space with a reference frame, which is usually the TCP since the camera is attached to the robot arm. Furthermore, this transformation can be estimated manually by measuring the camera center position to the TCP or a known frame at a fixed position. However, this

process might affect the localization step by producing inaccurate results since the center point of the camera is an internal dimension that is hard to measure. An alternative method is an estimation of this transformation matrix using an eye-in-hand calibration process illustrated in **Fig. 3.8**, which can result in a more accurate estimation of the camera pose. This process is accomplished by registering the TCP pose and camera center at several locations and solving equation **(3.8)**.

$$R = R_z(\gamma)R_y(\beta)R_x(\alpha)$$
$$= \begin{bmatrix} cos(\gamma) & -sin(\gamma) & 0 \\ sin(\gamma) & cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos(\beta) & 0 & sin(\beta) \\ 0 & 1 & 0 \\ -sin(\beta) & 0 & cos(\beta) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\alpha) & -sin(\alpha) \\ 0 & sin(\alpha) & cos(\alpha) \end{bmatrix} \quad \textbf{(3.7)}$$
$$t = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

$$AX = XB$$
$$A = A_i\, A_j^{-1} \quad , \quad B = B_i\, B_j^{-1} \quad \textbf{(3.8)}$$

Where A and B are the transformation matrices of the camera center and the robot end-effector respectively. The X matrix is the transformation of the camera center to the robot end-effector. Lastly, i and j are the first the second movement of the robot arm.

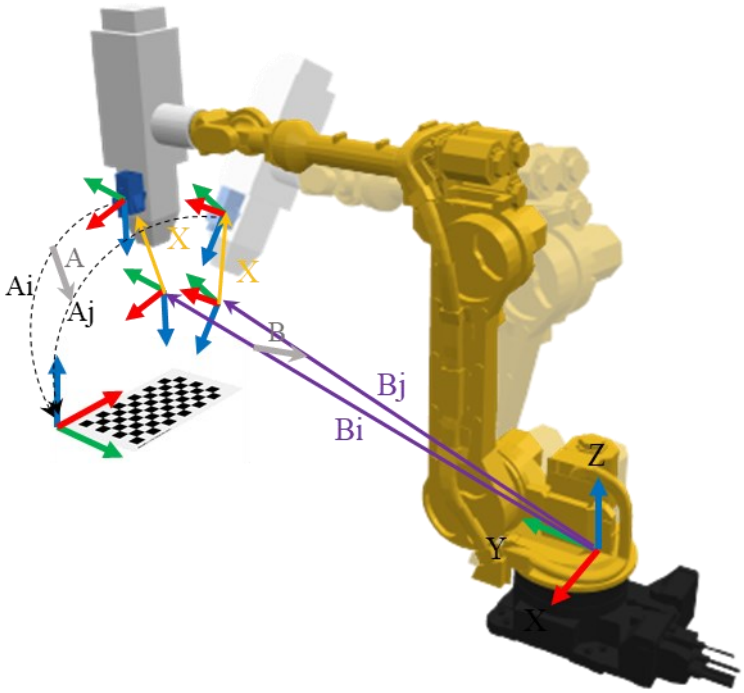


Fig. 3.8: Eye-in-hand calibration process illustration

3.2.3.3 Image-spatial surface mapping

In this step, a view of the unfolded workpiece's surface is obtained from the collected series of images in the inspection step. This process is called the pixel stitching process, where pixels or images are concatenated to form a single image. The purpose of this step is to emulate the line-scan camera type, which is widely adopted in the inspection sector. Although many stitching algorithms are proposed in the literature, those techniques focus on producing an optimal stitched image with minimal defects and artifacts, not considering true spatial reprojection. In addition, Schlagenhauf et al. [116] proposed a method for automatically stitching images of rotational components in the context of inspection based on image features. However, metal surfaces usually have fewer features to establish upon, which might cause poor performance of such an algorithm. Moreover, the focus of the developed algorithm is producing an acceptable stitched image with true and accurate spatial mapping.

An adaptive image stitching algorithm integrated with a spatial map is proposed in **Algorithm *3.2***. This algorithm constructs both an image of the damaged component to be

supplied to the defect detection step and a spatial map between the cylindrical component surface and the constructed image. The recorded video segments, positioner angular position, and camera center position of the scanning cycle are supplied to the algorithm. To stitch segments of videos together, each video segment is processed separately with a known recording camera position in space. The concept of the stitching process is extracting a middle line of pixels of each frame in the video segment to construct a single image of the part's circumference. The width of the scanning line depends on several factors, including capturing frame rate (fps) and the part's rotational speed (ω). Although, those factors are set to constant values, the positioner angular speed, in particular, exhibits some acceleration and deceleration due to the nature of the controlling system, e.g., damping system response, as shown in **Fig. *3.9***. Thus, an adaptive width of the scanning line is necessary to construct a true image of surface's part for an accurate defect detection process.

Based on the recorded angular position of the workpiece, the optimal scanning line width in pixels is first computed for each frame. Using the angular position, a change rate is calculated from the recorded data to obtain a vector representing the angular change rate for each frame. With the obtained angle, an arc length in the world unit is calculated using the workpiece radius $r\theta$. Mapping the arc length from the world unit to the pixel unit can be achieved using the camera calibration matrix. The linear mapping ratio from world unit to pixel unit is computed using equation **(3.9)**. Where Fx, fx are the focal length of the camera in world units and pixels, respectively, W is the arc length in world units, and w is the arc length in pixel units. Using this ratio, the estimated scanning line width is computed based on the angular position change rate. The obtained vector included float values that cannot be used to access pixels. Therefore, a cumulative approach is adopted to solve this issue. Stitching one video segment starts with a scanning line width of two pixels. However, the algorithm does not take the first frame directly. In fact, it tests and progressively collects the optimal scanning line width calculated previously for each frame. When the frame's scanning line-with or accumulated width is greater or equal to the optimal value, then pixels of this frame are captured using the optimal width; otherwise, the frame will be skipped for its overlap with another collected frame. Furthermore, the next even number is selected if the accumulated width exceeds the current optimal width. This means angular speed starts to increase, indicating more information is required per frame. Each image segment generated from video scans is concatenated vertically to obtain a single image of the workpiece.

$$\frac{Fx}{fx} = \frac{W}{w} \tag{3.9}$$

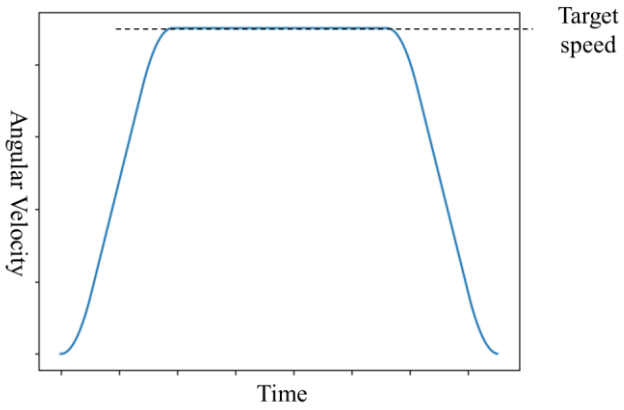


Fig. 3.9: Positioner angular velocity response

In the same loop, a spatial map is constructed of three values, including r, theta, and z. A 2D grid shown in **Fig. *3.10*** is developed where each point contains a 3D point. In this work, the radius is assumed fixed, and scanning is conducted on the workpiece's surface. For the angle, each frame is captured at a known angular position; thus angle/pixel ratio is calculated using **(3.9)** for a fixed radius. The longitudinal position of each pixel is computed using the camera calibration matrix developed in **(3.4)**. With this, an image-spatial map is developed. Overall, the adaptive stitching method is unique for further reduction in inspection time.

Algorithm *3.2* Integrated image-spatial mapping algorithm

- **INPUTs:** Video Segments (VS), Angular Position Vector (APV), Camera Center Position (CCP).
- **OUTPUTs:** Unfolded Surface Image (USI), Spatial Map (SM)

1 **for** VS **do:**

2 Extract video frames for the cycle.

3 Compute line scan width vector (**lsv**) and its optimal width (**max_width**), mid-frame (**mf**).

4 Construct **z** position vector using a camera matrix based on the CCP.

5 Initiate include array including all frames, $lw = 2$

6 **for** frame **do**:

if include[frame]:

  if line-scan-width[frame] or cumulative-width ≥ **lw**

    Extract line of pixels and concatenate to the local unfolded image map using **mid-frame** and **wl.**

    Compute angular position for each pixel on the line. θ

    Skip frames that overlap with the extracted line.

    Compute line width for the next iteration. $\forall\ lw\ \leq$ max_width

    Set cumulative-width = 0

  else:

    add line-scan-width[frame] to cumulative-width

end for

Concatenate scan cycle images to the global unfolded image map.

Construct a spatial global map of the scan cycle$[r, \theta, z]$.

end for

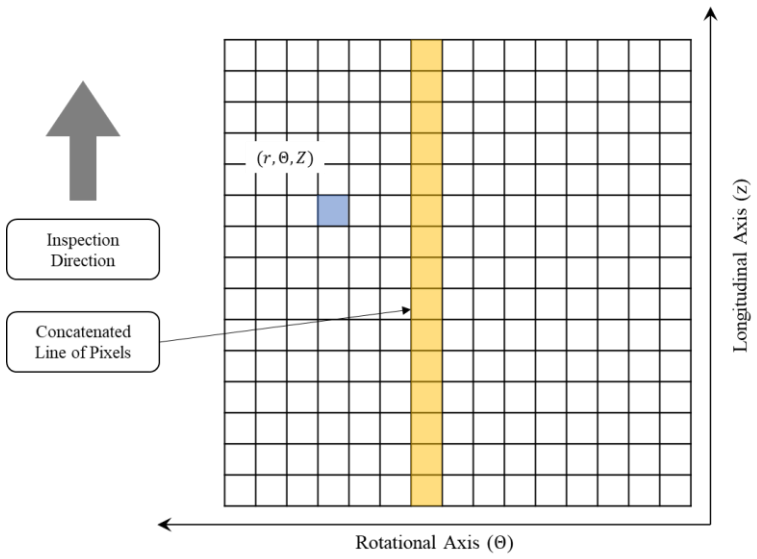


Fig. 3.10*:* Concatenated single image of the recorded video segments

3.2.4 Defect detection

The current trend of defect detection in images is based on deep learning approaches. This work follows the same approach for defecting defects on the image map of the workpiece. Several types of defects on metal surfaces are observed in the literature including cracks,

scratches, corrosion, pit, wear, etc. [106]. This study focuses on detecting wear-type defect since it is commonly repairable using additive processes. In addition, it is an external defect that does not extend to the internal layers of the component.

On the other hand, cracks, for instance, are difficult to repair using an additive process and might expand deeply in the damaged component. Furthermore, the current progress in metal defect detection tasks is adopting a well-established deep learning model developed for general object detection tasks into detecting defects on metal surfaces. This approach is called transfer learning, where a model is retrained on newly collected dataset images, leveraging the previously gained knowledge from other tasks. Based on this approach, this repair framework adopts the state-of-the-art model on the object detection task to be trained on the public metal defect dataset for recognition and localization of defective regions on the image map.

You Only Look Once (YOLO) version 7 is the state-of-the-art network for object detection tasks in terms of both speed and accuracy [44]. This architecture is a single-stage type object detector. The YOLO model series comprises three main components: the backbone, neck, and head. In the backbone, features of the image are extracted and passed to the neck. They are then combined and mixed in the neck stage to generate feature pyramids. Given those features, the head then predicts the location and class of each object in the image. The YOLOv7 model is the latest version of the YOLO detectors series that outperforms all real-time detectors models in speed and accuracy by introducing serval architectural improvements. YOLOv7 proposed an improved computational block in the backbone, Extended Efficient Layer Aggregation Network (ELAN), for enhanced learning ability of the model. In addition, a compound model scaling technique was introduced with YOLOv7 for better scaling of the model for different types of applications while maintaining the model's optimal structure. Moreover, serval trainable bag of freebies (BoF) is explored in the YOLOv7 paper, which are methods for increasing the model's performance without increasing the computational cost, including rep-parameterized convolution method and coarse for auxiliary and fine for lead loss technique.

In this study, the base model of the YOLOv7 is selected for its balance between accuracy and speed. The size of the selected version is shown in **Table 3.2** with its performance in the COCO dataset, which is a benchmark for object detection from Microsoft containing 80 classes. The model is small compared to previously developed models,

including Cascade Mask R-CNN and Swin Transformers [117]. In addition, the YOLOv7 model surpasses previously mentioned models by about 2% in accuracy and 551% in speed. The AP metric is one of the model performance evaluation gages, which is the measure of the averaged area under the precision-recall curve over all classes using a range of Intersection over the Union (IoU) threshold between 0.5:0.05:0.95. The base model of YOLOv7 achieved 51.4% mAP on the COCO dataset, which is currently the highest score on the object detection task.

Table *3.2* YOLOv7 base model parameters and performance on COCO

Model	#Param.	FLOPs	Size	FPS on V100	AP test	AP val
YOLOv7-base	36.9M	104.7G	640	161	51.4%	51.2%

There are several metal defect datasets available for the public. One of those datasets is the no-service rail surface defects (NRSDs) dataset, which contains images of damaged metal surfaces with different types of defects [55]. This dataset is selected for this work for several reasons, including RGB type images, different types of defects, higher resolution, higher defect/non-defect ratio, polygon annotation type, and similar texture to the actual damaged component. The NRSD dataset contains 3,793 images of 600x600 resolution with only one class but different types of defects, including natural and man-made defects. Samples of the annotated NRSD dataset are shown in **Fig. *3.11***. The dataset images are split randomly into three groups: 70% (2653) training set, 20% (758) validating set, and 10% (379) testing set. All images are resized to a standard size of 640 for width and height dimensions.

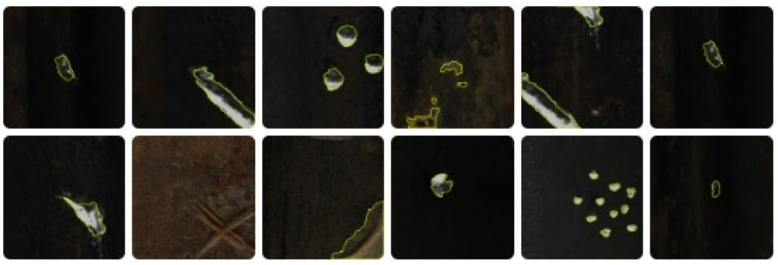


Fig. 3.11: NRSD Dataset Annotated Samples

The selected model is trained on the NRSD dataset with pre-trained weights of the COCO dataset for faster convergence of the training process. The training system has the following specification: AMD Ryzen 3970X 32C CPU, Nvidia RTX 3090 24GB GPU, 11.8 CUDA, 128 GB RAM, 1.10.1 PyTorch. The training process took about 180 minutes to

complete 100 epochs with the training parameters in **Table 3.3**. Moreover, the training metrics for both the bounding box and mask of the detected object are shown in **Fig. *3.12***. After 100 steps of training, the YOLOv7 achieved 83% and 69.6% in mAP for the bounding box and mask, respectively. This trend is predictable as the segmentation prediction is difficult since the model has to classify areas at the pixel level. The training curves in **Fig. 3.13** show a stable training process without over or under-fitting issues, which will produce a reliable prediction in the next steps.

Table *3.3* Model training parameters

Parameter	**Learning Rate**	**Momentum**	**Weight Decay**	**Batch Size**
Value	0.001	0.937	0.0005	24

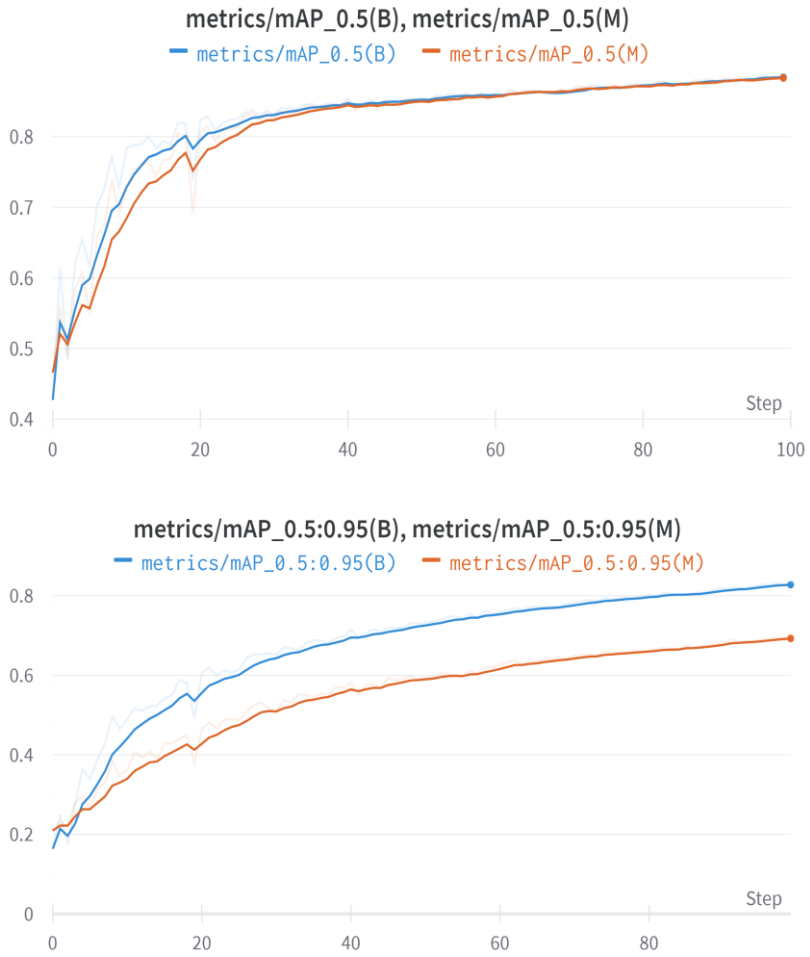


Fig. 3.12: YOLOv7-base model training process accuracy, B: box, M: mask

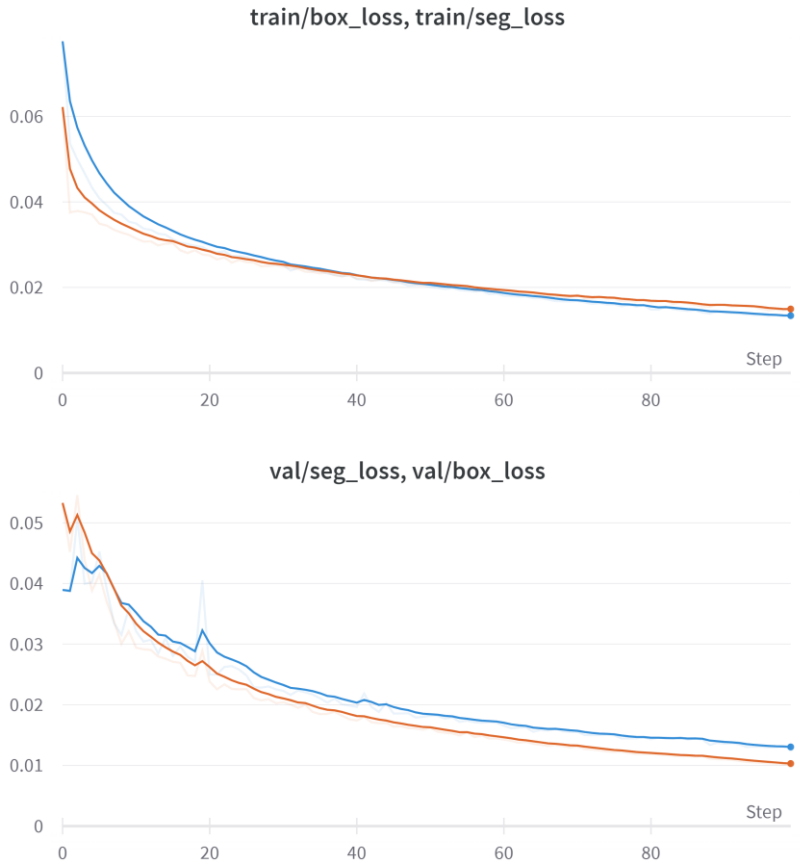


Fig. 3.13: YOLOv7-base model training process losses, B: box, M: mask

With this, the unfolded single image of the damaged component is passed along to the trained YOLOv7 model for detecting defects on it. The model returns bounding boxes containing four corners of the defective regions. Using the reconstructed spatial map, a location of the corners on the 3D cylindrical space is obtained and passed to the scanning process.

3.2.5 3D defect scanning path planning

As the location of the defective regions is recognized, topography information of the defective areas is essential for the next stage in the repair process. In this work, an automatic scanning path planning method is developed to acquire 3D points of the defective regions on revolving workpieces. The focus of this method is minimizing scanning time and robot traveling

distance. The zigzag path type is selected to achieve an optimal scanning process for the aforementioned factors.

This algorithm, shown in **Algorithm *3.3***, requires several inputs to generate the desired scanning paths, including bounding box corners of the defective regions, scanning step, and scanning resolution. Defect bounding box corners are obtained in the localization step in the cylindrical coordinate system. Those corners are unwrapped by converting angle (θ) to an arc length ($r\theta$) as the first step in this algorithm. With this step, the path planning problem is now in a 2D space, where the ($r\theta$) is the X axis, Z is the Y axis, and r is ignored since points are at the same radius. From the two axes extremes, a middle line segment is generated. Furthermore, points are sampled on the middle line based on the scanning step, which is the distance between two scanning trances. Sampled points are then projected on each bounding box edge based on segment intersections calculation. The projection vector direction is always perpendicular to the longitudinal axis of the workpiece. This is to ensure an optimal scanning motion by leveraging the positioner axis motion. At this stage, motion points are generated, and then the next step is creating a motion plan. By default, the lowest point in the two axes is selected as a starting point, as shown in **Fig. 3.14**. From the selected point, a search for a colinear point is conducted, that is, the point with an equal z value to the selected point. The colinear point means a path or a motion inside the bounding box of the defect which is considered a scanning path by triggering a scanning flag. This flag represents a scanning trace moving from the current point to the next point. On the other hand, where no colinear point is found, a k-d tree algorithm is executed to obtain the nearest point to the selected point in the 2D grid. This indicates a linking path where motion is a transition between two scanning traces. While the algorithm steps over each point, the processing points are extracted from the original list and concatenated into a new list for the algorithm to keep track of the processed points. The final step in this process is to remap the point's angle to the unwrapped space.

Algorithm *3.3* Zigzag scanning path planning algorithm

- **INPUTs:** Bounding Box Corners (BBCs), Scanning Step (SS), and Scanning Resolution (SR).
- **OUTPUTs:** Scanning paths, Scanning triggers.

