2020-12-01

Real-Time Intelligent And Multi-Spectral Inspection Of Structural Components

Mst Mousumi Rizia
University of Texas at El Paso

REAL-TIME INTELLIGENT AND MULTI-SPECTRAL

INSPECTION OF STRUCTURAL COMPONENTS

MST MOUSUMI RIZIA

Doctoral Program in Mechanical Engineering

APPROVED:

Angel Flores-Abad, Ph.D., Chair

Ahsan R. Choudhuri, Ph.D., Co-Chair

Joel Quintana, Ph.D.

Virgilio Gonzalez, Ph.D.

Stephen L. Crites, Jr., Ph.D.
Dean of the Graduate School

Dedication

To Ammu & Abbu

for all their love, endless support, and encouragement.

REAL-TIME INTELLIGENT AND MULTI-SPECTRAL INSPECTION OF STRUCTURAL COMPONENTS

by

MST MOUSUMI RIZIA, M.Sc.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

Department of Aerospace & Mechanical Engineering

THE UNIVERSITY OF TEXAS AT EL PASO

December 2022

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my advisor, Dr. Angel Flores-Abad, for his advice, kind encouragement, enduring patience, and constant support. He never ceased believing in me, even when I doubted my abilities, and always gave me his time throughout my doctoral journey; I am so thankful.

I am forever grateful to my mentor and co-supervisor, Dr. Ahsan Choudhuri, for the opportunity he has given me. I express my deepest gratitude for his continued guidance, support, and inspiration, allowing me to fully recognize my interest and potential as a researcher. I have learned so much from you.

I also thank the other committee members, Dr. Joel Quintana and Dr. Virgilio Gonzalez, for their invaluable suggestions, comments, and guidance in completing this work. As a special note, I thank Dr. Sergio Luna Fong, who graciously shared his expertise and gave valuable guidance I needed to complete the chapter on Inspection Payload Development.

I want to acknowledge the help, support, and cooperation from my teammates, Angel Ortega, Julio Reyes, and Noshin Habib. I recognize and am thankful to the team for useful discussion and collaboration in trajectory optimization, simulation, and flight tests and for sharing synthetic data. I also want to thank Andrea Vargas for her assistance in data annotation.

I thank my lovely sisters, for always being there for me and constantly motivating me. I thank my beloved husband, *Mustafa*, for putting up with me during the development of this work, who showed nothing short of unconditional love and support without complaint. And finally, I must thank my dearest parents, Ammu & Abbu, who instilled in me the importance of education, discipline, and perseverance necessary to achieve any goal. Who I am today has only been possible thanks to your selfless love, contributions, and sacrifices. You all mean the world to me.

Abstract

Conventional, manual inspection methods are the most commonly used inspection approaches to this day; that cause downtime and can be erroneous due to their repetitive nature, heavy workload, and human error. The overarching goal of this work is to advance structural inspection with intelligent and autonomous techniques across infrastructures. In particular, this project will develop path-planning schemes for close navigation around the structures and intelligent algorithms for crack and corrosion detection.

The introduced novel navigation method uses advanced manufacturing techniques to generate aerial inspection trajectories in GPS-denied areas. The proposed method is validated using the 'Gazebo' robotics simulator; the results confirm the usability for close-quarter inspection of any structural components with complex geometry.

The intelligent inspection algorithms are developed by leveraging Artificial intelligence's (AI) Deep Neural Networks (DNNs). Custom data sets are acquired and appropriately prepared for the specific model and anomaly classes, 'crack' and 'corrosion.' The models are further optimized to a lighter, lower latency version for real-time deployment at the edge. Developed custom models are tested for validation in industrial compounds, and they competently identify and localize the defects at the scene.

Lastly, an integrated multi-spectral inspection capability with a user interface (UI) is developed to advance and supplement the inspection method. It generates overlayed multi-spectral scopes fusing color and infrared sensor feeds, read temperatures & displays thermal profiles to the UI. Experimental studies are conducted to demonstrate the usability and advantage of the system in infrastructural defect detection. The proposed approaches are validated in laboratory and industrial setups.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and Problem Statement

Apart from safety-related issues, regular inspection and maintenance of infrastructures are critical for a stable economy in any country. Structural components, from power plants and industries to civil infrastructures such as buildings and bridges, suffer from environmental effects, aging, and usage decay. The power plants (gas & coal) produce some 80% of global electricity as of 2020, as per DNV GL's Energy Transition Outlook [7]. Therefore, these plants' correct health and operational condition are extremely critical to the global economy. Environmental and operational-related defects, e.g., crack and corrosion, are stringent safety and financial concern all across the globe and can be significantly costly through incidents, downtime, or irrevocable damage and loss if not timely detected. The global estimated cost of only corrosion was US$276 billion, equivalent to 3.1% of the US gross domestic product (GDP) in a study done by the US Federal Highway Administration (FHWA) in 2002 alone [8]. Scheduled preventive maintenance is key to timely anomaly detection, monitoring, and taking measures to resolve an issue, if necessary, to maintain structural integrity. However, inspection and maintenance of these massive numbers of ever-expanding and aging infrastructure require enormous national resources from federal and state agencies.

Inspection of the structural component is a frequent and repetitive task. While infrastructure anomalies, e.g., cracks and corrosion, have caused many accidents taking precious lives, and businesses are losing millions in time and money, inspection methods have changed little over time to identify them, relying heavily on human discernment. The con-

ventional inspection method is still in place as human inspectors perform manual inspections, sometimes with handheld equipment. Although the manual inspection is effective, it is limited to human capacity regarding reach and speed; and heavily relies on the experience and expertise of inspectors. Depending on the inspectors' perception, it may also be subjective, inconsistent, and inaccurate due to human error. It is also unrealistic to have the same inspector inspect all plants owned by a single corporation. The inspection could also be dangerous and challenging when performed in hazardous and hard-to-reach areas, imposing heightened risks on personnel. Not to mention manual inspection can be costly due to downtime.

Applying advanced technology to automate the inspection process and to attain consistent results to assist human inspectors is a practical and suitable alternative motivated by the limitations of the traditional approach. Technology adoption transforms the repetitive maintenance task into an easier, more consistent, and significantly inexpensive one. With the preventive maintenance philosophy, there have been major improvements in inspection methods, and ongoing research will speed up the inspection process to reduce downtime even further. Cost savings can be significant through proper maintenance and management over the lifetime of an asset. A rapid rise of robotic technology incorporation for inspection can be observed today.

1.1.1 Structural Defects

The defect of different infrastructures is a diverse topic, and it also depends on not only the type of structure and its use but also the system dynamics and environment. One of the main common faults that maintenance inspectors look for in any infrastructure inspection is cracks. Crack detection is a critical and common structural maintenance check since it directly reflects the structure's safety, durability, and applicability. It significantly influences the bearing capability and integrity of the structural component. There could be multiple reasons for crack formation, starting from something internal, like material shrinkage, to an external influence, such as extreme tensile stress. It could also be a warning

sign, indicating undiscovered problems and potential failure. Environmental conditions such as humidity and chemical interference also cause cracks and sometimes can lead to corrosion of metallic reinforcement underneath, ultimately leading to failure. As such, inspecting cracks is a vital maintenance task to ensure serviceability and prevention of bigger problems.

Another common problem, especially in industries and civil infrastructures, is corrosion. Like crack propagation, corrosion is a product of the environment. It is caused by the chemical reaction of metal while interacting with its surrounding environment. Not always easy to detect or visible to the naked eye, it has rapid propagation and causes accelerated decay. It is a severe safety and financial concern. If left unchecked can cause catastrophic event endangerment to precious lives. On the other hand, it causes major financial setbacks if not addressed in time. In 2013, the estimated corrosion cost was US$2.5 trillion, equivalent to 3.4% of the global GDP. Worldwide estimated savings could be between US$375 and $875 billion annually using available corrosion control practices [9]. Accurate and timely detection of cracks and corrosion in infrastructures is crucial to economic efficiency by initiating appropriately managed maintenance processes that will save lives.

1.1.2 Challenges

Preventive maintenance is pivotal for the safe operation of vital infrastructures. As discussed above, the major challenges considered are- time constraints, slower speed of manual inspection, and issues related to difficult-to-reach structural components due to height, accessibility, or hazardous condition. Human inspection is also limited by human capacity. There is only so much workload humans can handle. Using different inspectors results in inconsistency in analysis, and sometimes, cracks and corrosion are difficult to detect with human vision. The fundamental human difference makes it difficult to standardize the results and decide on corrective or preventive measures.

1.1.3 Available Technologies for Visual Inspection

Digital technologies have been assisting in maintenance planning, monitoring, and inspection for decades, including but not limited to the use of image processing techniques [10], robotics [11], and computer vision [12]. Using robotics is one of the most popular visual inspection capabilities for collecting data and remote monitoring. It addresses the safety concern of human inspectors, particularly in hazardous conditions. The use of aerial vehicles reaps additional benefits, as it gives access to hard-to-access areas [13]. With a well-planned and trained team, it can significantly reduce inspection-related downtime. Available aerial inspection systems usually are component-specific and heavily depend on the operator for remote-controlled navigation or GPS. The navigation methods may cause complications in areas with limited or no GPS signal or direct line of sight.

Vision-based methods are recently being extensively researched for surface defect detection, including machine learning and convolutional neural networks (CNN) [14]. Deep learning (DL), a popular branch of machine learning (ML) where deep neural networks (DNN) with deep layers is used for relevant feature extractions by itself [15]. Numerous DL-based studies have been proposed, which are categorized as (i) image classification (IC), (ii) object detection (OD), and (iii) semantic segmentation (SS). Recently, DNNs have proven to surpass human-level accuracy on the ImageNet classification [16]. The groundbreaking success of these methods has inspired several recent studies to propose AI-based algorithms for automated crack and corrosion detection.

At times, however, the defects can be hard to notice with the naked eye, depending on the defect location, dimension, surface noise, e.g., color and texture, lighting condition, and even the time of the day. In such conditions, using a thermal sensor gives an edge over visual inspection using cameras. The radiometric information can be useful for locating components with abnormal thermal profiles, and undetected anomalies can be analyzed and resolved with early detection.

1.2 Research Objectives

This research presents an improved and streamlined inspection method applicable to diverse infrastructures. The goal is to provide an intelligent autonomous inspection system with a real-time application by developing an inspection payload (IP). The research objectives are as follows:

1. Provide a close-quarter trajectory generation method for aerial inspection in GPS-denied areas.
2. Automatically detect & localize 'crack' and 'corrosion' using a custom DNN model.
3. Develop intelligent algorithms to deploy at the edge (embedded platform) for real-time inspection.
4. Develop a multi-spectral capability and interface for thermal analysis of structural components.

The research should include building structures such as vertical walls and industrial structural elements, such as boilers and chimney stacks, to achieve the said objectives. Different structures, materials, and surface conditions should also be considered, e.g., textures and background noise. The inspection environments, anomaly types, and locations of defects vary from industry to industry. Therefore, data from different locations should be collected considering these different conditions. The trajectory generation method should be able to customize and modified according to the location. The developed crack and corrosion detection solutions should be able to identify and localize defects at different surface images and structures. The multi-spectral capability should provide a user-friendly way to inspect industrial components' temperature and heat profile at the required operational condition of power plants. Through application studies and experimental work in these three areas, crack, and corrosion detection methods for different infrastructure components, practical suggestions, and recommendations using DNN-applied industrial infrastructure inspection should be provided.

1.3 Methodology

Automatic real-time detection of structural defects is a much-required technology to enable rapid, accurate, and on-site inspection. This study has developed an inspection payload (IP) capable of deploying across diverse infrastructures for inspection. The IP is a complete setup that can be mounted on platforms like aerial vehicles or used as a handheld setup to inspect cracks and corrosion. It is equipped with necessary sensors for autonomous visual and multispectral inspection.

Task 1. Current aerial robots, i.e., unmanned aerial vehicles (UAVs) (commonly termed drones), mostly rely on human pilots to operate within a limited line of sight, and a reduced perception, preventing close-quarter inspection in intricate, structurally complex and GPS-denied environments. This work introduces a novel offline inspection trajectory generation method based on advanced manufacturing techniques for close-quarter inspection in known, static but GPS and/or communication-denied areas. The two approaches are based on computer-aided manufacturing (CAM) and additive manufacturing (AM) techniques. The drone would fly along the path generated using a toolpath of a CAM machining tool or a 3D printer's extruder, enabling it to fly close to the structure, especially in complex geometry using the custom trajectory. These developed trajectories are validated in the Gazebo robotics simulator and tested to fly in the complex geometry of an industrial environment. The results demonstrate the proper performance of the method and confirm that this approach can be used for close inspection of structural components.

Task 2 & 3. For automating defect detection, this work uses advanced Computer Vision (CV) techniques of Artificial Intelligence (AI), leveraging Deep Neural Networks (DNN). In this study, custom DNN algorithms were developed to obtain an intelligent inspection system capable of detecting structural problems in power plant components, such as 'cracks' and 'corrosion' in offline and online modes, i.e., at the edge. The methodology consists of creating a custom data set, training using transfer learning of the state-of-the-art model

'You Only Look Once' (YOLOv4), testing and comparing models' results, and using the best model in a handheld device for detection at the edge.

Data sets were created by combining existing and newly collected own images for both anomaly types since no suitable data set was available for this specific case. For automating corrosion inspection, one major challenge is that it does not have consistent, salient features needed by a DNN model. The diversity of corrosion and scarcity of target-specific data makes things difficult to develop a well-performing model. Considering that, synthetically generated data is added for corrosion data set development. Next, customized DNN models for offline inspection modes are developed by conducting several trials and experimentation with the data set, hyper-parameters, and different models. The custom offline model is further optimized using quantization techniques to attain a lighter, lower latency model. The lighter custom model is appropriately adapted to the embedded platform for edge deployment. Doing so makes the models capable of deployment on board a single-board computer (SBC) (i.e., NVIDIA's Jetson Nano), all the while maintaining the model's mean average precision (mAP) during real-time inference.

Both offline & online (real-time) models' performance is tested on the test set industrial images never seen by the model taken by aerial and handheld setups. Cracks and corrosion inference results show that the system can identify and localize it within the camera's field of view (FOV). The best model's evaluation metric for crack detection, mAP, is 98.44% for the crack detection algorithm. For corrosion detection, several studies show improvement in model performance with improvement in quality and an increasing number of specific corrosion images. The best mAP achieved is 44.54% on real data, while the model yielded 72% when trained on the increased number of images combining real and synthetic ones. Model performance improves by 28% in mAP with increased data, specific data augmentation, and hyper-parameter adjustments. Real-time inference latency test on images of an old power plant stack shows 2.5 FPS (frames per second) inference speed for crack and corrosion detection.

Task 4. This study adds a multispectral capability for the visual inspection of power plants by combining RGB (color) and infrared images. A system is developed to capture infrared images, display temperatures in the display scope, and generate overlayed multi-spectral display of the powerplant using Python code and a custom user interface (UI). This system helps inspectors identify or observe areas of concern, develop a structural health monitoring (SHM) system, and collect multispectral images as operation and maintenance records that can later be useful for detecting historical thermal-related anomaly detection.

1.4 Research Contributions

The research contributions are:

1. A novel trajectory generation method using advanced manufacturing techniques for close-quarter aerial inspection of any infrastructure with complex geometry and in GPS & communication-denied areas.
2. A 'crack' and 'corrosion' detection database is developed
3. Custom DL-based 'crack' and 'corrosion' detection algorithm is developed and tested with adequate performance.
4. Custom models are optimized & adapted to embedded SBC (Single Board Computer) platform for real-time detection at the edge.
5. Multispectral analysis UI (user interface) is developed for thermal and radiometric inspection.

1.5 Dissertation Outline

The dissertation is organized as follows. In Chapter 2, the IP is explained along with the hardware specifications and system interfaces. Chapter 3 describes advanced manufacturing-based techniques to generate UAV inspection trajectories for aerial inspection. Chapters

4 and 5 present database development, custom DL-based anomaly detection algorithm development, model optimization, evaluation, and online-offline deployment for inference experimentation using a different platform for crack and corrosion, respectively. Chapter 6 describes the development of multispectral inspection and analysis capability and introduces the UI for inspection data post-processing and other details. Each chapter describes its literature review, method, experimental results, and discussions. Finally, the conclusions and concluding remarks are presented in Chapter 7.

Chapter 2

Inspection Payload Development

2.1 System Requirements

The system's main objective is to advance structural inspection with autonomous techniques to overcome the associated limitations of available manual inspection methods. The system should be able to identify common structural defects with adequate precision and accuracy.

An inspection payload (IP) needs to be developed with the necessary computational power, hardware, software, and developing custom intelligent algorithms to detect and localize 'crack' and 'corrosion' autonomously in structural components for the advanced inspection system.

General Requirements

The requirements of the inspection system and IP were reviewed using the Requirements Verification Matrix (RVM) and determined as follows.

SR-1. IP shall perform autonomous visual inspection of structural components.

SR-2. IP shall successfully identify defects common to infrastructures.

SR-3. IP shall aid inspectors by locating areas of interest.

SR-4. IP shall perform inspection faster to lower system downtime.

Table 2.1: IP Functional Requirements RVM

No	**Requirements**	**Source**	**Verification Method**	**Status**	**Verification Result**
FR-1	The IP shall be able to detect industrial defects autonomously.	SR-1	Demonstrating	Complete	Passed
FR-2	The IP shall be able to detect cracks from 3 ft away from the surface.	SR-2	Testing	Complete	Passed
FR-3	The IP shall be able to detect corrosion from 3 ft away from the surface.	SR-2	Testing	Complete	Passed
FR-4	The IP shall be able to detect and localize corrosion and cracks in an illuminated or daylight environment.	SR-3	Demonstrating	Complete	Passed
FR-5	The IP shall be able to detect corrosion and cracks when moving at 1 m/s or less.	SR-4	Testing	Complete	Passed

Table 2.2: Mechanical Design RVM

No	**Requirements**	**Source**	**Verification Method**	**Status**	**Verification Result**
MD-1	The IP shall not occupy more than 110 x85 x45(75 with sensor mount) mm^3 in volume.	Internal	Inspect	Complete	Passed
MD-2	The IP shall not exceed 150g of mass budget.	Internal	Inspect	Complete	Passed

Table 2.2: Mechanical Design RVM

No	Requirements	Source	Verification Method	Status	Verification Result
MD-3	IP sensors shall be able to set up facing the inspection surface(s) on any of the four/six faces of the drones/inspection platform.	Internal	Inspect	Complete	Passed
MD-4	IP shall be able to operate within –5-45°C.	Internal	Inspect	Complete	Passed

Table 2.3: Electrical RVM

No	Requirements	Source	Verification Method	Status	Verification Result
ER-1	RGB sensor/camera shall be connected to IP board with Mobile Industry Processor Interface (MIPI) Camera Serial Interface Type 2 (CSI-2) connector for power & data transfer.	Internal	Inspect	Complete	Passed
ER-2	IR sensor shall be connected to SBC (Single Board Computer) with USB 3.0 Type A connector for power and data transfer.	Internal	Inspect	Complete	Passed
ER-3	IP shall use a switching regulator to regulate required input voltage.	Internal	Inspect	Complete	Passed
ER-4	IP shall control cooling fan with 4-pin fan control header.	Internal	Inspect	Complete	Passed

Table 2.4: Power RVM

No	Requirements	Source	Verification Method	Status	Verification Result
PR-1	IP shall have the option to use the power source of the inspection platform or external source.	Internal	Inspect	Complete	Passed
PR-2	IP total power shall not be more than 20W at peak usage.	Internal	Inspect	Complete	Passed
PR-3	IP shall have additional cooling option with a cooling fan powered from the common power source with 4-pin fan control header.	Internal	Inspect	Complete	Passed
PR-4	RGB camera shall not demand more than 200-250mA.	Internal	Inspect	Complete	Passed
PR-5	IR sensor shall have between 5mW-650mW power consumption.	Internal	Inspect	Complete	Passed

Table 2.5: Data RVM

No	Requirements	Source	Verification Method	Status	Verification Result
DR-1	IP shall be able to store data locally on microSD card (minimum of 64 GB physical storage on board).	Internal	Testing	Complete	Passed
DR-2	IP shall use local Wi-Fi connection to communicate with ground station.	Internal	Testing	Complete	Passed
DR-3	Data collection & inspection of detection shall be analyzed remotely for usage at remote computer.	Internal	Testing	Complete	Passed

Table 2.5: Data RVM

No	Requirements	Source	Verification Method	Status	Verification Result
DR-4	RGB sensor should have at least 4 Gbps bandwidth data lane.	Internal	Testing	Complete	Passed
DR-5	RGB sensor should support RGB & user defined input data formats.	Internal	Testing	Complete	Passed
DR-5	RGB sensor should have at least 8 Megapixels resolution and 3 mm focal length.	Internal	Testing	Complete	Passed
DR-6	IR sensor shall have between 5mW-650mW power consumption.	Internal	Testing	Complete	Passed
DR-7	RGB sensor should have 60°C × 40°C FOV.	Internal	Testing	Complete	Passed
DR-8	RGB sensor should support 1080p, 720p and 640 × 480p video modes.	Internal	Testing	Complete	Passed
DR-9	IR sensor should have at least -10° to +400°C temperature reading range (at room temperature).	Internal	Testing	Complete	Passed
DR-10	IR sensor shall support YUV & USB UVC data formats.	Internal	Testing	Complete	Passed
DR-11	IR sensor shall have at least 160h x 120v pixels resolution.	Internal	Testing	Complete	Passed
DR-12	IR sensor shall have at least 4 Hz frame rate.	Internal	Testing	Complete	Passed

Table 2.6: Algorithm & Software RVM

No	Requirements	Source	Verification Method	Status	Verification Result
ASR-1	The IP shall run on Linux-based OS (Operating System).	Internal	Testing	Complete	Passed
ASR-2	The IP shall include pre-loaded firmware and software compatible with the sensors.	Internal	Demonstrating	Complete	Passed
ASR-3	IP shall use the pre-trained & optimized, custom deep learning (DL) model for damage detection.	Internal	Demonstrating	Complete	Passed
ASR-4	Pre-loaded algorithm/model shall leverage IP's GPU to reduce inference latency.	Internal	Testing	Complete	Passed
ASR-5	The pre-trained DL model shall be optimized on each IP (quantization is platform specific).	Internal	Testing	Complete	Passed
ASR-6	The IP shall be compatible with ROS (Robotic Operating System).	Internal	Testing	Complete	Passed

2.2 Architecture

As part of the inspection system, the IP is developed, comprising a Single Board Computer (SBC) as the computational tool, sensors, required software, firmware & the custom intelligent inspection algorithms developed for autonomous intelligent inspection installed in the SBC platform. The IP product Breakdown Structure (PBS) is shown in Figure 2.1); it includes both physical (SBC & hardware) and logical parts. The logical elements include but are not limited to required software, firmware, libraries, custom codes & models developed for defect detection (seen in the PBS tree in green fonts). The interface and architectural components are described in this section.

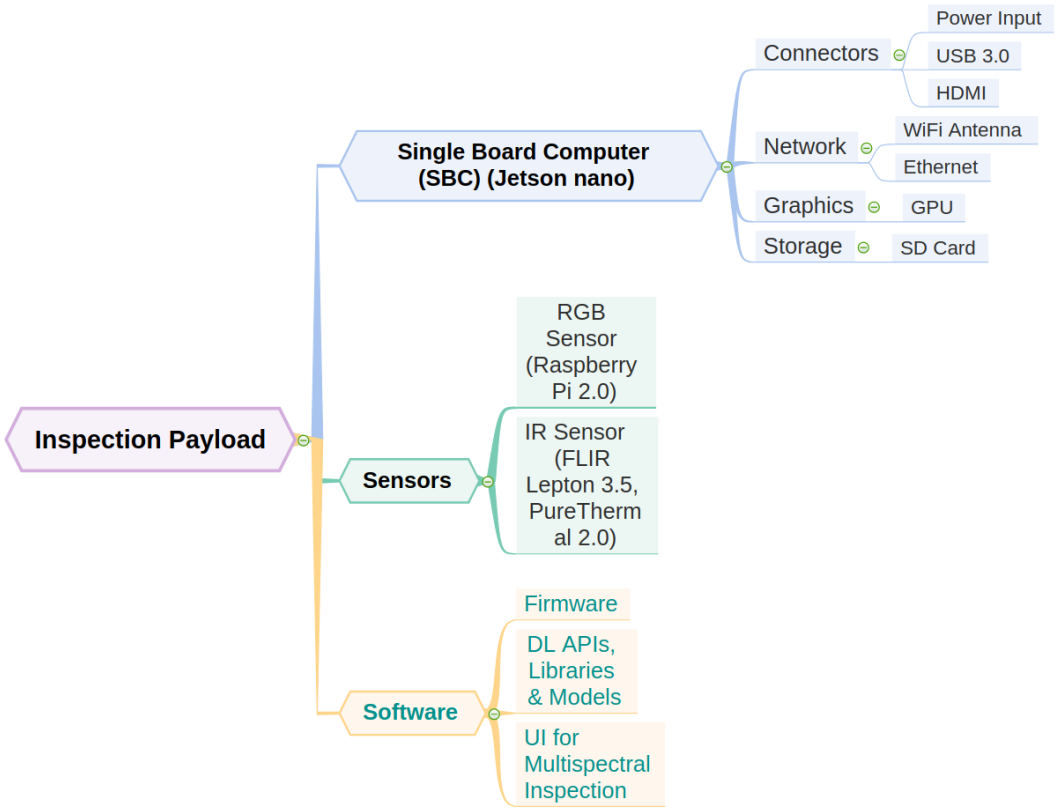


Figure 2.1: Product Breakdown Structure of the Inspection Payload

2.2.1 Interface Requirements Document

The following sections outline the interfaces available on the IP and detail how the user interacts with and controls the IP interface. Figure 2.2 demonstrates the IP's interfaces and high-level architecture.

The IP (in purple) includes Jetson Nano, the Single Board Computer (SBC), software, and sensors. The SBC is pre-installed with the required firmware, custom models, and codes. The two sensors are the RGB camera- Raspberry Pi 2.0, and the infrared (IR) sensor, Lepton 3.5, with Purethermal Board 2.0. Together, the IP can be accessed via different interface combinations by the user at the remote workstation (in gray).

Electrical Interface

1. IP shall be powered by 5V DC at 4A (or 20W max) using a 5.5mm x 2.5mm barrel connector.

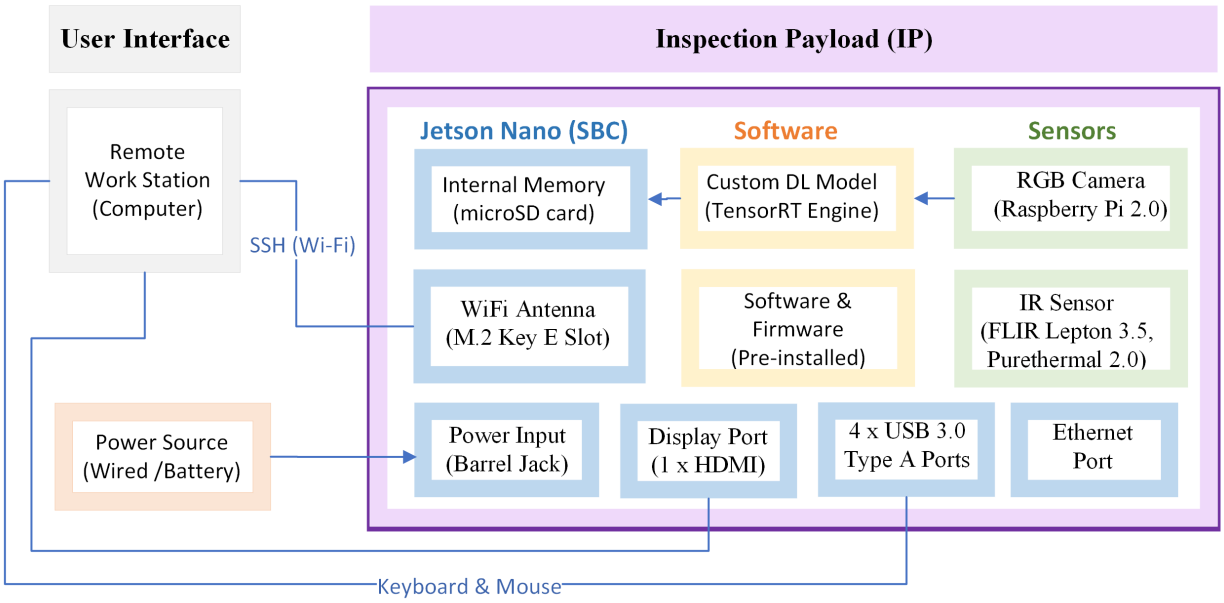


Figure 2.2: High-level Inspection Payload Architecture and Interfaces

2. IP shall be powered with the common inspection platform power source.
3. IP shall have the option to have a separate power source using a 5.5mm x 2.5mm barrel connector depending on requirements.
4. IP shall have common ground with platform electronics.
5. The IP shall be powered with 14.8V Li-Poly battery, through a UBEC (10A peak) with a XT60 to barrel connector for edge deployment.

Mechanical Interface

1. IP shall be bolted on inspection platform with 4×M3 screws.
2. IP shall include dampers to mitigate vibration.
3. Sensors of IP shall be mounted to custom 3D printed mounts specifically designed to be placed closest to each other to ensure maximum field of view overlap using eight M2×8 screws & bolts.

Data Transfer

1. IP shall have M.2 Key E connector for wireless networking interface.

2. IP shall use wireless Secure Shell Protocol (SSH) to communicate with & send data to ground station computer over Wi-Fi.
3. IP shall have wired internet connection option using ethernet jack per IEEE 802.3af.
4. IP shall have data display option connected with HDMI port or USB 3.0 Type A.
5. IP shall use USB Bluetooth keyboard & mouse direct computer access.

Documents

This is the latest revision or most recent version of interface document. Documents referenced are [17, 18, 19, 20, 21]

2.2.2 Hardware

2.2.2.1. NVIDIA Jetson Nano

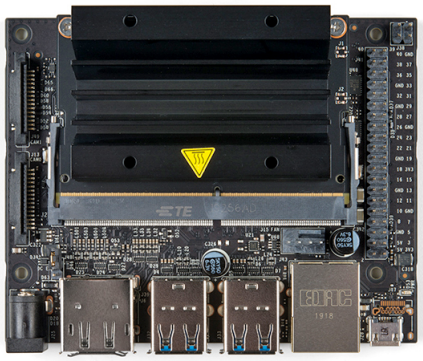


Figure 2.3: SBC Jetson Nano (Top View)

A single board computers (SBC), Nvidia's Jetson Nano is selected to use as the embedded platform of the IP. It is a compact, low voltage System on Chip (SoC) designed to carry out programmed instructions. It has 4GB of RAM, a quad-core ARM A57 processor with a 128-core NVIDIA Maxwell™ architecture-based GPU 2.2.2. Some of the specifications are as follows:

Table 2.7: Mechanical Design RVM

GPU	128 Core Maxwell
CPU	Core ARM A57 @ 1.43 GHz
Memory	4GB 64 bit
Storage	16 GB, Expanded to 128 GB
Camera	12 (3x4 or 4x2) MIPI CSI-2
Display	HDMI 2.0 or DP1.2

2.2.2.2. RGB Sensor

For Color images, RGB sensor Raspberry Pi Camera Module 2.0 (Sony IMX219 Sensor) is used [19].

Table 2.8: RGB Camera Raspberry Pi 2.0 Specifications

Sensor Name	Sony IMX219 Sensor
Sensor Resolution	3280 × 2464 pixels
Focal Length	3.04 mm
Field of View (FOV)	Horizontal: 62.2\textdegree Vertical: 48.8\textdegreeDiagonal: 74.2\textdegree
Weight	3 g
Still resolution	8 Megapixels
Video Modes	1080p30, 720p60 and 640 × 480p60/90
Linux integration	V4L2 Driver

2.2.2.3. Infrared (IR) Sensor

For Infrared images, GroupGets LLC's IR sensor FLIR Lepton 3.5 with Purethermal 2.0 Board is used for thermal & radiometric analysis [20, 22].

Table 2.9: IR Hardware Specification

Sensor Name	FLIR Lepton 3.5
Sensor Range	8 - 14 microns (nominal) Long Wave Infrared (LWIR)
Rediometric Dynamic Range	-10° to +450°C (typical)
Thermal sensitivity	< 50 mK (0.050°C)
Resolution	160h x 120v pixels
Radiometric Accuracy	High Gain Mode: Greater of +/- 5°C or 5% (typical) Low Gain Mode: Greater of +/- 10°C or 10% (typical)
Frame Rate	8.7 Hz (effective)
Lens Type	f/1.1
Field of View (FOV)	Horizontal: 57°
Sensor Size (w x l x h)	10.50 x 12.70 x 7.14 mm
Sensor Weight	0.9 grams
Power Consumption	150 mW typical, 650 mW during shutter event & 5mW standby
Optimum Operating Temperature Range	-10°to + 80°C
Sensor Board Name	Purethermal 2.0 (Embedded)
Sensor Board Interface	I2C, USB
Voltage Supply	5V USB
RoHS Status	RoHS Compliant

2.2.3 Software, Firmware & DL APIs

For 'Trajectory Generation,' CAD software 'Fusion 360', Additive manufacturing slicer software 'Ultimaker Cura 3.6.0', and for trajectory optimization & Gcode extraction algorithm, 'MATLAB' is used.

For 'Autonomous Intelligent Inspection,' all the Deep Learning (DL) models were implemented using Python 3.6 version and 'TensorFlow 2.0' DL API. The Darknet framework is selected as the backbone for the base model 'YOLOv4' (You Only Look Once version 4) 'Object Detection'

model.

For 'Multi-Spectral Visual Inspection,' IR sensor board Purethermal 2.0 needs the UVC Purethermal firmware version 1.2.2 or greater. It is needed to communicate with the IR sensor Lepton 3.5. It also needs Linux Kernel versions 4.0 or greater. For UI development, Python 3.6 is used.

A summary of some of the specifications of the software, libraries, and other tools used for the development of autonomous DL-based & IR-based inspection & analysis methods are given in Table 2.10.

Table 2.10: Software, DL APIs, Libraries & Other Tools

Operating System (OS)	Linux Ubuntu 18.04 LTS
CUDA	10.0.326
CuDNN	7.6.3
Pycuda	2019.1.2
Protobuf	3.8.0+
GNU	7.5.0
Cmake	3.10
Python	3.6.9
OpenCV	4.5.1
TensorFlow	2.0
TensorRT	6.0.1
ONNX	1.4.1
NumPy	1.19.5
Matplotlib	3.3.3
ROS	Melodic
Jetpack (Jetson Nano)	4.3

2.2.4 Computational Tools

Training and Offline Testing. The DL-model training needs higher computational power to train the model faster. A paid cloud-based service, Google Colaboratory Pro+ consisting of Tesla K80 graphics processing unit (GPU) card with Intel Xeon® (with two cores) central processing unit (CPU) and 11 GB of graphics memory is used to train the custom model.
To test the performance of the trained models, a laptop with a 64-bit Ubuntu operating system (OS), 32 GB RAM, Intel® Core™ i7-9750 CPU with a clock speed of 2.60 GHz and NVIDIA GeForce GTX 1650 is used.

Model Optimization & Online Testing. Model optimization, and online testing using live camera feed from the onboard camera (Raspberry Pi 2.0) is done using the SBC, Jetson Nano. The model optimization using TensorRT quantization technique is hardware specific; Therefore, it must be done in the platform itself, which will deploy the model at the edge. The major specifications of Jetson Nano are mentioned in Table 2.7.

2.3 IP Platforms

2.3.1 Handheld Setup

Handheld Setup 1. The IP can be used in different ways, leveraging the interfaces needed by the user. Figure 2.4 shows the basic 'handheld setup 1' of IP with the necessary sensors attached. The RGB & IR sensors can be placed in different ways. A custom 3D-printed mount is designed and fabricated for this study. This mount places both sensors as close as possible, giving the highest overlap of the RGB and IR frames for multispectral display.

Handheld Setup 2. This setup is another way if using the IP. A 3D printed hand-held setup with SBC and Raspberry Pi camera like shown in Figure 2.5) can also be used for real-time inference experimentation at the edge.

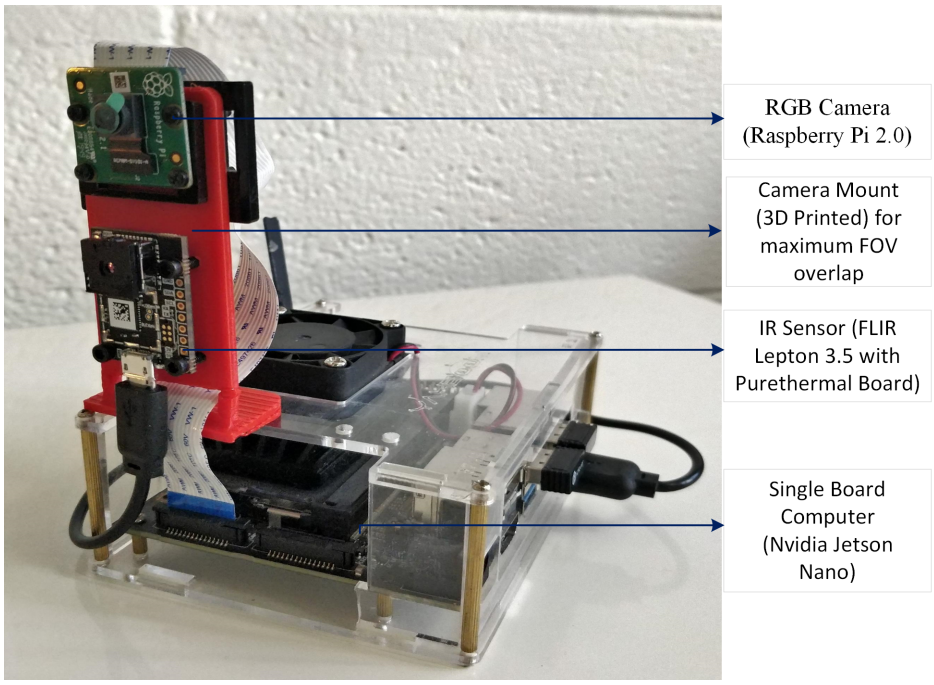


Figure 2.4: Handheld Setup 1 for Visual Inspection, Analysis & Online Inference Experimentation at the Edge

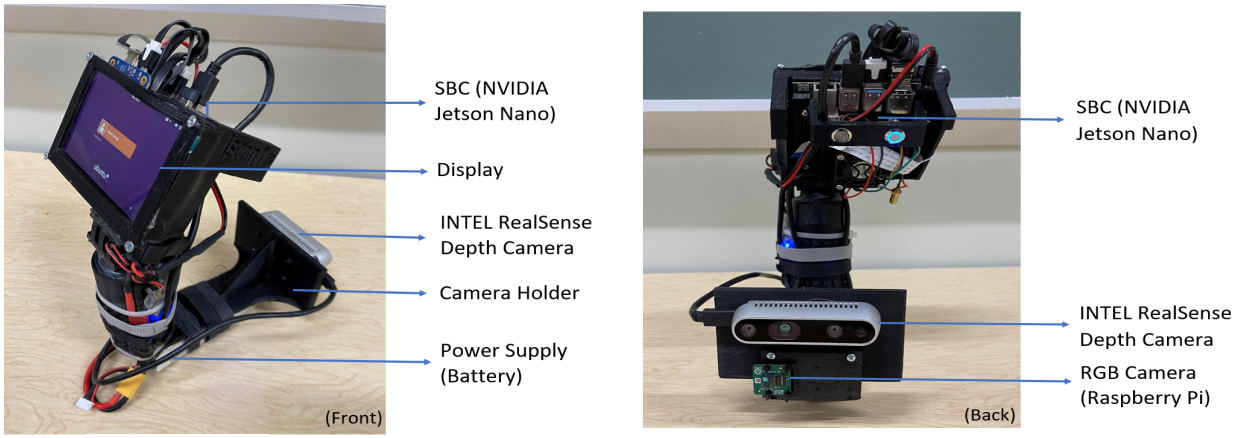


Figure 2.5: Handheld Setup 2 Used for Visual Inspection and Experimentation

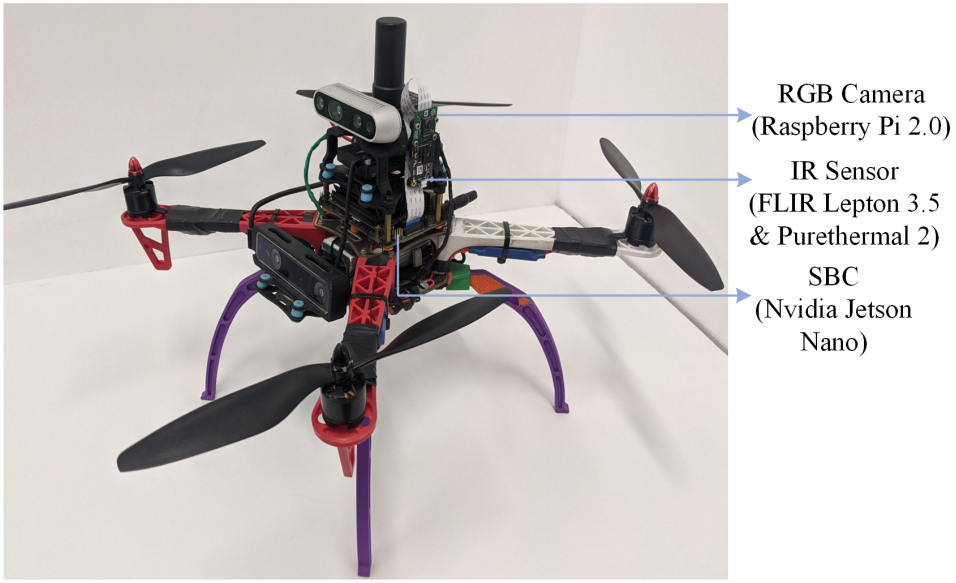


Figure 2.6: UAV Equipped with Inspection Payload

2.3.2 UAV Setup

The IP can also be mounted on a UAV for aerial inspection. Figure 2.6 shows a quadcopter UAV built with a DJI frame and Jetson Nano as its companion computer. Along with the other flight-related components, the three major components of the IP, the SBC (Jetson Nano), Raspberry Pi, and FLIR Lepton IR sensor, can be installed in appropriate locations. These two sensors are vertically placed to the custom camera mount, similar to the Handheld setup 1, to get the maximum frame overlap.

2.4 Test Facilities

1. Initial test runs shall be conducted at the remote testing site of the Aerospace Center's Fabens Airport.
2. Data collection shall be done at industrial compounds for relevant data sourcing in collaboration with local industries, e.g., Vinton Steel LLC & El Paso Electric).
3. IP system testing shall be done at industrial compounds for verification of method in collaboration with local industries (Vinton Steel LLC & El Paso Electric).

Chapter 3

Advanced Manufacturing Technology-based Trajectory Generation

3.1 Related Work

Robotic Inspection Industrial inspection systems have been evolving per the development of technologies that enable efficient, safe, and economical inspection. Robotic technologies have been proven useful, especially as it addresses the safety and accessibility concerns of manual industrial inspection by human inspectors. For example, robots have been used to remotely surveil and inspect hazardous areas in different types of power plants [23, 24] and nuclear power plant facilities [25]. However, these inspection systems can only be applied to a specific module or component of the power plant.

Aerial Robotic Inspection With advanced research and the development of hardware systems, aerial robotics is more frequently used to inspect larger areas. However, most of these UAVs (Unmanned Aerial Vehicles) are manually controlled by human pilots through remote control and operated within their FOV (Field of View). Effective aerial inspection trajectory generation, several works are being done to generate flight trajectories. In an experimental research work by Burri et al., an aerial visual inspection system is developed using a MAV (micro aerial vehicle) for thermal power plant boiler inspection. Their system uses a predictive model controller (MPC) based efficient flight controller targeting confined spaces [26]. The proposed control strategy is less susceptible to external disturbances or delays, but the MPC bias estimation is sometimes off,

which results in position offset. In another approach by Shan et al., the UAV compares imagery data to the Google Maps image database for navigation. However, possible inconsistencies in the map database and a lack of information in the third dimension cause concerning limitation [27].

Trajectory Generation Otherwise, the UAV trajectory waypoints are either using GPS (global location services) [28, 29] or using SLAM (Simultaneous Localization and Mapping) generated maps [30, 31]. GPS-generated flight paths need to maintain a safe distance from the inspection surface for safety, due to their reduced accuracy, especially in industrial compounds with metallic interference that hinder robust signal connectivity. Also, available GPS resolution is low and unreliable, and spotty signals influence and affect finer flight trajectory controls. The maps generated from SLAM, used by local planners, are also not as accurate, especially for inspecting components or intricate structures with tiny details, surfaces without enough features, and different lighting conditions. All these lead to poor mapping and prove difficult to conduct a close-quarter aerial inspection in complex and reduced structural environments; In these conditions, the proposed method rises as a suitable option to be applied based on advanced manufacturing techniques. This work proposes a method to closely inspect components of a structure in a static and known environment, two common features in actual inspection campaigns. The main contribution of this work lies in the unique waypoint generation method for path planning in aerial inspection tasks. The introduced approach generates flight paths at a short separation distance from the inspection surface, enabling close-range image acquisition and capturing more details of possible defects. Moreover, the approach allows the generation of flight paths and navigation in complex structural geometries and reduced spaces.

Generic Trajectory Generation As for the GPS and communication-denied environments, the works mostly use generic flight paths, i.e., a linear or a spiral flight path. Zheng et al. describes an image acquisition system combining two quadcopters [32]. One uses a generic spiral flight path to inspect inside a chimney without GPS, and the other hovers on the chimney's top center for data reception and forwarding to the ground station (using GPS). This method works for a uniform surface inspection, e.g., a circular area of a chimney. In this work, the trajectory enables the UAV to go close to the surface of any shape to carry out inspection tasks, such as image

acquisition. Bolourian et al., in another aerial inspection work, used a LiDAR-equipped UAV for scanning cracks in bridge inspection [33]. They employ a Genetic Algorithm (GA) to generate an obstacle-free 3D point cloud-based flight path to scan the targeted critical areas, minimizing flight time and maximizing inspection visibility. A bridge reference model needs to be given to BrIM/BIM (Bridge Information Modeling) as a reference. They vary the inspection overlap based on VPI (viewpoint of interest) calculation, depending on the critical area, whereas in this work, a uniform grid method with 30% overlap is used throughout. Several flight missions are considered to divide the complete infrastructure into several areas for full coverage.

Manufacturing Technology-based Approaches Manufacturing technique-based work has been proposed for path planning and other tasks like robots in the manufacturing industries using CAD (Computer Aided Design) and CAM(Computer Aided Manufacturing). In earlier work, an algorithm was developed using CMM (coordinate measuring machine) to automate a collision-free probe path planning for CAD-based dimensional inspection [34]. CAM-based robot trajectory planning is seeing popularity for different autonomous machining operations. For example, Cerit et al. used a CAM-based approach for trajectory generation to create a rapid prototyping tool path [35]. In another work, Zeng et al. used the coordinates information extracted from the G-code of a grinding operation using a path generator of the KUKA robot [36]. Using a similar approach, Chen et al. demonstrated an offline path generation technique for an industrial robot based on the CL (cutter location) data for an automatic finishing operation [37].

3.2 Proposed Method

This chapter introduces a novel CAM and AM-based methodology of trajectory generation for UAVs to allow close-quarter aerial inspection in structurally complex and intricate environments. At first, the model of the complex structure to be inspected is obtained/designed using Computer-Aided Design (CAD) software. This 3D CAD model is then uploaded to CAM software to generate a 3D printing or milling operations tool path. The G-code (numerical control programming language) files consist of the coordinates/location. The machining operations and the necessary

parameters are appropriately set in the CAM software environment while considering the UAV and inspection payload constraints, so a uniform toolpath is generated. The XYZ Cartesian coordinates of the toolpath are exported into excel CSV (Comma-separated values) format, representing the waypoints the UAV must follow. For trajectory generation, time information is needed that is later introduced in the MATLAB software via the time step. A simulation of the UAV inspection trajectory around a CAD model is obtained in Gazebo open-source robotics simulator software for verification. Later, the trajectory generation method is used to fly in industrial space as proof of concept.

Contributions of this work are as follows:

1. Proposal and subsequent verification of an aerial flight path generation method from a CAD model of the structure to inspect. Both internal and external inspection trajectories can be generated.

2. The approach uses the tool path generation technique from advanced manufacturing (CAM and AM) to generate the UAV inspection trajectories.

3. This method can operate autonomously in reduced-space environments for close surface visual inspection and does not rely on a GPS signal.

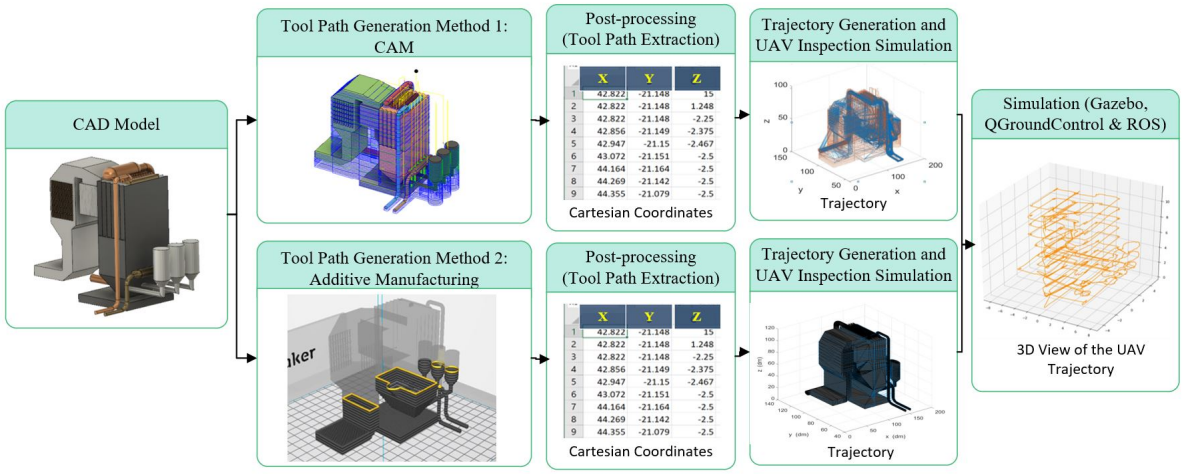


Figure 3.1: Methodology of the CAM/AM-based UAV Inspection in a GPS-denied Environment

3.3 Technical Approach

Autonomous UAV systems can potentially reduce the risk associated with human intervention in inspection tasks and provide means to automate the system and enable the collection of precise inspection data. However, there are several challenges, such as payload limitation, maneuvering capabilities, and reduced flight time. One of the main difficulties is ensuring collision avoidance while flying close to the structure. In this part of the study, offline trajectories are generated for the UAV to follow during the inspection, which allows flying close to the surface in complex environments for internal and external inspection of structural assets. The flight path is generated from the tool path coordinates of two advanced manufacturing techniques, CAM & AM. Using the machine toolpath coordinates from G-code to create a flying trajectory for UAV navigation is a novel idea. The four-step methodology is as follows (Fig. 3.1):

1. System setup and modeling
2. Tool path generation methods
    (a) Method 1: CAM
    (b) Method 2: AM/3D printing
3. Post-processing/Tool path extraction
4. Trajectory generation

3.3.1 CAD Model

The first step is obtaining a CAD model of the component to inspect. An updated CAD model is a crucial part of this research, as based on it, the UAV trajectory will be generated to inspect the concerned areas effectively. If the CAD model of the plant is not available, one needs to be generated using any available CAD software. Some of the latest CAD software provides additional features like highly accurate dimensional analysis, design optimization, assembly of parts, simulation, mechanical design automation, detailed documentation, project management options, and integrated manufacturing environment, i.e., CAM and Additive Manufacturing (AM)

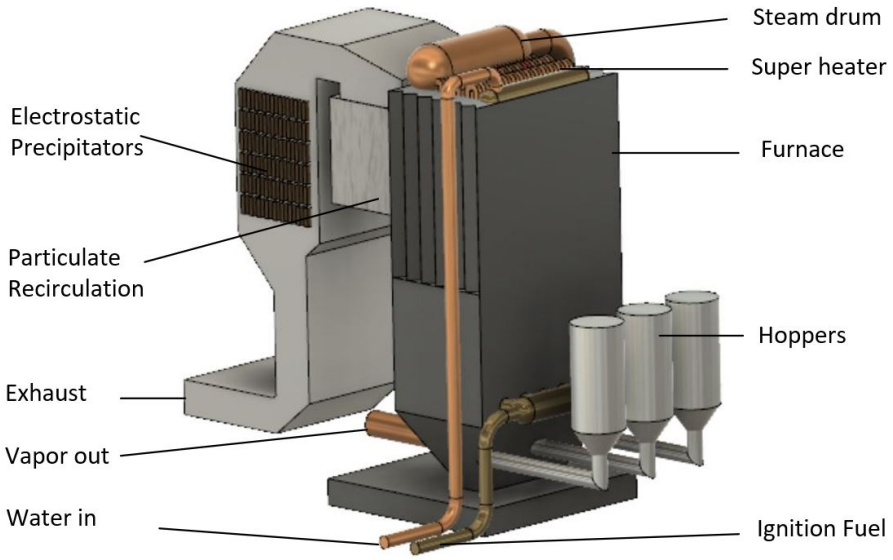


Figure 3.2: 3D CAD Model of a Boiler and its Components

features. These directly connect the design environment to the manufacturing one, and from there, manufacturing instructions can be sent to hardware such as 3D printers and CNC machines.

The fully integrated system helps to integrate and apply rapid design iterations without needing manual modification each time a design gets updated. Fusion 360, open-source CAD software with the incorporated manufacturing features stated above, has been used in this project. Figure 3.2 shows a Fusion 360 generated 3D CAD model used for this study. However, any CAD software with needed features and file formats, e.g., STL and G-code, can also be used.

Model Scale. The CAD model to be inspected is appropriately scaled to fit the software working environment. Model scale, S_L is the ratio of the height of the infrastructure L_p to the CAD model height, L_m.

$$S_L = \frac{L_p}{L_m} \tag{3.1}$$

Commonly, Boiler heights range between 12–30 m. For this work, a 17 m high boiler is designed. Due to the tool (CAM tool) and extruder (3D printing tool) size limitations, the overall power plant's CAD model height is scaled to 170 mm. From equation 3.1, S_L calculates to 100:1.

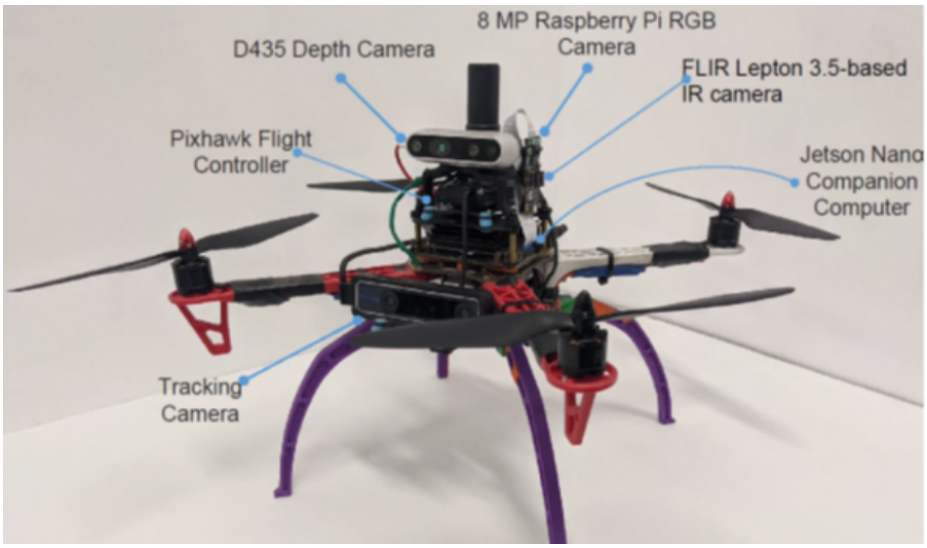


Figure 3.3: UAV Equipped with Inspection Payload

Once path planning is done in the model phase, the coordinates, i.e., the UAV waypoints, are converted to real-life scale using the 100:1 model scaling ratio. From the conversion, a 100 mm model distance between coordinates converts to a 10 m distance for the real use case.

3.3.2 System Description:

The system shall perform a close-quarter inspection to catch any surface detected at a close range of 1 ft in a GPS-denied environment. For trajectory generation, the aerial platform used here is a quadrotor, as seen in Figure 3.3. All data and parameters are selected from the said UAV system for the corresponding calculations. However, this study focuses on offline trajectory generation and validation for proof of concept of the proposed approach.