1 Convert angle (θ) to arc length ($r\theta$) to unwrap angles.
2 Generate mid-line (ML) from the angle extremes (θ_{min} , θ_{max}) and longitudinal position extremes (Z_{min} , Z_{max}) parallel to the workpiece longitudinal axis (z).
3 Sample points on ML based on SS.
4 Project sampled points on bounding box edges perpendicular to the workpiece longitudinal axis (z).
5 Select (**SP**) the path planning starting point and extract it from the Projected points list.
6 **While** point in **PL:** > planning list = projected points list
7 Search for a colinear point with **SP** in **PL.**
8 **If** found colinear point:
9 Update **SP** with the new point.
10 Extract point from **PL**.
11 trigger=True > scanning trigger.
12 **else:**
13 Search for the nearest point in **PL**.
14 Update **SP** with the new selected point.
15 Extract point from **PL**.
16 trigger=False
17 concatenate planned points and their triggers.
18 **end while**
19 Wrap angles of the planned points.

An illustration of the developed algorithm is shown in **Fig. *3.14***. This algorithm was developed to generate scanning paths of polygon shapes; however, bounding boxes are important to this study for now. This method is simple yet effective for generating optimal scanning paths for an autonomous robotic cell. With this scanning and transition, paths are passed to the next step to compute the robot poses in a dynamic frame configuration using the method explained in section **3.2.2**. Furthermore, the robot program is generated and sent to the robot for acquiring points clouds of the defective regions.

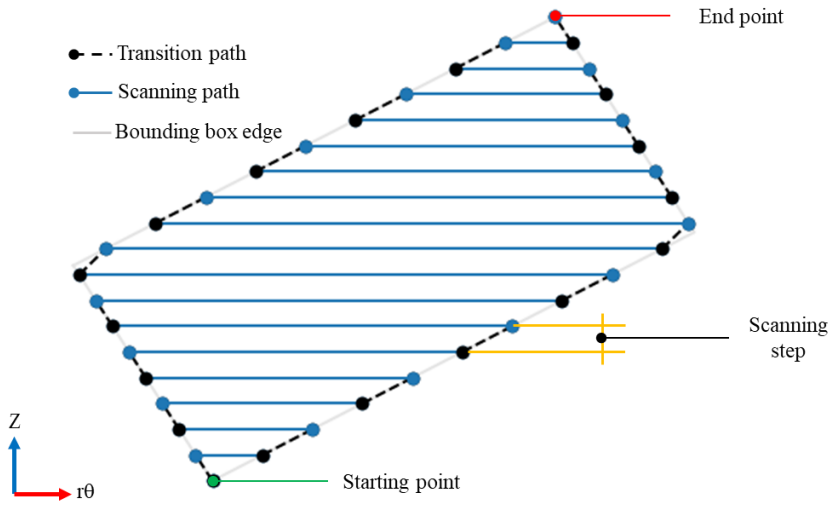


Fig. 3.14: Defect scanning path planning

3.3 Results and Discussion

A virtual robotic cell was designed in the CoppeliaSim environment as a replication of the actual system setup to perform a virtual experiment of this work before carrying out the actual experiment. The CoppeliaSim environment is capable of simulating both kinematic and dynamics. In addition, it has a built-in plug-in for various sensors and components, including vision sensors and proximity sensors. The development of the virtual setup is for several reasons, including the all-time availability of the virtual system for testing and debugging before the actual experimental verification of the framework. Another reason is the availability of various specimens for virtual experimenting that are similar to actual damaged workpieces. Furthermore, an actual experiment will be conducted in the future to verify the proposed methodology.

The virtual cell shown in **Fig. 3.4** has a 2000iA/80F Fanuc robot arm that has a mid-range payload and a 2-axis Fanuc positioner. In addition, the actual laser cladding head is attached to the robot arm with a simulated version of the DFK 33GP006 IMAGINGSOURCE camera and a Keyence IL-300 ToF laser sensor installed on the side of the laser head. The camera and the laser sensor are fixed using a 3D printer bracket in a static position. Furthermore, synthetic damaged workpieces were created with similar geometric and texture characteristics to the actual future workpieces. In this cell, the ToF laser sensor is simulated with a proximity sensor that has a range of 290 mm and a starting reading point at 160 mm.

In addition, the DFK camera is simulated using a perspective-type vision sensor with approximated 15° degrees FoV, 1280x720 resolution, and a working distance of 800 mm. Those specifications are estimated using the DFK33GP006 actual camera with a TCL-3520 lens. The angular field vertical FoV is estimated using **(3.10)**.

$$FoV = 2\tan^{-1}\left(\frac{sinsor\ size}{2\times\ focal\ length}\right) = 2\tan^{-1}\left(\frac{d_{img\_pix}\times d_{obj}}{2\times O_d\times d_{obj\_pix}}\right) \qquad \textbf{(3.10)}$$

Where sensor size and focal length are in world unit, d_{img_pix} is the vertical/horizontal size of the image in pixels, d_{obj} is the object dimension in world units, O_d is the object's working distance in world unit, and d_{obj_pix} is the object size in pixels. Those parameters can be estimated using a camera calibration pattern.

To set up the autonomous system a camera calibration process and an eye-in-hand calibration process is essential for the localization step of this method. While the camera calibration process is to find the intrinsic parameters, eye-in-hand calibration is to estimate the camera center pose to the tool reference frame. A square calibration pattern board was designed for the camera calibration process with a 7x12 square pattern of size 14 mm. Several pictures were taken of the calibration board using the virtual camera. Unfortunately, the CoppeliaSim simulation assumes an ideal camera lens without any distortion, as results skewness, radial, and tangential distortions are ignored for the virtual experiment. A sample of the collected images is shown in **Fig. 3.15**. In, addition, multiple robot poses are recorded for the eye-in-hand calibration process. Furthermore, the OpenCV library is used to obtain the intrinsic and extrinsic matrices reported in **(3.11)**. Where the z value in the extrinsic matrix depends on the scanning position, which is implemented in long_step variable.

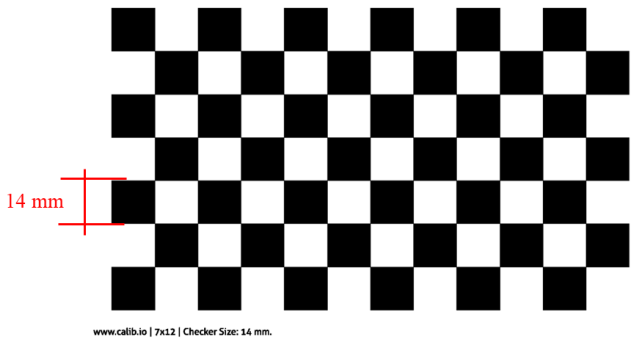


Fig. 3.15: A captured image of the calibration pattern using the virtual camera.

$$z\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 4914 & 0 & 380 \\ 0 & 4914 & 640 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 864 \\ 0 & 1 & 0 & 105 + long\_step \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad \textbf{(3.11)}$$

3.3.1 Case Studies Results and Analysis

With the system calibrated, the damaged workpiece is loaded into the system for defect inspection. The virtual damaged workpiece is designed in Blender software to integrate metallic texture on its surface based on the actual damaged samples shown in **Fig. 3.16**. A real metallic texture is used to further enhance this experiment. Two general case studies are generated to better generalize the proposed algorithm and investigate its sensitivity to different defects. In the first case study, a single wear defect is introduced on the surface of the part, as shown in **Fig. 3.17**. This type of defect represents a worn area or dents on the surface of the components, which is commonly observed in actual workpieces. The second case study contains groove-like damage on its surface. Both defects can be repaired using additive manufacturing processing, e.g., the laser cladding process. Some details of the two case studies are provided in **Table *3.4***. Different sized and lengths for the two cases are selected to observe the sensitivity of the developed algorithm. In addition, a measured location of the center of each defect is provided as a reference to evaluate the accuracy of the proposed method.

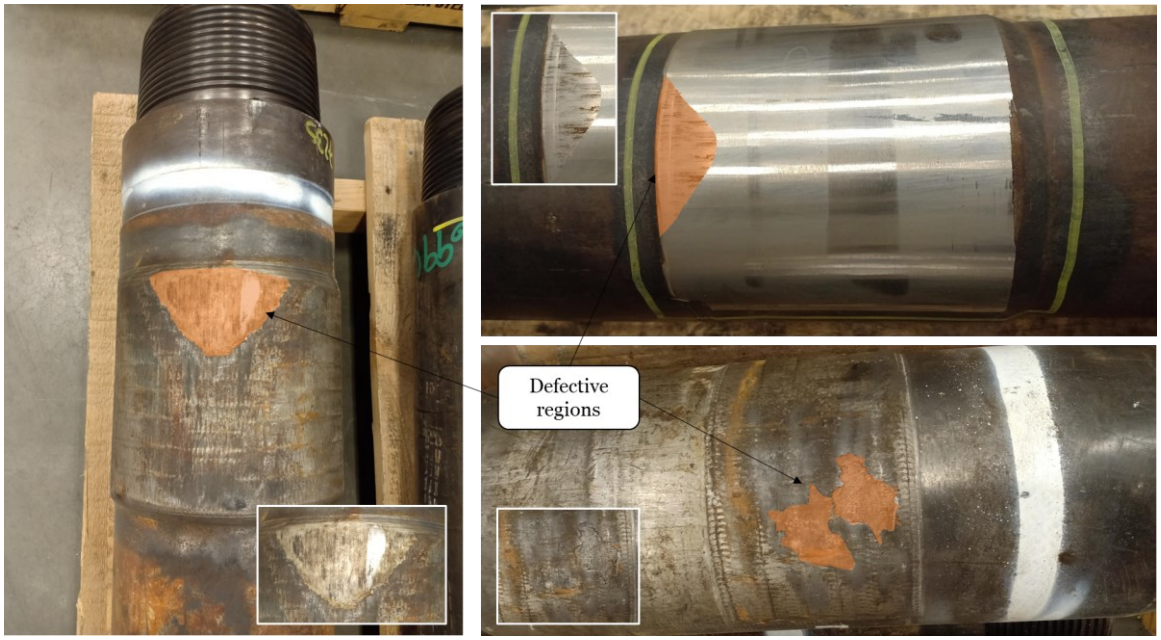
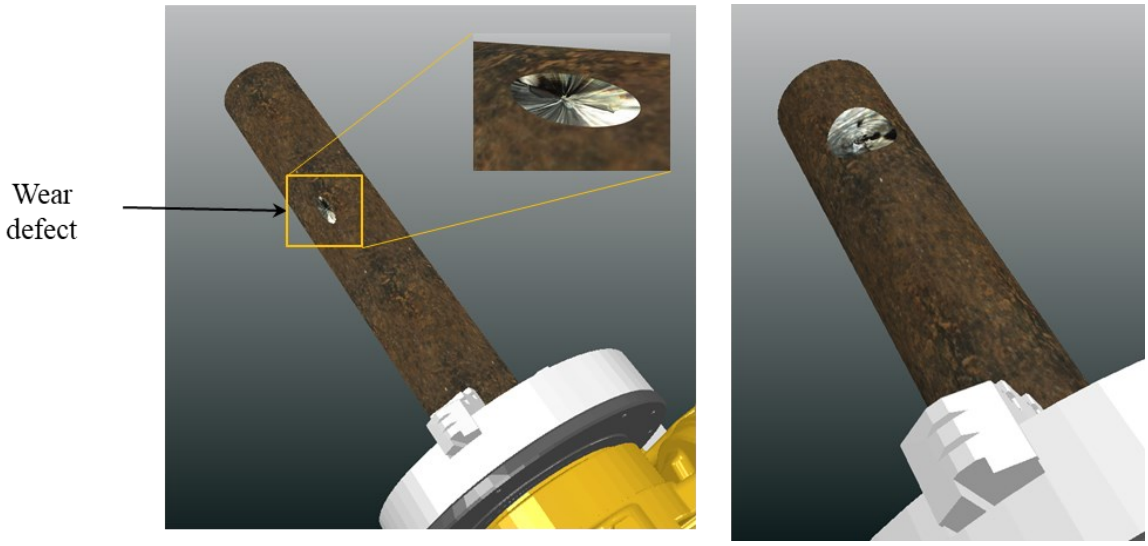


Fig. 3.16: Actual damaged samples.



Case study 1 Case study 2

Fig. 3.17: Damaged case studies.

After providing the workpiece information by the user, some values need to be computed before starting the inspection process, such as the number of scans per the workpiece length, longitudinal field of view, camera standoff distance, fps, and positioner rotational speed. For the first case study, the workpiece is divided into four cycles of scans, while case study 2 contains only three cycles. The sectioning calculation is based on approximately 210 mm longitudinal FoV using **(3.10)** for both samples. In addition, scanning standoff distance depends on the workpiece diameter, which is 654 mm for the first case and 642 mm for the second case. The video rate is set to 20 Hz for both runs. It is easier to visualize the targeted arc length per frame; thus, a maximum of 1.35 mm arc length per frame or 1.3 degrees per frame is captured to minimize and avoid any effect of the circular curvature on the map.

Table *3.4* Case studies information

Parameter	**Case study 1**	**Case study 2**
Length	700 mm	450 mm
Diameter	120 mm	95 mm
Number of defects	1	1

Center locations [r,θ,z]	[60, 235, 400]	[47.5, 31, 349]

The initialization of the cell is now completed, and inspection paths are generated with their recording triggers. In **Fig. 3.18**, samples of the dynamic motion poses of the robot TCP and the positioner angles are plotted for one cycle of the visual inspection process. The scanning pose of the robot with the positioner is shown in the right image of **Fig. 3.18**. As shown in the graph, the first axis of the positioner is fixed while the second axis is rotating one cycle in a unique trend. Opposite to this trend is the roll angle of the robot TCP to maintain a vertical pose of the robot to the workpiece. It is also important to note that the angle in CoppeliaSim is the range between $(-\pi, \pi)$ which is the reason for the sharp drop in the J2 curve moving from $-\pi$ to π. In addition, the yaw and pitch angles are not changing as expected in this configuration. This method results in optimal scanning paths where the robot and positioner are handled smoothly. In addition, an estimated scanning time per cycle is 30 seconds and 120 seconds in total for 4 scans. Whereas the second case study requires less time, about 19s per cycle and 3 cycles of scans in total.

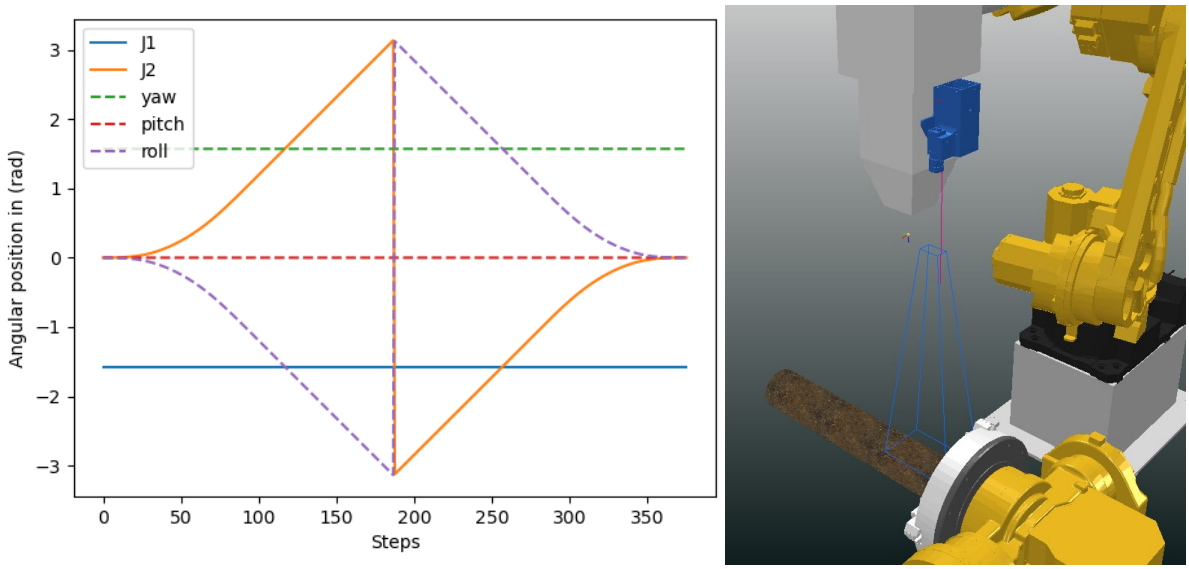
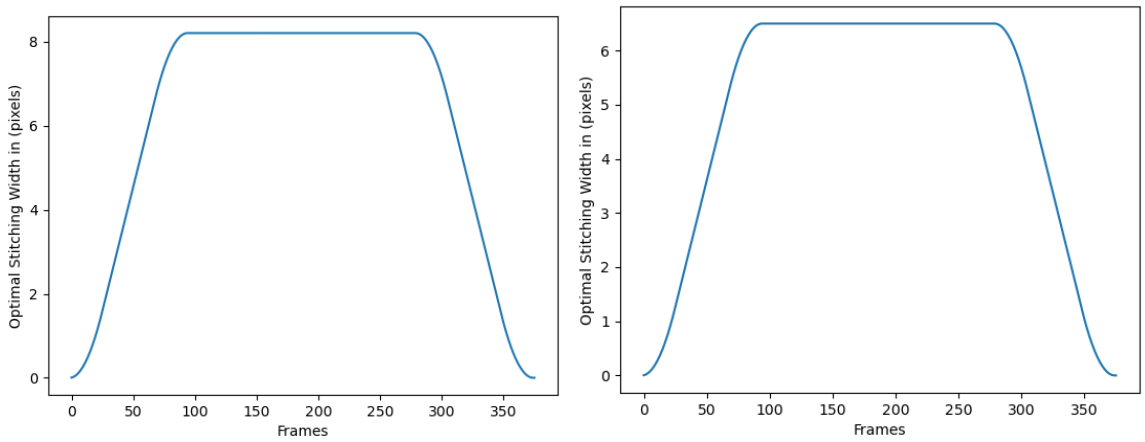


Fig. 3.18: Visual scanning paths for the first cycle of case study 1.

With the collected video segments, the next step is to stitch the extracted frames to obtain a single image mapping the workpiece's surface while retaining spatial information of each pixel. Initially, mapping the angular position change rate from world units to the pixel domain is essential for the adoptive stitching algorithm. The linear mapping ratio (LMR) is calculated using **(3.9)** and reported to be 152/25, which means a millimeter in the world

equals around 6 pixels in the image, which is constant for both workpieces. Then, the optimal scanning line width is calculated for each frame based on the arc length generated by the angular position change rate. The optimal stitching vector is plotted in **Fig. 3.19**, which illustrates the pattern of the controlling response system. As shown in the graph, around 1/3 of the frames are captured during the ascending and descending motion of the positioner, which highly depends on the system itself and might differ from the actual system. It is important to note that scanning line width can only be an even number thus, numbers are rounded to the nearest even number, which might cause the loss of a small portion of information in the image map. However, this issue can be addressed by selecting a resolution in **(3.3)** where it maps for an even number of pixels.

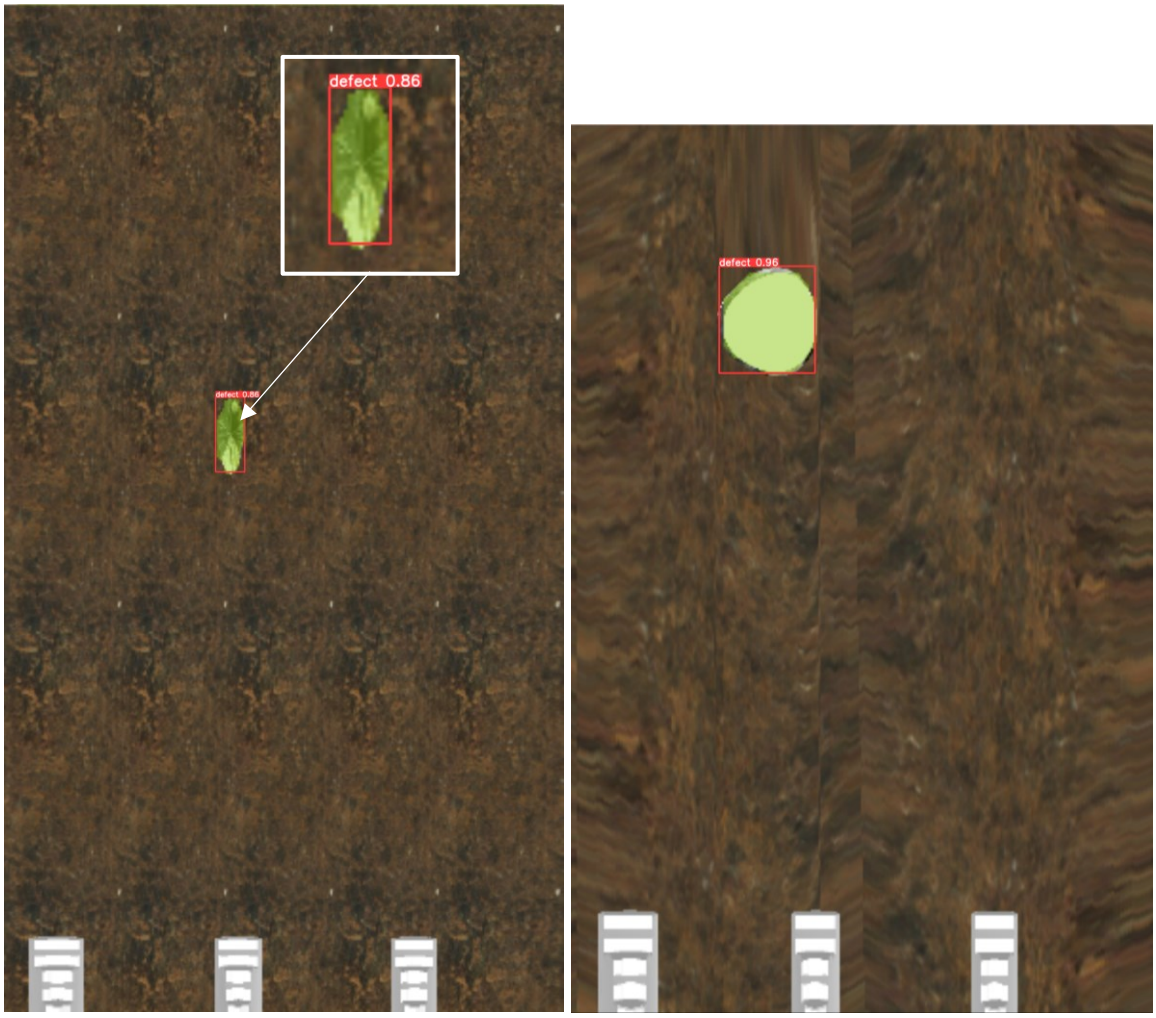


Part#1 Part#2

Fig. 3.19: Optimal targeted pixels of the scanning line width.

With the scanning line width vector calculated, the image-spatial map generation is carried out. Based on four scanning cycles for part 1, a UV map with a width of 2362 pixels and a height of 5120 pixels is generated out of the total scans shown in **Fig. 3.20**. For part 2, the map has a size of 1804 x 3840 in width and height respectively. Along with the pixel map, a spatial map is obtained based on the camera calibration matrix. The UV map is then passed along to the pre-trained YOLOv7 model for the prediction of defects. The YOLOv7 correctly detected the synthetic defects on both parts with an accuracy of 86%, and 96% on mAP for 1 and 2, respectively. The reported inference time is 1.265s for the first UV map, which is high compared to the original dataset that is due to the large size of the image map. For the second UV map, the model took 1.878s for the inference process, which is worse than for part 1.

Fortunately, the YOLOv7 model runs only once on the entire map, unlike the traditional method where deep learning models run on each frame of video. The prediction process returns defective region annotation and its bounding box.



Case 1 Case 2

Fig. 3.20: Generated stitched maps using the developed method.

The bounding box pixels and their spatial locations are listed in **Table *3.5*** and **Table *3.6***. The center point of the defective region is computed by averaging the four corners of the bounding box. In **Table *3.7***, points localized using the developed method are compared to the measured points. Furthermore, the root means square deviation is evaluated for the localized centers of defects to determine the performance of the tested method using **(3.12)**. The comparison of the localized points with the measured points showed a promising

localization result. The achieved mean accuracy of about 98% indicates the progress of this method in the 2D localization process. Although the method showed good results, it is important to increase the validation samples and to carry out an actual experimental to verify it is performance under world conditions.

Table *3.5* Case 1 detected defect results

Points#	Pixel domain (u, v)	Spatial domain (r, θ, z)
1	(885, 2513)	(60, 222.8, 428.6)
2	(885, 2845)	(60, 222.8, 374.1)
3	(1021, 2845)	(60, 244.6, 374.1)
4	(1021, 2513)	(60, 244.6, 428.6)
Center	(953, 2679)	(60, 233.7, 401.35)

Table *3.6* Case 2 detected defect results

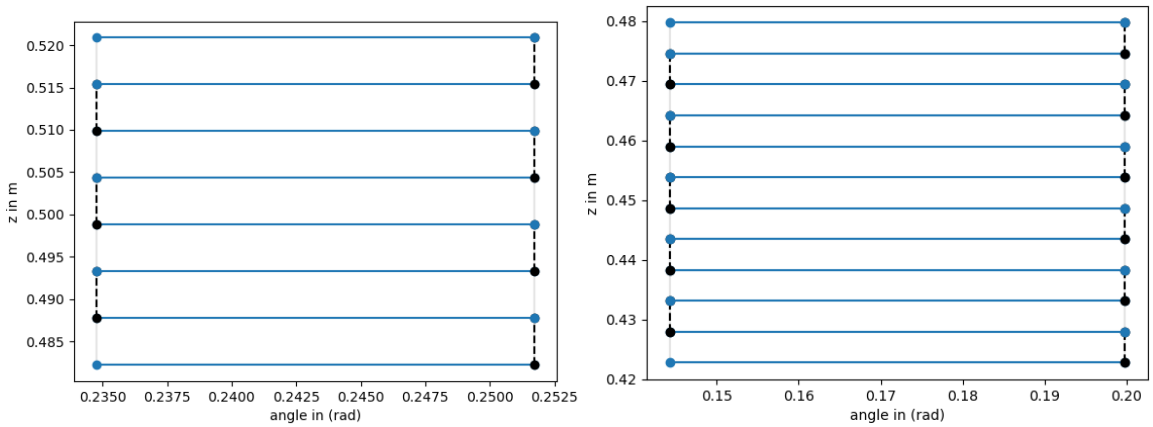
Points#	Pixel domain (u, v)	Spatial domain (r, θ, z)
1	(451, 1531)	(47.5, 173.9, 379.7)
2	(451, 1877)	(47.5, 173.9, 322.7)
3	(763, 1877)	(47.5, 241, 322.7)
4	(763, 1531)	(47.5, 241, 379.7
Center	(607, 1704)	(47.5, 207.45, 351.3)

Table *3.7* Measured and localized points

Points#	Measured points	Localized points	δ (mm)	r
1	(60, 235, 400)	(60, 233.7, 401.35)	1.933	0.478%
2	(43.6, 211, 349.9)	(47.5, 207.45, 351.3)	4.73	1.15%
Mean	-	-	3.3315	0.775%

$$\delta = \sqrt{(x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2}$$
$$r = \frac{\delta}{\sqrt{x^2 + y^2 + z^2}} \times 100\% \quad \textbf{(3.12)}$$

In the last step of the localization pipeline, 3D scanning paths of the predicted defective regions are generated automatically and executed to collect the points clouds. The four corners of each defective area are passed along to the automatic algorithm with a scanning pitch selected as 5 mm and point cloud resolution given as 0.005 degrees. A 2D view of the generated paths is given in **Fig. 3.21**, containing scanning and transition traces for case study 1. In addition, the acquired point clouds are visualized in the same figure. The number of scanning paths depends mainly on the size of the defective area. As shown in the figure below, more scanning traces are generated for the second case study since it is larger in size. The pattern of those scanning paths is to support the oscillating type motion to minimize unnecessary motion. The point cloud acquisition process took around 80-100s each. With this, topography information of the defective surface is acquired for further generation of repair paths of the laser cladding process.



Case 1 Case 2

Fig. 3.21: Scanning path planning process.

The proposed method has addressed several issues in the traditional approach and the recently developed methods. One is the generalization of the pipeline to process a wider range of workpieces and defects. Another is the development of a fully integrated framework for the autonomous repair system. Furthermore, a robust detection and localization approach for enhanced performance has been achieved. Lastly, a full run of this pipeline task is around 5-10 minutes which is about %50 reduction in time of the previously reported timing here [18] depending on several factors, including workpiece size, defect size, system speed, and scanning resolution.

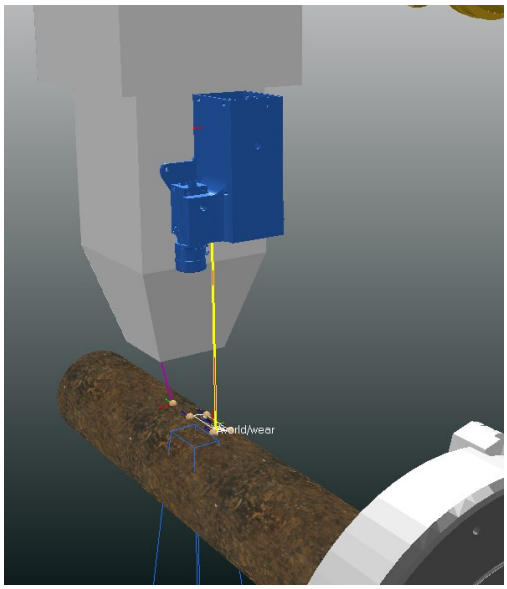


Fig. 3.22: ToF sensor acquiring point cloud using the generated paths.

Chapter 4
A path planning method for surface damage repair using a robot-assisted laser cladding process

4.1 Introduction

Energy and resource consumption in the manufacturing sector accounts for a significant portion of global consumption, making it a serious pollution source [118]. Thus, extending a product's life beyond its expectancy is important to reduce resources wastes. This means reusing, repairing, or refurbishing used products, which all fit under the umbrella of remanufacturing processes [119]. In general, remanufacturing process is a technique that restores a product's status from used to a new condition ensuring its functionality and quality [120]. The restoration of worn products has been proven to save energy, reduce emissions, and cost less than newly manufactured products [65]. Despite this, the repair process has been a challenging and crucial step in remanufacturing due to its dependency on a skilled worker or advanced robot system to process complex damaged components [121]. Moreover, various traditional metal processing techniques are utilized to restore damaged products, such as different types of conventional welding, thermal spray, and material deposition processes [72].

Single or multiple processes can achieve a repair and restoration process of a product. Many researchers have recently focused on integrating additive and subtractive processes, described as a hybrid approach, that leverages the benefits of both technologies as in [122]. Usually, the subtractive processes, including milling, turning, and grinding, are used as pre or post-processing steps to clean or manipulate the defective area as introduced in [75,77]. Furthermore, intelligent algorithms for an automatic suitable process selection and automatic feature extraction were developed in [123,124]. Since the additive process is the main process for repairing components, this work focuses on utilizing one of the additive manufacturing processes that is the laser cladding (LC) process, to automatically repair

damaged surfaces. Additionally, improved object modeling and optimization are achieved for the longer life of the restored component [125].

Laser Cladding is a type of metal deposition process, defined as a technique of feeding a continuous flow of a metal to a metallic molten pool of a substrate surface that is generated using a heat source of a laser beam [71] as illustrated in **Fig. *4.1***. In addition, deposited material can be pre or co-deposited into the substrate and formed in either wire or powder form, which is delivered in different configurations, including coaxial, and off-axial flow [126]. According to the literature, this technology has several applications, including wear resistance coating, welding, and metal additive manufacturing [127]. Further, this technology is reported to be of the most effective and reliable techniques for surface damage repair due to its ability to control the input heat, achieve high dimensional accuracy, and attain minimal surface properties modification [128]. Practically, this process is carried out using a robotic arm in an enclosed environment due to the high risk of exposure or direct contact with a high-power laser beam, fine metallic powder, and hot surfaces [129–131]. For those reasons, researchers have been focusing on automating the repairing process, which has proven to increase reliability and repeatability and decrease processing time and cost [67].

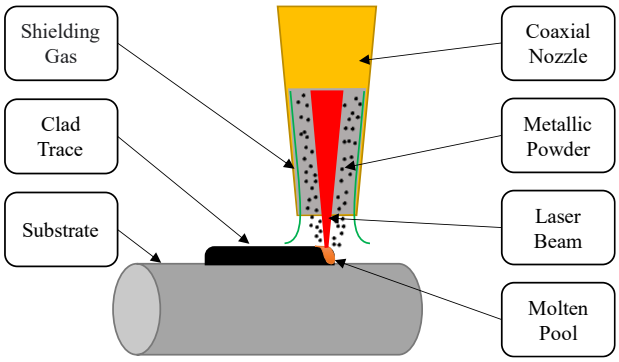


Fig. 4.1: *Laser Cladding Process*

A digital representation of the damaged surface topology is the base for restoring worn products, which is enabled by reverse engineering techniques [132]. One common approach for model reconstruction is to compare the nominal model with a three-dimensional (3D) scan of the damaged part to extract the repair volume, as presented in [133–136]. Registration and alignment of the damaged model with the nominal computer-aided design (CAD) model are necessary for an accurate repair volume extraction. Although this method is efficient and reliable, a nominal model is constantly required. However, Y. Zheng et al. [137] introduce a

Step-RANSNC method that reconstructs primitive shape models from point cloud data based on B-spline curves. This method eliminates the requirements for a nominal model, but it is still constrained to a full scan of the damaged part. Recently, with the advanced enhancements in computer vision and artificial intelligence technologies, the damaged area can be recognized and localized using a deep learning object detection method [26]. The intelligent approach will eliminate some of the current method limitations, including long scanning and processing times, large data, as well as a nominal model requirement. In addition, those intelligent strategies help towards collision-free and safe tool path planning [138–141].

LC processing paths significantly affect the repair quality, machine efficiency, and processing time [142]. There have been many efforts and studies to develop an optimal tool-path planning method to repair damaged surfaces. Imam et al. and Zheng et al. [143,144] proposed a method for direct tool-path generation from point cloud data. Imam's technique is based on grouping spatial points into different layers sorted by height. For each group of layers, 3D points are then connected to generate tool-paths. This was found to be an effective algorithm in reducing processing time, yet it requires an interpolation process for a different point cloud and path resolution. Another constraint of this method is that paths are generated in the same orientation of the point cloud scanning direction; thus, it lacks the flexibility to generate oriented paths. It is also limited to cylindrical components only. While Imam's method was developed for a layer-by-layer building approach, Zheng's method was designed to produce a single-layer path. In addition, paths were smoothed after slicing the point cloud data using a B-Spline curve, which is also found to improve the robot movement for an industrial static and mobile manipulation system [145].

Moreover, Wang et al. [146] proposed a method that follows the same approach as Imam's and Zheng's methods, with an extra step to compute processing point vectors. Those vectors were generated based on the fitted NURBS surface of the point cloud. Nevertheless, Wang's method was only capable of generating a single-layer path that provides insufficient topology surface information. Furthermore, Liu et al. proposed an algorithm in [147] for trajectory planning of curved surface cladding of a CAD model. Specified planes intersected with the CAD model to generate the tool-path. The same approach was discussed in [148] by Li et al. with a different laser cladding configuration. While this approach was designed and optimized to follow surface topology, i.e., coating, it was not developed for damage repair. A

recent method was proposed by [149] Flores et al. that achieved useful results in additive manufacturing using laser cladding. A major improvement of this method over others is the adoption of the non-planar adaptive layers; however, this method is aimed for CAD model building. From the literature review, the authors recognized the need for an optimal automatic path planning method for repairing surface damage using the laser cladding process from a local defective scan.

This paper proposes a method shown in **Fig. 4.2** to repair freeform damaged surfaces automatically using the laser cladding process. This procedure aims to generate optimal tool-paths for the cladding robot to repair a wide variety of damages. This repair method requires information about the surface topology, which can be accomplished using one of the reverse engineering techniques. In this work, the 3D points are acquired using a digital depth sensor that is based on the time-of-flight (ToF) principle. This ToF sensor is used to obtain spatial measurements of the defective surface of a component, instead of a full part scan, for lower scanning and processing times. The first step of this method is to rebuild the damaged surface after generating the freeform mesh of the collected point cloud data using the Delaunay algorithm. The rebuilding process is based on a unique freeform mesh slicing algorithm to generate mesh slices for layer-by-layer build paths to be computed in the second step. Then, a tool-path planning algorithm is adopted which is based on the intersection points of a plane (clipper) and mesh faces. This clipper is a vertical hyperplane that is oriented at the desired clads' angle with respect to the fixed axis. One feature of this method is the flexibility to produce angled traces of clads, which can be obtained by the clipper's orientation and layers' cross-hatching pattern. Normal vectors are also computed using a ratio calculation method in this step. The third step is to determine processing path points and their normal vectors through a linear interpolation process that is based on the maximum allowable laser defocusing range. Additionally, a custom tool-path processing method is adopted to generate the robot tool center point (TCP) orientations for optimal robot cell movement. Lastly, the generated laser paths were then sent to simulation software to generate a robot program, including the robot's position and attitude. In addition, a detailed description of the proposed method supported by several case studies was discussed in this paper. A summary of this work and future directions was mentioned in the conclusion section.

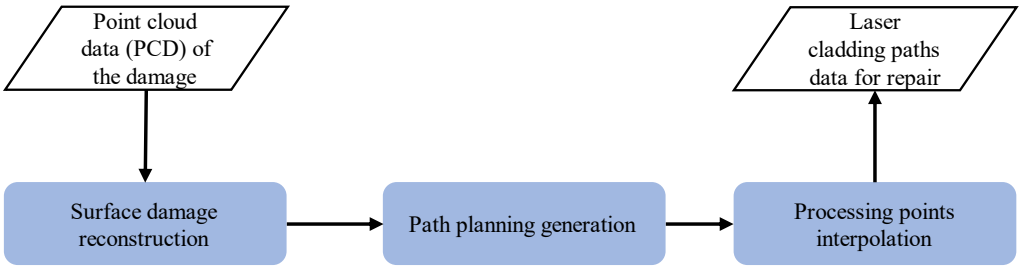


Fig. 4.2: Overall proposed damage path planning procedure

4.2 Experimental Setup: Laser Cladding System

The laser cladding system used in this work consists of two independent groups, as shown in **Fig. 4.3**. In this configuration, the leader group, which holds the workpiece, is changeable and has two different rotatory tables named headstock and positioner. The headstock has one degree of freedom (DOF) which is the rotation around its axis. Unlike the headstock, the positioner has two DOFs defined as rotations around two of its three axes. Both tables are configurated to operate with the Z-axis of the user frame (UF) pointing out of them as shown in **Fig. 4.3**. The main objective of using those two different tables is to process both cylindrical and spherical components and planer parts effectively. The robot arm, mounted on a moving rail, is utilized as the follower group, and attached to this arm is the laser cladding process equipment. These are laser cladding head as an end-of-arm tooling (EOAT), dual metallic powder and shielding gas feeding systems, and digital laser depth sensor. The depth sensor is installed on the laser head for convenient and automated spatial measurements acquisition. In addition, the rail with headstock adds the ability to process long parts. Ultimately, the follower group can be used with either the headstock or the positioner leader groups.

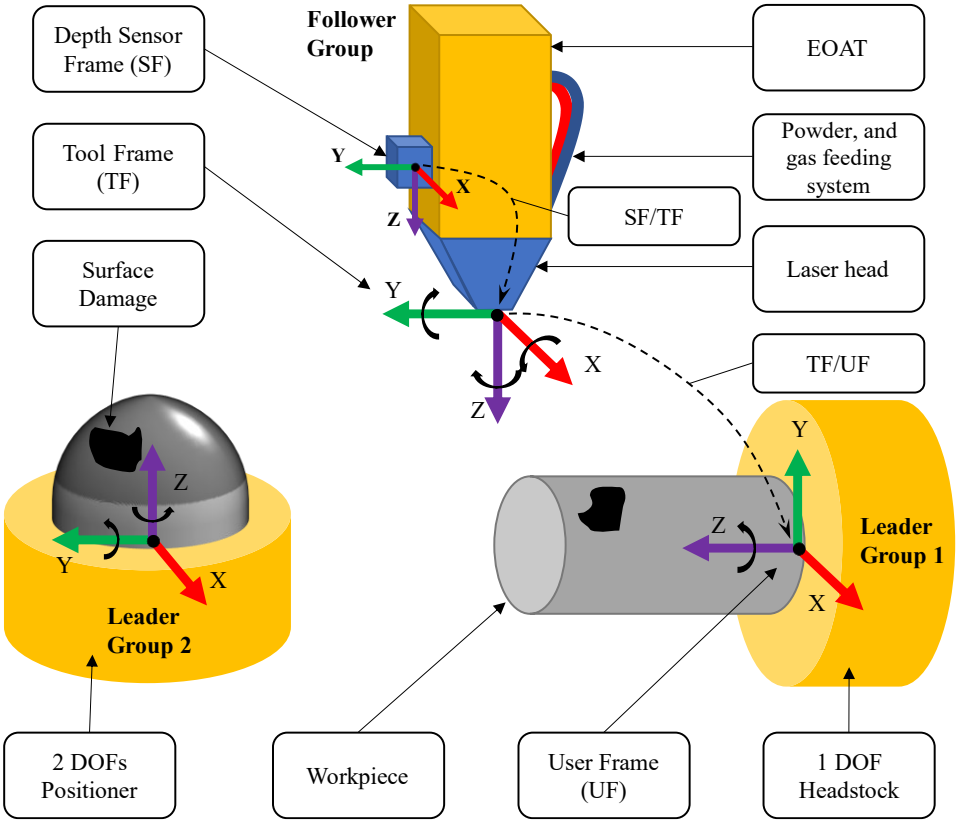


Fig. 4.3: Laser cladding system configuration

4.3 Methodology: Laser Cladding Path Planning

4.3.1 Damaged Surface Reconstruction

A spatial measurements acquisition of the actual worn surface is an essential step to represent it in a digital 3D point cloud form. This is considered a step on the reverse engineering technique and achieved by any means of depth-sensing equipment. Since the cladding process is performed only on the damaged area, it is efficient to obtain a 3D scan of that section only. This approach is adopted here, with promising results, by applying deep learning models to detect and localize a product's damaged area automatically. Detection and localization of the damage using AI are out of the scope of this research. In this work, a laser time-of-flight sensor is used to collect 3D points of the damaged surface of a part. Practically, the robot arm is fixed in a perpendicular pose to the damaged workpiece while maintaining the targeted scan area within the sensor range for a convenient point cloud collection process. For each collected point, a transformation from the Sensor Frame (SF) to the User Frame

(UF), illustrated in **Fig. 4.3**, is required for local representation of the damaged surface with respect to the UF. The SF/UF transformation is constructed using two transformations shown in **Fig. 4.3**, including transformation from (SF) to (TF) and transformation from (TF) to (UF) as described in **(4.1**), where *pcd* is one point cloud data, *r* and *R* are rotations matrices, and *d and D* are translations between different frames. Those two transformations can be combined into one transformation (SF/UF) as long as the sensor's position and orientation are fixed with respect to the tool frame.

$$\begin{bmatrix} pcd_x \\ pcd_y \\ pcd_z \\ 1 \end{bmatrix}^{UF} = T_{TF}^{UF} \cdot T_{SF}^{TF} \cdot pcd^{SF} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & d_x \\ r_{21} & r_{22} & r_{23} & d_y \\ r_{31} & r_{32} & r_{33} & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} R_{11} & R_{12} & R_{13} & D_x \\ R_{21} & R_{22} & R_{23} & D_y \\ R_{31} & R_{32} & R_{33} & D_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} pcd_x \\ pcd_y \\ pcd_z \\ 1 \end{bmatrix}^{SF} \quad (4.1)$$

The collected points are then converted to the suitable coordinate system depending on the workpiece and targeted working configuration using **Table 4.1**. This conversion is performed to enable the unwrapping of cylindrical and spherical shapes. The flattening process allows for a planar slicing process and layer-by-layer building. After the conversion process, the building direction, i.e., the damage axis, is selected and aligned with the system's Z-axis. For example, a worn surface of a cylindrical component gives the geometry radius as a direction of build. This is illustrated in

Fig. 4.4, where a section of the cylindrical component was scanned, followed by a conversion and unwrapping process.

Table 4.1: Cartesian, Cylindrical, and Spherical Coordinate Systems Conversion

To / From	**Cartesian (CART) (x, y, z)**	**Cylindrical (CYL) (r, θ, z)**	**Spherical (SPH) (ρ, θ, φ)**
Cartesian	-	$r = \sqrt{x^2 + y^2}$ $\theta = \arctan2(y/x)$ $z = z$	$\rho = \sqrt{x^2 + y^2 + z^2}$ $\theta = \arctan2(y/x)$, $\varphi = arccos(z/\sqrt{x^2 + y^2 + z^2})$

Cylindrical	$x = r\cos\theta$ $y = r\sin\theta$ $z = z$	-	$\rho = \sqrt{r^2 + z^2}$ $\theta = \theta$ $\varphi = arccos(z/\sqrt{r^2 + z^2})$
Spherical	$x = \rho\ \sin\varphi\cos\theta$ $y = \rho\ \sin\varphi\ \sin\theta$ $z = \rho\cos\varphi$	$r = \rho\ \sin\varphi$ $\theta = \theta$ $z = \rho\ \cos\varphi$	-

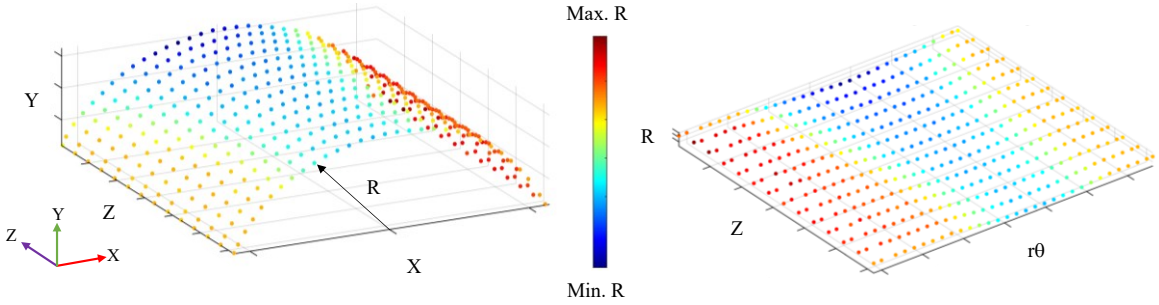


Fig. 4.4: Point Cloud in Cartesian Coordinate (left), and in Unwrapped Cylindrical Coordinate (right)

Some of the cladding parameters are in a unit length, such as cladding height, width, and pitch (clads spacing distance) which are discussed later in detail. Those parameters are later used to generate the tool paths based on the spatial position of the point cloud. However, the flattening process results in inconsistent units of some axes in the example above θ and Z. For instance, calculating the cladding edge length results in a length that is a combination of two units of angle and length. Thus, a consistent, correct length measurement requires matched axes units. This is accomplished by converting the angle θ to an arc length using the nominal radius of the rebuild. This is achieved using **(4.2)**, with the angle in radian. Similarly, two angles are converted to an arc length for the spherical coordinate.

$$\text{arc length} = r\theta \tag{4.2}$$

After preparing the point cloud data, a meshing process is applied to represent the topology of the damaged surface from the point cloud, which generates triangulated mesh, as shown in **Fig. 4.5**. One reliable meshing method is the Delaunay, which is based on the triangulation of the convex hull such that no point is inside the circumcircle of any triangle

in the mesh. The created mesh is structured as faces that contain the indices of the provided points in space. Furthermore, a bounding box of the mesh is determined using the mesh limits. This box is used later to generate the layers' slicer planes and calculate the length of the building edge, that is, the vertical edge following the suitable building approach. For this method, a bottom-up rebuild approach is selected for an outer damaged surface; however, a top-down approach can be used for inner damaged surfaces. Since the laser system configuration used in this work is aimed at out defects, repairing inner damage is not discussed in this study.