The UAV carries an inspection payload (IP) with an RGB camera, Raspberry Pi 2.0. Relevant specifications of the camera are given as Annex 2.2.2, collected from the sensor module documentation [19]. For close-range inspection, the surface to UAV's camera distance is selected at 500 mm (1.5 feet) as the IP, including the camera, is mounted at the body frame of the platform. The camera takes images or video of the surface at the pre-defined offset as it flies around the structure at a certain height, tracing its perimeter. Once it has covered the area, it moves to the next layer by a specific height. To properly cover all surfaces, the UAV flight path will fly from

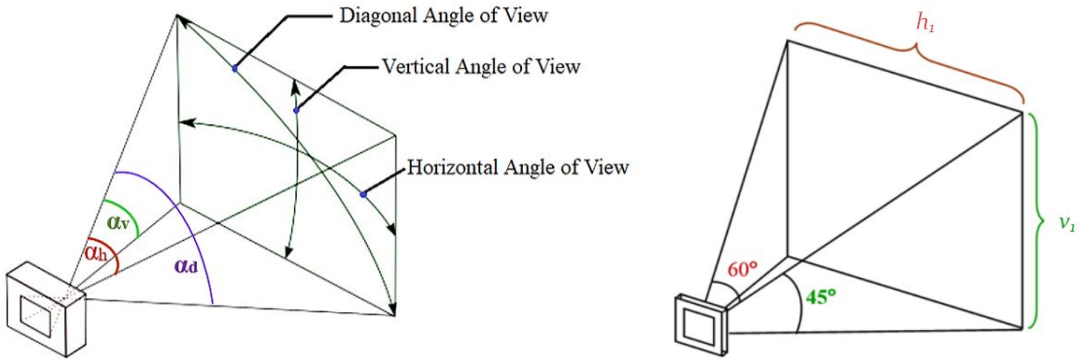
the bottom to the top of the structure, going one layer at a time. During the flight, the inspection sensors shall always face toward the inspection surface. The camera field of view, focal length, frame grid, and frame rate are considered while planning the trajectory.

Camera FOV (field of view)

From the following relation, for an effective focal length f and the angular field of view α, correlated dimension d can be calculated and vice versa.

$$\alpha = 2 \arctan\left(\frac{d}{2f}\right) \tag{3.2}$$

Figure 3.4a shows the camera fields of view. It can be measured horizontally, vertically, and diagonally. Relevant specifications, including the angular camera fields of view, can be found in Annex 2.2.2. For inspection grid calculation, the horizontal α_h, vertical α_v and diagonal α_d angles are approximately taken as 60°, 45°, and 70°, respectively.



(a) Angular FOV (b) Inspection Grid

Figure 3.4: Inspection Camera's Field of View (FOV)

For calculation, the camera angular FOV is taken. The sensor has a focal length of 3.04 mm. In close-quarter inspection, for a minimum 500 mm inspection surface distance, with $\alpha_h = 60$ °, $\alpha_v = 45$°, and $f = 5$ mm, the inspection grid horizontal length, h and vertical length, v are calculated as follows:

$$h = 2f \tan\frac{\alpha_h}{2} \Rightarrow h = 5.8 \tag{3.3}$$

$$v = 2f \tan \frac{\alpha_v}{2} \Rightarrow v = 4.1 \tag{3.4}$$

A 30% overlap of adjacent image/frames is selected, considering the flight time internal and external perturbation for a high-quality inspection profile. The inspection camera's frame grid is then calculated as,

$$h_1 = 70\% of h = 4.06 \approx 4 \tag{3.5}$$

$$v_1 = 70\% of v = 2.87 \approx 2.5 \tag{3.6}$$

Hence, the model scale FOV calculates to a (v_1) at 2.5 mm, which correlates to the maximum step down/feed for machining tool of CAM and the layer thickness for AM; and the number of parallel tool-path or slices is 68. The real-life application FOV calculated is 400 mm x 250 mm approximately, and the 17m boiler can be covered with 68 horizontal layer or flights with a jump of 400 mm or 1.3 feet to the next layer (Fig. 3.4b)

Drone Velocity Range:

Average drones for photography have a velocity range between 30–50 mph, with a suitable FPS (frames per second). For an average speed, $v_{max} = 20$ mph = 8.9408 m/s , the FPS for the drone camera is the ratio of the camera grid horizontal length (h_1), i.e. when the drone is travelling 8940.8 mm in a sec (8.94 m/s),

$$\begin{aligned} \text{FPS} &= \frac{8940.8}{400} \\ &= 22.352 \end{aligned} \tag{3.7}$$

The proposed camera has 40-200 FPS for a size resolution of 640×480, which is well within the calculated range. However, considering close-range flight in a constrained area, the performance of the low-budget camera, power rationing for other additional inspection sensors (IR sensor), and overall flight, one-fourth of the lowest FPS, is chosen for velocity calculation to be cautious. The system can effectively perform photography and video inspection, flying at 0.5 m/s -4 m/s with the IP.

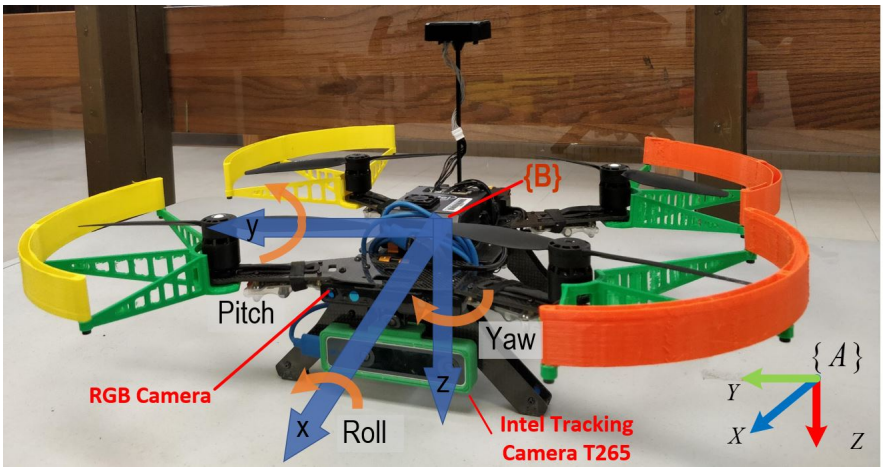


Figure 3.5: Drone Reference Frames

3.3.3 Inspection Kinematics

For analyzing the motion of the drone, an inertial frame {A} (or world frame) and a body coordinate frame {B} are established. {A} acts as the global reference for the plant to be inspected. Following the aerospace convention, {B} has its origin coincided with the center of gravity of the quadrotor, the z-axis pointing downward, and the x-axis in the direction of the flight. Figure 3.5) shows the frames on a drone as example. Besides, the vector $\boldsymbol{r} \in \mathbb{R}^3$ gives the position of the drone in inertial frame {A} as depicted in Fig. 3.6.

The inspection cameras are mounted on the UAV chassis so that the camera location can be taken as the point of origin or zero point for it. The UAV propellers go beyond the origin by a_d. Again, from the endpoint of the propeller to the nearest inspection surface point can be assigned as d_s. Therefore, the total inspection offset from the camera at any time is,,

$$d = d_s + a_d \tag{3.8}$$

For a safe flight path planning, a minimum drone offset distance d_s is introduced. Also, due to the design of the drone, the rotor blades expand a certain distance from the camera that also needs to be considered along with the camera focal distance itself. Therefore, the offset distance should be, $d \geq d_s + a_d$ and $d > f$ at any time (Fig. 3.6). As such, the position vector for the UAV camera is,

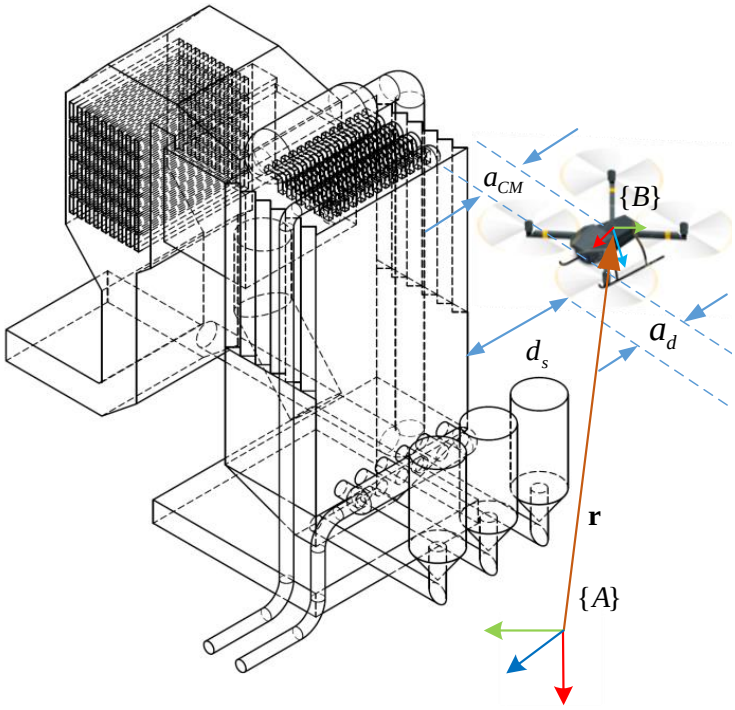


Figure 3.6: Kinematic Diagram: Frames Definition and Inspection Distances in a Boiler (Case Study)

$$\boldsymbol{r} = \begin{bmatrix} x+d & y & z \end{bmatrix}^T \tag{3.9}$$

3.3.4 Toolpath Generation Methods

The proposed method introduces an approach to model the UAV flight path from the path of a tool of advanced manufacturing machines, e.g., a CNC machine or 3D printer. The tool, cutter (CAM), or extruder (3D printer) will commonly be addressed as a 'tool.' Two approaches are introduced, one using CAM and the other using AM. For both methods, the tool path is set with a predetermined offset for the UAV for collision avoidance, as carefully calculated in the previous steps.

The tool path data is generated for a selected machining operation using the manufacturing environment of the CAD/CAM software package. From each operation type, e.g., milling, the nodal coordinates along the path of the tool, tool feed rate, and type of motion are obtained as

G-code. The tool path followed in a CAM milling operation and AM process is essentially similar; The former is a 'subtractive' process where the tool travels in the -Z axis (top to bottom), and the latter is 'additive' where the tool advances towards the +Z axis (bottom to top). During both operations, the tool moves by layer. These signs are adapted to the flight trajectory at the post-process step by coding targeted programming. Aside from these, both approaches are well-aligned. In both methods, the tool paths are predetermined. The tool can follow, for example, a zigzag or a peripheral path. Some relevant and important pre-set global parameters are tool radius, depth of cut, and step-over cut (layer-to-layer distance).

These paths are translated to the UAV flying path, encompassing a structure to be inspected with an appropriate offset. CAM or AM methods can be used interchangeably to generate the UAV trajectory successfully.

Method 1: Computer Aided Manufacturing (CAM) Based

A. Toolpath Generation with 2D Contour Operation: At first, the machining properties are set in the "Manufacture" environment of Fusion 360, for example, choosing an appropriate type of manufacturing operation, choosing an apt stock offset, defining the working coordinate system, and post-processing setup.

A stock block is chosen relevant to the inspection study structure. The stock offset is the surface to quadrotor inspection offset, 'd' (equation (3.8)). A 2D contour milling operation is chosen as it can generate a uniformly distributed toolpath to obtain a given geometry through machining. The machining parameters are aptly chosen, which gives a uniform toolpath well wrapped around the powerplant model (Table 1) to make it acceptable as a proper UAV flight trajectory (Fig. 3.7). A 2.5 mm diameter ball cutter is chosen; its endpoint machines out the model from the stock. The toolpath is captured to extract coordinates.

Two important parameters are stock offset maximum step down. 'Stock offset' corresponds to the linear distance from the inspection surface, which correlates to the camera to inspection surface distance. 'Maximum step down' is the drop of the cutting tool as it travels to the next layer. These parameters are determined, and their values can be amended or chosen depending on the inspection path and profile as required. For example, the minimum safe offset distance has been selected to be 500 mm. However, it can be increased or decreased depending on the type of

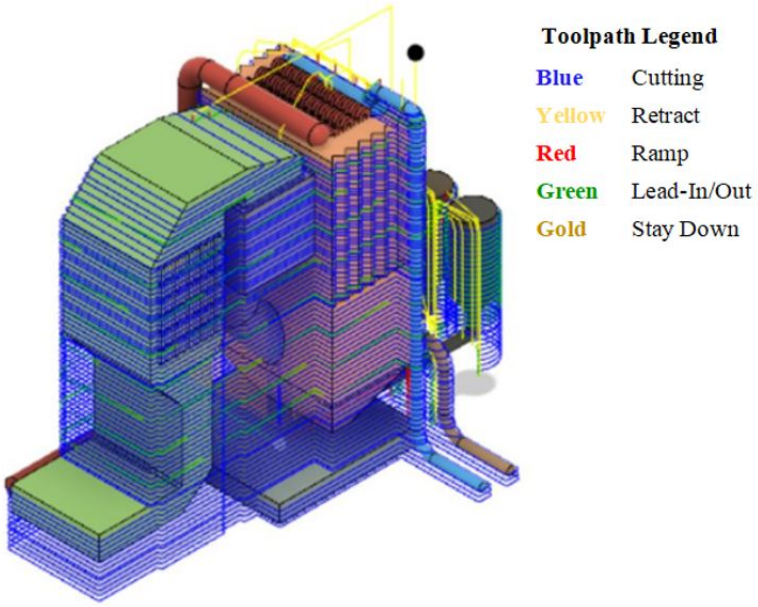


Figure 3.7: Tool Path Generated Around the Boiler CAD Model for 2D Contour Operation

Table 3.1: CAM Contour (2D) Machining Parameters

Operation type	**2D Contour Milling operation**
Cutting Tool	ϕ2.5 mm ball end mill
Maximum Step Down	2.5 mm
Tolerance	0.01 (low)
Stock Offset	5 mm
Cutting Feed Rate	1000 mm/min
Machining Distance	71.03 m
Feed/Revolution	0.0667 mm/rev
Surface Speed	39.27 m/min
Machining Time	1:10:14

inspection platform and the IP, especially the camera capabilities, to realize the desired outcome.

In Figure 3.7, the 'blue' toolpath (lines) describes the machining operation (cutting the stock), whereas the 'red' and 'yellow' lines delineate tool ramping and retraction, respectively. Given tool

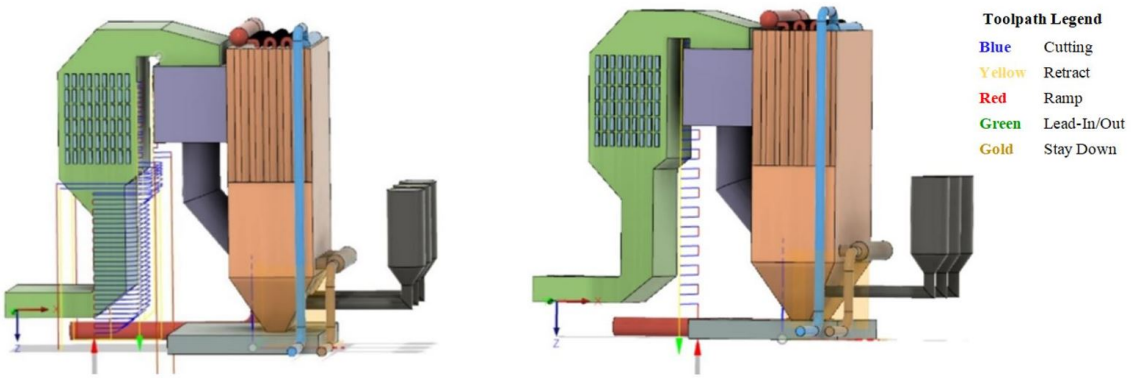


Figure 3.8: Tool Paths Generated with Modified Tool Orientation and Operation for the Component Gaps and Inward Slope of the Structure

orientation and the principle of CAM operation, the gaps, model overhang, and inward slopes are avoided as expected. Considering the camera's superior zoom capabilities, the parallel toolpath around the inward slope of the exhaust is acceptable. However, tool orientation is changed to get the best possible wrapped toolpath of the said areas.

B. Toolpath Generation with Modified Tool Orientation: For the inner gap of the powerplant model, again 2D Contour milling operation is chosen, other parameters the same (Table 3.1). The Z axis of the tool orientation along with Y axis are flipped 180°(Fig. 3.8(a)). Z axis directs the tool entry to the workpiece. For this case, it is the same as assuming that the workpiece has been flipped for operating from the bottom surface of the setup, whereas the X axis is setup according to the manufacture operation coordinate principle. The inner gap between the boiler and exhaust can also be addressed with "Slot" milling operation (Fig. 3.8(b)). This is a step toolpath that requires a different set of parameters (Table 3.2).

Similar to the operation earlier, the flipped Z axis points downwards. Although the toolpath can reach the shallow gap found on the upper chamber of the exhaust, it is not considered due to the aerodynamic restriction of a UAV. The operation stops near the constraint area of the "particulate recirculation" component of the plant.

For all the operations, 'smooth transition' and 'plunge in' options were selected to avoid

Table 3.2: CAM Slot Machining Parameters

Operation type	**2D Slot operation**
Cutting Tool	ϕ2.5 mm ball end mill
Maximum Step Down	5 mm
Tolerance	0.01 (low)
Stock Offset	1.52 mm
Cutting Feed Rate	10000 mm/min
Machining Distance	5 mm
Feed/Revolution	0.0667 mm/rev
Ramp Type	Plunge
Surface Speed	78.5 m/min
Machining Time	1:00:50

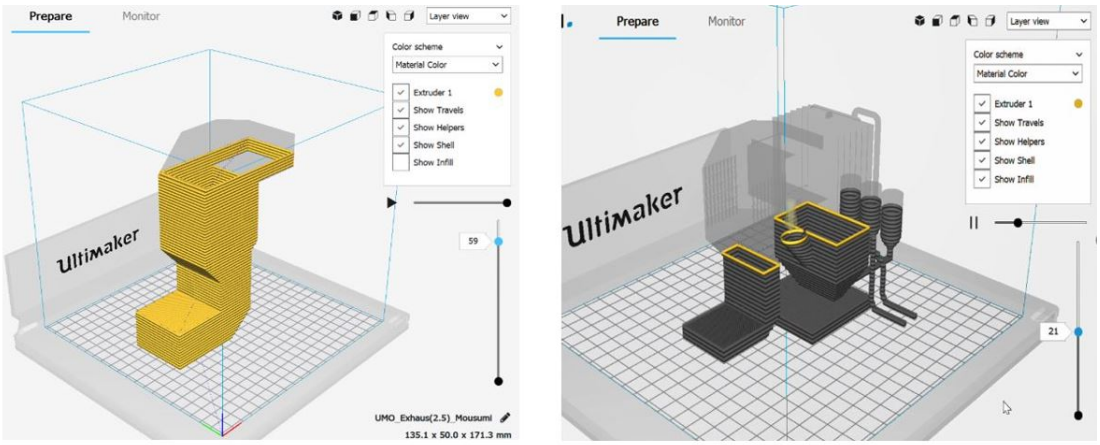
sharp changes in the heading of the UAV during flight. Given the CNC tool safe retrievals/lead-ins/lead-outs, additional noise is observed from the generated toolpath. Mentionable that the tool time information, e.g., feed rate, revolution, and surface speed, are ignored in the final trajectory generation. Only the tool's end point coordinates are pulled as a point cloud. Afterward, a UAV-appropriate time-step is set up.

Method 2: Additive Manufacturing (AM) Based Method

In order to obtain the external coordinates with a preset offset, a CAM based approach was used previously. The CAM base method has some challenges that are inherent to the CAM characteristics, for example:

1. Machining of certain features is difficult, e.g. inward slopes larger than 45°, or convex surfaces.

2. Difficult to reach shallow gaps between design parts due to tool size limitation. The tool tilt angle is also limited due to physical constraint and workpiece orientation etc.



a) Exhaust of the powerplant b) The powerplant

Figure 3.9: 3D Printing Layers Preview of the Model

To overcome these challenges, an "Additive Manufacture (AM)" based method is introduced with significant advantages. Essentially, a path that is followed in a CAM milling operation and a 3D printer (in AM) are very similar, one being subtractive, tool travelling in -Z axis (top to bottom) and the other being additive i.e. tool travels in +Z axis (bottom to top) (Fig. 3.9). During the operation of both manufacturing methods, the tool moves one layer at a time.

The second method uses the AM tool (extruder) path's coordinates at a pre-set offset to generate a UAV trajectory. The first CAM-based approach has some challenges due to inherent CAM characteristics, for example:

1. Certain features, e.g., inward slopes larger than 45°, and convex surfaces can not be machined as the tool cannot reachable without breaking the surface integrity.

2. Shallow gaps between design parts are difficult to access due to tool size limitations. The physical constraint and workpiece orientation limit the tool tilt angle.

The AM-based method can overcome these challenges introduced along with several other benefits. The 3D printer (in AM) path is additive i.e. tool travels in +Z axis (bottom to top) (Fig. 3.9). During the operation, the tool moves one layer at a time.

A commercially available AM 'slicer' software, 'Ultimaker Cura 3.6.0' is used to generate a 3D shell of the model with the pre-selected offset of 5 mm (model scale), similar to CAM

method (Table 3.3). The hatch distance (layer height) in an AM slice corresponds to the vertical displacement of the UAV during the inspection flight. Structure support is ignored, and some default parameters have been applied to get a general shape of the model. This operation generates the toolpath coordinates of the shell at the selected offset. AM method covers all surface areas, including slopes, overhangs, and shallow gaps. The G-code generates a layer-by-layer, bottom to upward Z axis values, with associated XY coordinates (Fig. 3.9). This method is preferred between the two, as it is much more straightforward and covers all surfaces without much modification for generating the final trajectory

Table 3.3: Additive Manufacturing Parameters

Printer	Ultimaker Original+
Material	PLA
Profile	Normal (0.15 mm)
Layer Height	2.5 mm
Wall Thickness	0.5 mm
Top Thickness	0.1 mm
Bottom Thickness	0 mm
Infill Density	1%
Print Speed	700 mm/s
Travel Speed	700 mm/s

3.3.5 Post-processing

The 'post-processing' step is all about exporting the tool path coordinates. The G-code is a file containing all detailed machining instructions. It contains the required nodal coordinates (tool endpoint), types of motions, and necessary instructions. It also contains information regarding different cutting modes and tool information, e.g., tool speed, tool type, tool orientation, machining direction, operation specification, and other critical information specific to the manufacturing machines. However, not all information is needed in this project for generating a UAV trajectory.

In 'Post Process,' the XYZ coordinates are exported from the G-code file, clearing unnecessary data. The 3-axis milling operation in the CAM-based method uses a software add-on to extract the tool coordinates directly. For the other approaches, G-code-specific programming is used to decode the coordinates. Once the coordinates are collected, they are modified using MATLAB code to get an optimized trajectory.

Direct Data Extraction

In this first step of path planning, the Cartesian coordinates of the tool endpoints are exported using a post-process add-on (Fig. 3.10). This method is only applicable to the 3-axis milling operation. Due to the machining characteristics, the point cloud is very dense, as seen in said figure. There are about 109,575 points for the 'exhaust' component alone. Here, all points are organized and brought into one global coordinate system. In the second and third operations under the CAM method, the tool axis (Z and Y axis) were flipped. In this step, those coordinates are adjusted, and data is reset and re-arranged such that the Z values increment. Final values are exported in a CSV file.

Targeted G-code Decoding with MATLAB

As mentioned, G-code files contain detailed machine instructions, including the required tool path/nodal coordinates. G-codes are machine specific, i.e., it varies depending on the manufacturing machine it is programmed to run. This work uses custom MATLAB coding depending on the type of G-Code (CAM/AM) to extract coordinates and discard other useless information. The complete model attains a dense plot of 36525×3 double points, including all the tool path coordinates (Fig. 3.11A).

Tool Path Optimization

The targeted programming successfully extracts all the coordinates. However, such a dense cloud is unnecessary (Fig. 3.11A). Depending on the model's geometry, e.g., flat vs. curved, the point cloud density also varies. There are arbitrary jumps between layers that take the given trajectory paths through the surface mesh itself. In AM method, there is minimal support generation at

	1 Var1	2 Var2	3 Var3
1	0	-4	0.2800
2	-1.5717	-1.9680	0.2800
3	-1.5833	-1.9678	0.2800
4	-1.5948	-1.9672	0.2800
5	-1.6062	-1.9662	0.2800
6	-1.6177	-1.9648	0.2800
7	-1.6291	-1.9630	0.2800
8	-1.6404	-1.9608	0.2800
9	-1.6516	-1.9582	0.2800
10	-1.6627	-1.9553	0.2800
11	-1.6737	-1.9519	0.2800
12	-1.6846	-1.9481	0.2800
13	-1.6954	-1.9440	0.2800
14	-1.7060	-1.9395	0.2800
15	-1.7164	-1.9347	0.2800
16	-1.7267	-1.9294	0.2800
17	-1.7367	-1.9238	0.2800
18	-1.7466	-1.9179	0.2800
19	-1.7563	-1.9116	0.2800
20	-1.7657	-1.9050	0.2800
21	-1.7749	-1.8981	0.2800
22	-1.7839	-1.8908	0.2800
23	-1.7926	-1.8833	0.2800
24	-1.8010	-1.8754	0.2800

Figure 3.10: A Sample of the Initial Cartesian Coordinates (xyz) Exported in Excel

the bottom of the structure. Also, due to the safety feature of machining operations, there are additional tool movements during retraction at the beginning and end of operations in the CAM method. These noise and conflicts are removed, and the number of points is reduced in this step using MATLAB code to attain a uniform external trajectory pattern (Fig. 3.12). The significant reduction of the point cloud (about 1/100th) results in a uniform trajectory which is also computationally less expensive when fed to the inspection platform (Fig. 3.13).

Trajectory is further modified, and an improved trajectory is achieved that has significantly smaller number of points (373x3 double) as well as a conflict-free trajectory (Fig. 3.12). Also, the significant reduction of point cloud (almost 1/100th) will result in better trajectory and computationally less expensive when fed to the inspection UAV (Fig. 3.13).

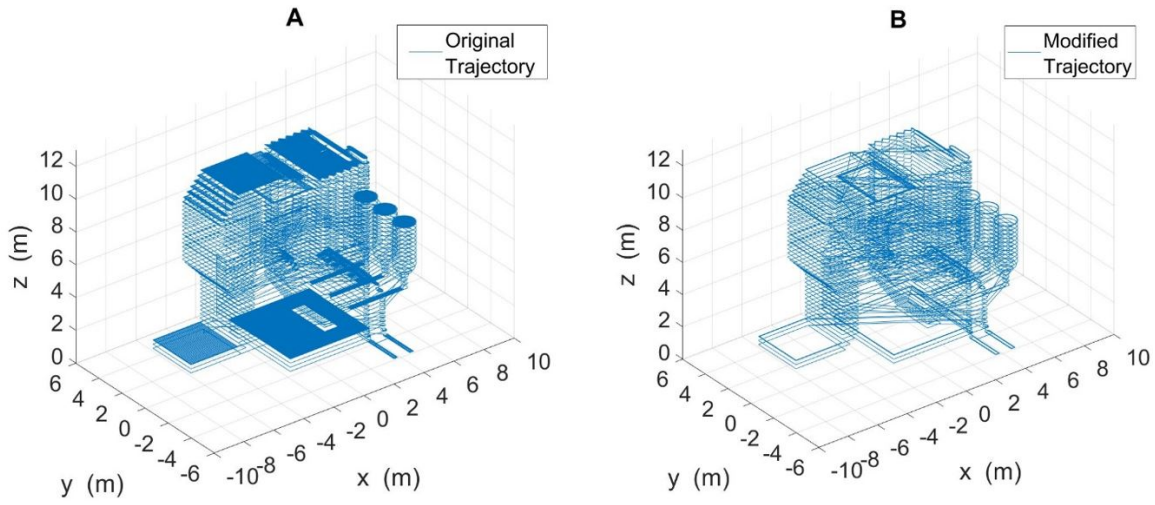


Figure 3.11: Coordinates Obtained from a G-Code (AM Method) of the Structure Using MATLAB.