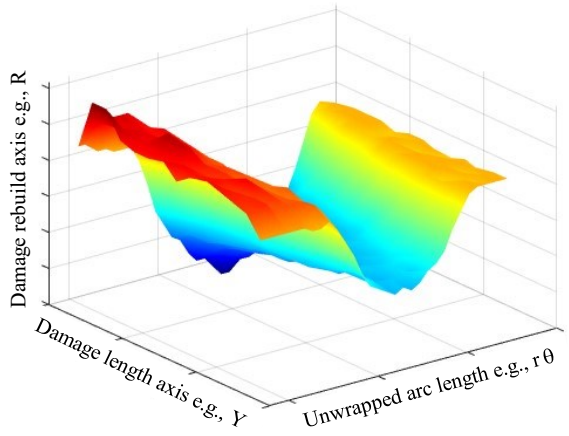


Fig. 4.5: Triangulated mesh of the damage

Since the laser cladding process is a type of additive manufacturing technique, the building sequence of such a process follows the layer-by-layer concept by stacking the deposited material in layers. Thus, the proposed method shown in **Fig. 4.6** follows the same approach for damage mesh reconstruction by slicing the raw mesh using a plane (planar slicer). The mesh slicing process results in a group of meshes that describes the rebuild of the damage in layers approach. An initial step of this method is to calculate the layer thickness, described as layer pitch, using (4.3).

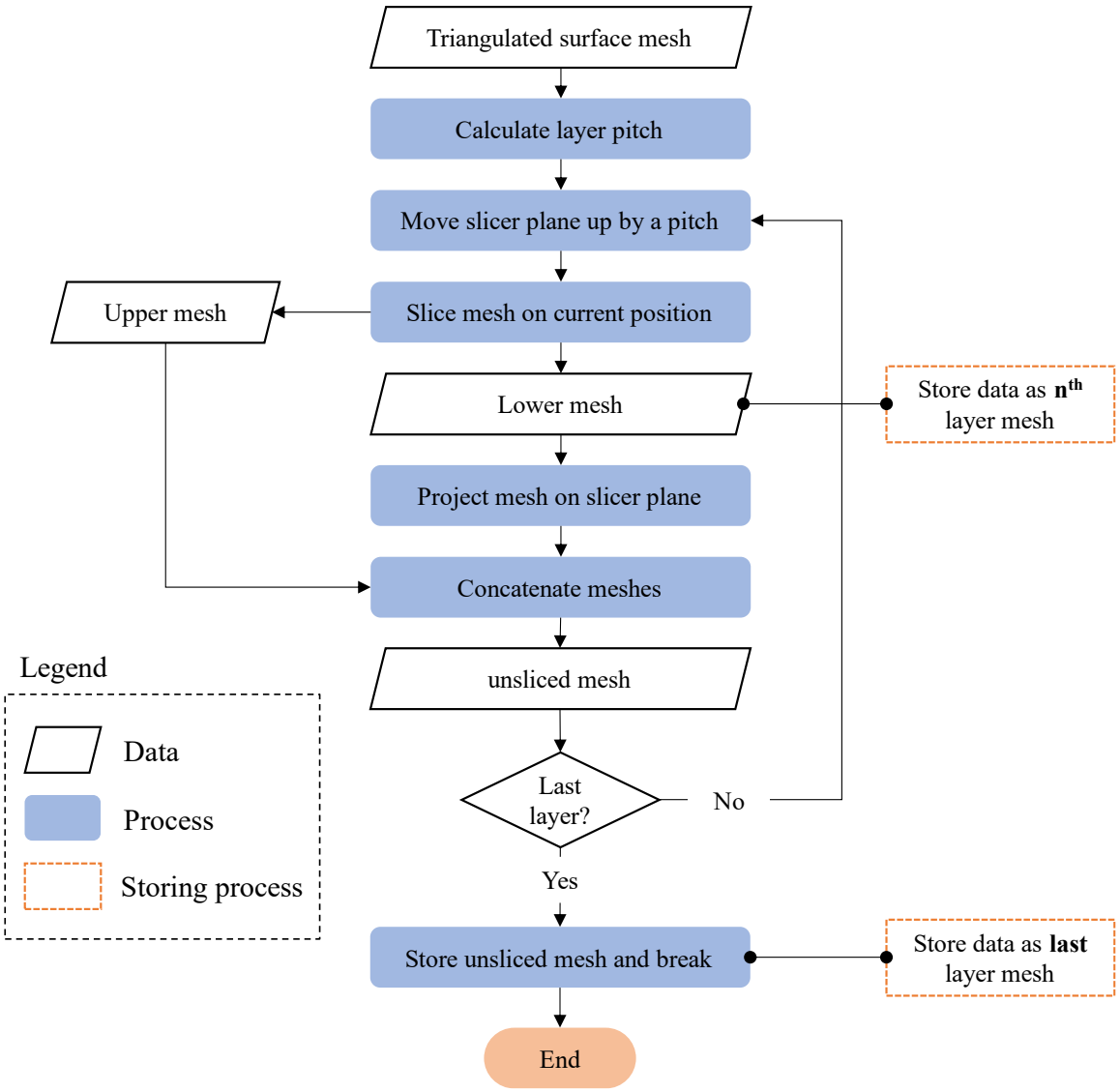


Fig. 4.6: Proposed damaged surface reconstruction method

$$Number\ of\ layers = Building\ edge\ length\ /\ Maximum\ allowable\ pitch \qquad (4.3)$$

As mentioned above, the building edge length is computed using the mesh top and lower limits for maximum rebuild or a reference radius for a specified rebuild. Using the maximum layer thickness that the cladding processing can achieve gives the minimum number of layers using (4.3). Then, given the number of layers for rebuild, the thickness of the last layer is tested for two cases: If the last pass is greater than half-pitch, then the number of layers is rounded towards plus infinity. On the other hand, the number of layers is rounded towards minus infinity, except if there is only one layer to be built.

After obtaining the number of layers and pitch, a slicer, hyperplane, is created at the center of the mesh with its normal points towards the build-up direction. The triangulated mesh is then sliced using that plane, which generates two detached meshes. This process is achieved by looping over every face in the mesh to find the intersection between a face and the slicer. Moreover, the lower section is stored as the first layer mesh, and the upper section is passed to the next step. Furthermore, a projection of the lower layer onto the slicer plane is generated and concatenated with the upper mesh, as illustrated in **Fig. 4.7**. The concatenated mesh is then used as a raw mesh for the next iteration of slicing. In the next iteration, the slicer is moved by a pitch in the rebuild direction to generate the second layer mesh. Additionally, this process continues as layers are generated until the remaining mesh is the last layer. The final output is a group of open surfaces passed to the tool-path planning stage and processed independently, as shown in **Fig. 4.8**.

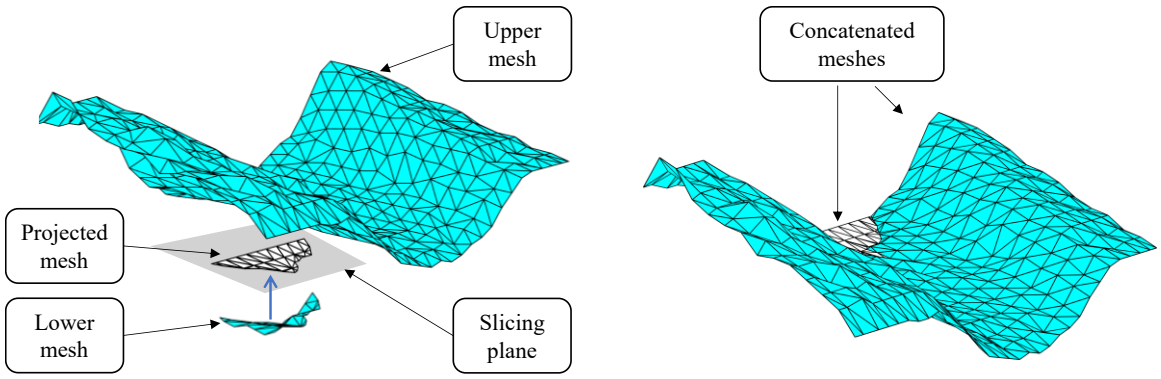


Fig. 4.7: Surface rebuild process illustration (left), and concatenated mesh (right)

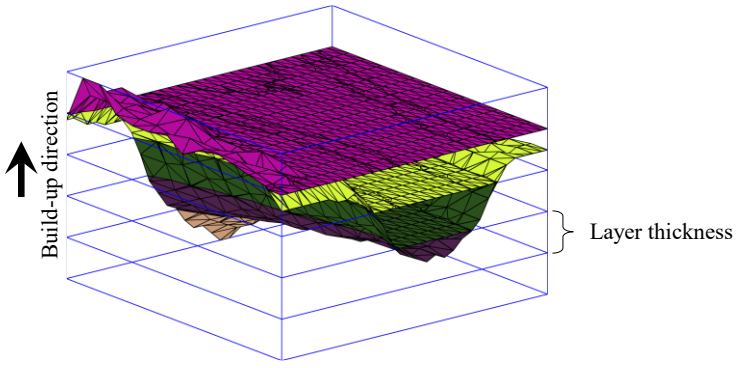


Fig. 4.8: Reconstructed damage

Furthermore, a metallic molten pool is generated during the laser cladding process, which is affected by several factors, including gravitational force. Moreover, one of the other

factors for an effective and predictable bounding between the metallic powder and substrate is the tilt angle of the laser head, which is the rotation of the TCP around the x-axis as in **Fig. 4.3**. A study by [150] showed predictable and acceptable clads (weld bead) for less than 30 degrees for a tilting angle of the xy-plane. Thus, a perpendicular laser head to the damaged surface can be accomplished by surface following the technique. For those reasons, information about the tool orientation during the cladding process is essential for ensuring surface following. For this reason, mesh normal vectors are computed and used in the next steps. A normal of the face is computed using the cross product of any two edge vectors for each triangular face. A normal vector is also calculated for every vertex in the mesh by averaging normal vectors coming from every adjacent face. With this, normal vectors are used to compute normal paths in the next step.

4.3.2 Laser Cladding Path and Normal Generation

Several parameters affect the laser cladding paths, including cladding height, width, pitch, overlap, and orientation, as shown in **Fig. 4.9**. Those parameters are calculated for each layer to generate the optimal number of paths per layer; however, those parameters are fixed within the layer of the build. This means the height, width, and pitch of clads are the same per layer. Unlike other studies, this feature adds flexibility and functionality to this method. The cladding pitch is the distance from one clad center to another, i.e., stepping distance. Also, the overlap of clads can be measured in percentage to the clad width. Moreover, the angle measured from the cladding traces to the x-axis in the xy-plane of the unwrapped format is described as the cladding orientation illustrated in **Fig. 4.9**, which is constant per layer. This feature enables generating cross-hatched layer paths. Furthermore, path layers are aligned together when zero layer-offset is used. Nevertheless, a half layer-offset produces interlocked path layers that are shifted by a half-clad width.

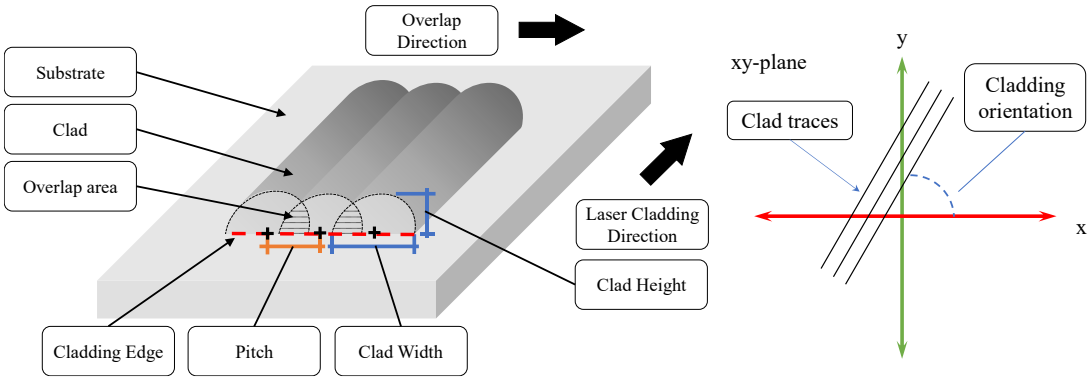


Fig. 4.9: Laser Cladding Parameters Illustration

Additionally, cladding pitch and overlap are calculated based on the maximum allowable spacing distance and overlap. To compute the optimal pitch and overlap, the cladding edge, which is the limit of stacked clads per layer, is determined first. Using the original triangulated mesh, a plane is created at the center (x_c, y_c, z_c) of the mesh with a normal (i, j, k) angled at the cladding orientation using (4.4). Then, a projection of each vertex in the mesh onto this plane is calculated and sorted spatially. The lower and upper limits of those points are selected as the start and end points of the cladding edge, with its vector computed from the difference between these two points. After the cladding edge length is computed, the same process as in (4.3) is followed to calculate the optimal pitch and number of clads per layer. Ultimately, the relation between pitch and overlap ratio is described by using (4.5).

$$(x - x_c)i + (y - y_c)j + (z - z_c)k = 0$$
$$n(i,j,k) = [\cos(cladding\ orientation\ angle),\ \sin(cladding\ orientation\ angle),\ 0] \quad (4.4)$$

$$overlap\ ratio = 1 - \frac{pitch}{width} \quad (4.5)$$

After the cladding parameters calculation, the algorithm for generating paths, in **Algorithm 4.1,** starts iterating over the mesh layers formed in the reconstructing process. For each layer, a clipper, formed as a hyperplane shown in **Fig. 4.11**, is created at the starting point position and rotated at the cladding orientation using (4.4**)**, as described in the previous paragraph. Using this clipper, each face in the layer is tested for an intersection with the current clipper. If a clipper/face intersection exists, points of intersection and their normal vectors are calculated and stored. Thus, this allows for fast processing of mesh/plane

intersection by processing influenced faces only. Moving the clipper forward by a pitch step in the overlapping direction shown in **Fig. 4.9**, a new clipper is created and tested as before. This process ensures a uniform distribution and spacing of clads in each layer. The same procedure is repeated for the number of clads in one layer. Further, the generated line segments, from one clipper, are split into groups of connected paths. Ultimately, iterating over layers generates the arranged paths for the rebuild process. However, those paths require cleaning and optimization to be processed by the robot efficiently. An example of the generated paths is shown in **Fig. 4.10**, where each layer of paths is plotted in a different colour. Practically, the laser cladding rebuild process starts from the bottom layer and going up for the external damage repair. One trace of a clad is represented by a set of points and shown as connected line segments in **Fig. 4.10**.

Algorithm 4.1: Tool-path generation algorithm of a reconstructed surface based on plane/face intersection.

VARIABLES:
pitch = distance between two clad centres
pitch_vector = clads overlap direction vector
starting_points = first clad point position
nClads = number of clads per layer
Cp = a two-dimensional plane clipper of a layer
layer_mesh = one slice of the reconstructed mesh slices
face = one triangle face of a reconstructed sliced mesh
path = a line segment resulting from clipper/face intersection
paths/grouped_paths = unsorted and sorted group of paths respectively
layer_paths = all groups of paths of the rebuild process, arranged in layers
INPUT: RECONSTRUCTED MESH SLICES
FOR n = number of mesh layers per model
  Calculate clad ***pitch, pitch_vector, clipping_starting_position,*** and ***nClads***, using given cladding parameters for each ***layer_mesh***
  / cladding orientation, and layer-offset affect the cladding vector and starting point respectively. /
  FOR ***C*** = *1 :* ***nClads***{n} (number of clads per layer)
    p = ***pitch***{n} * ***clipping_starting_position***{n} * ***pitch_vector***{n} * ***C*** */ position of the clipper /*
    Cp= create clipping plane at **p** position, with the normal vector of ***pitch_vector***.

20 **FOR *faces*** in current ***layer_mesh***{*n*}

21 **IF face** of ***faces*** passes through the clipper ***Cp***, **THEN**

22 ***path*** = intersect clipper ***Cp*** with the current **face**.

23 */ normal vectors associated with path are also computed here, as discussed below/*

24 Add ***path*** to ***paths***

25 **END FOR**

26 Sort paths of the same clipper spatially.

27 Split paths of the same clipper at separation points.

28 Add sorted and split groups of ***paths*** to ***grouped_paths***.

29 **END FOR**

30 Add ***grouped_paths*** to ***layer_paths***{n}

31 **END FOR**

32 **OUTPUT:** TOOL-PATHS and NORMAL VECTORS

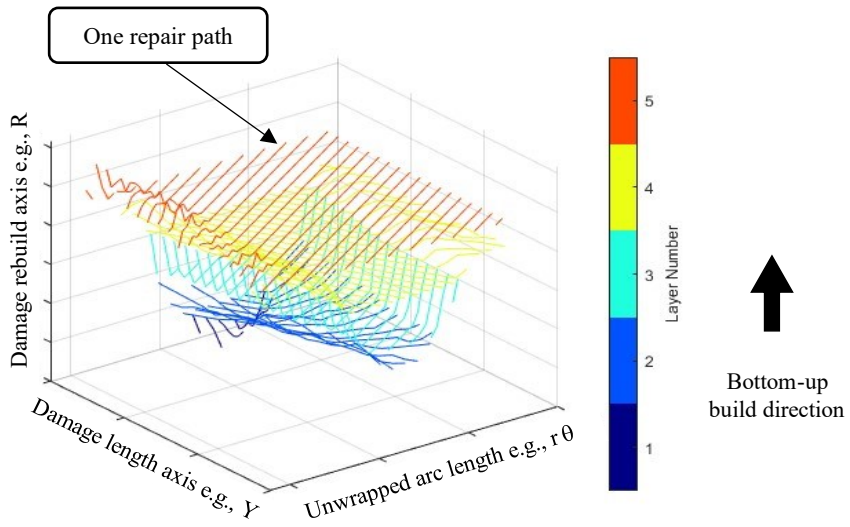


Fig. 4.10: An example of the cladding paths generated using the proposed algorithm in unwrapped CYL format

Providing the vertices normal vectors computed in the reconstruction section, a face consists of three edges that are used to calculate the clipper/face intersection. For each point of intersection in an edge, a normal of that point is computed using the edge length ratio as in (4.6). As shown in **Fig. 4.11**, n represents normal of a vertex, l represents the length of a line segment. When the clipper is passing through a face vertex, the intersection point is the same as the face vertex, which results in an

equal normal for the intersection point. The same process is applied to each edge of the intersection. With this method, an accurate representation of the normal vector at any point can be obtained easily.

$$n = \frac{l_1}{L}\, n_1 + \frac{l_2}{L}\, n_2 \tag{4.6}$$

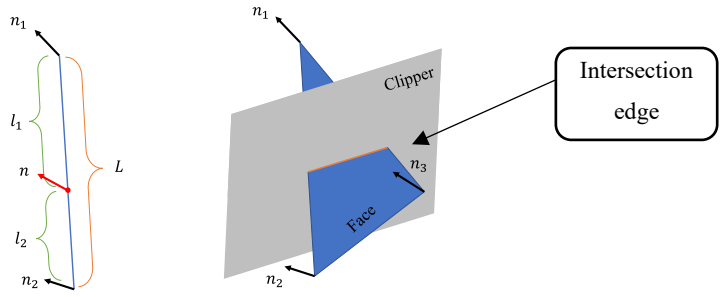


Fig. 4.11: Computation process of the tool-path normal vectors

4.3.3 Interpolated Robot Processing Points Calculation

The generated paths are represented by line segments, which approximate the real curve line. It is neither efficient nor practical to send those paths directly to the robot since they usually consist of very dense and large data. Additionally, they are often spaced unevenly since they are based on the spacing of the triangulated mesh vertices. Furthermore, robot cell has limited computing resources. Thus, sending a very large program file will require a longer time for execution and processing. Thus, processing such a large file leads to a concentration of the laser power for a longer time, which allows for more energy to be absorbed by the substrate. This is found to increase the dilution rate during the cladding process, which changes the clad performance and composition [151]. Therefore, a linear interpolation process, shown in **Fig. 4.12**, of the raw tool paths generated previously, is adopted to reduce the number of processing points sent to the robot to ensure an optimal cladding process. The interpolation interval, which is an arc length, is based on the laser defocus range. This interval ensures a generation of processing points that are within the limits of the laser focus. The same process is also applied to the normal vectors; an interpolated normal vector is computed from the nearest normal vectors. This reduces the points in processing paths while maintaining the laser focus when the surface height changes. In addition, very short paths are sometimes generated depending on the cladding orientation, which is not efficient to process if it is less than a threshold. Hence, a minimum length of the clad is used as a threshold to remove any short and impractical paths.

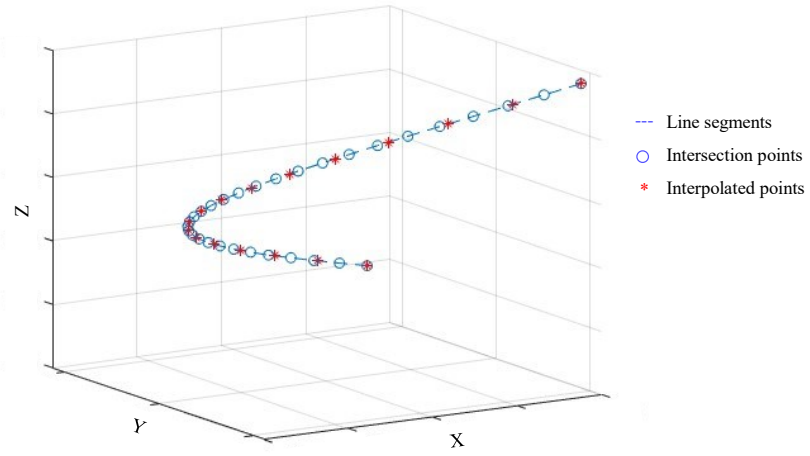


Fig. 4.12: Tool-paths interpolation process

A reduced paths file is generally sent to programming and simulation software for further kinematic calculation of the robot joints' movement. However, matching the coordinate systems of the generated paths with the coordinate system used to carry out the cladding process is an essential step. Same as before, **Table 4.1** is used to perform a system coordinate conversion for both processing points and their normal vectors.

4.4 Results and Discussion

In this work, a laser cladding system is used with a configuration mentioned in section 4.2. The robot arm used is Fanuc-R-1000iA/80F, which handles high speed with a medium payload. Attached to the robot is a line beam laser head with dual off-axial powder feeding nozzles. In addition, a Keyence IL-300 single-point depth digital sensor, with a range of 160 to 450 mm, is fixed to the EOAT that is based on the time-of-flight (ToF) technology. This type of sensor provides several benefits over others, including high accuracy, compact size, cost reduction, and ease of operation.

4.4.1 Case Studies

The developed method is tested on three practical case studies, shown in **Fig. 4.13**, to verify its effectiveness in different conditions. The point cloud scans of those case studies are obtained from an industrial partner. A common factor between those cases is they all

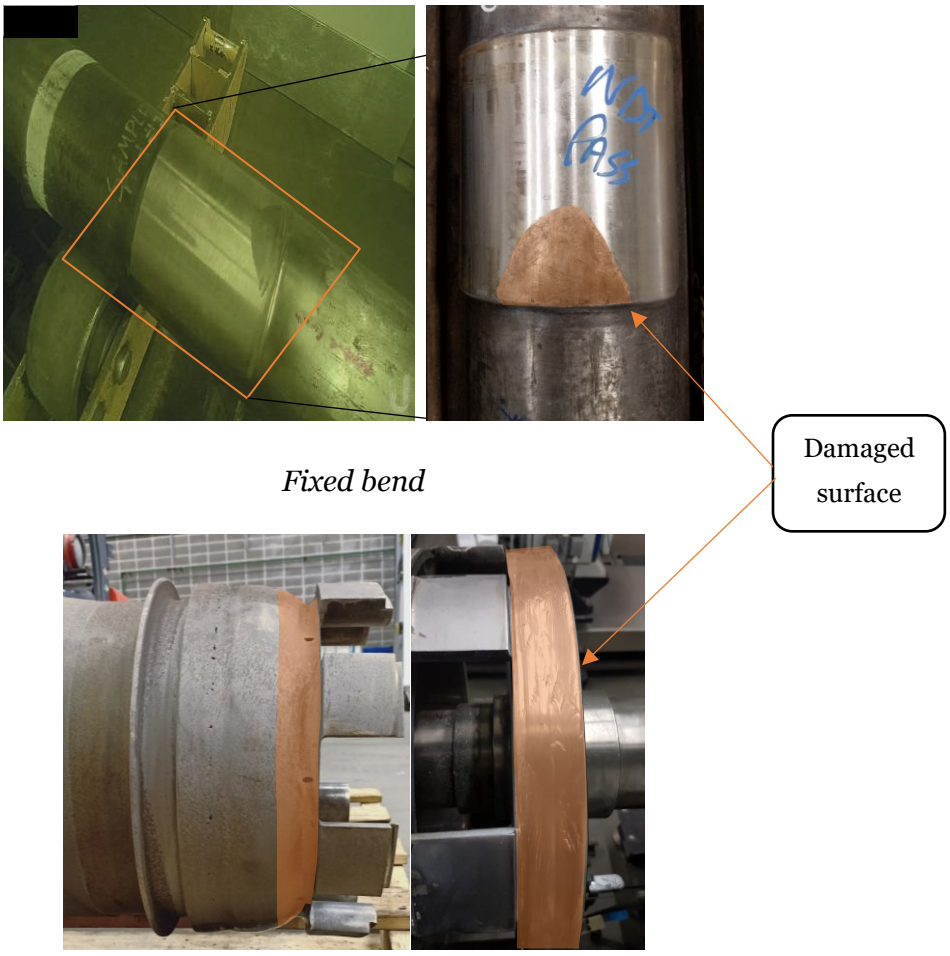
represent external surface damage. Moreover, the properties of those samples are listed in **Table 4.2**.

Table 4.2: Actual damage case studies

Property	**Case 1 (fixed bend)**	**Case 2 (ring)**	**Case 3 (band)**
Point Cloud	375 points	9,125 points	344 points
Surface Mesh	672 faces	17,498 faces	510 faces
Processing System	Cylindrical Coordinate	Cylindrical Coordinate	Cylindrical Coordinate

The first case study is an example of an abrasive wear cavity that exists on a fixed bend resulting from the friction of the fixed bend surface with the stones during its drilling operation. In addition, the ring case (case 2) has damage created around the part forming ring-like damage. In the third case, a band contains multiple scattered cavities across its circumference. This case tests the ability of the developed method to handle scattered damaged areas. All samples can be processed in either system configuration, headstock, or positioner. In all cases, the headstock group is used since the damage axis is in the radial direction, and 1 DOF is sufficient for this repairing process.

The reconstruction and path planning algorithm is programmed in MATLAB® 2021 language with the support of the Geom3D library [152] an external library that enables 3D primitive shapes manipulation. The developed program allows for automatic path generation and manipulation, as well as exporting processing points and their normal vectors into a CSV file for further processing. The exported points and normal are then used to calculate robot kinematics using online/offline programming and simulation software such as FANUC ROBOTGUIDE®, and RoboDK®. A robot program is generated and verified using such tools to be sent to the actual robot system for laser cladding.



Fixed bend

b. Ring *c. Band*

Fig. 4.13: Actual case studies

The acquired spatial points of the case studies using the ToF sensor are transformed from the sensor frame to the user frame using a separate program in the robot controller. It is then stored in a PCD file format that is efficient for managing the point cloud. The path planning program receives the raw point cloud and performs an initial step to clean the 3D points from any redundant and noise in the point cloud. A list of the selected parameters for path planning generation of the three case studies is listed in **Table 4.3**.

Table 4.3: Laser path planning parameters

Parameter	Case 1	Case 2	Case 3
Clad height (mm)	1	1.3	1.5

Clad width (mm)	3	7	10
Maximum allowable pitch (mm)	1.85	5.5	8.25
Clads orientation (deg.)	45	0	60
Cross-hatching pattern	Enabled	Disabled	Disabled
Layer offset	0	½ cladding step	0
path segment range (mm)	3-6	4-7	3-5
Build-up direction	Radial	Radial	Radial
Processing System	Cylindrical	Cylindrical	Cylindrical

The representation of the point clouds in the cartesian coordinate system shows no indication of surface damage, as shown in **Fig. 4.14**. Therefore, the point clouds were converted to a suitable processing coordinate system, cylindrical coordinate for all cases. Then, those surfaces were unwrapped for the reconstruction process to be performed and better damage visualization. Further, unwrapping angles in all cases were accomplished using the nominal rebuild radius. This nominal radius is either extracted from the point cloud as the maximum radius or provided by the user manually. In cases 1 and 2, the repairing process is performed to the maximum radius since the scanning procedure was performed such that a portion of the nominal surface is included in the acquired data. On the other hand, Case 3 is selected to be repaired to a reference radius, which shows the capability of the algorithm to build to a specified dimension. For cases 1, 2, and 3, the rebuild radii were 100, 342, and 75, respectively, all in mm.

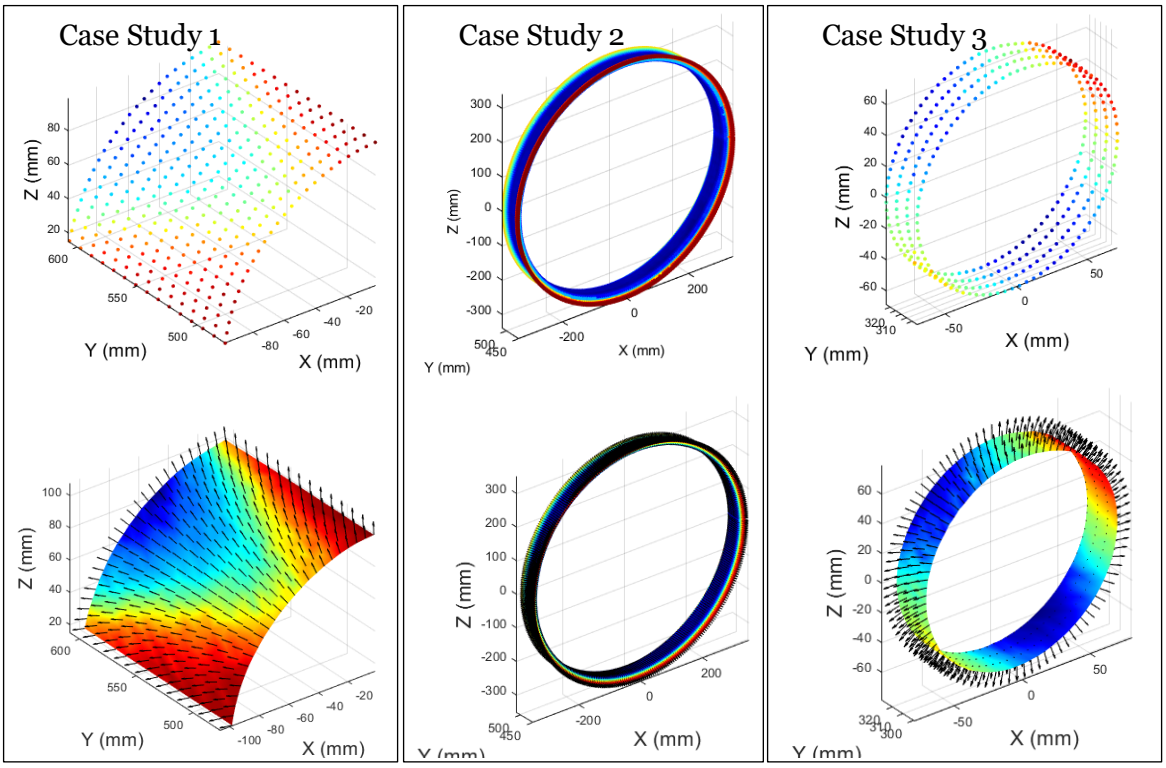


Fig. 4.14: Case studies point clouds, and meshes in the cartesian coordinate system

The converted and unwrapped point clouds and meshes are shown in the first two figures of **Fig. 4.15**, **Fig. 4.16**, and **Fig. 4.17**, which enables the true visualization of the surface damage. One cavity damage existed on the fixed bend case, but multiple cavities were in the ring and band cases which showed the effectiveness of this algorithm in handling different types of wear. For three case studies, the automatic path planning algorithm results are listed in **Table 4.4**. For the reconstruction process of the fixed bend case, four layers of meshes were generated with a thickness of 0.945 mm, as shown in **Fig. 4.15**c and d. Moreover, the obtained layer thickness is calculated based on the maximum clad height, which is 1 mm. In addition, normal vectors were computed for each mesh layer. With the selected parameters in **Table 4.3**, four layers of three-dimensional tool-paths were generated using the unwrapped cylindrical meshes from the reconstructed step. As shown in **Fig. 4.15**e, a various number of paths per layer is observed. Further, a 1.846 mm clads spacing distance is calculated as an optimal step to fit as many clads as possible for each layer based on the cladding width and maximum cladding step. This spacing distance resulted in a 38.47% overlap of the clads. As shown in **Fig. 4.15**e, paths are generated in two orientations, 45° and -45° angles, because the cross-hatching feature was enabled. This feature allows for

a composite like a rebuild, which can be of focus in future investigations. Shown in **Fig. 4.15**f is a cartesian representation of the generated tool-paths and their normal vectors that were later exported to the programming and simulation software.

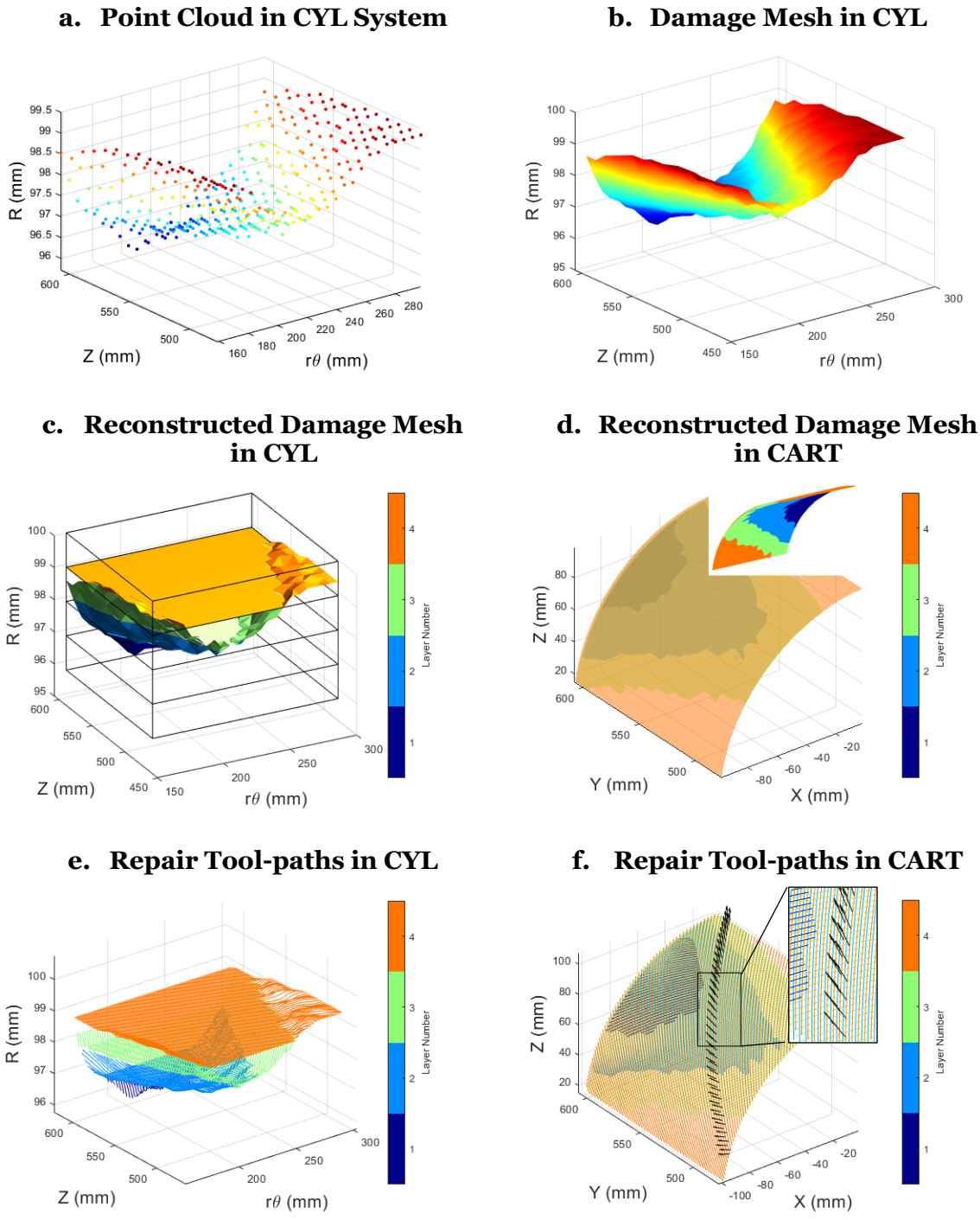


Fig. 4.15: Case 1, generated tool-paths using the proposed method

Table 4.4: Cases path planning results

Parameter	Case 1	Case 2	Case 3
Build radius (mm)	100	342	75
Number of layers	4	14	5
Reconstructed mesh vertices	1,326	283,120	1,877
Reconstructed mesh faces	2,302	540,907	2,847
Layer thickness (mm)	0.945	1.2593	1.45
Spacing distance (pitch) (mm)	1.846	5.4715	8.2478
Overlap ratio	0.3847	0.2184	0.1752
Paths' raw points	8,398	122,422	2,886
Number of processing paths	283	167	221
Number of processing points	7,875	69,960	2,092
Average processing time (s)	8	345	6

Unlike the first case, the second case represents a full damaged part's circumference that features ring-like damage. Moreover, more spatial points are acquired to accurately rebuild this case. This case also shows the capability of this method to handle partial and full damaged components. The procedure results of the automatic path planning algorithm are shown in **Fig. 4.16**. With the given parameters, fourteen layers of the reconstructed meshes resulted in a full rebuild of the damaged part shown in **Fig. 4.16**c and d. Further, the optimal layer thickness was 1.2593 mm. As mentioned before, various paths were generated per layer depending on the parameters' selection. For instance, 6 paths were generated to repair the first layer, but 11 traces were calculated to rebuild the last layer, as shown in **Fig. 4.16**e. Moreover, a 21.84% overlap was calculated with a spacing distance of 5.4715 mm between clads. In addition, the path orientation was given as 0° with the x-axis and resulted in the path's orientation as expected. Wrapping, converting those paths and their normal vectors to the cartesian coordinate system, visualize the actual repair process movements shown in **Fig. 4.16**f.

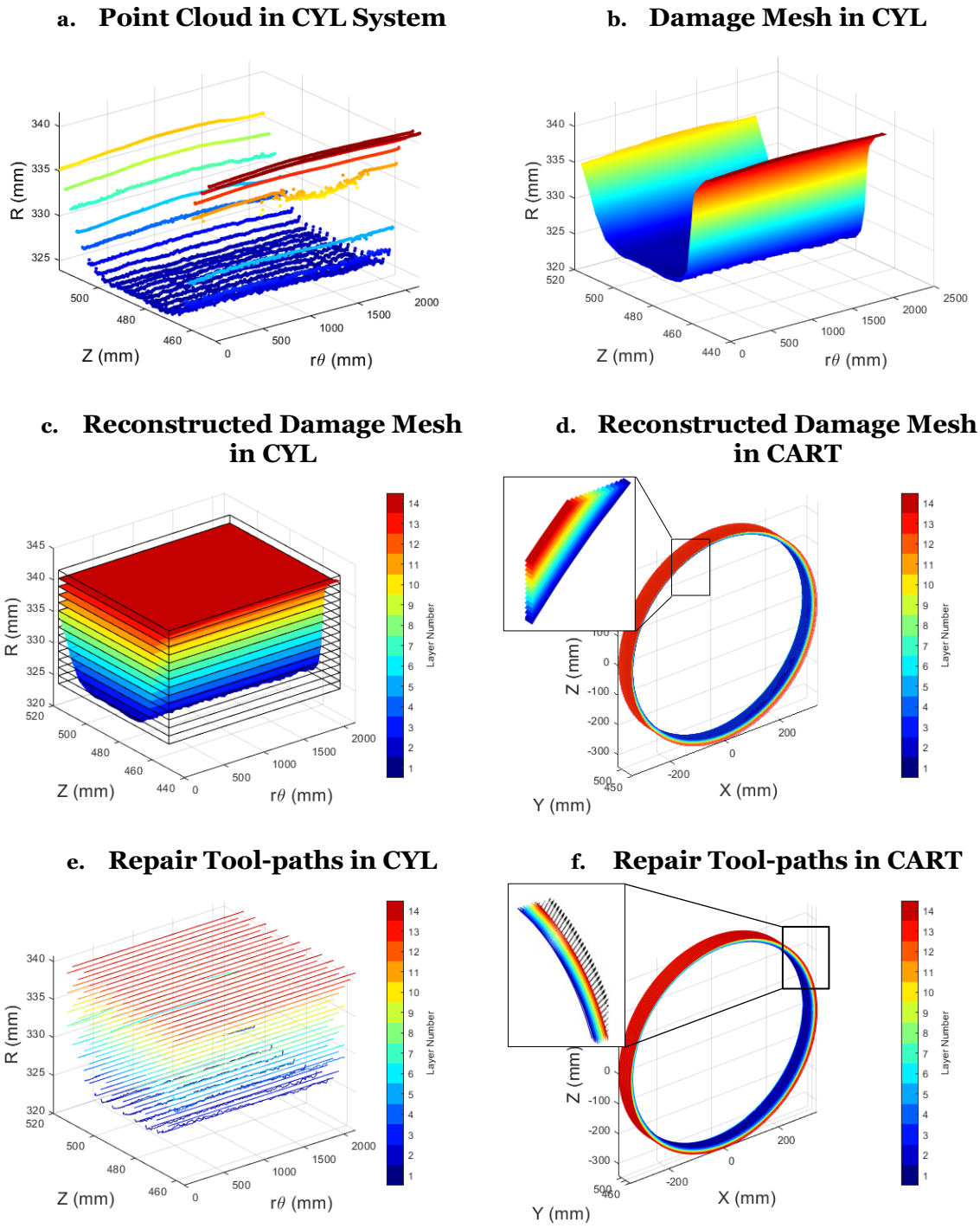


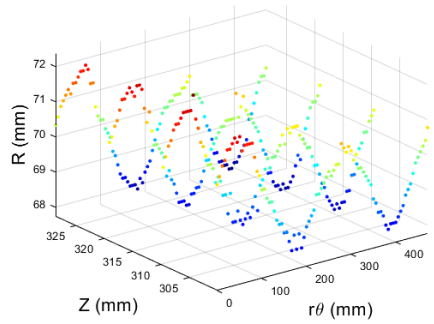
Fig. 4.16: Case 2, generated tool-paths using the proposed method

The third case also shows ring-like damage that contains multiple damaged cavities. With the given cladding parameters in **Table 4.3**, five layers for the rebuild were constructed. In addition, layers exceeded the maximum limit of raw mesh in **Fig. 4.17**c to

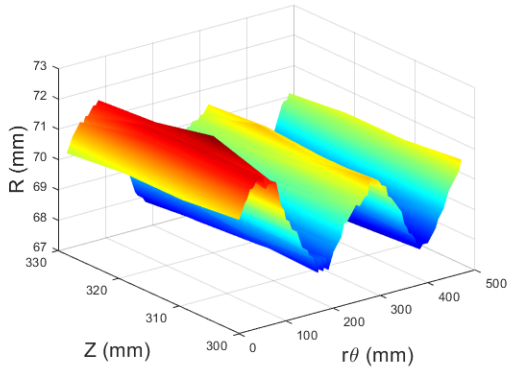
build to a 75 mm radius as expected. This shows the algorithm's capability to rebuild to various reference dimensions. As seen in **Fig. 4.17**c, the first layer of reconstruction contains two different groups of mesh that are disconnected. This is important to note since those groups result in separate paths. Each constructed layer has a thickness of 1.45 mm and is illustrated in different colours. In addition, paths were produced with a spacing distance of 8.24 mm and an overlap of about 17.5%. Overall, over two hundred laser cladding paths were generated to build the damaged surface to 75 mm. However, fewer paths were produced to build the first layers due to the less surface area to be built compared to the last layers. As shown in **Fig. 4.17**f, paths are orientated at a 60 degrees angle with the x-axis.