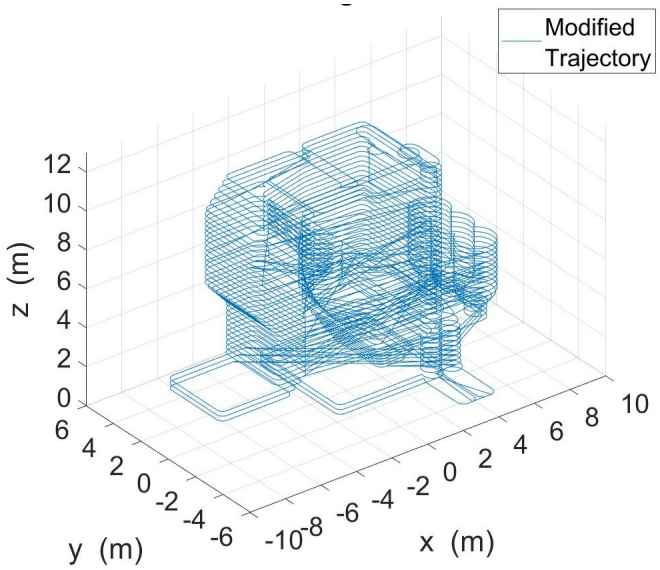


Figure 3.12: Final Modified Trajectory the Structure (Case Study) Obtained Using MATLAB Code

3.3.6 Aerial Inspection Trajectory Generation

After post-processing, both methods deliver a set of Cartesian coordinates of the tool path way points in 3D space. Time steps are added to these coordinates and further processed to generate

Figure 3.13: Significant Reduction in the Number of Points from the Original Slicer G-code ('toolpath1'), a Modified Trajectory ('toolpath2') to the Final Trajectory Generation ('toolpath3') Using MATLAB Code

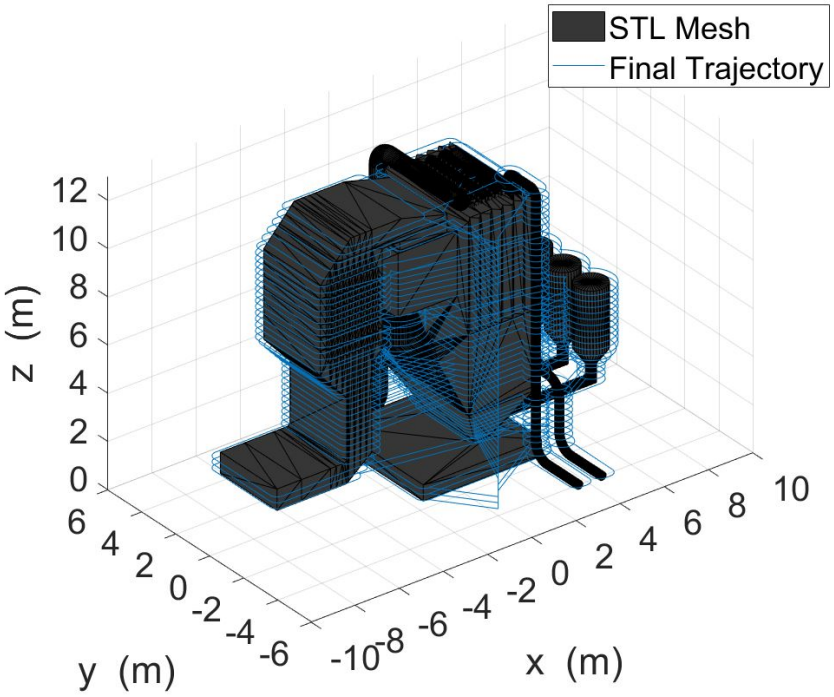


Figure 3.14: Flight Path Generated in MATLAB for AM Method

an inspection trajectory or a flight path. In Figure 3.14, black corresponds to the surface, and the blue lines portray the trajectory. The AM method's projected trajectory is much more uniform, smoother and effectively covers all surfaces irrespective of the geometry compared to the CAM method. Hence, AM method is preferred over the other. Mentionable that if an original CAD model of a structure is used, there may be some discrepancies with the real setup, e.g., loose pipes, hanging wires, or moved parts due to the operation of the power plant over time. This information will be absent in an old CAD, and even with an updated CAD, the stakeholders may

not know about these unforeseen deviations at remote locations. To avoid collision caused by such deviation, the UAV must have an anti-collision system utilizing other onboard sensors. The final AM method-generated UAV trajectory around the power plant is validated in simulation.

3.4 Trajectory Verification

Flight paths are verified with simulation software Gazebo, a powerful robotics simulator with a robust physics engine of a 3D environment. PX4, a professional autopilot software, is used to implement different flight modes and safety features to control the UAV. This software is run on simulation in a personal computer, and 'QGroundControl,' a graphical interface, is used to interface with PX4 to set up the drone, i.e., update flight parameters, create and execute flying missions, and visualize real-time telemetry data. A program-defined API, 'DronecodeSDK,' is used to provide the functions for interacting with the virtual drone through PX4. Using Gazebo and PX4, the experimental setup is virtually built with a general quadcopter model to show how a real drone would act, given the AM-generated coordinates. The proposed simulation steps consist of the following steps:

1. A C++ code is used to read and reformat the CSV file comprising the trajectory point

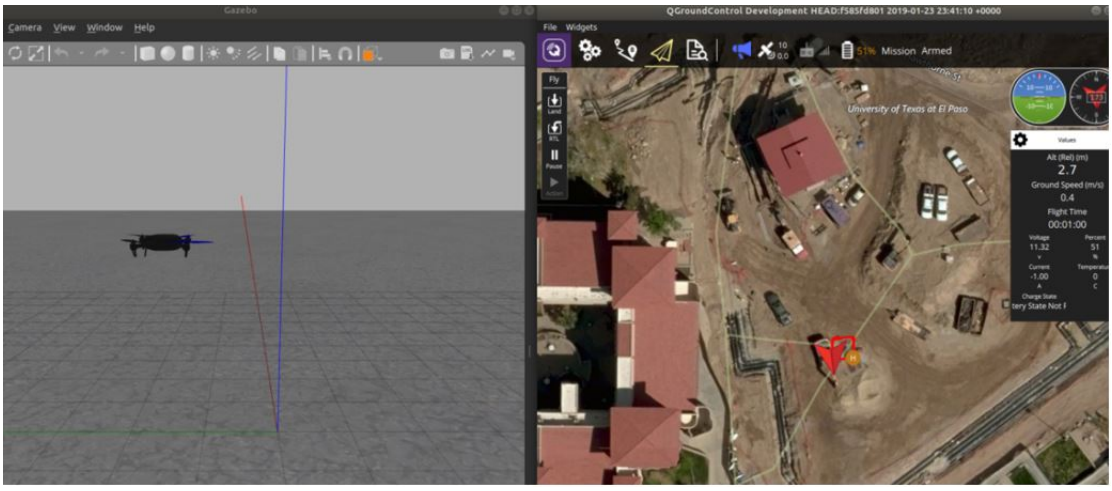


Figure 3.15: Gazebo Simulation at Run-time and QGroundControl Plotting the UAV trajectory of the Test Mission in Real-time

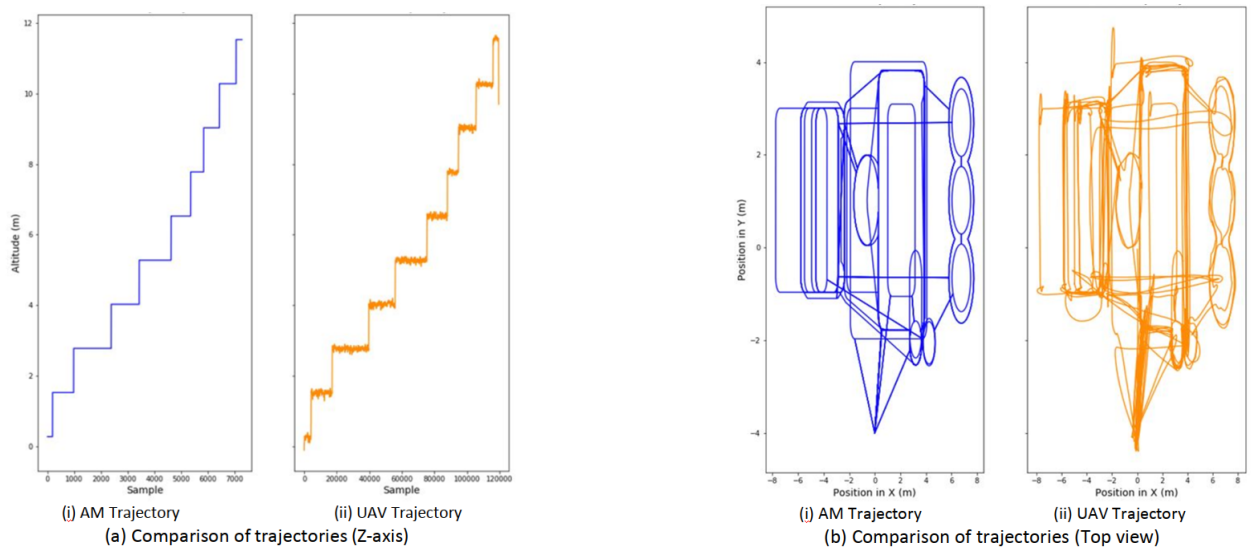
coordinates.

2. Using the DronecodeSDK libraries, a mission item is created based on the trajectory points.

3. Mission is uploaded and executed on the simulated quadcopter.

4. The mission is visualized in real-time using QGroundControl and RVIZ visualization environment of ROS (Robotic Operating System) framework while getting flight logs of the UAV's trajectory for offline analysis.

For verification, the AM-generated trajectory of the power plant CAD model is used to formulate the drone flight mission. The flight mission is studied with real-time visualization (Fig. 3.15). As seen in the figure, the flight mission and the telemetry data can be visualized in Gazebo and QGroundControl, respectively, and the logs are saved using ROS.



(a) (b)

Figure 3.16: Comparison of: a) The Z Position of the Tool Based on (i) the AM, and (ii) the Altitude of the UAV During Flight, b) the Top View of Both (i) the AM Generated Trajectory, and (ii) the UAV Inspection Path.

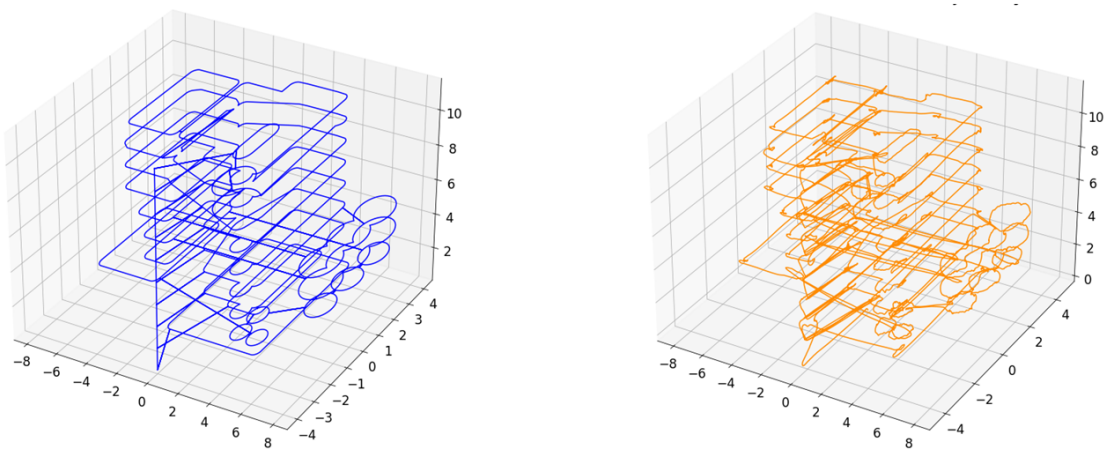
From the flight log, the real-time/actual drone trajectory is studied and compared with the

AM one (Fig. 3.16a and Fig. 3.16b). The results show that the trajectory pattern closely follows the given trajectory, with some variation. From the top view, the compared trajectory shows the general shape of the power plant, with some smoothening in real trajectory at some points due to the UAV cutting corners. The front and right side views show the simulated trajectory following closer to the AM-generated trajectory, with some noise caused most likely due to the flight controller corrections. Some shapes are lost, which indicates the importance of post-processing the coordinates to preserve the relevant points of the mission.

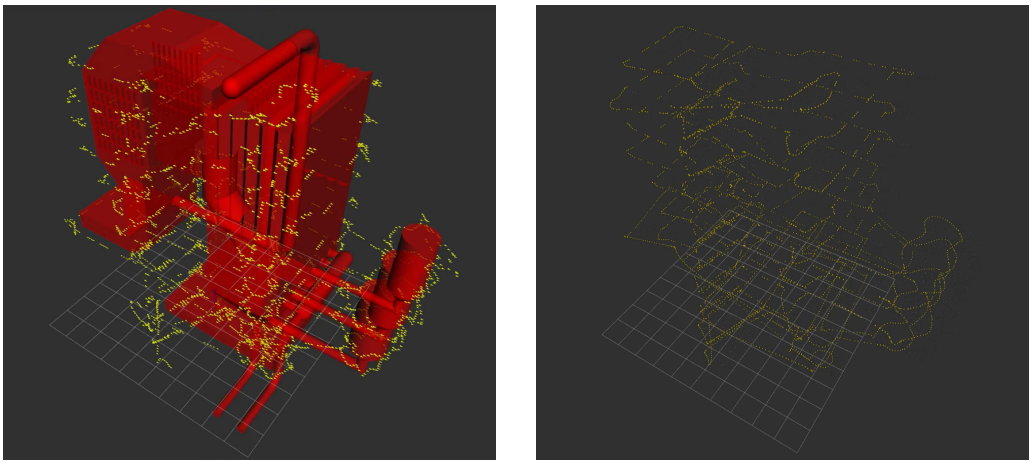
Figure 3.17 (a) & (b) compare the 3D views of the trajectories side by side. The same is visualized in Figure 3.17c using RVIZ, with and without the surface of the boiler model (red). The simulated flight path overall follows the general shape of the model, even though it is a bit wavy and smooth at some points. Mentionable that the number of layers in the simulation study has been reduced from the original number for better visualization, and as a proof of concept of the adaptability and validity of this method. This study validates the AM and CAM-based methods for generating trajectories, demonstrating that these paths can generate an actual UAV flight mission.

3.5 Discussions

The waviness, non-linearity, and offshoots from the given flight path are mainly caused by the drone control parameters, e.g., attitude control, gain, and recovery, which can be adjusted in the simulation software with the drone engine. The mission parameters and drone navigation can be optimized further for this application, so it does not invalidate the proposed approach. The implementation of collision avoidance algorithms should allow the trajectory paths to be even more similar to the ideal AM-generated trajectory, which can be explored in future work. The simulation outcome proves that the proposed advanced manufacturing technology-based aerial inspection trajectory generation method can certainly be implemented using a real UAV controlled by a professional flight stack.



(a)



(b) (c)

Figure 3.17: Comparison of the (a) Ideal Inspection Trajectory Generated by AM and (b) the Actual Flight Inspection Path Followed by the UAV and (c) Inspection Flight Path Point Cloud Visualized with the RVIZ Environment With (Left) and Without (Right) Inspection Surface (Boiler)

3.6 Conclusion

A novel advanced manufacturing technique-based method is introduced to inspect structural components of power plants and industrial infrastructure using aerial robotics technology. Two approaches proposed were based on the tool path of the Computer Aided Manufacturing (CAM) and Additive Manufacturing (AM) machines. Among the two methods, AM offers a better and more straightforward method as it can include all types of geometries, including overhangs and shallow gaps in components. The proposed trajectory generator seeks to permit close-quarter visual inspection to acquire more details in infrastructures with complex geometries with reduced flying space, even without a GPS signal to perform a visual inspection. This approach needs precise navigation to localize the vehicle, which would not be suitable for a GPS-based system due to the precision it requires. Also, the GPS signal will be intermittent or non-existent in most cases when inspecting the interior of an industrial compound. The proposed method would need several flights to cover large areas due to the limited flight time of UAVs. Simulations conducted in the open-source robotics simulator Gazebo and with flight controller PX4 demonstrated the feasibility of the methods to construct UAV offline trajectories. Due to the offline definition, this method would require an up-to-date CAD model of the infrastructure. The obstacle detection and avoidance systems will complement the system to account for un-modeled and unforeseen objects in the facility to inspect.

Chapter 4

Intelligent Crack Detection Using Deep Learning

4.1 Introduction

Automated inspection and preventive maintenance have been pushing the research community for a while now. The available conventional infrastructure inspection methods for cracks suffer from human error and accuracy issues, take time and cause downtime. It is also heavily reliant on the experience and expertise of the inspector. The observation and analysis may differ from one to another based on their perception. There have been occasional incidents when human inspectors missed identifying cracks while inspecting large areas with inaccessible parts, e.g., underneath a structure or components at a height. This work proposes, develops, and applies a DL-based algorithm as part of an inspection payload (IP) to resolve these complications. IP takes images of the infrastructure as input and performs intelligent inspection of cracks in offline (remote) and online (real-time) modes at the edge. The intelligent and real-time capability speeds up the inspection process and gives a consistent performance.

4.2 Related Work

There are significant advantages to automating the process compared to a manual one. However, the automated process needs to be intelligent and perform the task satisfactorily. Adopting the latest and relevant AI-driven solutions could be a promising approach that will pay dividends.

4.2.1 CV-based Approaches

Structural Health Monitoring (SHM), a maintenance system, demands timely damage identification and its subsequent characterization to monitor structural integrity at the locations of interest. There are four steps required in classical machine learning (ML) algorithms to develop a structural health monitoring system, including (i) operational evaluation, (ii) data acquisition, (iii) feature extraction, and (iv) statistical modeling [38]. Operational evaluation determines if it is justified to develop the system economically and from a safety perspective, including deciding on the scope of damage detection. The data acquisition step determines the means to acquire data, especially considering the time required and financial aspects. Feature extraction is the meaningful extraction of information and necessary values from raw data to facilitate training. Finally, the statistical modeling includes training and evaluation of the ML algorithm. These steps can be adopted for crack detection in light of the inherent difference between the modeling of the classical ML to that of a DL algorithm.

Depending on the feature extraction technique, image-based crack detection can be broadly divided into two categories, manual and automatic. The low-level traditional computer vision (CV) based approach is the former kind, and the model performance depends entirely on pertinent feature selection. Otherwise, it could lead to an inaccurate and unexpected results. Unlike CV-based techniques, DL uses high-level feature extraction techniques similar to the neurons of the human brain to identify & learn relevant information automatically [15]. From manual handcrafted feature extraction to the usage of DL architecture for automatic feature extraction has happened in different areas, e.g., video analysis [39], image reconstruction [40], medical imaging [41], medical diagnosis [42], and also in crack detection [1, 43]. Due to the promising result in other applications, the DL-based approach is opted for in this work.

DL provides a plethora of models as options. There were two conditions to consider, the model should adequately identify and localize the model, and it should be deployable in real-time. The requirement is drawn from the rising demand to make the inspection process time-sensitive, i.e., reducing the system downtime by automating the process and deploying it at the edge of the inspection. In this section, existing work related to crack detection in SHM is reviewed under three sub-sections: (1) Image Classification, (2) Object Detection, and (3) Semantic Segmentation. The

main difference between each type of DL technique is the level at which the detection is performed, e.g., image, image patch, and pixel level, respectively. These are briefly discussed below.

4.2.2 DL-based Models

4.2.2.1. Image Classification Technique

The Image Classification (IC) technique is a binary approach that determines whether there is a crack in the image. Another patch-based approach is the network detecting patch-level cracks and only showing patches with positive crack identification. The overall IC architecture consists of two parts, first is layer-by-layer meaningful feature extraction from raw images, performed by successive convolutional and max-pooling layers. The second part is responsible for extracted feature classification by the first part using fully connected layers [1]. One of the major conclusions of work by Pauly et al. is the effect of the number of deep layers (depth) on the detection capability of the network, which determined that deeper networks enable the architecture to learn more information [44]. However, a well-known fact is that deep architectures require a large quantity & quality of annotated data, making their application in different areas with limited available data challenging. One proven and effective method is transfer learning, which uses pre-trained networks trained on large-scale annotated image data sets, e.g., ImageNet [45]. Several work that used application of transfer learning, used different pre-trained architectures such as VGG-16 [46, 47, 48], AlexNet [49], Resnet [50] etc. trained on ImageNet. Another interesting research in the IC category would be using the sliding window technique using MatConvNet [51] to perform crack detection [1]. In another work same technique is used for a crack detection method using UAV with geo-localization [52]. Other architectures such as LeNet [53] and ANIVOS have also been employed to detect crack [54].

However, from the problem point of effectiveness of crack detection, localization is also highly important for defect identification and characterization. Although the IC setting provides the crack images/image patches, the main limitation is that the overall outline obtained by stacking the positive patches together of cracks in the images is coarse and does not provide detailed localization of the cracks appropriately.

Figure 4.1: Input (Left) and Typical Output (Right) of a Deep Learning-based Image Classification (IC)

Figure 4.2: Input (Left) and Typical Output (Right) of a Deep Learning-based IC by Stacking Positive Class Patches [1]

4.2.2.2. Object Detection Technique

In computer vision, the object detection (OD) technique detects an object of interest in an image and localizes it by encapsulating the area in bounding boxes [55]. There are several families of state-of-the-art OD architectures available, such as 'Region-based Convolutional Neural Networks' (R-CNN), 'Single Shot Detectors' (SSD), and 'You Only Look Once' (YOLO). In the crack detection test, an OD-inferred image has the class label and a bounding box around the crack.

R-CNN: All three members of the R-CNN family, R-CNN, fast R-CNN, and faster R-CNN, have been used for crack detection. The R-CNN architecture has only been applied once for crack detection. One of the first works was to compare and conclude that using a faster R-CNN model to perform crack detection improves their previous study for using an IC setting with the sliding window technique. In another work, CrackDN used a pre-trained CNN to extract deep features to increase feature detection sensitivity. In addition, they added a different, improved version of

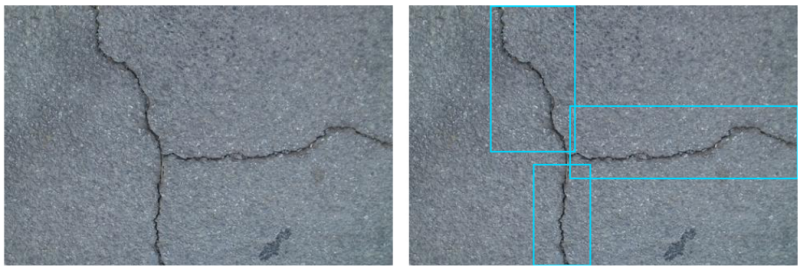


Figure 4.3: Input (Left) and Typical Output (Right) of a Deep Learning-based Object Detection (OD) [1]

faster R-CNN for crack detection. This concatenation divided the feature extraction and detection phase, resulting in a faster training procedure [56]. In another study, Faster R-CNN uses realistic occlusion and interference in images to simulate real scenarios, e.g., surface markings, writings, and other realistic scenes, to obtain a robust crack detection model [57].

SSD: Liu et al. first proposed SSD, which outperformed R-CNN in terms of speed on the Pascal VOC2007 dataset in OD [58]. As the name suggests, it does all computation in a single network without the additional step of finding a candidate region of interest, resulting in easy-to-train and straightforward integration with other frameworks for detection. It was used by Maeda et al. to perform crack detection on top of Inception V2 [59] and MobileNet [60] as the backbone feature extractors [61].

YOLO: 'YOLO' or 'You Only Look Once' is one of the popular OD architecture families. It performs OD tasks in one forward propagation, as its name suggests. The model was first introduced in 2016 and garnered attention for its faster and more accurate computer vision algorithms. There have been several models in the family, e.g., YOLO (2016), YOLO9000 (2016), YOLOv3 (2018), YOLOv4 (2020), YOLOv5 (2020), and YOLOX (2021). YOLO9000 was used by Mandal et al. [62] to perform different types of defects, including 'crack' detection, using a diverse data set. In recent work, Ghosh et al. using YOLOv3 [63] and Peraka et al. using pre-trained YOLOv4 [64] model with transfer learning developed a multi-distress detection system for pavement from high-resolution 3D images and video images respectively. These models have specific areas of application, e.g., pavements or bridges. Whereas this work concentrates on the 'crack' detection

of diverse infrastructures.

4.2.2.3. Semantic Segmentation Technique

A pixel-level classification is known as Semantic Segmentation (SS) Technique in CV [65]. The application of SS has the apparent benefit of precise pixel-level localization of the object in the image than OD application, and IC settings for the same purpose [66]. In other words, SS is a natural progression from coarser inference (such as patch-based, region-based, and object detection) to finer inference. Like surface crack detection work, Mask R-CNN, an updated SS version of the R-CNN family, has been used by Truong et al. to detect road distress using UAV imagery [67]. A typical output is shown in Figure 4.4.

Figure 4.4: Input (Left) and Typical Output (Right) of a Deep Learning-based Semantic Segmentation (SS) [2]

4.3 Model Selection

OD has been proven more accurate in detection and localization than the IC sliding window technique mentioned previously. One known limitation of OD is the undesirable performance and coarse localization, i.e., overlapping bounding boxes when it fails to capture the entire crack as a single object is problematic. For a vision-based approach to identify areas of interest for crack detection in industrial infrastructure, OD is fast, decently accurate, and has comparatively easier data preparation steps than SS. Therefore, this work selects OD technique-based approach, as accuracy is the leading goal. DLs are known to be computationally hungry with millions of

parameters, making them unsuitable for edge applications. The model quantization technique is used later in this work to resolve the issue.

4.3.1 Model Architecture & Configuration

The popular state-of-the-art, real-time object detection model 'YOLO' is used here. YOLOv4 (released in early 2020) is one of the popular versions in the YOLO family; in some studies, it was found that its accuracy is higher than that of the newer YOLOv5 model [68]. It is a single-stage DL algorithm that detects objects using a deep convolution neural network (CNN) called DNN (Deep Neural Network). There are different DL algorithms, but they cannot detect an object in a single run. YOLO can detect with a single forward propagation through a DNN, making it suitable for real-time applications. It is an OD-type network, so it also localizes multiple instances simultaneously. This single-run detection & localization capability of YOLO is why it is popular among other DL algorithms. The generic network framework is shown in Figure 4.5.

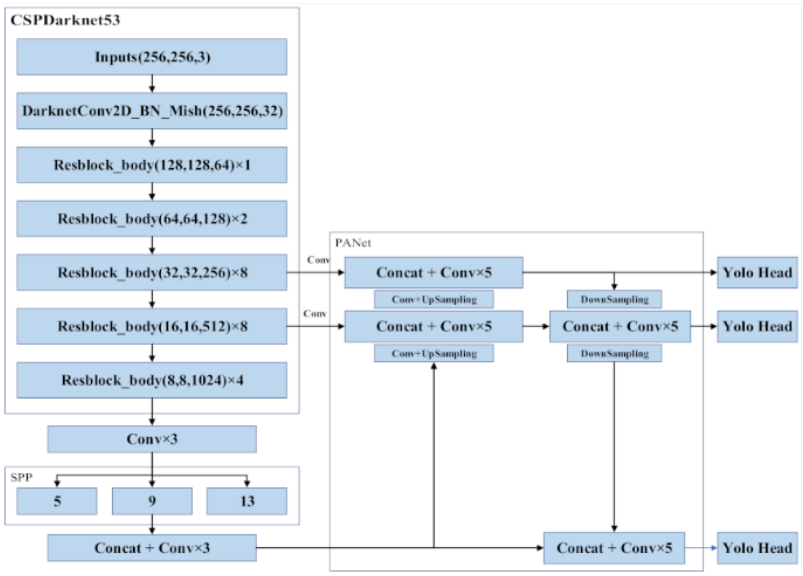


Figure 4.5: YOLOv4 Network Architecture [3]

The fourth improved version of YOLO or 'YOLOv4' uses a new feature extractor backbone called 'CSPDarknet53', which uses the 'Cross Stage Partial Network' (CSPNet) in the 'Darknet' framework. The 'ResBlock body,' the residual block of CSPDarknet53, extracts the target fea-

tures of the image and reduces the computational bottleneck and memory cost. Its architecture is based on modified 'DenseNet' and for GPU version, it outperforms the other backbones (Table 4.1) [69]. It carries one of the useful characteristics of 'Densenet,' that is, a copy of the feature map is transferred from the base layer to the next layer through a dense block. Doing so resolves some of the key issues of DNN, which include the infamous diminishing/vanishing gradient problems. It also removes computational bottlenecks, boosts back-propagation, and, by extension, improves learning. Certain backbones are more suitable for classification than for detection, e.g., CSPDarknet53 exhibits better OD performance than CSPResNext50 [3]. The issue to be addressed is the millions of model parameters and the BFLOPS (Billion Floating point Operations per Second) count at the edge devices for inference latency.