According to the resulted paths and their calculated parameters, the algorithm generated optimal paths and their normal vectors as expected without defects. In addition, the interpolation method reduced the processing points significantly, which ensures an optimized and effective laser cladding path. Unlike Imam's [143] method, this method was able to produce tool-paths for different types of damages, and with a flexible path orientation. Those features increased the path planning processing times compared to Imam's method. Moreover, this method enables the repair of damaged components without the need for nominal CAD models, which was required in the method developed in [136]. In addition, repurposing parts might require building to a different dimension than the nominal surface. This is achieved in this method, which was not developed in [147]. Furthermore, the processing time depends on many factors, including computation resources and selected cladding parameters such as building reference and path orientation. According to the processing time results in **Table 4.4**, the average processing time was 2 milliseconds per face for mesh processing, damage reconstruction, and path generation on a regular desktop computer with a 6^{th} generation i5 CPU and 16 GB of RAM. This method was found to be 75% faster than Li et al. [148] method for generating paths for laser cladding coating application from a CAD model.

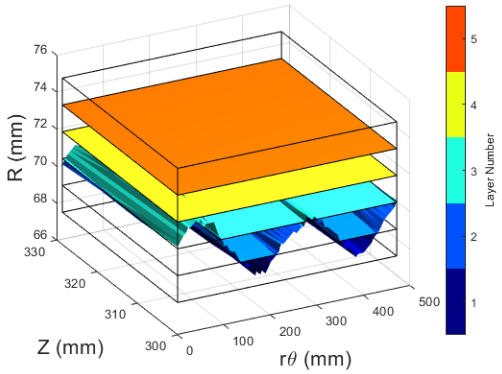
a. Point Cloud in CYL System



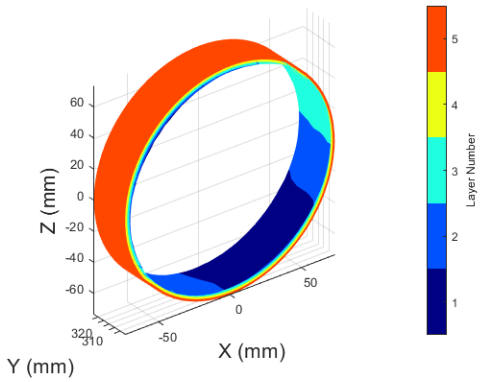
b. Damage Mesh in CYL



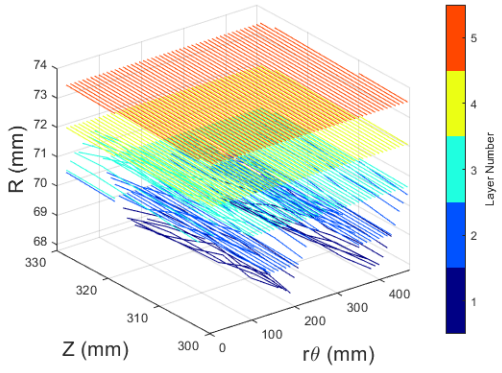
c. Reconstructed Damage Mesh in CYL



d. Reconstructed Damage Mesh in CART



e. Repair Tool-paths in CYL



f. Repair Tool-paths in CART

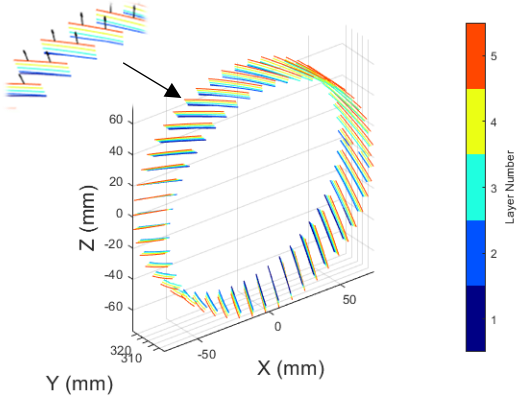


Fig. 4.17: Case 3, generated tool-paths using the proposed method

4.4.2 Tool-path Planning Results Validation in the Virtual Environment

A digital representation of the actual laser cladding cell was developed in an online/offline programming software named RoboDK, as shown in **Fig. 4.18**. There are several methods to import and interpret paths in this software. One way is to construct curves from those paths with the normal vectors and apply a curve following project. Despite the fact that the robot program can be generated automatically using the curve following project method in RoboDK mentioned above, this method was found to be unreliable and unpredictable. In addition, maintaining a vertical pose of the robot to the workpiece is not considered in the curve following project. Furthermore, cladding direction is crucial for the metallic powder flow; this is achieved by maintaining counter-clockwise rotation of the rotary table for the processing points, which is not guaranteed for the curve following project. For those reasons, a new method, described in the next paragraph, was developed to interpret the cladding tool-paths and generate transition paths based on the TCP orientation that is computed using the normal vectors of the processing points. Therefore, a Python script was written to automatically interpret and simulate tool-paths, and eventually generate the robot program for the actual laser cladding process.

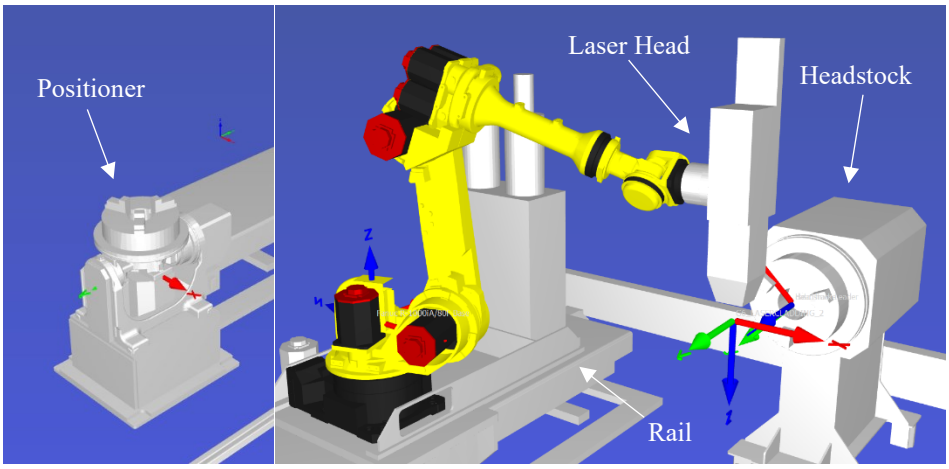


Fig. 4.18: Virtual Robot System in RoboDK software

After the tool-paths are generated in the MATLAB program, they are exported in a CSV file format to be processed by the RoboDK Python script. A sample of the exported paths is listed in **Table 4.5**, including the layer number that contains the processing point is listed in the first column. In addition, a code for each processing point is listed in the second column as 1, 0, and 2 that donate the start, end, and middle points of a path, respectively. This is used

to turn the laser ON and OFF, as well as the approach and retraction movement of the robot. The third column represents the orthogonal distance from the top plane of that layer. In laser cladding, the various layer thickness is achieved by controlling the laser beam scanning speed during the laser cladding process. For the targeted layer thickness build, processing point speed is calculated using point depth. The X, Y, and Z columns represent the spatial position of the processing points in the cartesian coordinate system. This position is with respect to the part frame (user frame).

Table 4.5: Exported paths sample for case 1

Layer number	Point Code	Point Depth mm	X mm	Y mm	Z mm	W deg	P deg	R deg	J1 deg
1	1	0	-44.91186	84.68597	607.301	88.05781	45	27.51275	-27.93853
1	2	0.2575816	-42.03393	86.51348	603.7844	87.53432	45	27.29976	-25.91352
1	2	0.4806255	-39.03753	88.15022	600.2639	88.58002	45	25.97807	-23.88627
1	0	0.806829	-35.99167	89.71467	596.7444	89.12558	45	24.29436	-21.85963

Furthermore, the orientation of the TCP frame is given in W, P, and R columns that represent the rotation around X, Y, and Z with respect to the part frame, respectively. To maintain a vertical orientation of the laser head, the orientation of the EOAT around Z with the part frame must be updated at every processing point. This approach is achieved by updating the TCP frame orientation for each rotation of the headstock, which is indicated in the last two columns, where R is the rotation of the TCP frame around Z, and J1 is the headstock rotation around Z. Thus, this results in the laser head following paths while ensuring a perpendicular attitude to the surface and a vertical posture of the laser head. In **Fig. 4.19**, a screenshot was provided to show the laser head orientation with the headstock for various layers and case studies. Additionally, a sample of the robot program is provided in **Fig. 4.20**.

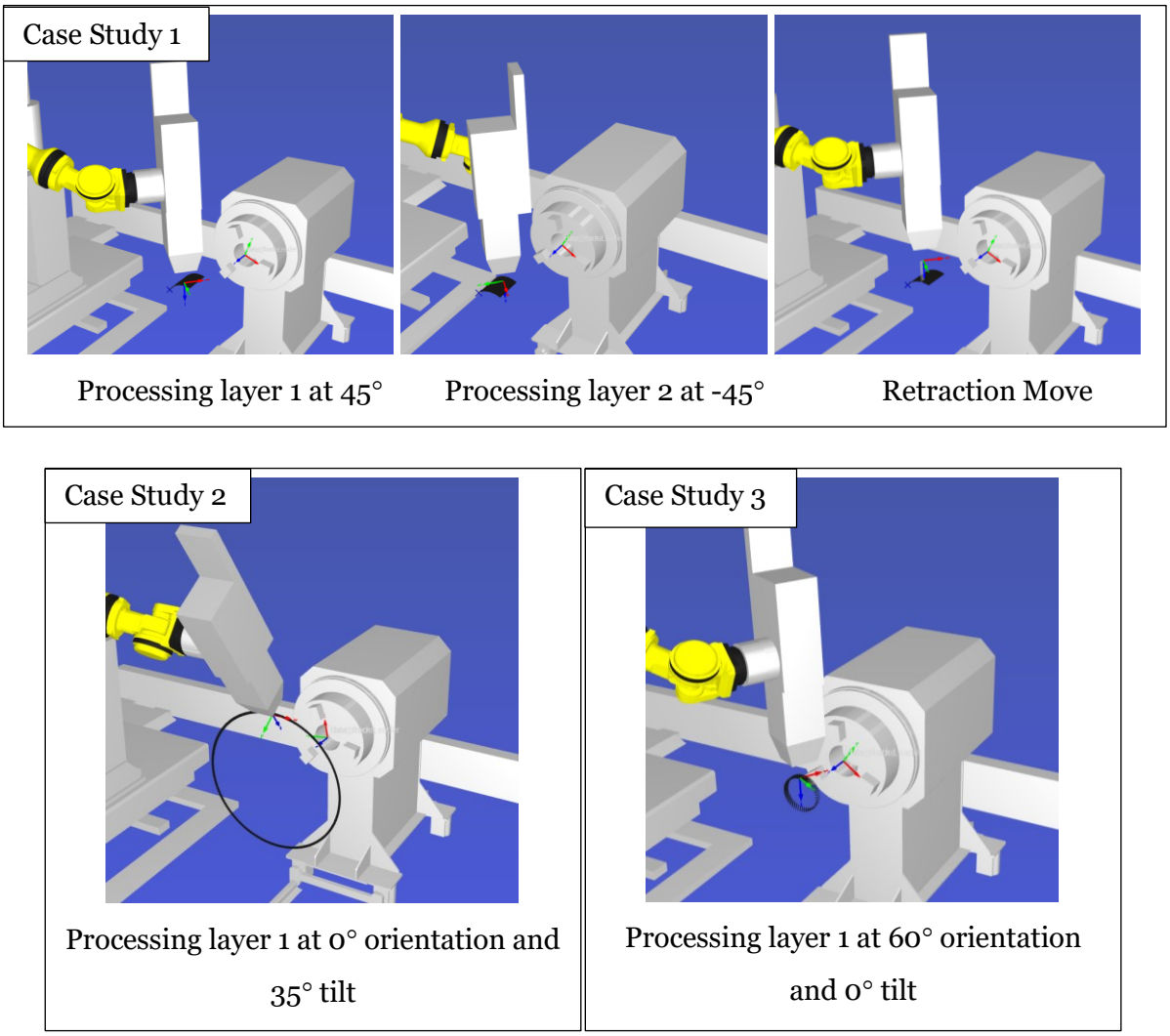


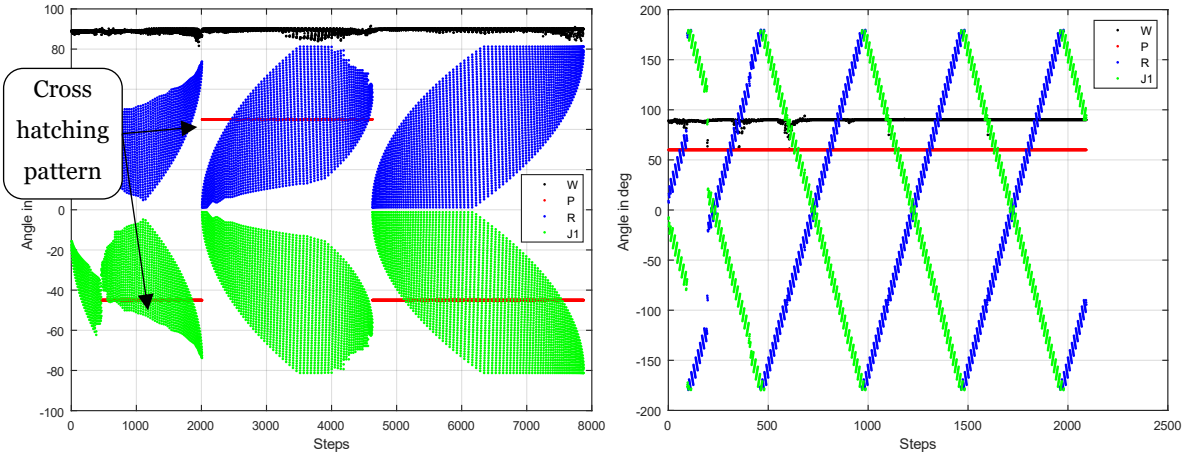
Fig. 4.19: Tool-path simulation in RoboDK

```
24:  ! ** PASS 1 ** ;
25:  LBL[101:pass1] ;
26:L P[6] 100mm/sec FINE Offset,PR[58] ;
27:  WAIT   0.20(sec) ;
28:  TIMER[3]=START ;
29:L P[6] 21mm/sec CNT100 COORD TA   0.00sec,CALL RUN_LASER_START Offset,PR[25] ;
30:L P[6] 21mm/sec CNT100 COORD TA   0.00sec,DO[50]=ON Offset,PR[69] ;
31:L P[7] 18mm/sec CNT100 COORD TA   0.00sec,DO[50]=ON Offset,PR[69] ;
32:L P[8] 16mm/sec CNT100 COORD TA   0.00sec,DO[50]=ON Offset,PR[69] ;
33:L P[8] 16mm/sec CNT100 COORD TA   0.00sec,CALL RUN_LASER_STOP Offset,PR[26] ;
34:L P[8] 16mm/sec FINE COORD Offset,PR[59] ;
35:  TIMER[3]=STOP ;
36:L P[9] 100mm/sec CNT100 ;
37:L P[10] 100mm/sec CNT100 ;
```

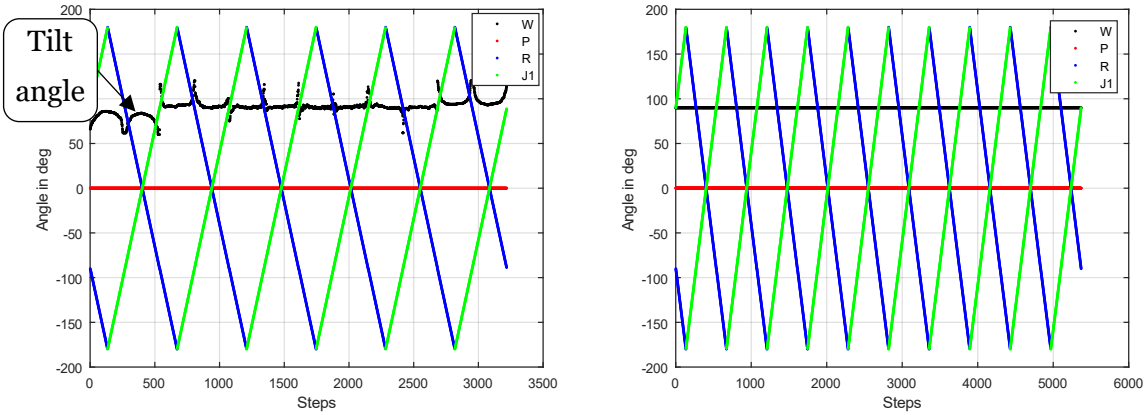
Fig. 4.20: Sample of the generated robot program

In addition, the TCP angles throughout the process for the three cases are plotted in **Fig. 4.21**, which shows the TCP orientation (WPR) and headstock (J1) angles. Some details of the simulated paths in RoboDK are also provided in **Table 4.6**. Each layer of paths is represented in a clustered group of angles. Usually, the first batch of layers has fewer paths since those layers target holes and cavity filling. As shown in all cases below, fewer cladding points existed in the first layers, and more paths as moved forward. A mirrored copy of the Z-axis rotation was observed in the diagram plotted in blue and green colours. This ensures a vertical posture of the cladding tool to the workpiece's surface as it rotates. In addition, it utilizes the system extended axis, headstock in this case, for minimal cladding tool movements. Moreover, the cross-hatching pattern in the desired path was noticed in case 1 plotted in red colour points. In this case, the angle is alternating for each layer of repair between 45 and -45 degrees, which was not found in the literature. Unlike the first case, cases 2 & 3 showed a unidirectional path oriented at 0 and 60 degrees, which was as expected. Another observation is the slight variation of the rotation around X, plotted as W, which represents the tilt angle of the laser head with the workpiece's surface. A higher variation was observed in the lower layers due to a normal vector change indicating a proper orientation of the tool following the damaged surface, as shown in the first layer of case study 2 in **Fig. 4.21**. As the cladding rebuild progressed, a tilt angle shows a minimal change due to the smooth topology of the rebuilt layer beneath as shown in last layers of case 2 and 3. A threshold of 30 degrees was used to prevent and limit an aggressive tilt angle. This is shown in the second case diagram.

For further validation, detailed information on the simulated tool-paths was collected from RoboDK software and listed in **Table 4.6**. The cycle time provided in the table below is an estimation of the laser cladding process time, including any transition movement, that is, the movement when the laser is off. As shown in the table, estimated cycle times are 45 minutes, 5 hours 48 minutes, and 13 minutes for cases 1, 2, and 3, respectively. This time depends on several factors, including the robot travel distance, robot speed, robot controller, and powder flow rate. Further, the estimated travel distances of the three cases are 124, 308, and 64 meters for cases 1, 2, and 3, respectively, which is reflected in the processing times above. Moreover, case 2 has the most change in tilt angle, especially for the first layers of the rebuild, because the laser cladding head requires a perpendicular pose to the substrate surface for proper and effective clad bounding. For the other cases (1 and 3), minimal change is observed.



Case 1: all four layers **Case 3: all five layers**



Case 2: first layer **Case 2: 12th layer**

Fig. 4.21: Robot TCP orientation, and extended axis angles throughout the laser cladding process

In addition, the unidirectional resulted paths in cases 2 and 3 are due to the disabled cross-hatching functionality, whereas Bi-directional paths are observed in case 1. Furthermore, processing speed is mainly affected by targeted rebuild thickness. Processing speed is higher for less cladding build, which means as the layer thickness increases, processing speed decreases. This was observed for all cases, respectively, comparing the targeted layer thickness in **Table 4.4**, as 0.945, 1.2593, and 1.45, all in mm, with average processing speed in **Table 4.6**, as 10.7, 5.95, and 5.52, all in mm/s. In addition, more transition paths are generated for the partially damaged parts, as shown in case 1. This is

because the laser cladding process is limited to one direction of paths per layer. This means the laser head needs to move to the starting point of every clad trace before starting to clad. In cases 2 and 3, the starting point of a new trace is usually close to the previous trace; thus, fewer transition points are required. Lastly, simulation results will be verified and compared with the actual cladding process experiment in future work.

Table 4.6: RoboDK simulation results

Parameter	Case 1	Case 2	Case 3
Estimated cycle time	45 min.	5 hrs. 48 min.	13 min.
Tilt angle range (deg) (around x) (yaw) *	81.6 – 91.6	60 – 120	80 – 93.8
Path angle (deg) (around y) (pitch) *	+45/-45	0	60
Minimum processing speed (mm/s)	8	5.32	3.23
Average processing speed (mm/s)	10.7	5.95	5.52
Maximum processing speed (mm/s)	21.27	21.26	21.26
Transition speed (mm/s)	100	100	100
Number of processing points	7,875	69,960	2,092
Number of transition points	1,023	32	36
Estimated travel length (m)	123.9	308.206	64.65

* following XYZ Euler angles rotation sequence.

Chapter 5
Conclusion and Future Work

5.1 Conclusions

The circular economy is gaining more attention for its advantages towards sustainability and economic growth. One significant pillar of this model is remanufacturing and repair of used components to restore them to their original state. Restoring a used or damaged component passes over serval steps, including depositing material through an additive manufacturing process. The current repair process relies heavily on a professional worker to achieve this goal. In addition, this process requires a high level of customization since damages and defects exist in a wide variety of geometrical shapes. The current progress of automating this process aims to automatically recognize damaged areas and automatically generate repair paths for the additive manufacturing process, such as laser cladding. In this context, this research presents a semi-autonomous repair system to repair cylindrical components on a robotic cell using a laser metal deposition process. This proposed process pipeline begins with components visual scanning, then defective area detection and localization, and ends with repair path planning of those regions. The presented work has achieved three main goals and is summarized as follows.

One technique to detect defects on a metal surface is using a deep learning model, which has been reported to achieve reliable and accurate results. In addition, those models are highly customizable for different types of defects. With the rapid development in the artificial intelligent sector, several object detectors are emerging every year, thus, a preliminary investigation on the behavior of the state-of-the-art model in object detection for metal defect detection is necessary. In this work, the YOLOv7 model was trained and tested on public datasets containing images of actual damaged surfaces. First, an in-depth study was conducted on all variants of the YOLOv7 using the NEU-DET dataset for a selection of the optimal variant. With the base variant of the YOLOv7 model selected, a study of the detection and segmentation of defects on NRSD and KolektorSDD2 datasets are conducted, and performance was reported. The YOLOv7 has achieved 88.5% mean average precision accuracy of the detection on the inference process when trained and tested on the NSRD

dataset. However, the model struggled to detect defects in the KolektorSDD2 dataset. Going forward, the NRSD dataset was selected for its similarity to our real-world damaged components for the development of the autonomous localization method.

With the bases of defect detection knowledge established previously, the YOLOv7 model is used to detect defective regions in an image of the damaged component's surface and obtain a bounding box of the defect. Further, a novel method to automatically locate the detected bounding of a defect in a spatial space. This method is based on pixel-spatial mapping using a vision sensor. After the location is recognized, a 3D scan of the damaged area is obtained using the proximity sensor for the repair path generation. The developed method is a fully integrated pipeline containing the visual scanning path planning method, image-spatial mapping method, defect detection method, and 3D scanning path planning method. Lastly, the proposed method was tested on two virtual case studies that replicated two actual damaged components.