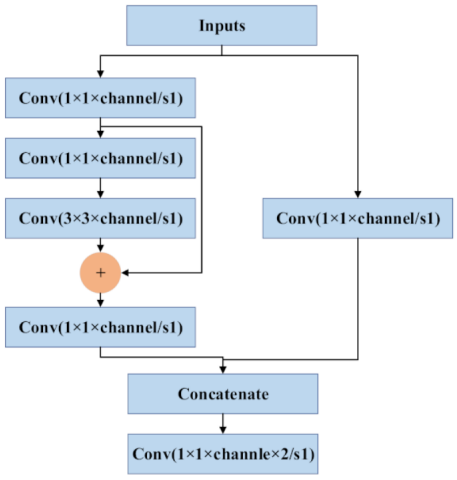


Figure 4.6: Residual Block Module Structure [3]

Table 4.1: Parameters of Neural Networks for GPU Versions with Different Backbones

Backbone model	Parameters	BFLOPS (512x512 network resolution)	FPS (GPU RTX 2070)
CSPResNext50	20.6 M	31	62
CSPDarknet53	27.6 M	52	**66**
EfficientNet-B3	12.0 M	11	26

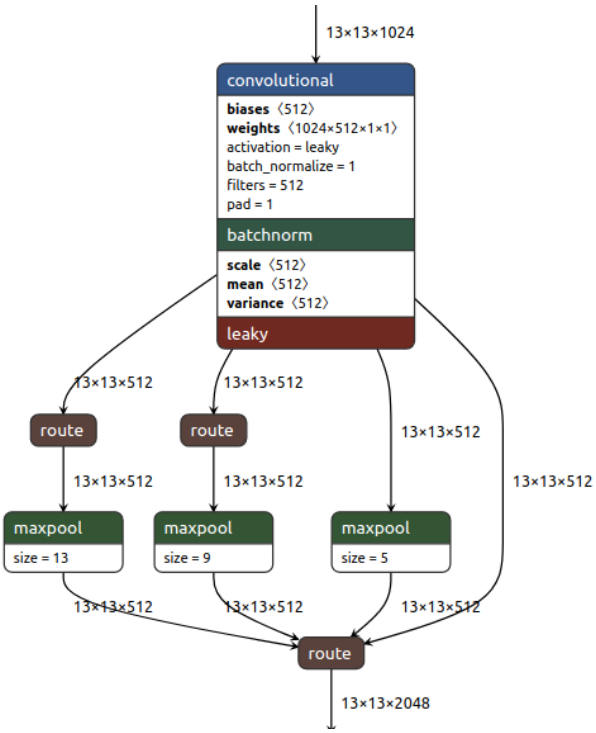


Figure 4.7: SPP Observed in 'yolov4.cfg' (Visualization App: Netron)[4]

The 'spatial pyramid pooling' (SPP) layer and the 'path aggregation network' (PANet) compose the 'neck.' They are used for feature aggregation to short out important features from the backbone and improve the receptive field. Visualizing the SPP module in Figure 4.7, one can observe that it performs 'max pooling' over the 13*13*512 feature maps with different kernel sizes k = 5, 9, 13 and same 'padding' to preserve the spatial size. The four corresponding feature maps are concatenated to form a 13*13*2048 volume, increasing the neck's receptive field and improving the model accuracy with a slight increase in inference time.

The 'Head' contains the YOLO layer. As the image input is given, the CSPDarknet53 extracts the feature and then sends it to PANet for fusion. Lastly, the YOLO layer generates the results. Figure 4.8 shows the three heads applied at different network scales for detecting different-sized objects. For each detection, localization is generated as the four corners of the bounding box. As such, there are four coordinate values for each detection instance in an image. In this study for

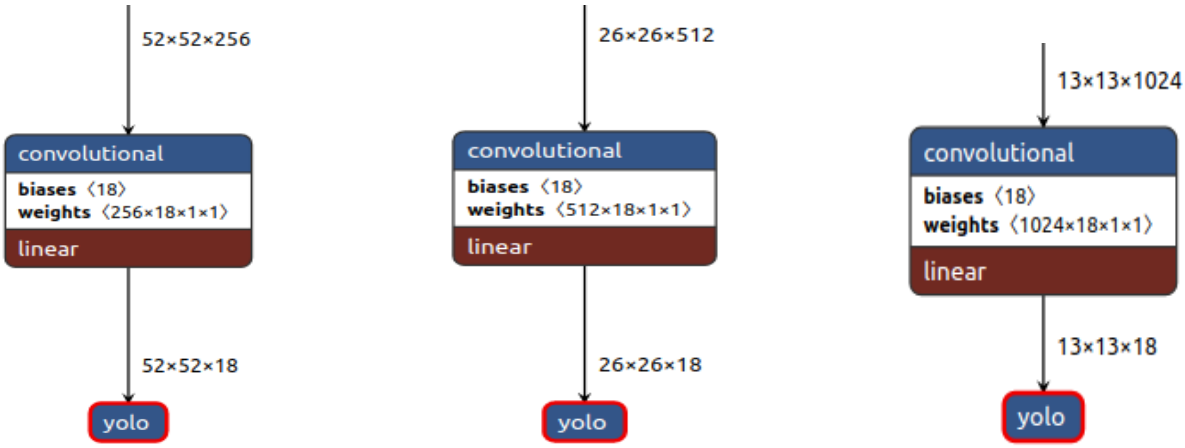


Figure 4.8: YOLO Heads Applied at Different Scales of the Network

the single-class, single-object OD model, the number of channels can be calculated as,

$$\begin{aligned} No.ofChannels &= (Classes + Objects + Coordinates) * Anchors \\ &= (1 + 1 + 4) * 3 \\ &= 18 \end{aligned} \tag{4.1}$$

YOLOv4 introduces additional model features called 'Bag of Freebies' (BOF) and 'Bag of Specials' (BOS) to improve the algorithm performance [70], [3]. BOF includes drop block regularization, Complete IOU loss (CIOU), and different augmentation techniques. Bags of specials includes mish activation function, Diou-NMS, and modified path aggregation networks [71]. Using the transfer learning technique, the weight of a pre-trained YOLOv4's layer-137 begins the training model on the custom data set.

4.3.2 Model Optimization

TensorRT is a DL platform that optimizes the trained neural network models and speeds up performance by different optimization techniques [72, 73]. DL models are computationally heavy, making it difficult to deploy at the edge as is without any optimization. In order to speed it up, several lighter versions of different models have been proposed; one of the well-known limitations of these lighter models is lower accuracy than the full-size (larger) models. Instead of a lighter version, this study proposes using the full model and quantizing it for faster inference without

much compromise in accuracy. Some of the quantization techniques used are:

- Utilization of GPU memory,
- Fusing nodes,
- Freezing the convolution layers of the model,
- Reusing memory for tensors efficiently, and
- Generating a TensorRT engine that enables faster model deployment.

Using these techniques of TensorRT, the custom model is optimized by maximizing throughput with INT8 while preserving accuracy and reducing overall memory requirement.

4.3.3 Performance Metrics

The DL pipeline uses performance metrics to judge the model's performance. This study uses objective evaluation metrics to measure the custom DL model's performance quantitatively. Among the different objective evaluation metrics, 'precision,' 'recall,' 'intersection over union' (IoU), and 'mean average precision' (mAP) are used to demonstrate detection performance. 'Precision' is the proportion of positively predicted samples to all true positive samples, whereas 'recall' is the proportion of samples that are positively predicted among all true positives (Equations 4.2 & 4.3).

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{P} \tag{4.3}$$

IOU is the ratio of the intersection and union between the bounding box predicted by the model and the real bounding box, also known as the 'Jaccard index.' As shown in equation (4.4), mAP is calculated, where Average precision (AP) is the average precision. The mean AP over all classes and overall IoU thresholds is the measure of mAP. Like most popular literature, the IoU threshold is set to 50 for the single class.

AP is measured by the area under the precision-recall curve. It is averaged over all categories and calculated for all the objects in the image. The 11-point method is used, which is done by segmenting the recalls evenly to 11-points, e.g., 0, 0.1, 0.2,...0.9, 1. The definition of AP is shown in equation (4.5).

$$mAP = \frac{1}{|Q_R|} \sum_{Q_R}^{q=1} AP(q) \tag{4.4}$$

$$AP = \int_0^1 p \, dr \tag{4.5}$$

One of the major challenges of using a DL model at the edge is that it is computationally expensive because of its deep layers. It is one of the complications this work looks to address. Therefore, in addition to the mAP, the model size and computational complexity (FLOPs) are also considered for model algorithm evaluation. The model's size is closely related to its parameters, which can be related and therefore used to measure the performance of the YOLOv4 model. FLOPs reflect necessary algorithm calculation; their unit is BFLOPS (Billion Floating-Point Operations per Second) or GMacs (Giga Multiply–Accumulation Operations per Second). It represents the floating-point operations per second, which can reflect the algorithm's calculation performance. Furthermore, the final detection FPS (Frames Per Second) improvement from the original trained model (custom YOLOv4) to that of the optimized model (custom TensorRT) is another parameter that is considered a performance metric.

4.4 Technical Approach

The objective is to develop a DL algorithm for the intelligent inspection system to determine if there is a 'surface crack' in the inspected area, and subsequently, it is marked (localized). The objective is to develop a low-power, low-budget & compact IP using an SBC that can be used effectively across different inspection platforms, including a UAV and a handheld setup. The IP consists of the target edge device, NVIDIA's Jetson Nano 2, but this model can also be deployed from any computer given the required software, sensors & firmware match. Final models are deployed and tested from both laptops & SBC mounted on the platforms mentioned. The true

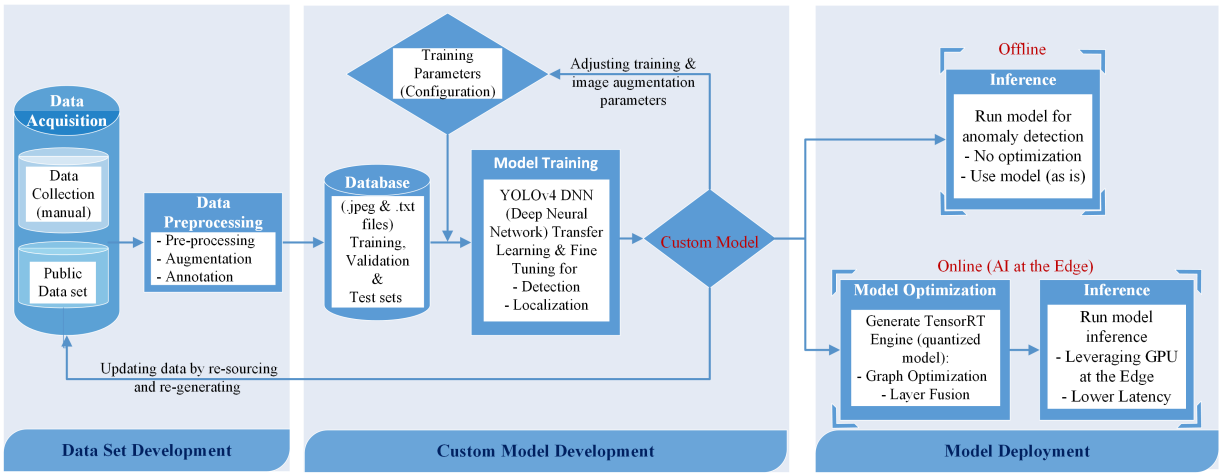


Figure 4.9: Deep Learning-based Crack Detection Pipeline

color camera, Raspberry Pi 2.0, is used for image acquisition and further analysis. The IP-related specifications and other relevant information can be found in chapter 2. As mentioned in chapter 3, the inspection parameters are set by studying the camera FOV, and a 70% adjacent frame overlap is considered to ensure a full inspection coverage [74, 75]. The images can also be used for system or critical parts of infrastructure monitoring as necessary for an SHM system.

The intelligent crack detection methodology comprises three major steps: (i) data set development, (ii) custom model development, and (iii) model deployment (Figure 4.9). The custom OD model for crack detection is developed using Python & DL API TensorFlow 2.2.3. YOLOv4 base model with 'CSPDarknet53' backbone is used as the base model to develop the custom OD algorithm. Finally, NVIDIA's TensorRT is used to optimize the model for the embedded platform at the edge. The process is explained as follows:

4.4.1 Data Set Development

4.4.1.1. Data Acquisition

It is essential to obtain a good quantity and quality dataset for a good-performing DL model. The focus is to develop a model to inspect common defects across different infrastructures, and the anomaly to detect here is a single class 'surface crack.' Hundreds of color images of surface cracks

are manually collected with handheld cameras or mobile devices at different locations. However, more data is needed. Aside from manual data acquisition, other popular publicly available data sets, e.g., 'CFD' and 'Mendeley' have been added [76], [77]. The JPG image file format is used, but other popular extensions readable by Python will also work. Table 4.2 shows the data set summary, including data split, number of bounding boxes, and average area at a glance.

4.4.1.2. Data Pre-processing

Augmentation. The images are sourced from different sources, so it needs preliminary analysis to check for the balance of different types of crack to remove any bias from the data. After analyzing the acquired data set, the next step is to augment it to provide more variety.

As mentioned in section 4.4.1, augmentations are done to make the model more robust and improve accuracy in general to avoid over-fitting. In the OD domain, augmentation may have an unpredictable effect on the model; as such, it should be applied carefully. It is also dependent on the attributes of the object to detect. First, image orientation does not matter much for corrosion, so the 'geometric transformation' augmentations can be applied without negative ramifications. As such, in the first batch, images are re-sized. However, since the feature and color are important attributes, following 'random color adjustments' are applied with caution for the model to learn to distinguish corrosion from misinterpretation. In the rest of the batch training, experimentally augmentation techniques are added and analyzed as part of manual pre-processing. The techniques used are as follows:

- Geometric transformation, such as random horizontal flip, vertical flip, and angular rotation.
- Random color adjustments, including brightness, hue, saturation, and contrast.

Annotation. OD model needs annotated datasets. For this purpose, 1110 images of cracks under different lighting conditions were chosen as a training dataset and manually annotated using the open-source software 'LabelImg.' Annotation is carried out manually and also checked if it appropriately covers all the crack instances in the images. The annotation should cover the crack instance with a tight box. The overlapping annotation must be avoided for multiple instances as it reduces or affects the model precision. At the same time, no instances should be missed,

Table 4.2: Database Summary

Data Split	No of Images	Bounding Boxes	Average Area (Pixels) of BB
Training	835	919	23141.44
Validation	166	186	23103.61
Test	109	117	26177.28

which leads to incorrect training. Manually annotating thousands of images is time-consuming; nevertheless, the annotation must be done as best as possible. During training, if certain batches give problematic evaluation scores, the batch is re-analyzed, and annotation is corrected if any issues are found. This step is revisited to make amends as needed. Figure 4.10 shows an annotated image for training.

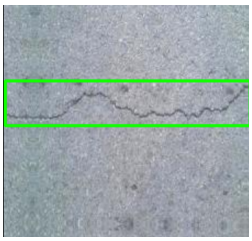


Figure 4.10: Example of an Annotated Image for 'Crack' Detection

Annotation is a crucial step, as the training and detection are highly dependent on the precision of the labeling of the images. After basic pre-processing, the images are split into training, validation, and test sets [table 4.2].

4.4.2 Custom Model Development

Training

The training was conducted over the cloud, as mentioned in the tools section. Table 4.3 shows the YOLOv4 custom model training summary. YOLOv4 took around 6.5 hours for 6000 iterations for the input size of 416. The 'mish' activation function used for training is a low-cost function. It has useful properties like unbounded above and bounded below properties, improving its performance

compared with other popular functions like ReLU (Rectified Linear Unit) and Swish functions. Mish, a smooth, non-monotonic activation function that is defined as,

$$f(x) = x.tanh(ln(1 + e^x) \tag{4.6}$$

where, $ln(1 + e^x)$ is a 'softmax' activation function [78]. Among other BOF tools used for the YOLOv4 model are the 'mosaic' augmentation technique and the use of saturation, exposure, and hue. The learning rate was initially chosen to be 0.001 with 0.0005 decay and 0.949 momentum.

Table 4.3: Training Outline: Custom YOLOv4 Model

Training Hyper-parameter	**Value**
Iterations	6000
Batch size	64
Momentum	0.949
Decay	0.0005
Saturation	1.5
Exposure	1.5
Hue	0.1
Learning rate	0.001
Bag of Freebies (BOF)	mosaic
Activation Function	mish
Training time	6.5 Hours

4.4.3 Model Deployment

Offline Model Deployment

The model is deployable right after training. It can be deployed over the cloud platform, where it is trained to check how the model behaves. It can also be loaded on other machines with the required software and DL APIs. This direct use, without any amendment to the model, is called

offline deployment. The model was also tested on a laptop (section 2.2.4). The model can be deployed and used from these machines with high enough computational power to run inference. It was tested on the test data set. Results are presented and discussed in section 4.5.

Online Model Deployment

The best-trained model is checked in offline mode. The best model is selected for optimization to adapt it to the embedded platform Jetson Nano. Using TensorRT, model is quantized as discussed in section 4.3.2. After quantization, a TensorRT engine is generated, which is the lighter version of the model, deployable at the edge. It is tested using the IP RGB sensor, Raspberry Pi, for real-time inference. The results are also presented and discussed in the following section 4.5.

4.5 Result & Discussion

4.5.1 Model Evaluation

For validation, the generated algorithm is checked on the test data set (Table 4.2) (unknown to model) and also on different image qualities. Some probable cases are- i) hairline crack, ii) images with no surface cracks or defects, iii) low light images, and iv) distorted images of surface cracks. In all these cases, the algorithm holds well, gives an acceptable result, and successfully flags the anomalies (cracks) that need attention from an inspector.

The overall precision and recall is 97% and 96% respectively with a F1 score of 0.97 (Table 4.4). The 11-point AP measurement is used to calculate the mAP at IoU 50. In Figure 4.11, the blue line shows the training loss or the error on the training data set (specifically 'Complete Intersection-Over-Union' or CIoU loss for YOLOv4). The red line is the mAP at 50% IoU threshold (mAP@0.5), which checks the model's performance on a never-before-seen validation data set. The mAP dips between 1200 & 2400 iterations are likely caused due to the particular mini-batch of the dataset compared to other mini-batches. The overall best mAP@50 attained is 98.44%, with an average loss of 0.7897%, as the model converged over the 6000 iterations.

The validation inference is analyzed against PascalVOC 2007 and MS COCO data sets (Figure 4.12) at 50% and 75% IoU thresholds. The mAP for the custom data set at the 50% threshold

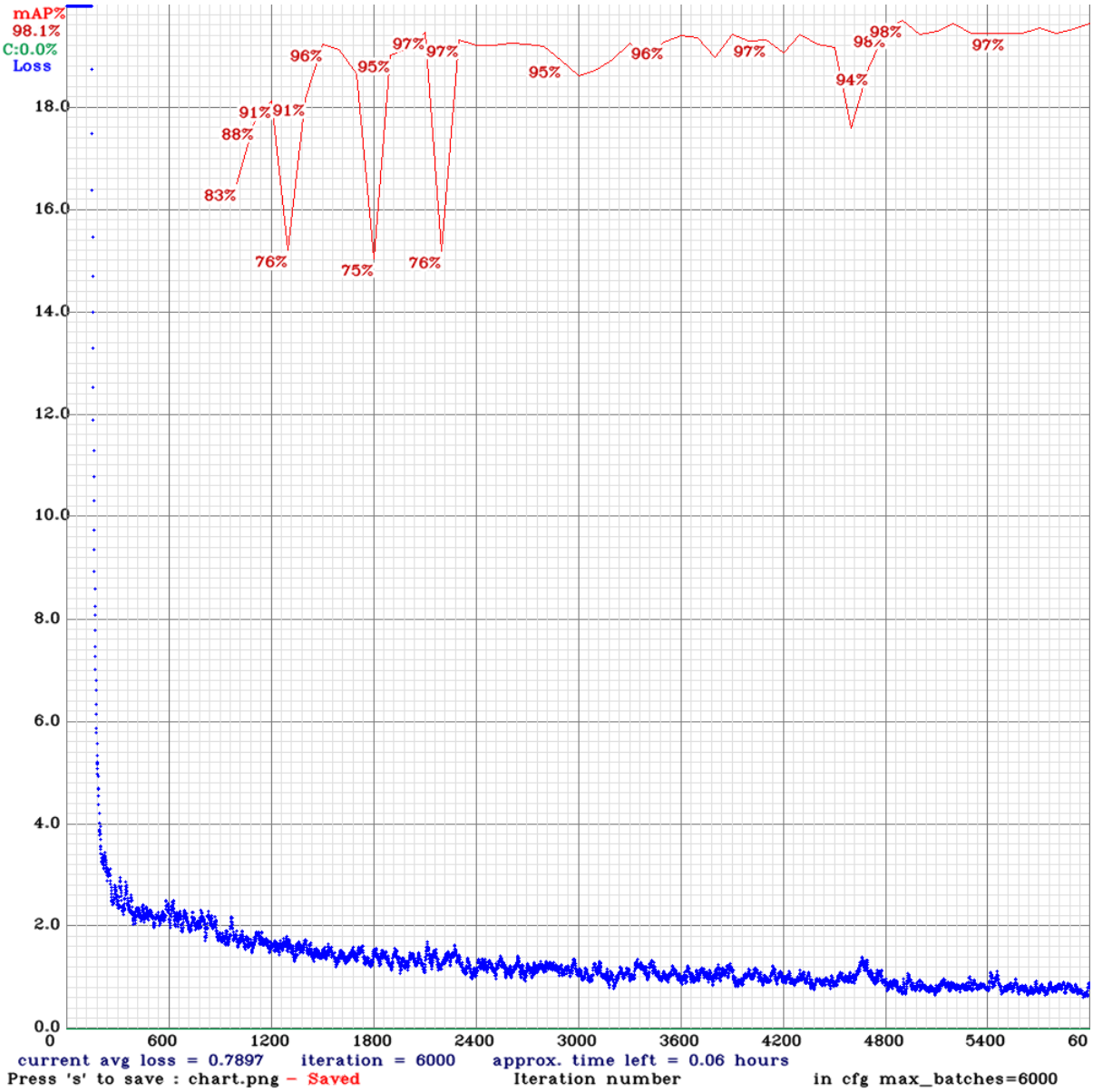


Figure 4.11: Training Chart Showing 'mAP' and Average Loss Over 6000 Iterations

is closely followed by the MS COCO dataset, with a drop in mAP for Pascal VOC. For IoU threshold 75%, all three data sets are approximately the same, with our custom data set scoring the highest mAP@75 of 0.764114 or 76.41%.

Images in Figure 4.13 were randomly selected from the database for testing, and these intuitively demonstrate the detection performance of the improved model.

Table 4.4: Custom YOLOv4 Model Scores

Evaluation Parameter	**Value**
Mean Average Precision, mAP@0.50	98.44%
True Positives (TP)	180
False Positives (FP)	5
False Negatives (FN)	7
precision	0.97
recall	0.96
F1-score	0.97
Average loss	0.7897
Best weight size	244.2 MB
BFLOPS (Billions Floating Points per Second)	59.563

Table 4.5: Training Outline: Custom YOLOv4 Model

Model	**Avg Training /Conversion Time**	**Frame Per Second (FPS)**	**Mean Average Precision (mAP)**	**Average Loss**
Custom YOLOv4 (Darknet)	9 hrs	1	98.44	0.7897
Quantized Custom YOLOv4 (TensorRT)	1 hr	2.5	-	-
YOLOv4-tiny	5 hrs	1.2	90.98	0.512

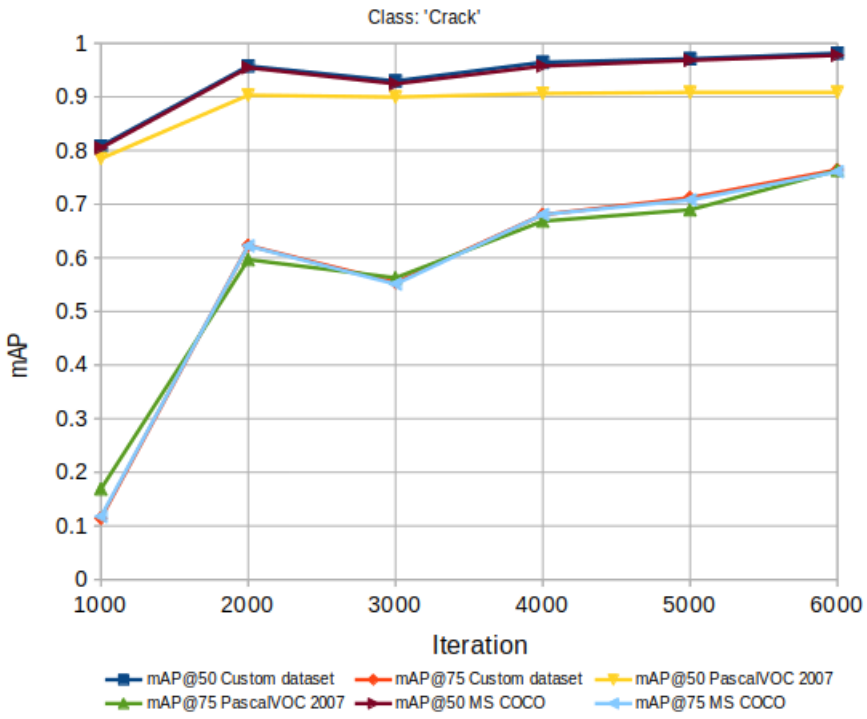


Figure 4.12: Iteration vs. 'mAP' Score of Custom YOLOv4 Model

4.5.2 Experimental Result at the Edge

For further validation of the processing capability of the improved model in mobile devices, the trained model is deployed to the Jetson Nano, an embedded platform. The processor is small in size, low in power consumption, and strong in computing performance. The performances of the custom YOLOv4 model, lighter model YOLOv4-tiny, and the quantized custom YOLOv4 model are compared in terms of the objective evaluation indicators mAP and FPS, respectively, as shown in Table 4.5. Between the trained full-size custom YOLOv4 and the lighter YOLOv4-tiny models, the former has 8% higher accuracy, whereas the latter has slightly faster inference speed (FPS 1 & 1.2 respectively). The full-size YOLOv4 model is slower due to its complex structure, which cannot meet the needs of mobile devices for real-time crack detection. The full model weight is frozen and optimized to a lighter version using TensorRT, which increases the FPS from 1 to 2.5. The optimized model is tested with different setups and images to test its performance. The quantized YOLOv4 model retains a higher accuracy and achieves a faster processing speed, which

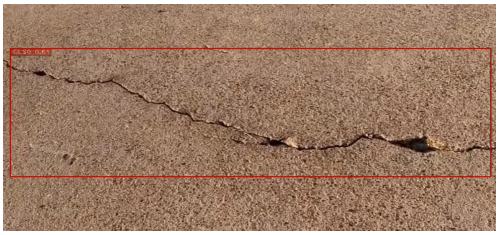
(a) Input Image

(b) Detection Against Patterned Surfaces

(c) Input Image

(d) Detection on Uneven Concrete Surface

(e) Input Image Taken in Late Afternoon Sunlight



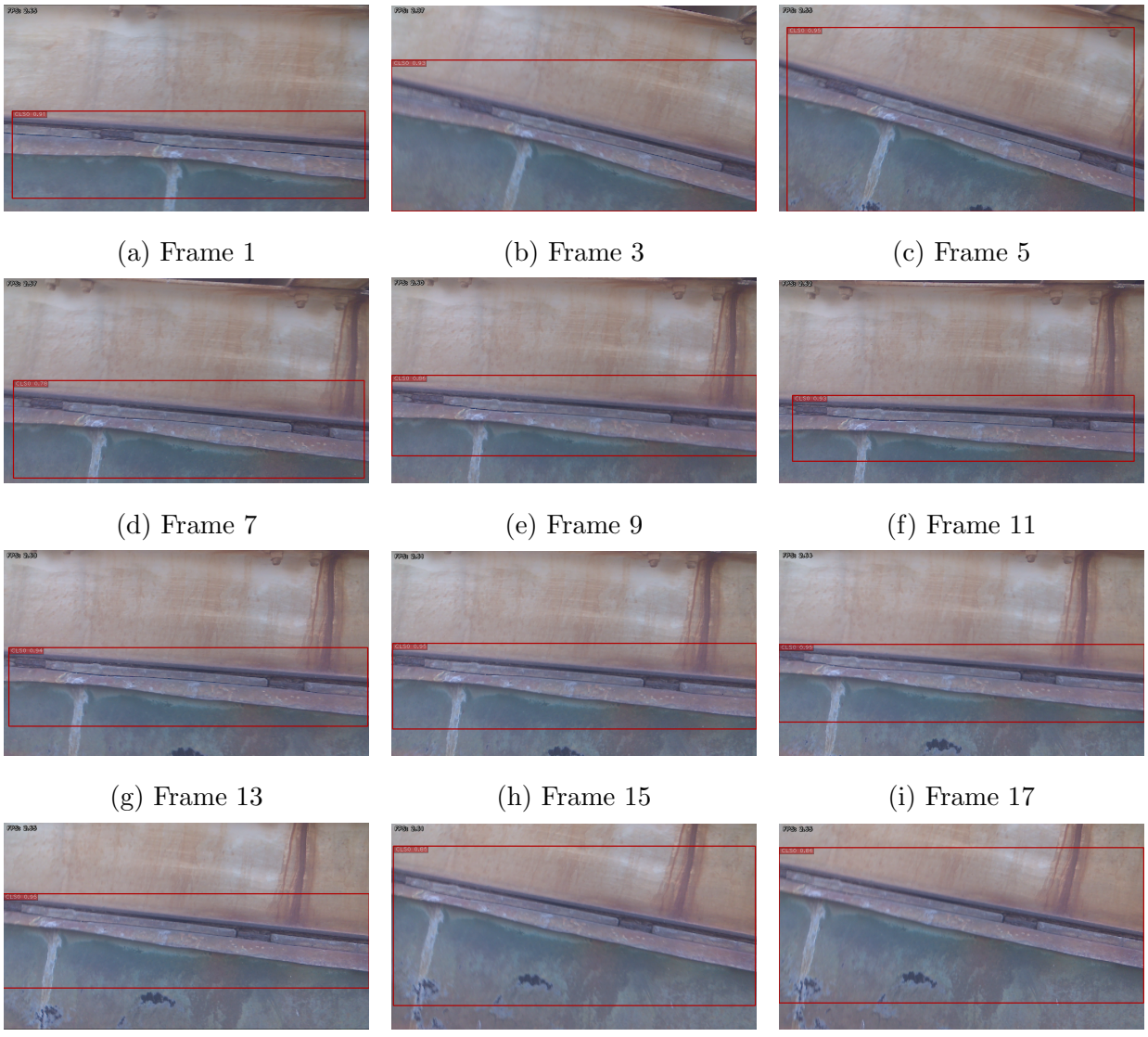
(f) Detection on Image (e)

(g) Input Image in Low Light Condition

(h) Detection on Image (g)

Figure 4.13: Detection Results of Surface Crack Under Different Lighting Conditions on Videos

meets the requirements of a real-time object detection system (Figure 4.14).



(a) Frame 1 (b) Frame 3 (c) Frame 5

(d) Frame 7 (e) Frame 9 (f) Frame 11

(g) Frame 13 (h) Frame 15 (i) Frame 17

(j) Frame 19 (k) Frame 21 (l) Frame 23

Figure 4.14: Surface Crack Detection at the Edge Using Hand-held Setup with FPS Shown at the Top Left Corner of Each Frame

4.6 Conclusion

Remote monitoring and inspection methods have been used since the 1960s, especially in industry scenes such as pipeline inspection in the oil and gas industries. The challenge of automating the interpretation of complex inspection measures to replace the trained inspectors is highly debatable due to reliability concerns. This method uses custom DNN for inspection on images taken from an edge device, identifies areas of interest, and flags it for further inspection by one such experienced inspector without having to reach inspection areas physically. Therefore, it is an addition to the visual inspection methods with a higher reach. Essentially, this method reduces the inspection and analysis period and operational downtime, saving costs without compromising reliability.

Chapter 5

Intelligent Corrosion Detection Using Deep Learning

5.1 Introduction

'Corrosion' is a troubling infrastructural defect that is harder to detect and propagates fast due to the environmental catalyst. Metal structural components are usually under some surface (e.g., concrete) or coating (e.g., painting), causing visual occlusion, which makes it difficult to uncover. As different statistics presented in earlier chapters, corrosion can cause serious damage if not detected in time. It has also been an expensive problem worldwide in the infrastructure inspection scene, incurring billions of dollars in loss. The conventional approaches cause downtime and, occasionally, accuracy issues due to human error. As part of Task 3 of the proposed methodology, this chapter proposes, develops, and applies a DL-based algorithm for 'corrosion' detection as part of an inspection payload (IP) to resolve manual inspection-related issues. The onboard camera of the IP takes images of the infrastructure, which is used as inputs to the intelligent inspection of corrosion in offline (remote) and online (real-time) modes at the edge. The intelligent and real-time capability accelerates the inspection process and provides consistent analysis.

Challenges of Corrosion detection. Unlike 'crack,' corrosion presents different challenges, making it a harder class to deal with for the DL technique. This work proposes and develops an intelligent corrosion detection method leveraging the DL algorithm. The DNNs in the DL models look for a consistent pattern or shape and perform identification based on the consistent category. For example, a DL model detecting pedestrians in autonomous driving car navigation will have a human body with a head, body, arms, and legs. This consistent pattern is necessary so the

model can learn and perform well in detection. However, corrosion is diverse, and a spectrum of different types of corrosion throws the model off, resulting in poor performance.

Although several existing works have shown accurate corrosion detection with CNNs, they mostly have high computational requirements, and most of the work is geared toward civil engineering inspection. Hence, two major challenges become apparent; the first challenge is that there is not enough relevant data, specifically for the infrastructural corrosion scope and the other is the high computational cost of adopting these methods onto a low-cost and low-power embedded platform at the user end. This proposed DL-based corrosion detector shows improved performance by developing a data set incorporating synthetic data and using compression techniques to quantize the model to deploy it on the edge platforms.

5.2 Related Work

5.2.1 Low-level Feature-based Approach

CV-based studies have been proposed for fault defection in civil engineering for structural lifecycle service and maintenance. Especially corrosion-prone structures that may directly cause loss of life are the major area of concern. Visual corrosion detection studies using traditional CV mainly focused on identifying the key corrosion characteristics, e.g., color information and feature extraction from digital images. In earlier work, Wavelet analysis was used for feature extraction in order to assess damage or fault in structures, e.g., in aging aircraft structures [79], surface damage [80], and ship hulls [81]. In a project by the European project MINOAS (Marine INspection rObotic Assistant System), Bonnin and Ortiz used Hue-Saturation-Value (HSV) values of affected areas in the classifier algorithm for corrosion detection [82]. They used a combination of 2 classifiers, a weak classifier color-based corrosion detector (WCCD) and an AdaBoost-based corrosion detector (ABCD), to achieve better performance. Shape and size-based pitting corrosion detection have been proposed in work by Pereira et al. [83]. Hoang and Tran applied texture analysis to extract 78 features using the different color information and used a support vector machine (SVM) for further classification and detecting corrosion [84]. All these traditional approaches require extensive knowledge of corrosion and its optimal features.

The performance or accuracy of corrosion detection depends heavily on how well the features are defined in these preliminary and manual pre-processing steps [85, 86]. Determining optimal corrosive features is still challenging [87].

5.2.2 DL-based Approach

The breakthroughs in DL refocused researchers' interest in DL-based approaches for defect detection, including corrosion detection. Several research works have found that CV, coupled with DL, can identify corrosion in infrastructures, resulting in standardized results, speeding up the process, and omitting human error. There has been a significant rise in research investigating approaches based on the two cornerstones, human error reduction and speeding up the process. The DL-based approach uses convolutional neural networks (CNNs) that conduct automatic feature extraction, giving it an edge over the former method. The DL-based approach not only learns the important features automatically but out-performs state-of-the-art vision-based approaches, attaining a higher level of accuracy [88, 89, 90, 91, 92]. The DL-based approach has the capacity to be used as a solution to broader subject matters in just a single step. DL approaches for corrosion detection in various industrial settings have also proved successful. Atha et al. used CNN architectures with a sliding window of different sizes (i.e., 32×32, 64×64, and 128×128) to detect corroded areas within an image. They use two shallow CNN networks similar to ZF Net and VGG-net [88]. After determining CbCr to be the most robust color space for corrosion detection using wavelet decomposition, they use it in conjunction with the CNN algorithm over different sliding windows in an image. Mentionable that they used cropped-out images with clearly visible surfaces in their dataset, whereas in this work, images of complicated and hard-to-reach industrial spaces taken with a unmanned aerial vehicle (UAV) are used. Forkan et al. have developed a novel ensemble of deep learning CNNs base framework called CorrDetector for corrosion detection [93]. They evaluate several models and display one performing best among all in terms of segment-level and image-level predictions. Among the OD-based proposals, Cha et al. used 'Faster R-CNN' to develop a multi-damage detection algorithm, including steel and bolt corrosion. They used 6000 x 4000 high-resolution images as their data set [89]. Perhaps the closest and most related recent work is done by Yu et al., where they trained a modified version

of the lighter OD model YOLOv3-tiny to deploy on board an embedded platform of Micro Aerial Vehicles (MAVs) [94]. Unlike this approach of model quantization, they use a modified backbone, and their target corrosion types are bar, nubby, and fastener corrosion in oil wells. A thorough search of the relevant literature yielded that this is the first DL-based quantized model trained on real and synthetic data to test corrosion detection, deployable on board an SBC for real-time infrastructure inspection.

5.2.3 Synthetic Data for Neural Network Training

Collecting raw data, annotation, and further verification analysis is expensive and time-consuming. The principal success of DL models can be credited to the availability of large data sets to learn the transformation functions. Many studies endeavor to provide the necessary diversity of information for the model to learn a specific task. In the absence of required real-life data on the scope of corrosion, the model is bound to exhibit sub-standard performance. The pandemic made it even harder to collect data in such quantity and quality. Much research is focusing on overcoming the data scarcity challenge. Synthetic data has presented itself as a promising alternative that can generate data with acceptable quality and variance to develop a well-balanced data set for neural networks. Most of the work on using synthetic data for object detection models focuses on the autonomous driving domain to identify pedestrians and other obstacles [95, 96]. Nowruzi et al. report the advantages of mixing real and synthetic data in the same domain for training and fine-tuning that delivers better performance [97]. In recent literature, the useful and promising results obtained using synthetic and real data encouraged its use in this work. This work uses a mix of real & synthetic data to experiment with model training and post-process analysis. The synthetic data used in this work was obtained using the process described in [98]; As such, the synthetic generation process is beyond the scope of this work.

Scope. Most CNN and Deep CNN or Deep Neural Network (DNN) based DL approaches use an ensemble technique for identification and localization. They also differ in the type of corrosion to detect and their target inspection area. Model implementation time is a critical element for an intelligent inspection system. Most of the work uses computationally heavier models for edge deployment, or the detection algorithm for corrosion is specific to the component to inspect.

Different studies differ in the type of corrosion they aim to detect as well as their target inspection area, which most likely is civil structure-based. In this study, the training is done with a mix of real and synthetic data, using transfer learning and quantizing the best model to a lighter version to deploy from the SBC platform. The Scope of the defect is also scoped down to the three most common types of corrosion across infrastructure, i.e., crevice, pitting, and concrete corrosion.

5.3 Model Selection

DL provides a variety of models as an option. The two conditions to consider are that the model should adequately identify and localize corrosion and be deployable in real time. The requirement is drawn from the rising demand to reduce system downtime by automating and deploying the model at the edge for inspection. All three DL techniques- (i) image classification (IC), (ii) object detection (OD), and (iii) semantic segmentation (SS) have been used for 'corrosion' detection. It can be used and accepted as a standard practice as the results are based on the same data and are easily standardized throughout the similar infrastructure of different facilities. This work aims to identify and localize corrosion in images or videos gathered for inspection using DL- algorithm. The 'YOLOv4' OD technique is selected as discussed in sections 4.3. It identifies the corrosion instances in the image/camera frame and localizes them by drawing a box around it, which is the desired outcome. Among popular OD methods, only a handful of research has been done on corrosion detection compared to the other DL categories; however, none adapts the code on an embedded platform.

5.4 Technical Approach

The main focus of this work is to develop an intelligent inspection payload mountable and usable onboard embedded systems on different platforms, e.g., a handheld setup or with a UAV platform as mentioned in section 2.3 cost-effectively. The final model deployment experimentation is done on the inspection payload hardware, i.e., using Jetson Nano and a Raspberry Pi camera (section 2.2.2). Like the crack detection in chapter 4, the camera is considered for image acquisition and further analysis as an inspection of the plant. The inspection parameters are set similarly,

depending on the capability of the camera FOV (Field of View), considering a minimum of 70% overlap of adjacent frames for better image quality and ensuring a full inspection coverage [99]. The idea is to obtain RGB images to cover the area of interest of the infrastructure layer by layer, and each layer consists of grids of RGB images with overlap. The overlap ensures that all area is covered, and later, it is convertible to videos or other post-processing necessary for the structural health monitoring system.

The technical approach for the corrosion detection method can be summarized into three major steps: (i) data set development, ii) custom model development, and (iii) model deployment, as seen in Figure 5.1. Under data set development, the pre-processing consists of data set generation, image augmentation, and finally, detecting and localizing corrosion instance(s) in the FOV of the camera. The custom OD model to detect corrosion is developed using Python, DL API TensorFlow, the YOLOv4 category of the YOLO OD family, and TensorRT for model optimization and deployment at the edge 2.2.3. The process is explained as follows:

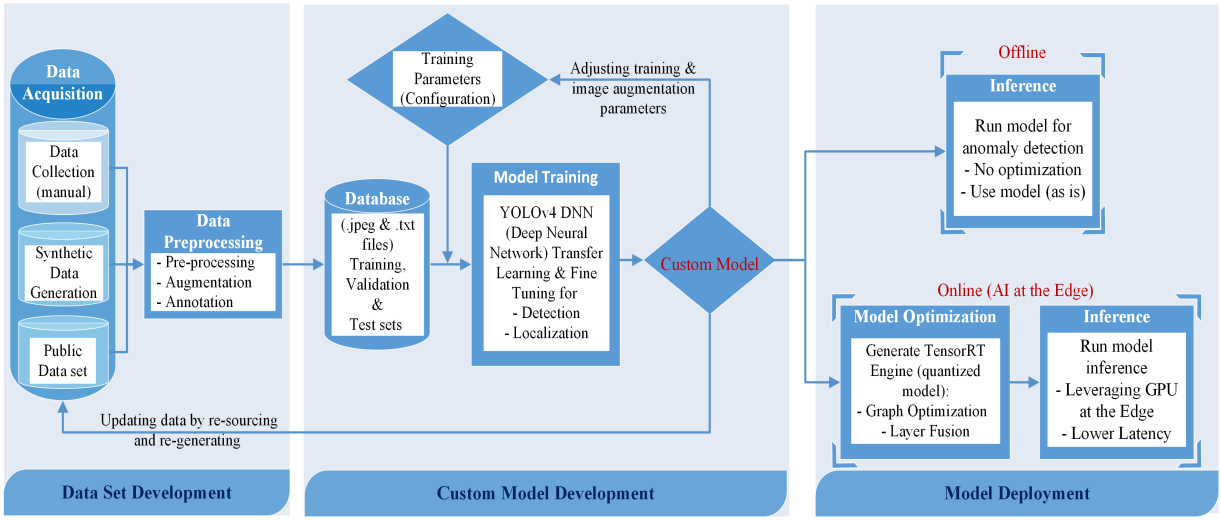


Figure 5.1: Deep Learning-based 'Corrosion' Detection Pipeline

5.4.1 Data Set Development

The DL model must be trained on an appropriate number and good quality data set to reflect similar learning representation. The key to developing a good quality data set is understanding

the problem, i.e., corrosion. Corrosion usually starts from coating loss, which differs in shape and size. Identifying corrosion is challenging since it has no consistent feature, e.g., edge, shape, or size, unlike other classes with specific features; for example, the 'human' class has consistent features, e.g., head, two hands, and legs.

On the other hand, corrosion has many different types that are different, depending on the structure, location, or point of interest. The diverse type of corrosion also makes things difficult for DL to learn each of these well. There should be an enormous amount of data combined as each type needs a substantial amount. Therefore, it is essential to scope it down to focus on specific types of corrosion. In this work, corrosion is narrowed down to three major infrastructure-related domains: i) Pitting corrosion, ii) Crevice corrosion, and iii) Concrete corrosion 5.2. The data set for this specific scope reduces the heterogeneity of corrosion and, at the same time, gives enough variety to each type to get enhanced performance.

(a) Pitting Corrosion (b) Crevice Corrosion (c) Concrete Corrosion

Figure 5.2: Defect Domain: Types of Corrosion.

One approach commonly used in industries is to detect corrosion on a particular component, such as a concrete slab, metal structure, or valve, which has a definitive shape and size; however, it will limit the algorithm to the specific component, which is not desirable. Another popular approach of using corrosion color for identification can also be misleading, resulting in misidentifying metallic or a similar colored surface as corrosion. Here, this complication is sorted out by getting the DNNs to focus on the texture and consider the image as a whole by training it on similar images. The advantage of this approach is that the model learns not to assume all background regions or the same colored surfaces as corrosion. However, in real-life applications,

this approach holds some risk of misinterpreting other faults, in this case, corrosion. Therefore, the larger the data set with all possible defect representations of the focused type of corrosion, the more robust will be the trained model, and the better and more accurate the outcome.

Data Acquisition

Hundreds of color images with handheld cameras or mobile devices were collected from different infrastructures at different locations. However, more is needed for DNNs. Therefore, besides the manual collection, one of the available data sets for 'corrosion,' 'COCO-Bridge-2021' with beams and under-bridge corroded surface images was used [100]. Like many, this publicly available data set is of civil engineering related structures such as bridges and roads. However, even after selecting the specific scope of corrosion, more data is needed, as demonstrated in the training section. So additional synthetically generated data were used to generate a mixed data set. The different data batches were used for training and testing in several trials and were updated as necessary.

Data pre-processing

Augmentation Once enough good-quality image data are obtained, the data is manually augmented as a pre-processing step to make it more robust. The images are sourced from different sources, so it is analyzed to check for balance across the corrosion data types. The file format used here is JPG. However, Python-readable other image formats can also be used.

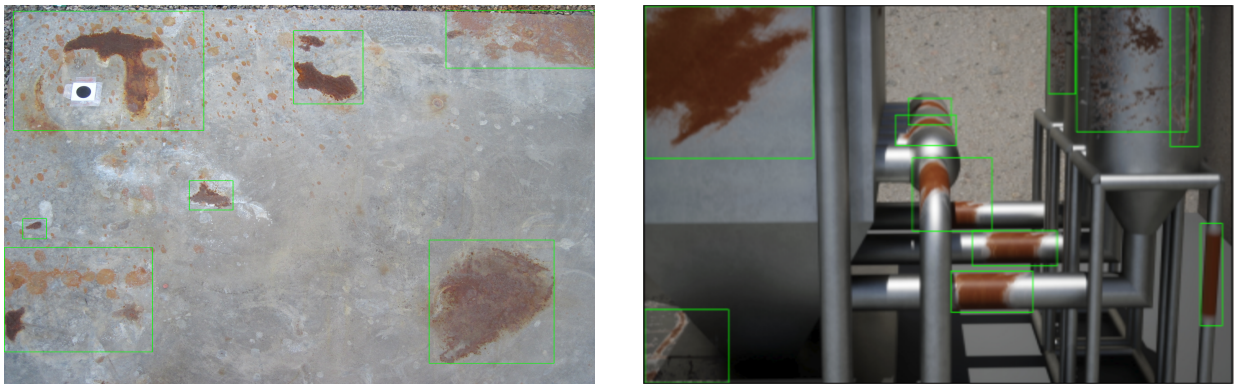
In an image classification problem, geometrical transformation is encouraged to randomly vary spatial characteristics on an image, e.g., rotation, cropping, random flip, etc. These augmentations are done to make the model more robust and improve accuracy in general to avoid over-fitting. It is more sensitive in the object detection domain, so image pre-processing techniques are cautiously applied depending on the object's characteristics to detect. Considering the nature or characteristics of corrosion, the orientation does not matter much. In the first batch, images are re-sized. In the rest of the batch training, image augmentation techniques are experimentally added and analyzed as part of manual pre-processing as follows:

- Geometric transformation, such as random horizontal flip and vertical flip, angular rotation,

etc.

- Random color adjustments, including brightness, saturation, and contrast.

Annotation OD model needs annotated data specific to the type of model used. Annotation refers to manually drawing bounding boxes around desired objects in the image. For this purpose, 868 images of corrosion under different lighting conditions were chosen and manually annotated using open-source software 'labelImg.' Synthetic data had its annotation done using Unreal Engine. The software-generated annotations are reviewed using a Python script; The decimal coordinates are rounded off, and the class id is rechecked to match the real image annotations.



(a) Annotation of Corrosion on a Real Image

(b) Annotation of Corrosion on a Synthetically Generated Image

Figure 5.3: Example of Annotated Images for 'Corrosion' Detection.

Figure 5.3 shows annotated images on real and synthetic data for training. After each training, the model evaluation scores are studied, and adjustments are made in the data set if needed to improve the model evaluation, especially mAP.

DL models are data-hungry, so the first approach is always to enrich the available data set. After the pilot run of batch 1, additional data were added for batch 2. The specific corrosion data were not easily available, and due to the global pandemic, it took much work to gain access to industrial compounds to collect more images. Open source data were added in batch 3 from the [100]. However, it is targeted toward under-bridge inspection for civil infrastructure, so some relevant images aligned with the target were sourced. In addition, in batches 3 and 4, 86 and

277 synthetically generated data were added respectively (Figure 5.4) based on the three types of corrosion.

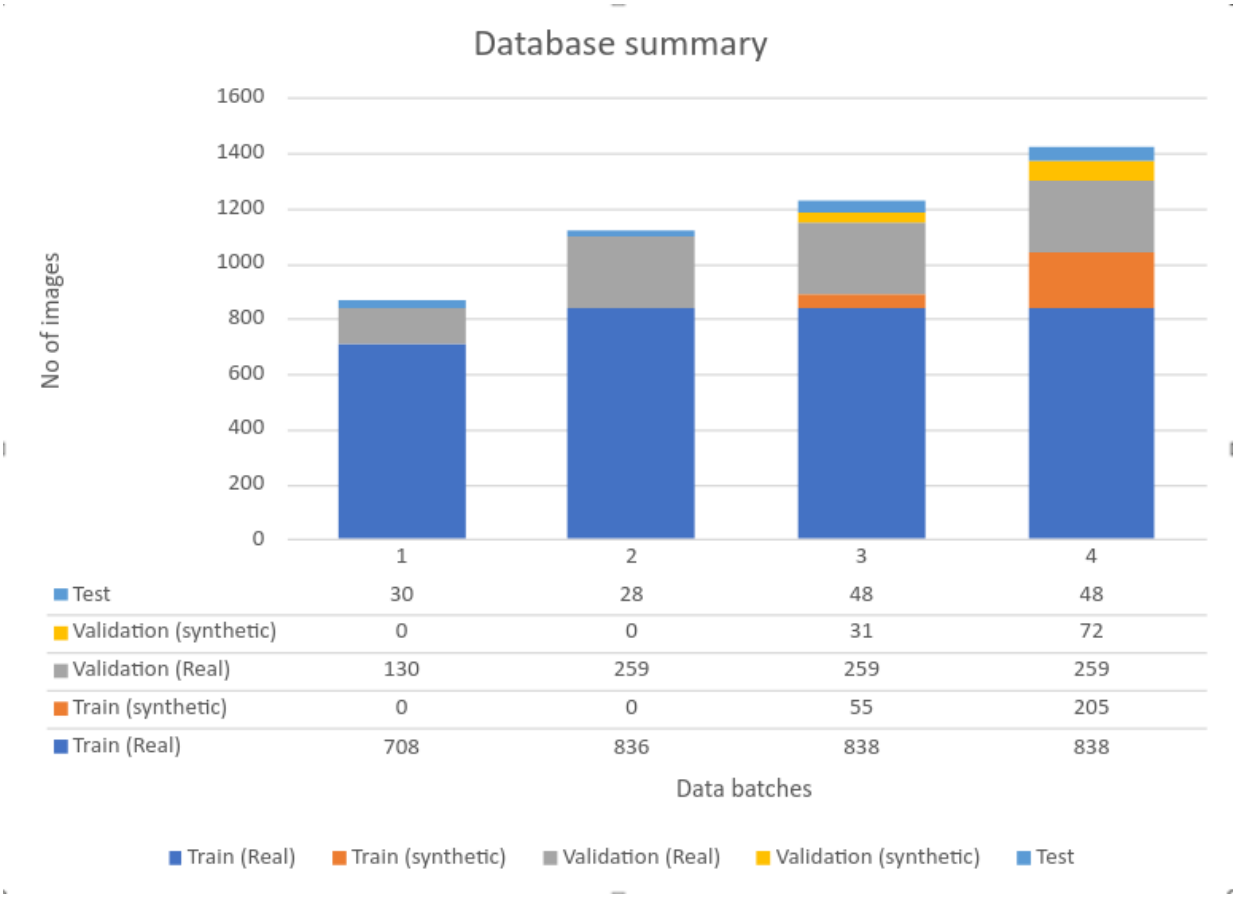


Figure 5.4: Database Summary for Real & Synthetic Data, Developed in Batches for 'Corrosion' Detection

Table 5.1: Data Set Summary

Batch	**Total No. of Images**	**Training Set**	**Validation Set**	**Test Set**	**Bounding Boxes**
1	868	708	130	30	1406
2	1123	836	259	28	1714
3	1231	893	290	48	1909
4	1422	1043	331	48	2022

Annotation is a crucial step, as the training and detection are highly dependent on the precision of the labeling of the images. After basic pre-processing (geometric transformation & random color adjustments), the images were split for training, validation, and test sets. The data summary table in Figure 5.4 shows the development of batches and data split, and Table 5.1 shows the number of bounding boxes by the batch at a glance for all real and synthetic images.

5.4.2 Custom Model Development

Training

The YOLOv4 model was used to train an OD DL model for corrosion detection. The model was trained over the cloud (section 2.2.4). Table 5.2 shows training configuration and combinations of different training parameters that have been tried and tested for custom model development. The model took around 7 hours for 6000 iterations for the input size of 416. Some of the BOS & BOF tools of YOLOv4 were used, e.g., the 'mosaic' augmentation technique and the use of saturation, exposure, hue, and 'mish' activation function. The hyper-parameters initially selected are a learning rate of 0.001 with 0.0005 decay and 0.949 momentum.

For the transfer learning, pre-trained YOLOv4's layer-137 weight is used to begin training the model on the custom data set. After the first trial, the best weights from previous custom-trained models were used for transfer learning to experiment with the model outcome in trials 2 & 3. Transfer learning uses pre-trained frozen layers already trained on feature extraction from the pre-trained weights to reduce model training time and improve accuracy drastically. The learning rate, momentum, and decay are important hyper-parameters selected and altered for experimentation. Data augmentation techniques from BoF were used to avoid network over-fitting issues. Different hyper-parameter combinations are used with BOF & BOS tools in different combinations to find optimal outcomes.

Table 5.2: Training Outline: Custom YOLOv4 Model

Training Hyper-parameter	**Trial 1**	**Trial 2**	**Trial 3**	**Trial 4**
Iteration	6000	6000	6000	6000
Batch	64	64	64	64
Momentum	0.949	0.949	0.949	0.8
Decay	0.0005	0.0005	0.0005	0.0005
Learning rate	0.001	0.001	0.001	0.005
Saturation (BoF)	1.5	1.5	1.5	1.5
Exposure (BoF)	1.5	1.5	1.5	1.5
Image Augmentation (BoF)	mosaic	mosaic	mosaic, angle = 30	angle = 45
Activation (BoS)	mish	mish	mish	mish
Data set	Batch 1	Batch 2	Batch 3	Batch 4
Data Addition	Base	Yes	Yes	Yes
Data type	Real	Real	Real & Synthetic	Real & Synthetic
Data augmentation (Pre-training)	No	Yes	Yes	Yes
Pre-trained weights (conv-137)	YOLOv4	Trial-1 (best)	Trial-2 (best)	YOLOv4
Training time	5.5 Hours	5.5 Hours	6 Hours	6 Hours

5.4.3 Model Deployment

Offline Model Deployment

The best weight of the trained model can be deployed as is after the training. It can be deployed over the cloud platform to check model performance on the test data set. It can also be

transferred to other computational tools (laptop or computer) with the required software and DL APIs (chapter 2.2.4). The model can be deployed and used from these machines with high enough computational power to run inference. Results are presented and discussed in section 5.5. However, the model weights have much training-related information that is not strictly needed for such deployment; as such, it is not a suitable form to deploy from an SBC for edge use. The best-trained model is checked in offline mode on saved images and videos.

Online Model Deployment

It is called an online deployment when the best model is deployed directly on the real-time camera feed for inference on a laptop or other setups. However, for edge deployment, the model needs to be optimized. The best model is selected for optimization to adapt it to the embedded platform Jetson Nano. as discussed in section 4.3.2, the model is quantized using TensorRT. During quantization, information regarding the inference is preserved, the convolutional layers are frozen, nodes are fused, GPU memory is used, and a TensorRT engine is generated. This engine is the lighter version of the model, which can be run at the edge with reduced latency. It is tested using the IP RGB sensor, Raspberry Pi 2.0, for real-time inference. The results are also shown and discussed in the following section 5.5.

5.5 Result & Discussion

The results of four experiments conducted to find the best model using a combination of real-synthetic data with different training hyper-parameter configurations are presented in this section. For the validation experiment, the custom algorithm is checked on data unknown to the model (Table 5.1) and on different image qualities. Some probable cases are- i) images with no corrosion or defects, ii) low light, iii) distorted images, and so on. In all these cases, the algorithm holds well, gives an acceptable result, and successfully flags the anomalies (corrosion) that need attention from an inspector.

5.5.1 Model Evaluation

The training was conducted over the cloud, as mentioned in the tools section. Table 5.2 shows the outline of the custom model training parameters of all four trials. Trial 1 is the initial model trained on the base, batch-1 real data (Table 5.1). The outcome gives a preliminary idea of the model performance. Corrosion detection needs a complex feature combined with texture and color. Due to its wide variety, inconsistent shape/size, diverse nature, and the small data set, the training results in 29% mAP with 0.55, 0.32, and 0.41 precision, recall, and F1-scores respectively (Table 5.3). The 11-point AP measurement is used to calculate the mAP at IoU 50.