An optimal repair tool-paths were then generated automatically using a novel developed method that uses the defective point cloud data as input. It introduces a new freeform surface-slicing method based on the mesh projection technique. With this, a reconstructed mesh of the damaged region is generated based on nominal dimension or user input. The output of the slicing method is mesh slices that are used to generate the interpolate processing paths and normal vectors. Then, the robot poses are computed using the processing points and the normal vectors with the surface following method. In addition, minimal robot transition paths were computed using a constrained robot movement method based on a dynamic reference frame. This work was verified using three case studies of actual damaged components providing their damaged regions point clouds.

5.2 Contributions

This research aims to automate the repair process of cylindrical mechanical components using newly developed methods and state-of-the-art technologies. The developed repair system provided promising results for an industrial adaptation and novel methods for the scientific community to build upon. Further, detailed contribution points are listed below.

- Investigated the behavior of all variants of the YOLOv7 model on the NEU-DET dataset for metal defect detection. A selection of the optimal variant was recommended based on the performance observed in this study. In addition, studied

the performance of the base variant of the YOLOv7 on the NRSD and KolektorSDD2 datasets for the detection and segmentation of defects on images. The YOLOv7 has provided promising detection results for different types of defects shown by the three tested datasets. One observation of this study is the YOLOv7 performed better in the NRSD dataset with 88.5% mAP, and worse in the KSDD2 dataset with 65% mAP. The YOLOv7 struggled in detecting crack defects with 54%; however, patches and inclusion defects were detected easily with 92% mAP. The performance is also compared with previously reported studies on the selected datasets. Although other models performed slightly better than the YOLOv7-base, many of those models are based on improved versions of the YOLO series. Thus, this study can provide valuable information for improving the YOLOv7 model to detect defects on metal surfaces.

- Developed a fully integrated pipeline for detecting, localizing, and quantifying defects on cylindrical components. This framework includes developing an automatic path planning method for visual scanning the damaged components on a robotic cell with a dynamic configuration system type utilizing a vision sensor. In addition, a spatial localization method for locating defective regions was also developed. This method is based on an image-spatial mapping technique using the camera calibration matrix. Furthermore, a zigzag path planning method for scanning the damaged areas on a cylindrical component's surface was proposed and verified. This method was based on point projection and k-d tree techniques, which generated optimal 3D scanning paths. A virtual experiment to verify the proposed framework is then carried out. Two different virtual case studies were generated and tested on the virtual cell. The method showed a localization mean accuracy of about 98% in localizing detects on damaged components. In addition, the proposed method reported a 50% reduction in processing time.
- Developed and tested an effective approach to reconstruct damaged surface models based on a mesh projection process to a specified reference dimension using a surface unwrapping technique. Optimal tool-paths and normal vectors were generated for repairing external damage to a product using specified parameters for the laser cladding process. Three case studies were used to illustrate the effectiveness of this algorithm in repairing and restoring different types of damage. In addition, carried out a virtual experiment to validate the generated laser cladding processing paths for

the three cases. The simulation results showed accurate and smooth handling of the repairing process using a minimal tool orientation and movement approach.

5.3 Limitation and Future Work

Despite that this system achieved promising results, there are some limitations that need to be addressed in future work. The list below works as expanding pathways from the author's perspective to enhance and improve this system in several aspects.

- For the detection of the defect on metal surfaces, the YOLOv7 showed a good performance in terms of accuracy and speed for one class of defect. However, degradation of the YOLOv7 performance has been observed when multiple classes of defects are used. As result, improving the YOLOv7 performance with multiple classes is required. According to the literature, integrating transformers into the backbone of the YOLO model series is one technique to improve and boost the accuracy of the model. Integrating the swin transformer with YOLOv7 is worth investigating for better performance on feature maps extraction of defects.
- Deploying the defect detection model to an industrial application requires more fine-tuning to the specific problem for reliable and accurate results. This tunning process is achieved by training the detection model on real-life defect samples. The performance of the detection model is sensitive and biased to the collected dataset, which might raise some challenges in achieving the desired results. For future development of this pipeline, a custom real-life dataset needs to be collected.
- In the localization process, one limitation of the system is assuming constant radius components, thus limiting the system from processing any further extended features. In practice, components with different 3D features are most common to encounter in real life; thus, processing cylindrical components with extended features are worth investigating in the future. One improvement to the localization process, specifically the image-spatial mapping process, is to estimate the radius of the scanned components using a deep learning model by estimating the depth in the collected images, where the depth is calculated from the component's radius provided by the user in the current implementation. Another enhancement to the localization process that might be considered investigating is confirming the depth component of the

detected defect bounding box using the ToF sensor before generating the scanning plan and paths for acquiring the point clouds.

- The current implementation of the defect quantification method is based on the spatial location of the four corners of its bounding box, which is limited to a rectangular shape containing the defective region and some surrounding areas. A primitive shape of segmentation can be obtained by 2D shape fitting to further reduce the scanning and repairing time by constraining the quantification of the defective region only. In addition, this step can be used for confirming repairable defectives by computing their area and aspect ratio.
- Adding flexibility to this framework by allowing the user through an interactive selection of specific ROI in the image-spatial map can boost its adaptation for different industrial applications and overcome some restrictions and limitations of the deep learning model.
- Furthermore, one major improvement that can be adopted is using a non-planar slicer and clipper, which is feasible using the curvilinear coordinate system. This will enable further control over the generated paths and orientations and, ultimately, the deposited material for enhanced mechanical performance. Furthermore, other planning techniques, such as spiral planning, thermal-dependent planning, etc. should be explored in the future for further optimized paths.
- It is also important to mention that this method is constrained to repair damaged components whose nominal geometry is locally planar, cylindrical, or spherical. Further investigation of repairing complex geometries is worth consideration in the future. Additionally, adding new 3D features or repairing damaged ones on cylindrical components is significant for the next step of this project. Extending the path planning algorithm to remanufacturing capabilities by generating additive paths for a newly added feature is an urgent need that will expand the framework's capabilities.
- This framework was developed to process both external and internal diameters meaning it is designed to be adopted for two different robotic cell configurations. However, the system was tested on repairing external damages only with an external robotic laser head cell explained in this thesis. Further examination and experiments on processing internal damages are required and considered in the future.

- The developed path planning method can generate paths of features that are based on plans, cylinders, and spheres; however, only planner and cylindrical features were tested on the path planning method, whereas spherical geometries will be considered for future work. This is mainly because the author currently does not have access to actual damage data for the spherical system configuration.
- The automatic tool path generation method produced good results in the virtual environment in terms of the smoothness of the robot movement and the effectiveness of the deposition paths. However, an actual repair process is required to verify the repaired geometry. Further, a metrological study of the repaired patch is important to investigate the material and mechanical properties, including porosity percentage.
- Achieving repairing and remanufacturing requirements for the processed components in terms of geometry and performance is challenging. A post-process clad inspection method should be adopted for the quality control step. A post-clad 3D scanning process can be performed to obtain topography information of the cladded region and inspect with the normal or target rebuild.

Bibliography

[1] A.P.M. Velenturf, P. Purnell, Principles for a sustainable circular economy, Sustain. Prod. Consum. 27 (2021) 1437–1457. https://doi.org/10.1016/j.spc.2021.02.018.

[2] P. Nohra, H. Ben Rejeb, S. Venkateswaran, Impact of automation during innovative remanufacturing processes in circular economy: a state of the art, (2022). https://arxiv.org/abs/2209.04040v1.

[3] S.S. Mad Yusoh, D. Abd Wahab, H.A. Habeeb, A.H. Azman, Intelligent systems for additive manufacturing-based repair in remanufacturing: a systematic review of its potential, PeerJ Comput. Sci. 7 (2021) 1–34. https://doi.org/10.7717/peerj-cs.808.

[4] A. Heshmati, A review of the circular economy and its implementation, Int. J. Green Econ. 11 (2017) 251–288. https://doi.org/10.1504/IJGE.2017.089856.

[5] T.A. Branca, V. Colla, D. Algermissen, H. Granbom, U. Martini, A. Morillon, R. Pietruck, S. Rosendahl, Reuse and recycling of by-products in the steel sector: Recent achievements paving the way to circular economy and industrial symbiosis in europe, Metals. 10 (2020). https://doi.org/10.3390/met10030345.

[6] S. Ponis, E. Aretoulaki, T.N. Maroutas, G. Plakas, K. Dimogiorgi, A systematic literature review on additive manufacturing in the context of circular economy, Sustain. 13 (2021). https://doi.org/10.3390/su13116007.

[7] Rahito, D.A. Wahab, A.H. Azman, Additive manufacturing for repair and restoration in remanufacturing: An overview from object design and systems perspectives, Processes. 7 (2019). https://doi.org/10.3390/pr7110802.

[8] J. Priyadarshini, R. Kr Singh, R. Mishra, M. Mustafa Kamal, Adoption of additive manufacturing for sustainable operations in the era of circular economy: Self-assessment framework with case illustration, Comput. Ind. Eng. 171 (2022) 108514. https://doi.org/10.1016/j.cie.2022.108514.

[9] V. Madhavadas, D. Srivastava, U. Chadha, S. Aravind Raj, M.T.H. Sultan, F.S. Shahar, A.U.M. Shah, A review on metal additive manufacturing for intricately shaped aerospace components, CIRP J. Manuf. Sci. Technol. 39 (2022) 18–36.

https://doi.org/10.1016/j.cirpj.2022.07.005.

[10] G. Piscopo, L. Iuliano, Current research and industrial application of laser powder directed energy deposition, Int. J. Adv. Manuf. Technol. 119 (2022) 6893–6917. https://doi.org/10.1007/s00170-021-08596-w.

[11] P.M. Bhatt, R.K. Malhan, A. V. Shembekar, Y.J. Yoon, S.K. Gupta, Expanding capabilities of additive manufacturing through use of robotics technologies: A survey, Addit. Manuf. 31 (2020) 100933. https://doi.org/10.1016/j.addma.2019.100933.

[12] P. Urhal, A. Weightman, C. Diver, P. Bartolo, Robot assisted additive manufacturing: A review, Robot. Comput. Integr. Manuf. 59 (2019) 335–345. https://doi.org/10.1016/j.rcim.2019.05.005.

[13] Meltio, Metal 3D Printing Applications, (2022). https://meltio3d.com/applications/.

[14] S. Arsova, A. Genovese, P.H. Ketikidis, Implementing circular economy in a regional context: A systematic literature review and a research agenda, J. Clean. Prod. 368 (2022) 133117. https://doi.org/10.1016/j.jclepro.2022.133117.

[15] M. Yang, L. Chen, J. Wang, G. Msigwa, A.I. Osman, S. Fawzy, D.W. Rooney, P.S. Yap, Circular economy strategies for combating climate change and other environmental issues, Environ. Chem. Lett. (2022). https://doi.org/10.1007/s10311-022-01499-6.

[16] X. Zhang, R. Liu, W. Yan, Y. Wang, N. Subramanian, Systematic Literature Review on Remanufacturing Trade Based on Bibliometric Analysis, Processes. 10 (2022) 1–18. https://doi.org/10.3390/pr10030596.

[17] H. Shekhar, R. Dumpala, Overcoming friction and steps towards superlubricity: A review of underlying mechanisms, Appl. Surf. Sci. Adv. 6 (2021) 100175. https://doi.org/10.1016/j.apsadv.2021.100175.

[18] H.Z. Imam, H. Al-Musaibeli, Y. Zheng, P. Martinez, R. Ahmad, Vision-based spatial damage localization method for autonomous robotic laser cladding repair processes, Robot. Comput. Integr. Manuf. 80 (2023) 102452. https://doi.org/10.1016/j.rcim.2022.102452.

[19] J. Yang, S. Li, Z. Wang, H. Dong, J. Wang, S. Tang, Using deep learning to detect defects in manufacturing: A comprehensive survey and current challenges, Materials.

13 (2020) 1–23. https://doi.org/10.3390/ma13245755.

[20] R. Mordia, A. Kumar Verma, Visual techniques for defects detection in steel products: A comparative study, Eng. Fail. Anal. 134 (2022) 106047. https://doi.org/10.1016/j.engfailanal.2022.106047.

[21] X. Tao, D. Zhang, W. Ma, X. Liu, De Xu, Automatic metallic surface defect detection and recognition with convolutional neural networks, Appl. Sci. 8 (2018) 1–15. https://doi.org/10.3390/app8091575.

[22] M.M. Parvez, M.F. Rahman, S.M. Galib, F. Liou, a Convolutional Neural Network (Cnn) for Defect Detection of Additively Manufactured Parts, ASME Int. Mech. Eng. Congr. Expo. Proc. 2A-2021 (2021) 1–11. https://doi.org/10.1115/IMECE2021-70500.

[23] F. Han, S. Liu, S. Liu, J. Zou, Y. Ai, C. Xu, Defect detection: Defect Classification and Localization for Additive Manufacturing using Deep Learning Method, in: 2020 21st Int. Conf. Electron. Packag. Technol., IEEE, 2020: pp. 1–4. https://doi.org/10.1109/ICEPT50128.2020.9202566.

[24] R. Xu, R. Hao, B. Huang, Efficient surface defect detection using self-supervised learning strategy and segmentation network, Adv. Eng. Informatics. 52 (2022) 101566. https://doi.org/10.1016/j.aei.2022.101566.

[25] H. Wu, Q. Lv, Hot-Rolled Steel Strip Surface Inspection Based on Transfer Learning Model, J. Sensors. 2021 (2021). https://doi.org/10.1155/2021/6637252.

[26] Y. Zheng, H. Mamledesai, H. Imam, R. Ahmad, A novel deep learning-based automatic damage detection and localization method for remanufacturing/repair, Comput. Aided. Des. Appl. 18 (2021) 1359–1372. https://doi.org/10.14733/cadaps.2021.1359-1372.

[27] H.Z. Imam, Y. Zheng, P. Martinez, R. Ahmad, Vision-Based Damage Localization Method for an Autonomous Robotic Laser Cladding Process, Procedia CIRP. 104 (2021) 827–832. https://doi.org/10.1016/j.procir.2021.11.139.

[28] Y. Zheng, H. Mamledesai, H. Imam, R. Ahmad, Deep Learning-based Automatic Damage Recognition and Spatial Localization for Remanufacturing/Repair, (2020) 381–385. https://doi.org/10.14733/cadconfp.2020.381-385.

[29] Z. Zheng, H. Qi, L. Zhuang, Z. Zhang, Automated rail surface crack analytics using deep data-driven models and transfer learning, Sustain. Cities Soc. 70 (2021) 102898. https://doi.org/10.1016/j.scs.2021.102898.

[30] I. Konovalenko, P. Maruschak, J. Brezinová, O. Prentkovskis, J. Brezina, Research of U-Net-Based CNN Architectures for Metal Surface Defect Detection, Machines. 10 (2022) 327. https://doi.org/10.3390/machines10050327.

[31] A. Litvintseva, O. Evstafev, S. Shavetov, Real-time Steel Surface Defect Recognition Based on CNN, IEEE Int. Conf. Autom. Sci. Eng. 2021-Augus (2021) 1118–1123. https://doi.org/10.1109/CASE49439.2021.9551414.

[32] H. Zahir, H. Al-musaibeli, Y. Zheng, P. Martinez, R. Ahmad, Robotics and Computer-Integrated Manufacturing Vision-based spatial damage localization method for autonomous robotic laser cladding repair processes, Robot. Comput. Integr. Manuf. 80 (2023) 102452. https://doi.org/10.1016/j.rcim.2022.102452.

[33] G. Li, R. Shao, H. Wan, M. Zhou, M. Li, A Model for Surface Defect Detection of Industrial Products Based on Attention Augmentation, Comput. Intell. Neurosci. 2022 (2022). https://doi.org/10.1155/2022/9577096.

[34] Y. Pan, L. Zhang, Dual attention deep learning network for automatic steel surface defect segmentation, Comput. Civ. Infrastruct. Eng. 37 (2022) 1468–1487. https://doi.org/10.1111/mice.12792.

[35] J. Wang, G. Xu, F. Yan, J. Wang, Z. Wang, Defect Transformer: An Efficient Hybrid Transformer Architecture for Surface Defect Detection, 14 (2022) 1–12. http://arxiv.org/abs/2207.08319.

[36] L. Gao, J. Zhang, C. Yang, Y. Zhou, Cas-VSwin transformer: A variant swin transformer for surface-defect detection, Comput. Ind. 140 (2022) 103689. https://doi.org/10.1016/j.compind.2022.103689.

[37] L. Alzubaidi, J. Zhang, A.J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M.A. Fadhel, M. Al-Amidie, L. Farhan, Review of deep learning: concepts, CNN architectures, challenges, applications, future directions, Springer International Publishing, 2021. https://doi.org/10.1186/s40537-021-00444-8.

[38] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, C. Liu, A survey on deep transfer learning, Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 11141 LNCS (2018) 270–279. https://doi.org/10.1007/978-3-030-01424-7_27.

[39] T.Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft COCO: Common objects in context, Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 8693 LNCS (2014) 740–755. https://doi.org/10.1007/978-3-319-10602-1_48.

[40] J. Li, Z. Su, J. Geng, Y. Yin, Real-time Detection of Steel Strip Surface Defects Based on Improved YOLO Detection Network, IFAC-PapersOnLine. 51 (2018) 76–81. https://doi.org/10.1016/j.ifacol.2018.09.412.

[41] X. Kou, S. Liu, K. Cheng, Y. Qian, Development of a YOLO-V3-based model for detecting defects on steel strip surface, Meas. J. Int. Meas. Confed. 182 (2021) 109454. https://doi.org/10.1016/j.measurement.2021.109454.

[42] Y. Xu, K. Zhang, L. Wang, Metal surface defect detection using modified yolo, Algorithms. 14 (2021). https://doi.org/10.3390/A14090257.

[43] Z. Guo, C. Wang, G. Yang, Huang, G. Li, MSFT-YOLO: Improved YOLOv5 Based on Transformer for Detecting Defects of Steel Surface, Sensors. 22 (2022). https://doi.org/10.3390/s22093467.

[44] C.-Y. Wang, A. Bochkovskiy, H.-Y.M. Liao, YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, (2022) 1–15. http://arxiv.org/abs/2207.02696.

[45] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 2016-Decem (2016) 779–788. https://doi.org/10.1109/CVPR.2016.91.

[46] J. Redmon, A. Farhadi, YOLO9000: Better, faster, stronger, Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017. 2017-Janua (2017) 6517–6525. https://doi.org/10.1109/CVPR.2017.690.

[47] J. Redmon, A.F. ön baskı arXiv:1804.02767, undefined 2018, Yolov3: Artımlı bir

gelişme, Arxiv.Org. (n.d.). https://arxiv.org/abs/1804.02767.

[48] X. Lhrg, YOLOv4: Optimal Speed and Accuracy of Object Detection arXiv:2004.10934v1, (n.d.).

[49] C.-Y. Wang, H.-Y.M. Liao, H. Kinsley, D. Kukieła, A. Meckel, Scaled-YOLOv4: Scaling Cross Stage Partial Network arXiv:2011.08036v2, (2017) 1–53. https://doi.org/10.48550/arxiv.2011.08036.

[50] Z. Ge, S. Liu, F. Wang, Z. Li, J. Sun, YOLOX: Exceeding YOLO Series in 2021, 5 (2021) 12. https://doi.org/https://doi.org/10.48550/arXiv.2107.08430.

[51] C.-Y. Wang, I.-H. Yeh, H.-Y.M. Liao, You Only Learn One Representation: Unified Network for Multiple Tasks, (2021) 1–11. http://arxiv.org/abs/2105.04206.

[52] P. Jiang, D. Ergu, F. Liu, Y. Cai, B. Ma, A Review of Yolo Algorithm Developments, Procedia Comput. Sci. 199 (2021) 1066–1073. https://doi.org/10.1016/j.procs.2022.01.135.

[53] E. Sari, M. Belbahri, V.P. Nia, How Does Batch Normalization Help Binary Training?, (2019). http://arxiv.org/abs/1909.09139.

[54] Severstal, Severstal: Steel Defect Detection, [Dataset]. (2022). https://www.kaggle.com/competitions/severstal-steel-defect-detection/data.

[55] D. Zhang, K. Song, J. Xu, Y. He, M. Niu, Y. Yan, MCnet: Multiple Context Information Segmentation Network of No-Service Rail Surface Defects, IEEE Trans. Instrum. Meas. 70 (2021). https://doi.org/10.1109/TIM.2020.3040890.

[56] J. Božič, D. Tabernik, D. Skočaj, Mixed supervision for surface-defect detection: From weakly to fully supervised learning, Comput. Ind. 129 (2021). https://doi.org/10.1016/j.compind.2021.103459.

[57] F.A.H. Matthias Wieler, Tobias Hahn, Weakly supervised learning for industrial optical inspection [Dataset], (2007). https://hci.iwr.uni-heidelberg.de/content/weakly-supervised-learning-industrial-optical-inspection.

[58] X. Lv, F. Duan, J.J. Jiang, X. Fu, L. Gan, Deep metallic surface defect detection: The new benchmark and detection network, Sensors (Switzerland). 20 (2020).

https://doi.org/10.3390/s20061562.

[59] K. Song, Y. Yan, A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects, Appl. Surf. Sci. 285 (2013) 858–864. https://doi.org/10.1016/j.apsusc.2013.09.002.

[60] Ball Screw Drive Surface Defect Dataset for Classification, Inst. Prod. Technol. (2021). https://doi.org/https://doi.org/10.5445/IR/1000133819.

[61] D. Tabernik, S. Šela, J. Skvarč, D. Skočaj, Segmentation-based deep-learning approach for surface-defect detection, J. Intell. Manuf. 31 (2020) 759–776. https://doi.org/10.1007/s10845-019-01476-x.

[62] J. Gan, Q. Li, J. Wang, H. Yu, A Hierarchical Extractor-Based Visual Rail Surface Inspection System, IEEE Sens. J. 17 (2017) 7935–7944. https://doi.org/10.1109/JSEN.2017.2761858.

[63] M. Geissdoerfer, P. Savaget, N.M.P. Bocken, E.J. Hultink, The Circular Economy – A new sustainability paradigm?, J. Clean. Prod. 143 (2017) 757–768. https://doi.org/10.1016/j.jclepro.2016.12.048.

[64] M. Barreiro-Gen, R. Lozano, How circular is the circular economy? Analysing the implementation of circular economy in organisations, Bus. Strateg. Environ. 29 (2020) 3484–3494. https://doi.org/10.1002/bse.2590.

[65] J.M. Wilson, C. Piya, Y.C. Shin, F. Zhao, K. Ramani, Remanufacturing of turbine blades by laser direct deposition with its energy and environmental impact analysis, J. Clean. Prod. 80 (2014) 170–178. https://doi.org/10.1016/j.jclepro.2014.05.084.

[66] J. Camacho-Otero, C. Boks, I. Pettersen, Consumption in the Circular Economy: A Literature Review, Sustainability. 10 (2018) 2758. https://doi.org/10.3390/su10082758.