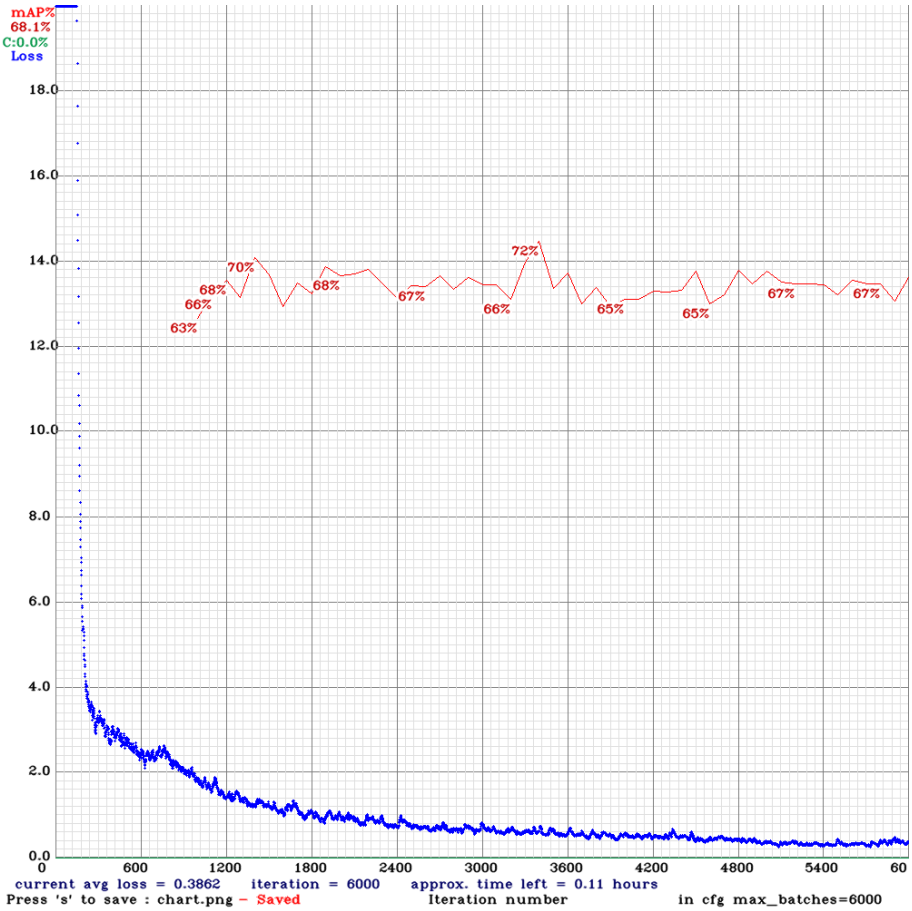


Figure 5.5: Training Chart Showing 'mAP' and Average Loss Over 6000 Iterations

For trial 2, 30% additional images is added creating a new data set batch 2 for training (Table 5.1). Before training, basic image augmentation techniques, i.e., geometric transformation and

Table 5.3: Custom YOLOv4 Model Scores: Corrosion Detection

Evaluation Parameter	**Trial 1**	**Trial 2**	**Trial 3**	**Trial 4**
Mean Average Precision, mAP@0.50	29.05%	47.47%	56.44%	72.33%
True Positives (TP)	77	156	182	599
False Positives (FP)	64	139	54	234
False Negatives (FN)	161	180	195	237
Precision	55%	53%	77%	72%
Recall	32%	46%	48%	72%
F1-score	41%	49%	59%	72%
Average loss	1.61	2.09	2.11	0.606
Best weight size (MB)	244.2	244.2	256	244.2

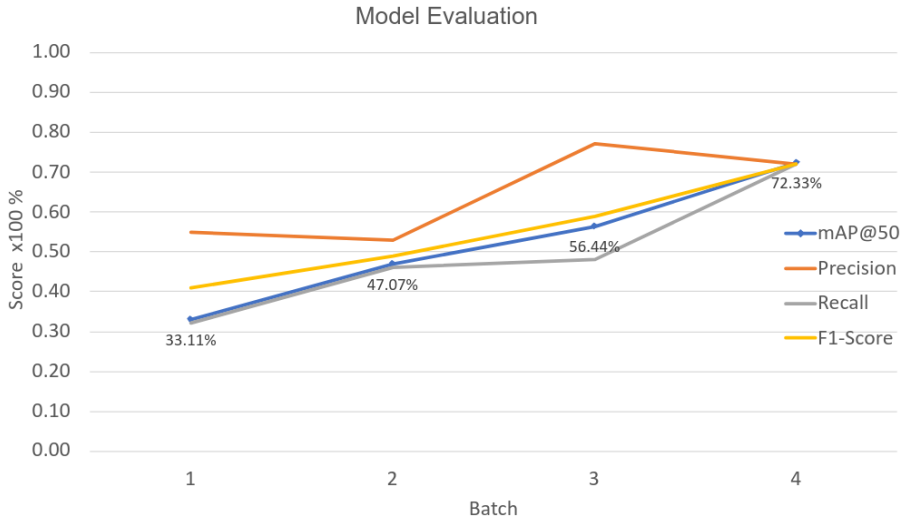


Figure 5.6: Model Training & Evaluation Scores

random color adjustment, are done to increase the data set size and add linear variation. For pre-trained weight, the frozen 'conv-137' layer of the best weight from trial 1 is used for transfer learning in this trial. An 18% mAP jump to 47.47% is observed. The precision falls to 0.53, whereas recall and F1 scores show improved scores at 0.46 & 0.49, respectively.

For further improvement, in trial 3, concentration is given to find ways to enrich the data set for the specific types of corrosion. In trial 3, the batch-3 data set is used as a mixture of real and synthetic data. For weight, the frozen conv-137 layer of the best weight from trial 2 is used to reuse its feature-learned layers. Additionally, one of the BoF image augmentation techniques, image rotation at a 30° angle, is added to the configuration. This amendment and adjustment yield a 13% improvement in mAP to 56.44%, as well as improved precision, recall, and F1 scores. The best model weight increased slightly to 256 MB.

Finally, in the fourth trial, the batch-4 data set is used with even more data (Figure 5.4). For pre-trained weight, YOLOv4 pre-trained weight conv-137 is used. The image augmentation technique 'mosaic' is removed which is not giving the desired result and the angular rotation is increased to 45°. Hyperparameters were also adjusted, such as decreasing the momentum from 0.949 to 0.8 and increasing the learning rate five folds to 0.005. This results in a 16% increase of mAP to 72.33% (Figure 5.5). There is also an improvement in recall and F1 score to 0.72 (Table 5.3), whereas the precision has a slight dip to 0.72. Another major improvement observed is in the training loss (specifically 'Complete Intersection-Over-Union' or CIoU loss for YOLOv4) as it drops continuously throughout the training phase to 0.606% (blue line in Figure 5.5) indication that the model is not over-fitting. The red line is the mAP at 50% IoU threshold (mAP@0.5), which checks the model's performance on a never-before-seen validation data set. Overall the mAP remains flat with little variations, but the average loss keeps dropping throughout the training phase, which is a good indication.

The overall improvement in model performance in terms of mAP@50 is from 29% to 47.47% with real data only. After adding additional target-specific synthetically generated data, the mAP@50 increases to 56.44% in trial-3 and 72.33% in trial-4. The mAP@50 significantly increases by 43% (a factor of 1.6), from 29% to 72%, as the data set increases from 868 to 1422 by 60% . The average loss falls below 1% as the model converges over the 6000 iterations. In the four trials, the data set is enriched, and different hyperparameters and augmentation combinations are used as training configurations; as such, the improvement cannot be correlated. However, the model performance definitely improves with the increase of data, specific data augmentation, and hyperparameter adjustments, with the fourth trial generating the best model with an accuracy of over 70%.

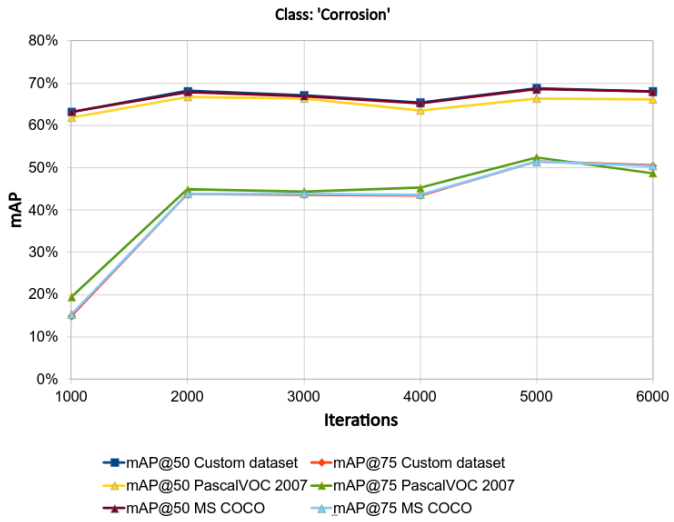


Figure 5.7: Iteration vs. 'mAP' Score Comparison in Different Standard Data Sets for the Best Custom YOLOv4 Model

The performance score is analyzed against Pascal VOC 2007 and MS COCO data sets (Figure 5.7) at 50% and 75% IoU thresholds. The mAP data set for the custom data set at 50% threshold is closely followed by the MS COCO data set, with a drop in mAP for Pascal VOC. For IoU threshold 75%, all three data sets are approximately the same, with the custom data set scoring the highest mAP@75 of 50%.

Images shown in Figure 5.8 were randomly selected by the algorithm from the database for testing to demonstrate the detection performance of the improved model more intuitively. The custom model is used to identify corrosion in images under different settings, and some of the detection results are shown in the said figure. Example images are taken under different angles and illumination conditions and with blurred and noisy (textured) backgrounds. The bounding boxes (red) display the model confidence score, and the detection FPS can be seen in the top left corner of each inferred image. It can be observed that in all four images, corrosion can be detected correctly.

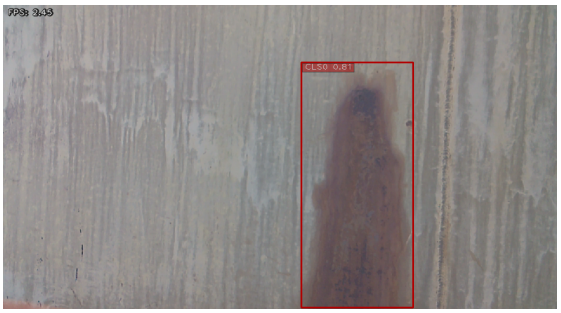
(a) Input Image

(b) Detection on Blurred Image

(c) Input Image

(d) Detection on Uneven Surface

(e) Input Image Taken in Late Afternoon Sunlight



(f) Detection on image (e)

(g) Input Image with Textured Background

(h) Detection on Image (g)

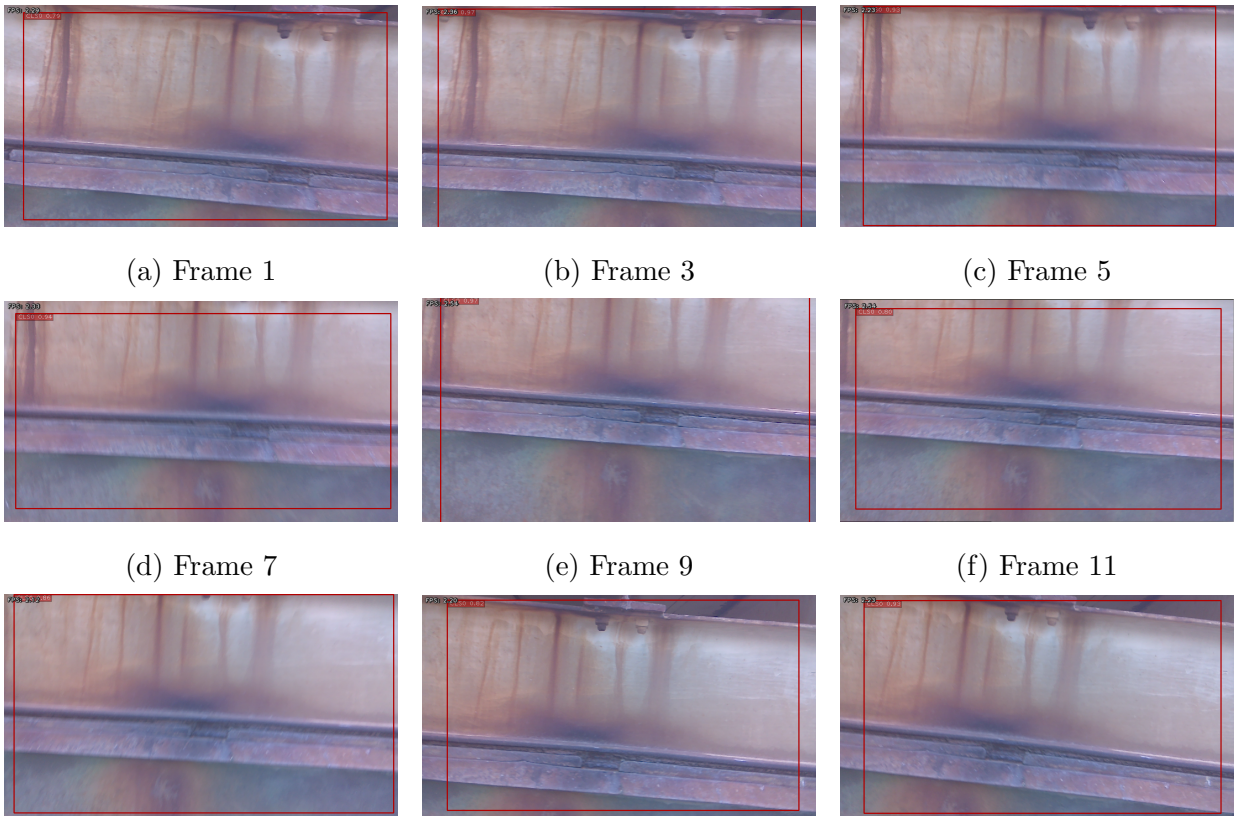
Figure 5.8: Detection Results of Corrosion Under Different Lighting Conditions

5.5.2 Experimental Result at the Edge

For further validation of the processing capability of the improved model in mobile devices, the trained model is deployed on the Jetson Nano. Its processor is small in size, low in power consumption, and strong in computing performance compared to its peers. The full-size YOLOv4 model is slower due to its heavy & complex structure, which cannot meet the needs of mobile devices for real-time detection. The optimized & quantized lighter version, model TensorRT engine's performance is tested by deploying it on images, videos, and live camera feed for inference. To compare the result, the lighter version of the model, YOLOv4-tiny, is trained with the batch-4 data set (Table 5.4). Among the models, the custom quantized model shows a faster deployment FPS of 2.5, as seen on the top left side of each inference image. The optimized model is tested by deploying it in the UAV setup at a local power station. Utilizing ROS bag files and custom Python script, the real-time camera feed is infered using the model (Figure 5.9). It retains the higher accuracy of the parent model and achieves a faster processing speed, meeting the demand of a real-time OD application at the edge.

Table 5.4: Training Outline: Custom YOLOv4 Model

Model	**Training/ Conversion Time**	**Frames Per Second (FPS)**	**Mean Average Precision (mAP, %)**	**Average Loss**
Custom YOLOv4 (Darknet) (Best)	6 hrs	1	72.33	0.606
YOLOv4-tiny	5 hrs	1.2	69.54	1.025
Quantized Custom YOLOv4 (TensorRT)	1 hr	2.5	-	-



(a) Frame 1 (b) Frame 3 (c) Frame 5

(d) Frame 7 (e) Frame 9 (f) Frame 11

(g) Frame 13 (h) Frame 15 (i) Frame 17

Figure 5.9: Corrosion Detection/Inference at the Edge (Old Power Plant Stack of El Paso Electric)

5.6 Conclusion

In this work, a deep learning (DL) based corrosion detection model is developed for automated visual inspection of industrial infrastructures. Among different types of corrosion, pitting, crevice & concrete corrosion are selected as the scope of this inspection, which reduces the variation, data-drift and also ensures enough data for each corrosion type to get a well-balanced and quality data set. A custom data set comprises both real and synthetically generated data to cater to limited corrosion data. YOLOv4 is used to build a custom model using transfer learning for its faster detection speed and significantly higher accuracy. This method conducts inspection

on images, videos and camera feed captured from the camera on an edge device, identifies areas of interest, and flags it for further inspection by an experienced inspector without having to reach inspection areas physically. The model is quantized for faster deployment and inference. Detection accuracy and model performance are improved using different combinations of data augmentation techniques and varying training hyper-parameters. Experimental results confirm that the proposed optimized detector obtains satisfactory corrosion detection results, achieving 72.33% mAP for corrosion identification in a complex environment and getting real-time performance (2.5 FPS) with an off-the-shelf commercial SBC platform. Essentially, this method reduces the inspection and analysis period and subsequently reduces operational downtime, saving costs without compromising reliability.

Chapter 6

Multi-Spectral Visual Inspection

6.1 Introduction

Different non-destructive testing (NDT) techniques provide the knowledge and skills to swiftly and effectively evaluate and monitor aging structures for engineers and stakeholders. These techniques are employed for local structural health monitoring and damage detection [101]. The most popular and common NDT technique, 'visual inspection,' is performed using true color or RGB images, which can identify a defect in the visible spectrum. The obvious limitation is that the acquired information is limited to the lighting condition, as a regular RGB camera inspection feed without proper or any lighting under-performs or fails in poorly lit or dark areas. Even with a lighting solution, a vision-based inspection can be done only on the visible spectrum. Adding another spectrum of information can add important information to the visual inspection approach. In this work, the infrared spectrum is added to the true color scope as a solution to such visual inspection to generate a multispectral inspection capability.

Multispectral imaging is one of the most exciting technologies that have been used in different areas for non-destructive quality and safety inspection. The agriculture and food processing areas have been extensively using a multispectral analysis by combining color with different spectrum, e.g., ultraviolet or infrared [102, 103, 104]. However, these methods had unsolved issues that resulted in incorrect estimations [105]. Some of the notable work done in other inspection areas using multispectral imaging are bridge inspection to detect concrete cavities, [106], building inspection using UAV [107], and undercarriage inspection of railroad equipment [108]. All the work stated uses off-the-shelf costly multispectral cameras. Mostly these works are limited to components of interest, making them component-specific. This work aims to develop multispectral capacity usable across different infrastructures cost-effectively.

A multispectral inspection capability can make the analysis or health condition monitoring more robust and well-rounded. Multispectral images are usually modeled as mixtures of a few spectral endmembers. The multispectral analysis produces images with pixel-wise overlayed information from another spectrum in addition to the visual one through appropriate pixel-level correlation and calibration. However, off the shelf, high-resolution multispectral cameras/sensors easily cost over a couple of thousand USD on average. This work aims to develop a low-cost analysis method using visible (RGB) & LWIR (Long Wave Infrared) thermal spectrums, each corresponding to a significant scene component to constructing the multispectral scene.

6.1.1 Thermal Spectrum.

There are three modes of heat transfer; Conduction, Convection, and Radiation, and three ways an object dissipates radiation heat energy, absorption, transmission, and reflection [109]. A material with zero transmissivity and reflectivity is called a black-body. The temperature and emissivity of an object are determined by how much IR radiation it emits. The Stefan-Boltzmann Law describes the total radiation energy that a surface can release is,

$$E = \epsilon \sigma T^4 \tag{6.1}$$

Where E is the radiation energy (W/m^2), T is the temperature (K), σ is the Stefan-Boltzmann constant, and ϵ is the object's emissivity.

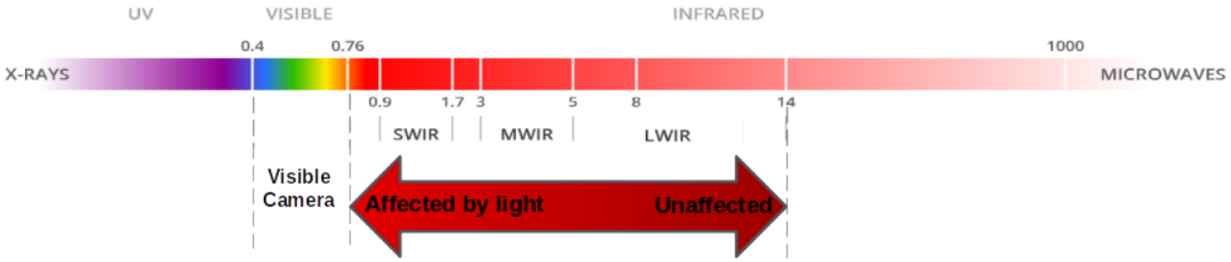


Figure 6.1: The Electromagnetic Spectrum [5]

Figure 6.1 shows the useful range of IR radiation in the electromagnetic spectrum, with wavelengths between 0.8 and 14μm, located between the visible and microwave spectrum. The

thermal spectrum can be broken down into the near-infrared region (0.8-1.5 μm), short-wavelength infrared region (1.5-2.5 μm), mid-wavelength infrared region (2.5-8 μm), and long-wavelength infrared region (8-14 μm). All object releases infrared radiation at temperatures over 273.15 °C (absolute zero temperature) in the MWIR and LWIR wavelengths in quantity proportional to the body's temperature.

Thermal & Radiometric Imaging. Thermal cameras 'see' the heat instead of light. They produce an image of a scene that portrays the temperature of the objects (infrared imaging) instead of the visible properties.

Figure 6.2: A Side-by-side Comparison of Near Infrared (NWIR) and Long Wave Thermal Infrared (LWIR) Cameras Display of a Scene [6].

'Thermal imaging' detects radiation and translates the temperature variations into grey scale, where brighter and darker shades of grey represent hotter and cooler temperatures to give a visual representation of the heat profile of the scene. In contrast, IR 'Radiometric imaging' can give pixel-wise temperature readings of the scene. If the temperature range is known in a frame of thermal image, then it can be called 'Radiometric' or 'Thermographic' representation. Some areas the IR inspection could be useful are in sub-surface defect detection, wear/friction detection, insulation problem detection, leak detection, electrical inspection, and fire prevention. 'LWIR' is chosen as it is representative of objects' temperature and it is not affected by ambient light. Figure 6.2 shows the difference between NWIR & LWIR spectrum, as the wavelength is much different from the visible spectrum range (figure 6.1). LWIR is useful as it can give low-light, day

& night vision with occlusion coverage and sub-surface inspection [110].

6.1.2 Applications.

Infrastructures may have sub-level issues that need to be determined effectively, which cannot be identified using only visual spectrum-based inspections. IR camera observes surface radiations (electromagnetic waves) connected to changes in temperature in the IR spectrum to find subsurface flaws [110]. Multiple literature confirm that emissivity is a crucial parameter for accurately measuring surface temperature with a maximum error in surface temperature of up to 7∘C. [111]. Most inspection guidelines recommend testing materials with emissivity higher than 0.6-0.7 [112]. The emissivity of most metals and materials on infrastructures usually exceeds that limit with some exceptions [111]. For example, concrete is one of the most common infrastructure materials with an emissivity higher than 0.92 [109]. The emissivity is typically influenced by the material's surface roughness, chemical composition, and moisture content. Other attributes on its surface which might cause visible changes in the image temperature are stains, water, and paint markings.

Observing the thermal profile can give useful clues to the sub-surface defects, usually hidden from the naked eye. For example, a concrete surface heats up by the increase in the surrounding temperature through heat absorption during the daytime, which then begins to radiate energy. If there are subsurface anomalies, the area inhibits heat conduction and warms up faster than the surrounding area of unaffected concrete. This difference in heat profile can be observed on IR imagery as a "hot spot" with a higher temperature. The sound concrete region loses heat slower than delaminated sections in comparison to at night when the ambient temperature typically drops. Thus, "cold spots" on the concrete surface with lower temperatures indicate underlying issues on thermal and IR images. IR thermography can be useful for assessing underlying anomalies of surfaces that obstruct and alters the general heat transfer profile. Therefore multispectral inspection can be effective for infrastructure inspection.

6.1.3 Thermal Sensor.

Among the IR cameras available in the market for this work, a budget-friendly LWIR camera, FLIR Lepton 3.5, is selected as mentioned in section 2.2.2. This inexpensive IR camera (about one-

tenth the cost of traditional IR cameras) solution has a small dimension and power requirement, which is usable in any of the handheld setups or onboard a UAV with limited power 2.3. This sensor is coupled with a Purethermal 2.0 smart I/O Module that is programmable and connects with micro-B type USB with the IP SBC, Jetson Nano [22]. The sensor comes pre-calibrated and streams temperature values in 0.01K pixel resolution. One of the prime features of the IR sensor is that it is radiometric, meaning that the temperature data is embedded in each pixel. The non-contact temperature data can be accessed with custom coding through the sensor board.

6.1.4 Challenges.

Normally, in a standard multispectral inspection system, two almost superimposed sensors are used; where one is sensitive to IR or some other spectrum, and the other is the RGB (true color) camera for visual image acquisition of the same surface. Since in this work, two different cameras are used, the acquired frames will vary in terms of resolution, FOV, and the number of images will depend on individual frame rates. A careful cross-system calibration is necessary for the pixel level overlap, which is time-consuming and presents several challenges, such as pixel level drift or inconsistent timing of the two frames. There are also hardware and software-associated configurations causing overlay deviation, depending on the hardware design, data encoding, and frame rate. Therefore, the two sensors need to be carefully configured, synced, and the data appropriately post-processed before they can be overlayed together on a pixel-by-pixel basis to get a consistent multispectral display. From an experimental point of view, thermographic techniques are classified into two major categories: 'Active' and 'Passive.' A transient heat transfer state is more effective in thermal inspection, as the thermal profile becomes obvious over time [113]; This can be achieved using an external excitation source for heating or cooling, e.g., photographic flashes and Halogen lamps. This method, known as 'active thermography,' needs a well-controlled lab environment. However, due to the onsite and periodical nature of the inspection, and considering IP mounted on platforms like UAV or handheld setups, this external heating cannot be achieved effectively. As such, the scope of this work is limited to passive thermography, which does not need external heating.

6.1.5 Contributions.

This work has developed a multispectral inspection capability by fusing RGB & IR feeds from the same scene. An integrated inspection system has been developed where a custom-built 'User Interface' (UI) can be used to perform data post-processing and multispectral inspection & analysis of infrastructure.

6.2 Technical Approach

Figure 6.3 shows the four steps of the technical approach of this work.

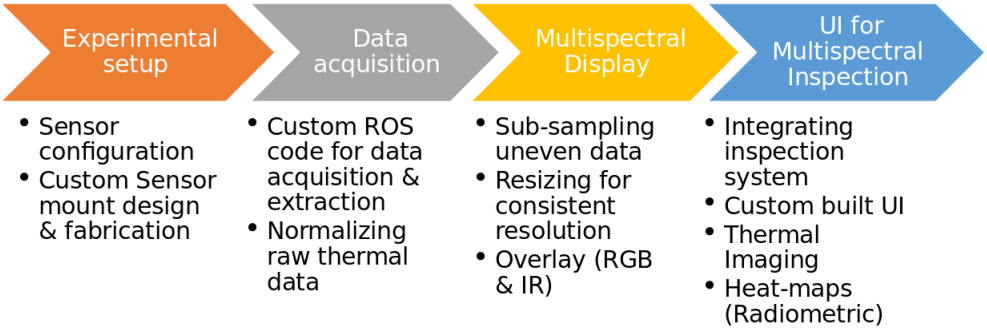


Figure 6.3: Technical Approach for Multi-spectral Inspection

6.2.1 Experimental Setup

The maximum resolution of the RGB camera is 3280 × 2464 pixels & 120x160 pixels for the IR. The focal length is also different; however, the field of view (FOV) of the RGB camera is 62.2°, which is pretty close to 57°, the FOV of the IR camera as mentioned in sections 2.2.2 & 2.2.2, which is a major reason for selecting this IR sensor.

For a close comparison of a scene, some adjustments regarding the resolution, frame rate, and frame saving rates must be made. However, some pixels will be lost due to the non-conformity of the aspect ratios and physical distance of the two sensors, as they cannot be physically put in the same location. A closer look at the RGB camera shows that the sensor is connected to the

board with a non-rigid connector, which may have a slight play during testing due to vibration caused by movement.

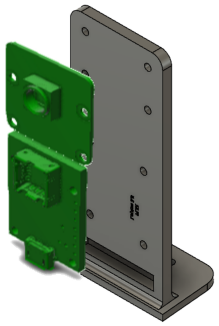


Figure 6.4: Expanded View of Custom Camera Mount with Two Sensors in CAD for Maximum Frame Overlap

The physical distance between the two sensors can be reduced as much as possible by putting the sensors as close as possible by designing custom camera mounts (Figure 6.4). The camera attributes are carefully selected considering IP computational power, power requirement, and best frame synchronicity. Some of the important parameters are as follows:

Table 6.1: Sensor Configurations

Sensor Configurations	**RGB Camera (Raspberry Pi 2.0)**	**IR Sensor (Lepton 3.5)**
Encoding	"bgr8"	"bgr8" or "mono8"
Resolution	1280x720	160x120
Frequency	14	8

One of the prime features of the IR sensor is that it is radiometric, meaning that the pixel-wise temperature can be accessed via the sensor board through custom coding. For IR images, either 'bgr8' or 'mono8' encoding can be used.