[67] Rahito, D. Wahab, A. Azman, Additive Manufacturing for Repair and Restoration in Remanufacturing: An Overview from Object Design and Systems Perspectives, Processes. 7 (2019) 802. https://doi.org/10.3390/pr7110802.

[68] J. Liu, Y. Zheng, Y. Ma, A. Qureshi, R. Ahmad, A Topology Optimization Method for Hybrid Subtractive–Additive Remanufacturing, Int. J. Precis. Eng. Manuf. Technol. 7

(2020) 939–953. https://doi.org/10.1007/s40684-019-00075-8.

[69] Y. Zheng, A.J. Qureshi, R. Ahmad, Algorithm for remanufacturing of damaged parts with hybrid 3D printing and machining process, Manuf. Lett. 15 (2018) 38–41. https://doi.org/10.1016/j.mfglet.2018.02.010.

[70] Y. Zheng, J. Liu, R. Ahmad, A cost-driven process planning method for hybrid additive–subtractive remanufacturing, J. Manuf. Syst. 55 (2020) 248–263. https://doi.org/10.1016/j.jmsy.2020.03.006.

[71] L. Zhu, P. Xue, Q. Lan, G. Meng, Y. Ren, Z. Yang, P. Xu, Z. Liu, Recent research and development status of laser cladding: A review, Opt. Laser Technol. 138 (2021) 106915. https://doi.org/10.1016/j.optlastec.2021.106915.

[72] D. Raj, S.R. Maity, B. Das, State-of-the-art review on laser cladding process as an in-situ repair technique, Proc. Inst. Mech. Eng. Part E J. Process Mech. Eng. 236 (2022) 1194–1215. https://doi.org/10.1177/09544089211044558.

[73] A. Saboori, A. Aversa, G. Marchese, S. Biamino, M. Lombardi, P. Fino, Application of directed energy deposition-based additive manufacturing in repair, Appl. Sci. 9 (2019). https://doi.org/10.3390/app9163316.

[74] M. Perini, P. Bosetti, N. Balc, Additive manufacturing for repairing: from damage identification and modeling to DLD, Rapid Prototyp. J. 26 (2020) 929–940. https://doi.org/10.1108/RPJ-03-2019-0090.

[75] X. Zhang, W. Cui, W. Li, F. Liou, A Hybrid Process Integrating Reverse Engineering, Pre-Repair Processing, Additive Manufacturing, and Material Testing for Component Remanufacturing, Materials. 12 (2019) 1961. https://doi.org/10.3390/ma12121961.

[76] X. Li, Q. Han, G. Zhang, Large-size sprocket repairing based on robotic GMAW additive manufacturing, Weld. World. 65 (2021) 793–805. https://doi.org/10.1007/s40194-021-01080-9.

[77] L. Li, C. Li, Y. Tang, Y. Du, An integrated approach of reverse engineering aided remanufacturing process for worn components, Robot. Comput. Integr. Manuf. 48 (2017) 39–50. https://doi.org/10.1016/j.rcim.2017.02.004.

[78] D. Liu, J.C. Lippold, J. Li, S.R. Rohklin, J. Vollbrecht, R. Grylls, Laser Engineered Net

Shape (LENS) Technology for the Repair of Ni-Base Superalloy Turbine Components, Metall. Mater. Trans. A. 45 (2014) 4454–4469. https://doi.org/10.1007/s11661-014-2397-8.

[79] M. Berger, A. Tagliasacchi, L.M. Seversky, P. Alliez, J. a. Levine, A. Sharf, C.T. Silva, A. Tagliasacchi, L.M. Seversky, C.T. Silva, J. a. Levine, A. Sharf, State of the Art in Surface Reconstruction from Point Clouds, Proc. Eurographics 2014, Eurographics Stars. 1 (2014) 161–185. https://doi.org/10.2312/egst.20141040.

[80] Y. He, B. Liang, J. Yang, S. Li, J. He, An Iterative Closest Points Algorithm for Registration of 3D Laser Scanner Point Clouds with Geometric Features, Sensors. 17 (2017) 1862. https://doi.org/10.3390/s17081862.

[81] Y. Liu, B. Fan, S. Xiang, C. Pan, Relation-shape convolutional neural network for point cloud analysis, Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 2019-June (2019) 8887–8896. https://doi.org/10.1109/CVPR.2019.00910.

[82] E. Grilli, F. Menna, F. Remondino, A REVIEW OF POINT CLOUDS SEGMENTATION AND CLASSIFICATION ALGORITHMS, Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. XLII-2/W3 (2017) 339–344. https://doi.org/10.5194/isprs-archives-XLII-2-W3-339-2017.

[83] X. Ruan, B. Liu, Review of 3D Point Cloud Data Segmentation Methods, Int. J. Adv. Network, Monit. Control. 5 (2020) 66–71. https://doi.org/10.21307/ijanmc-2020-010.

[84] A. Nguyen, B. Le, 3D point cloud segmentation: A survey, in: 2013 6th IEEE Conf. Robot. Autom. Mechatronics, IEEE, 2013: pp. 225–230. https://doi.org/10.1109/RAM.2013.6758588.

[85] M.A. Wani, H.R. Arabnia, Parallel edge-region-based segmentation algorithm targeted at reconfigurable MultiRing network, J. Supercomput. 25 (2003) 43–62. https://doi.org/10.1023/A:1022804606389.

[86] M. Breuß, A. Bruckstein, P. Maragos, eds., Innovations for Shape Analysis, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. https://doi.org/10.1007/978-3-642-34141-0.

[87] K. Koster, M. Spann, MIR: an approach to robust clustering-application to range image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (2000) 430–444. https://doi.org/10.1109/34.857001.

[88] R.B. Rusu, Z.C. Marton, N. Blodow, M. Dolha, M. Beetz, Towards 3D Point cloud based object maps for household environments, Rob. Auton. Syst. 56 (2008) 927–941. https://doi.org/10.1016/j.robot.2008.08.005.

[89] Y. Liu, Y. Xiong, Automatic segmentation of unorganized noisy point clouds based on the Gaussian map, Comput. Des. 40 (2008) 576–594. https://doi.org/10.1016/j.cad.2008.02.004.

[90] R. Schnabel, R. Wahl, R. Klein, Efficient RANSAC for Point-Cloud Shape Detection, Comput. Graph. Forum. 26 (2007) 214–226. https://doi.org/10.1111/j.1467-8659.2007.01016.x.

[91] F. Tarsha-Kurdi, T. Landes, P. Grussenmeyer, Hough-Transform and Extended Ransac Algorithms for Automatic Detection of 3D Building Roof Planes From Lidar Data, ISPRS Work. Laser Scanning 2007 SilviLaser 2007. XXXVI (2007) 407–412.

[92] A. Adam, E. Chatzilari, S. Nikolopoulos, I. Kompatsiaris, H-RANSAC: A HYBRID POINT CLOUD SEGMENTATION COMBINING 2D AND 3D DATA, ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci. IV–2 (2018) 1–8. https://doi.org/10.5194/isprs-annals-IV-2-1-2018.

[93] J. Zhang, X. Zhao, Z. Chen, Z. Lu, A Review of Deep Learning-Based Semantic Segmentation for Point Cloud, IEEE Access. 7 (2019) 179118–179133. https://doi.org/10.1109/ACCESS.2019.2958671.

[94] B. Wu, B. Yu, Q. Wu, S. Yao, F. Zhao, W. Mao, J. Wu, A Graph-Based Approach for 3D Building Model Reconstruction from Airborne LiDAR Point Clouds, Remote Sens. 9 (2017) 92. https://doi.org/10.3390/rs9010092.

[95] T. Hitchcox, Y.F. Zhao, Random walks for unorganized point cloud segmentation with application to aerospace repair, Procedia Manuf. 26 (2018) 1483–1491. https://doi.org/10.1016/j.promfg.2018.07.093.

[96] L. Chen, X. Yao, P. Xu, S.K. Moon, G. Bi, Surface Monitoring for Additive

Manufacturing with in-situ Point Cloud Processing, in: 2020 6th Int. Conf. Control. Autom. Robot., IEEE, 2020: pp. 196–201. https://doi.org/10.1109/ICCAR49639.2020.9108092.

[97] Z. Wang, D. Zhu, An accurate detection method for surface defects of complex components based on support vector machine and spreading algorithm, Measurement. 147 (2019) 106886. https://doi.org/10.1016/j.measurement.2019.106886.

[98] W. Xiao, G. Liu, G. Zhao, Generating the tool path directly with point cloud for aero-engine blades repair, Proc. Inst. Mech. Eng. Part B J. Eng. Manuf. 235 (2021) 877–886. https://doi.org/10.1177/0954405420970915.

[99] Y. He, W. Ma, Y. Li, C. Hao, Y. Wang, Y. Wang, An Octree-Based Two-Step Method of Surface Defects Detection for Remanufacture, Int. J. Precis. Eng. Manuf. Technol. (2022). https://doi.org/10.1007/s40684-022-00433-z.

[100] Y. Chen, Y. Ding, F. Zhao, E. Zhang, Z. Wu, L. Shao, Surface Defect Detection Methods for Industrial Products: A Review, Appl. Sci. 11 (2021) 7657. https://doi.org/10.3390/app11167657.

[101] P. Prasitmeeboon, H. Yau, Defect Detection of Particleboards by Visual Analysis and Machine Learning, in: 2019 5th Int. Conf. Eng. Appl. Sci. Technol., IEEE, 2019: pp. 1–4. https://doi.org/10.1109/ICEAST.2019.8802526.

[102] N. Ma, X. Gao, C. Wang, Y. Zhang, D. You, N. Zhang, Influence of Hysteresis Effect on Contrast of Welding Defects Profile in Magneto-Optical Image, IEEE Sens. J. 20 (2020) 15034–15042. https://doi.org/10.1109/JSEN.2020.3009478.

[103] R. Li, F. Tian, S. Chen, Research on Surface Defect Detection Method of E-TPU Midsole Based on Machine Vision, J. Comput. Commun. 08 (2020) 145–160. https://doi.org/10.4236/jcc.2020.811011.

[104] D.-M. Tsai, C.-K. Huang, Defect Detection in Electronic Surfaces Using Template-Based Fourier Image Reconstruction, IEEE Trans. Components, Packag. Manuf. Technol. 9 (2019) 163–172. https://doi.org/10.1109/TCPMT.2018.2873744.

[105] Q. Luo, X. Fang, L. Liu, C. Yang, Y. Sun, Automated Visual Defect Detection for Flat

Steel Surface: A Survey, IEEE Trans. Instrum. Meas. 69 (2020) 626–644. https://doi.org/10.1109/TIM.2019.2963555.

[106] T. Czimmermann, G. Ciuti, M. Milazzo, M. Chiurazzi, S. Roccella, C.M. Oddo, P. Dario, Visual-Based Defect Detection and Classification Approaches for Industrial Applications—A SURVEY, Sensors. 20 (2020) 1459. https://doi.org/10.3390/s20051459.

[107] A. Prahas Putri, H. Rachmat, D.S. Eka Atmaja, Design of Automation System for Ceramic Surface Quality Control Using Fuzzy Logic Method at Balai Besar Keramik (BBK), MATEC Web Conf. 135 (2017) 00053. https://doi.org/10.1051/matecconf/201713500053.

[108] Y. Liu, K. Xu, J. Xu, An Improved MB-LBP Defect Recognition Approach for the Surface of Steel Plates, Appl. Sci. 9 (2019) 4222. https://doi.org/10.3390/app9204222.

[109] X. Fang, Q. Luo, B. Zhou, C. Li, L. Tian, Research Progress of Automated Visual Surface Defect Detection for Industrial Metal Planar Materials, Sensors. 20 (2020) 5136. https://doi.org/10.3390/s20185136.

[110] X. Tao, D. Zhang, W. Ma, X. Liu, D. Xu, Automatic Metallic Surface Defect Detection and Recognition with Convolutional Neural Networks, Appl. Sci. 8 (2018) 1575. https://doi.org/10.3390/app8091575.

[111] Y. Huang, C. Qiu, X. Wang, S. Wang, K. Yuan, A compact convolutional neural network for surface defect inspection, Sensors (Switzerland). 20 (2020) 1–19. https://doi.org/10.3390/s20071974.

[112] H. Zhang, Z. Chen, C. Zhang, J. Xi, X. Le, Weld Defect Detection Based on Deep Learning Method, in: 2019 IEEE 15th Int. Conf. Autom. Sci. Eng., IEEE, 2019: pp. 1574–1579. https://doi.org/10.1109/COASE.2019.8842998.

[113] J. Zhang, G. Cosma, J. Watkins, Image Enhanced Mask R-CNN: A Deep Learning Pipeline with New Evaluation Measures for Wind Turbine Blade Defect Detection and Classification, J. Imaging. 7 (2021) 46. https://doi.org/10.3390/jimaging7030046.

[114] H. Wang, M. Li, Z. Wan, Rail surface defect detection based on improved Mask R-CNN,

Comput. Electr. Eng. 102 (2022) 108269. https://doi.org/10.1016/j.compeleceng.2022.108269.

[115] G. Bradski, The OpenCV Library, Dr. Dobb's J. Softw. Tools. (2000).

[116] T. Schlagenhauf, T. Brander, J. Fleischer, A stitching algorithm for automated surface inspection of rotationally symmetric components, CIRP J. Manuf. Sci. Technol. 35 (2021) 169–177. https://doi.org/10.1016/j.cirpj.2021.05.013.

[117] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin Transformer: Hierarchical Vision Transformer using Shifted Windows, (2021). http://arxiv.org/abs/2103.14030.

[118] G. Ingarao, Manufacturing strategies for efficiency in energy and resources use: The role of metal shaping processes, J. Clean. Prod. 142 (2017) 2872–2886. https://doi.org/10.1016/j.jclepro.2016.10.182.

[119] X. Xia, C. Zhang, The impact of authorized remanufacturing on sustainable remanufacturing, Processes. 7 (2019) 1–12. https://doi.org/10.3390/pr7100663.

[120] W. Huang, Z. Jiang, T. Wang, Y. Wang, X. Hu, Remanufacturing Scheme Design for Used Parts Based on Incomplete Information Reconstruction, Chinese J. Mech. Eng. 33 (2020) 41. https://doi.org/10.1186/s10033-020-00457-z.

[121] N.A. Aziz, N.A.A. Adnan, D.A. Wahab, A.H. Azman, Component design optimisation based on artificial intelligence in support of additive manufacturing repair and restoration: Current status and future outlook for remanufacturing, J. Clean. Prod. 296 (2021) 126401. https://doi.org/10.1016/j.jclepro.2021.126401.

[122] Y. Zheng, J. Liu, R. Ahmad, A cost-driven process planning method for hybrid additive–subtractive remanufacturing, J. Manuf. Syst. 55 (2020) 248–263. https://doi.org/10.1016/j.jmsy.2020.03.006.

[123] Y. Zheng, A.J. Qureshi, R. Ahmad, Algorithm for remanufacturing of damaged parts with hybrid 3D printing and machining process, Manuf. Lett. 15 (2018) 38–41. https://doi.org/10.1016/j.mfglet.2018.02.010.

[124] Y. Zheng, R. Ahmad, Automated Feature Extraction for Hybrid Additive-Subtractive Remanufacturing, Procedia CIRP. 93 (2020) 56–61.

https://doi.org/10.1016/j.procir.2020.04.092.

[125] J. Liu, Q. Chen, Y. Zheng, R. Ahmad, J. Tang, Y. Ma, Level set-based heterogeneous object modeling and optimization, Comput. Des. 110 (2019) 50–68. https://doi.org/10.1016/j.cad.2019.01.002.

[126] A.A. Siddiqui, A.K. Dubey, Recent trends in laser cladding and surface alloying, Opt. Laser Technol. 134 (2021) 106619. https://doi.org/10.1016/j.optlastec.2020.106619.

[127] N. Tamanna, R. Crouch, S. Naher, Progress in numerical simulation of the laser cladding process, Opt. Lasers Eng. 122 (2019) 151–163. https://doi.org/10.1016/J.OPTLASENG.2019.05.026.

[128] R.M. Mahamood, E.T. Akinlabi, M.G. Owolabi, Laser Metal Deposition Process for Product Remanufacturing, in: K. Gupta (Ed.), Adv. Manuf. Technol. Mater. Forming, Mach. Tribol., Springer, 2017: pp. 267–291. https://doi.org/10.1007/978-3-319-56099-1_12.

[129] R. Chen, H. Yin, I.S. Cole, S. Shen, X. Zhou, Y. Wang, S. Tang, Exposure, assessment and health hazards of particulate matter in metal additive manufacturing: A review, Chemosphere. 259 (2020) 127452. https://doi.org/10.1016/j.chemosphere.2020.127452.

[130] J.I. Arrizubieta, O. Ukar, M. Ostolaza, A. Mugica, Study of the Environmental Implications of Using Metal Powder in Additive Manufacturing and Its Handling, Metals. 10 (2020) 261. https://doi.org/10.3390/met10020261.

[131] S. Abreez Gillani, C. Singh, N. Raj, H. Al-Musaibeli, P. Panta, R. Ahmad, Implementation of Lean Tools to Improve Mass Production of a Laser Cladding Process; Implementation of Lean Tools to Improve Mass Production of a Laser Cladding Process, 2021 9th Int. Conf. Control. Mechatronics Autom. (2021). https://doi.org/10.1109/ICCMA54375.2021.9646209.

[132] R. Ahmad, P. Plapper, Generation of safe tool-path for 2.5D milling/drilling machine-tool using 3D ToF sensor, CIRP J. Manuf. Sci. Technol. 10 (2015) 84–91. https://doi.org/10.1016/J.CIRPJ.2015.04.003.

[133] C. Feng, J. Liang, C. Gong, W. Pai, S. Liu, Repair volume extraction method for

damaged parts in remanufacturing repair, Int. J. Adv. Manuf. Technol. 98 (2018) 1523–1536. https://doi.org/10.1007/s00170-018-2300-7.

[134] X. Zhang, W. Li, W. Cui, F. Liou, Modeling of worn surface geometry for engine blade repair using Laser-aided Direct Metal Deposition process, Manuf. Lett. 15 (2018) 1–4. https://doi.org/10.1016/j.mfglet.2017.11.001.

[135] X. Zhang, W. Li, F. Liou, Damage detection and reconstruction algorithm in repairing compressor blade by direct metal deposition, Int. J. Adv. Manuf. Technol. 95 (2018) 2393–2404. https://doi.org/10.1007/s00170-017-1413-8.

[136] X. Zhang, W. Cui, F. Liou, Voxel-Based Geometry Reconstruction for Repairing and Remanufacturing of Metallic Components Via Additive Manufacturing, Int. J. Precis. Eng. Manuf. Technol. 8 (2021) 1663–1686. https://doi.org/10.1007/s40684-020-00291-7.

[137] Y. Zheng, J. Liu, Z. Liu, T. Wang, R. Ahmad, A primitive-based 3D reconstruction method for remanufacturing, Int. J. Adv. Manuf. Technol. 103 (2019) 3667–3681. https://doi.org/10.1007/s00170-019-03824-w.

[138] R. Ahmad, S. Tichadou, J.-Y. Hascoet, Generation of safe and intelligent tool-paths for multi-axis machine-tools in a dynamic 2D virtual environment, Int. J. Comput. Integr. Manuf. 29 (2016) 982–995. https://doi.org/10.1080/0951192X.2015.1130258.

[139] N. Malik, R. Ahmad, M. Al-Hussein, Generation of safe tool-paths for automatic manufacturing of light gauge steel panels in residential construction, Autom. Constr. 98 (2019) 46–60. https://doi.org/10.1016/j.autcon.2018.11.023.

[140] R. Ahmad, S. Tichadou, J.-Y. Hascoet, 3D safe and intelligent trajectory generation for multi-axis machine tools using machine vision, Int. J. Comput. Integr. Manuf. 26 (2013) 365–385. https://doi.org/10.1080/0951192X.2012.717720.

[141] R. Ahmad, S. Tichadou, J.-Y. Hascoet, New computer vision based Snakes and Ladders algorithm for the safe trajectory of two axis CNC machines, Comput. Des. 44 (2012) 355–366. https://doi.org/10.1016/j.cad.2011.12.008.

[142] J. Jiang, Y. Ma, Path Planning Strategies to Optimize Accuracy, Quality, Build Time and Material Use in Additive Manufacturing: A Review, Micromachines. 11 (2020)

633. https://doi.org/10.3390/mi11070633.

[143] H.Z. Imam, Y. Zheng, R. Ahmad, An efficient tool-path planning approach for repair of cylindrical components via laser cladding, J. Remanufacturing. (2020) 137–146. https://doi.org/10.1007/s13243-020-00096-6.

[144] C. Zheng, D. Yang, W. Hao, T. Qiao, Z. Tan, J. Jin, Smooth path generation method of laser cladding bit repair robot based on 3D automatic measurement of wear surface point cloud, J. Phys. Conf. Ser. 1939 (2021) 012117. https://doi.org/10.1088/1742-6596/1939/1/012117.

[145] F. Jia, J. Tzintzun, R. Ahmad, An Improved Robot Path Planning Algorithm for a Novel Self-adapting Intelligent Machine Tending Robotic System, Lat. Am. Symp. Ind. Robot. Syst. 86 (2020) 53–64. https://doi.org/10.1007/978-3-030-45402-9_7.

[146] X. Wang, W. Sun, Y. Chen, J. Zhang, Y. Huang, H. Huang, Research on trajectory planning of complex curved surface parts by laser cladding remanufacturing, Int. J. Adv. Manuf. Technol. 96 (2018) 2397–2406. https://doi.org/10.1007/s00170-018-1737-z.

[147] J. Liu, W. Sun, Y. Huang, An algorithm for trajectory planning of complex surface parts for laser cladding remanufacturing, Proc. Inst. Mech. Eng. Part B J. Eng. Manuf. 235 (2021) 2025–2032. https://doi.org/10.1177/0954405420987712.

[148] Y. Li, T. Chen, D. Liu, Path Planning for Laser Cladding Robot on Artificial Joint Surface Based on Topology Reconstruction, Algorithms. 13 (2020) 93. https://doi.org/10.3390/a13040093.

[149] J. Flores, I. Garmendia, J. Pujana, Toolpath generation for the manufacture of metallic components by means of the laser metal deposition technique, Int. J. Adv. Manuf. Technol. 101 (2019) 2111–2120. https://doi.org/10.1007/s00170-018-3124-1.

[150] J. Hao, Q. Meng, C. Li, Z. Li, D. Wu, Effects of tilt angle between laser nozzle and substrate on bead morphology in multi-axis laser cladding, J. Manuf. Process. 43 (2019) 311–322. https://doi.org/10.1016/j.jmapro.2019.04.025.

[151] G. Lian, Y. Zhang, H. Zhang, X. Huang, C. Chen, J. Jiang, Investigation of Geometric Characteristics in Curved Surface Laser Cladding with Curve Path, Metals. 9 (2019)

947. https://doi.org/10.3390/met9090947.

[152] D. Legland, Geom3D Library, MATLAB Cent. File Exch. (2022). https://www.mathworks.com/matlabcentral/fileexchange/24484-geom3d.