6.2.2 Data Acquisition

ROS is used for data acquisition. Data acquisition includes extracting images from an RGB camera, temperature values from radiometric or 'raw' feed & normalized 'thermal' images from an IR sensor to view the thermal frame. The raw images are not discernible with the naked eye as the pixel or temperature variation in a raw feed is small (Figure 6.5b). As such, the raw data is normalized from 0 to 65535 and converted to the grey scale to view (Figure 6.5c). Below are images taken with the sensors:

(a) Visible Spectrum Image (Original Size 960x540)

(b) 8-bit Raw Thermal Image (Original Size 160x120)

(c) 8-bit Normalized IR Spectrum Image (Original Size 160x120)

Figure 6.5: Three Representations of an Instance from the Two Sensors

The sensor data are extracted and stored using custom ROS node 'image_extraction.py' coded in Python. All sensor data is saved in a ROS bag file with timestamps for post-analysis.

6.2.3 Multispectral Display

Sub-sampling

The timestamps are used to name the data frames for correlating between 3 different representations of an instance, i.e., RGB, raw & thermal (normalized). Raw & thermal images have the same number of images. As the frame rates of the sensors differ, the images are sub-sampled for synchronizing the multi-spectrum feed. Image numbers are compared as the first step of sub-sampling between the RGB and thermal images. RGB has more images due to higher image acquisition frequency, so it is sub-sampled to the thermal image. Next, During sub-sampling, the

closest float up to 2 decimal points are considered to look for a match. Once a match is found, these images are sub-sampled to another sub-folder called 'Extracted.' After sub-sampling, all 3 data representations will have an extracted folder with the same number of images. The directory should look like the following:

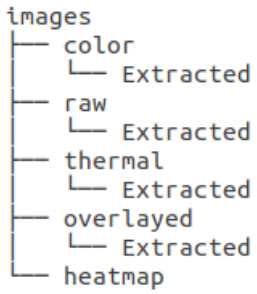
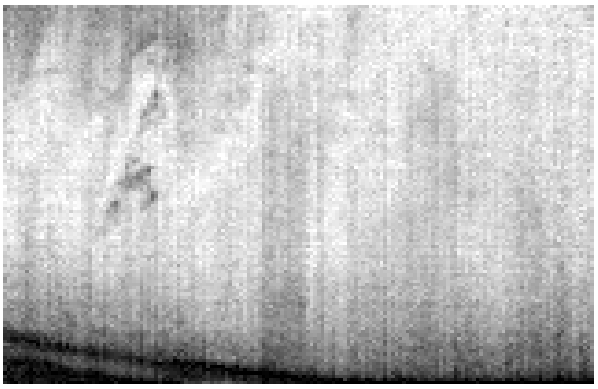


Figure 6.6: Image Directory Tree

Resizing

The two sensors' images need to be the same size to get the multispectral view by overlaying one over the other. As mentioned before, there is a difference in the aspect ratio and the FOV of the sensors. For this analysis, generic image processing techniques, i.e., cropping & resizing, are used to get the same size and aspect ratio. Given that constant frame resolutions, frame rate, sensor-to-inspection surface distance & FOVs are maintained during image acquisition, this generic technique would work adequately for this analysis. After studying the shapes, a suitable final image size, 1190 x 565 & aspect ratio, 2.1, is selected, so the least pixels are lost in the process. The RGB image is first resized from the original 960x540 to 1280x720, then cropped to 1190 x 565, ending with the desired 2:1 aspect ratio. The IR images are at first cropped to 145x69 to adhere to the 2:1 aspect ratio, then resized to the final size of 1190 x 565. The cropping ensures that the maximum overlap of the scene is achieved. The process is automated on the complete data set with a python script.

(a) Visible Spectrum Image



(b) Normalized IR Spectrum Image

(c) Multi-spectrum Image (Opacity 50%)

Figure 6.7: RGB, IR & Overlayed Multispectral View of an Instance
(Final Size 1190×565)

Overlay

All three RGB, IR, and raw have the same number of files and are resized to the same size. Using OpenCV, the IR & RGB images are blended from the index (0,0). The opacity of the blend is customizable. However, 50% opacity is taken as default. The figures below show the images together for a qualitative study of the Multispectral view, seen in Figure 6.7c.

6.2.4 UI for Multispectral Inspection

An integrated inspection system has been developed to perform data extraction, sub-sampling, multispectral display, and radiometric (temperature) reading of a scene using a UI. The custom-built UI titled "UI for Multispectral Inspection" aids users in easily executing data post-processing, inspection, and analysis in the multispectral domain with the click of a button.

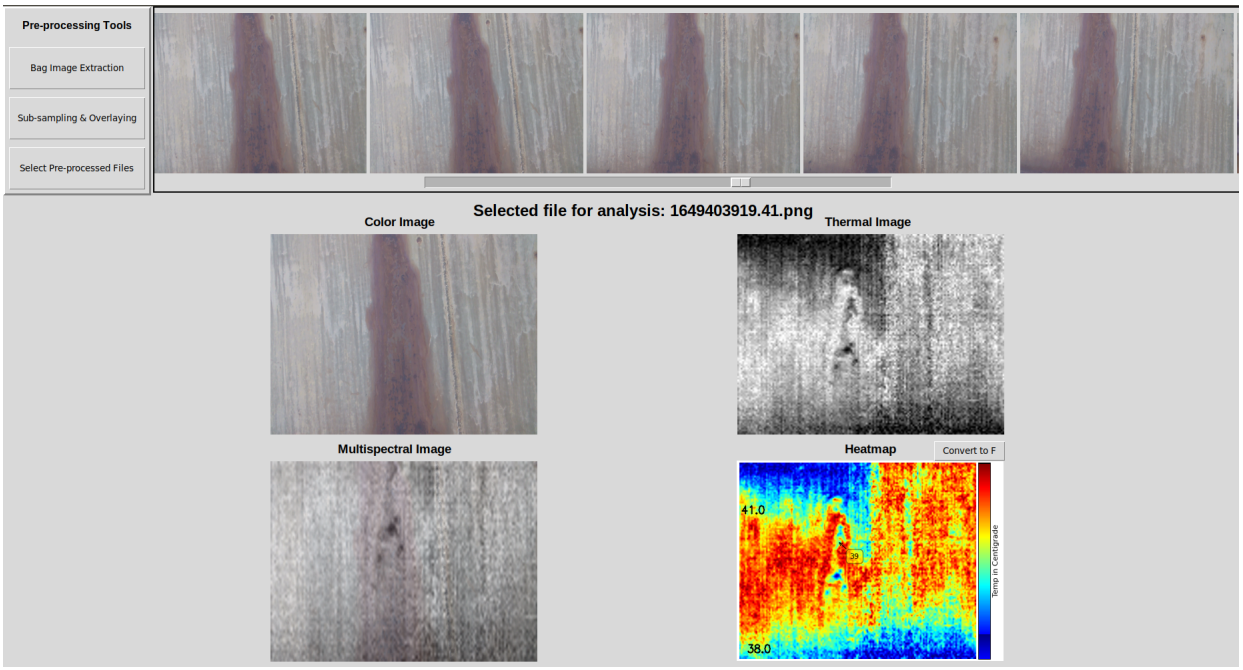
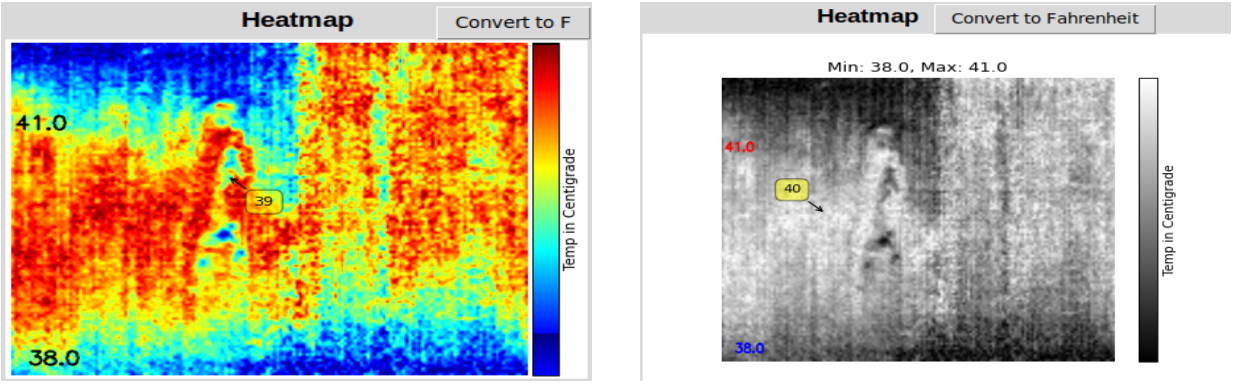


Figure 6.8: UI for Multispectral & Thermal Inspection

The functional buttons are placed on the top left corner of the UI window, and the names are expressive of the operation it performs, i.e., 'Bag Image Extraction,' 'Sub-sampling & Overlaying,' & 'Select Pre-processed files.' Clicking on any of these options opens up a file explorer in the system path to choose the file in the directory.

If the files have already been extracted, sub-sampled & overlayed, further analysis is done by clicking on the 'Select Pre-processed files' button. Once this button is pressed, and subsequently, the required 'image' directory containing the required sub-folders 6.6 is chosen via the file explorer, the UI will start populating all the color images on top, over the scroll bar. All images can be easily navigated by scrolling from left to right for each test set. The color image is correlated to sampled thermal and raw images. Selecting any of the color images selects the three representations for

multispectral analysis. The selected file's name is displayed just below the list of color images. The top row displays the 'Color' & the normalized 'Thermal Image,' and the bottom row shows the 'Multispectral Image' and 'Heatmap' (Figure 6.9b).



(a) Heatmap

(b) Heatmap (Grey scale)

Figure 6.9: Heat-map with Maximum, Minimum and Pixel-wise (Yellow Highlight) Temperatures (°C) Displayed

The multispectral image gives additional thermal information on the inspection surfaces. In Figure 6.8, the corrosion, although visible in the color image, is more prominent in the multispectral view due to the thermal profile seen in the LWIR spectrum. The exact location of metal loss can be found under the evenly corroded colored surface.

Along with the visual & multispectral views, a thermal heat map with temperature readings is generated. The temperature scale can be chosen from the adjacent conversion button in degrees Celsius (default) and degrees Fahrenheit. By default, the heat map shows the highest (red) and the lowest (blue) temperatures. A temperature scale displays the instance's temperature range (on the right), and pixel-wise temperatures can be read by hovering the mouse pointer above the display, as seen in Figure 6.8.

6.3 Experimental Results

6.3.1 Test Runs

Several experimental studies have been conducted in different test facilities mentioned in section 2.4 using IP on handheld setup-1 and UAV (section 2.3). Initial studies were conducted to try out the radiometric and thermal features. There were a total of 5 flight tests conducted at Vinton & EPE to acquire multispectral data from the two sensors. All relevant sensor data were saved in ROS bag files containing RGB, raw & thermal images from three ROS topics, i.e., /main_camera/image_raw, and /thermal_img. Here is the breakdown of the images extracted from the ROS bag file using the Python extraction code.

Table 6.2: Data Acquisition by IP Onboard UAV

Bag File Name	**RGB images (1280x720)**		**Thermal images (160x120)**		**Flight Duration (s)**
	Total	**Frequency (Hz)**	**Total**	**Frequency (Hz)**	
t1_vinton.bag	405	5.6	690	10.1	15.58
t2_vinton.bag	396	6.1	688	10.2	15.12
t1_therm_EPE_060922.bag	944	14.24	550	8.3	66.38
t3_therm_EPE_060922.bag	1550	14.53	921	8.63	106.67
t4_therm_EPE_060922.bag	1870	13.63	1076	8.2	137.41

6.3.2 Temperature Reading & Thresholding.

Using Heat-maps from the UI, the temperature measurement or thermographic study can be conducted at the region of interest 6.9b. Another functionality of the thermal imaging technique, 'thermal thresholding,' is added to get an augmented view of the temperature profile. The thermal thresholding adds pseudo colors to radiometric pixels within a custom preset temperature.

The pseudo-colored area of the radiometric feed has a temperature within the set threshold, making problem regions easily identifiable from the surrounding. It can easily detect hot-spots or

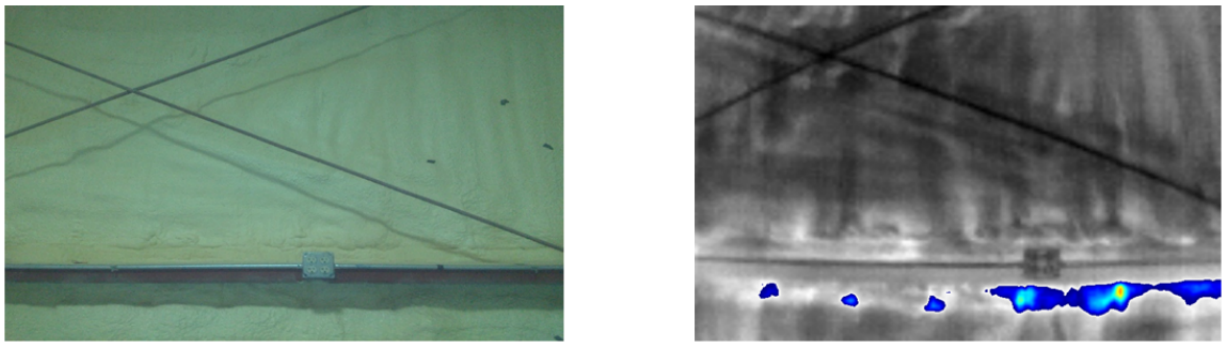
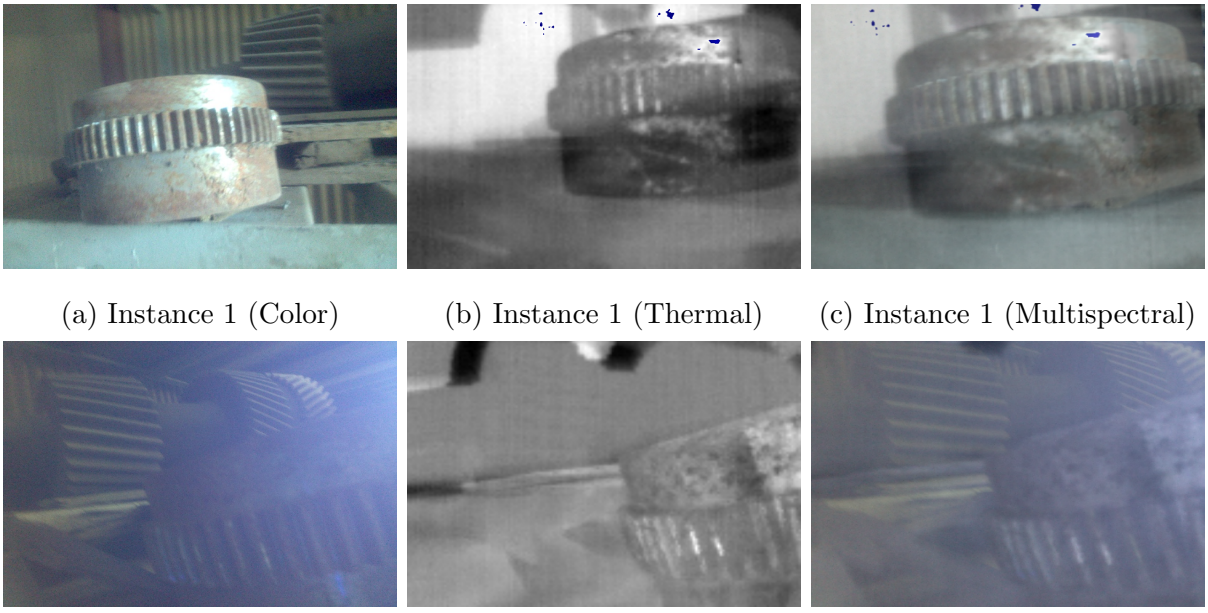


Figure 6.10: Thermal Thresholding Between 30-40°C

surfaces with a loss of insulation. In Figure 6.10, a thermal thresholding of 30-40°C is set while running the experiment indoors at a well-air-conditioned test site of ambient temperature 26°C (approximately). In comparison, the outside temperature was approximately 33°C that day.

6.3.3 Defect Detection.

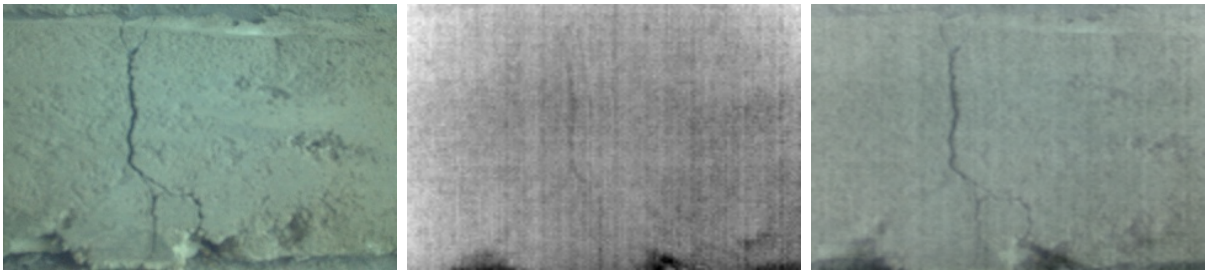


(a) Instance 1 (Color) (b) Instance 1 (Thermal) (c) Instance 1 (Multispectral)

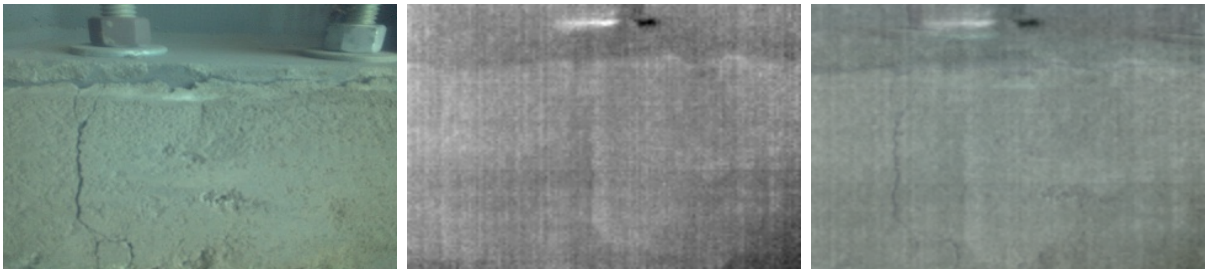
(d) Instance 2 (Color) (e) Instance 2 (Thermal) (f) Instance 2 (Multispectral)

Figure 6.11: Multispectral Analysis of Corrosion.

Some interesting damage spots are captured to evaluate the method's potential (Figures 6.11 & 6.12). In general, the majority of the damage types or features can be detected using color image data. Figure 6.11a shows a corroded gear in a tool shed at Vinton. There are visible rust spots, which indicate moisture penetration or corrosion. The gear teeth also show similar defect spots. However, the multispectral view shows more information as to the extent of corrosion. The darker region shows slower heat transfer due to the rust formation, whereas the lighter surface emits heat quicker, indicating a defect-free surface area. A closer look at the gear in Figure 6.11c shows the advantage of the IR spectrum that makes the damage detection discernible even in poorly lit areas.



(a) Instance 3 (Color) (b) Instance 3 (Thermal) (c) Instance 3 (Multispectral)



(d) Instance 4 (Color) (e) Instance 4 (Thermal) (f) Instance 4 (Multispectral)

Figure 6.12: Multispectral Analysis of Crack.

Similarly, cracks as thin as 1 mm can be detected from a distance of 3 ft (Figure 6.12c). Figure 6.12f shows another example of a crack on a concrete surface recognizable in the multispectral display.

6.3.4 Thermal Sensor Issues.

The low-resolution IR camera poses some concerns, and the defect cannot always be easily understood. Due to the in-built auto-calibration, the IR sensor performs Flat Field Correction (FFC) to re-calibrate when the camera changes temperature and periodically during operation as needed, which results in granular output (Figures 6.13) [114]. The thermal profile still exists. However, it would take a trained eye to pick the subtle change. Pseudo-coloring or thresholding can be used to augment the temperature profile.

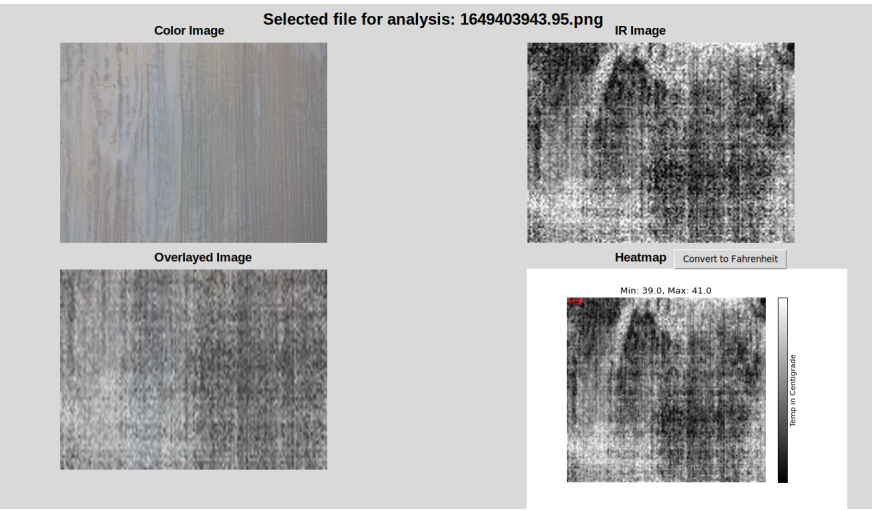


Figure 6.13: Granular Thermal Image due to Sensor's Periodic Flat Field Correction (FFC)

Another issue observed was the loss of data in some cases, observed in the IR image of Figures 6.14, most likely due to electrical or shutter issues. A manual pre-processing is needed to eliminate these erroneous data from the database.

The IR sensor does not have a built-in cooling system, which is needed to keep the sensor temperature within range. Continuous use in hot climates is another issue that sometimes generates shadow pixels (Figures 6.15); the shadow pixels are previous IR data getting locked in some of the sensor pixels, which needs the sensor to re-calibrate to resolve.

Given the results, the corrosion detection shows promising results over the crack detection with this budget-friendly multispectral inspection system. However, it can be assumed that other common defects, such as water leaks, semi-exposed subsurface metals, or foreign matters in

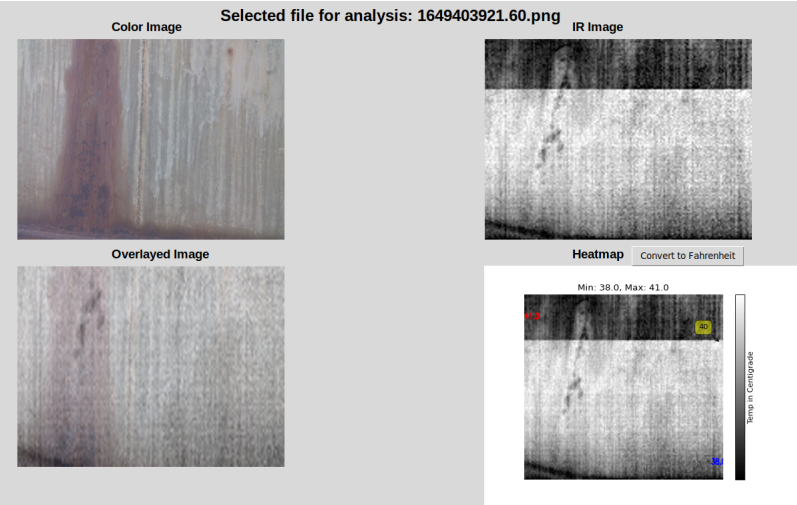


Figure 6.14: Thermal Data Loss at the Top of the IR Image

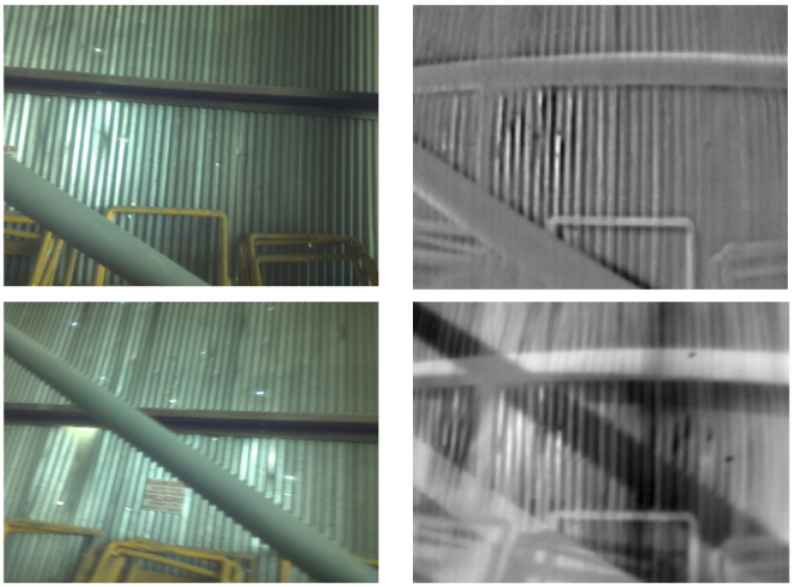


Figure 6.15: Temperature Drift Causes Shadow Pixels of the Pillar in the Bottom Right Thermal Image

infrastructures, can be easily detectable with a high-resolution multispectral setup.

6.4 Conclusion

The chapter presents a time- and cost-efficient integrated multispectral inspection system for infrastructure inspection tasks. Data has been acquired using IP on handheld setup and on-board UAV. The multispectral inspection system consists of an RGB & IR sensor to fulfill the requirements for detection and localization of the majority of damages on the inspection surface. Several practical studies were carried out to evaluate the system's performance. The results prove that the system is sufficient for crack and corrosion detection based on the investigation. At the same time, thermal imaging and radiometric capability add more functionality to detect known infrastructural defects, such as hot-spot identification or leak detection. Besides, the advantages of UAV-based monitoring and the other versatile and flexible functionalities added via the development of the UI make the inspection task easier at the user end.

Another fact is that, given the mentioned low-budget sensor limitations, it is not possible to solve all required inspection tasks. Image data can study and detect defect features on the surface of the infrastructure using multispectral vision. However, some defects, such as hairline cracks, can be better detected with an active thermography approach using external excitation in a lab setting. Nevertheless, UAV-based damage detection is a helpful and flexible assisting tool in infrastructure inspection. The multispectral analysis significantly facilitates the visual interpretation of infrastructure conditions monitoring and damage detection.

Chapter 7

Conclusion

7.1 Summary

The main objective of this dissertation was to develop a time- and cost-efficient inspection payload to perform intelligent inspection in infrastructures to detect cracks & corrosion. The methods can be applied in real-time or offline, and the inspection payload is inter-operable in different platforms, e.g., a handheld setup or onboard an aerial platform. The inspection task has four segments, and these are summarized below:

In the first part of the dissertation, a novel advanced manufacturing technique-based offline trajectory generation method was developed and verified for close-quarter aerial inspection of infrastructure using unmanned aerial vehicles (UAVs) in GPS and communication-denied environments in known & static settings. Two approaches presented are based on Computer Aided Manufacturing (CAM) and Additive Manufacturing (AM) machines. In both methods, the spatial coordinates were extracted from the generated machining tool paths of the infrastructure from the manufacturing environment of Computer Aided Design (CAD) software. The dense toolpath data was modified and optimized to construct a custom aerial trajectory that permits flight in reduced space without GPS or line of sight to perform a visual inspection. Of the two, the AM is the preferred method as it is much faster, more accurate, and has no machine-related limitation for intricate geometries. Simulations performed in the open-source robotics simulator Gazebo with a PX4 flight controller demonstrated the feasibility of the approaches. The proposed method requires multiple flights for large area coverage due to UAV's power constraint.

The second and third segments developed custom deep learning (DL)-based intelligent automated defect detection and localization algorithms for 'crack' and 'corrosion', respectively. Custom data sets have been developed using local and available resources. Corrosion provided some

challenges due to its multiplicity, inconsistent features, and inadequate data; as such, the study focuses on three common types: pitting, crevice & concrete corrosion. Additional synthetic data was added to create the standard-sized, balanced data set. A custom object detection model, 'YOLOv4', is trained using transfer learning and fine-tuning the hyper-parameters and configurations. The trained custom model is further optimized using model quantization and tested in a real-time application at the edge. Experimental results validate that the proposed algorithms and solutions can successfully detect crack and corrosion with satisfactory results, achieving 98.44% & 72.33% mean average precision (mAP), respectively, in a complex environment and getting real-time performance 2.5 frames per second (FPS) with an off-the-shelf commercial SBC platform.

In the last segment, an integrated multispectral inspection system is developed to detect infrastructural defects by fusing data from an RGB & an infrared (IR) sensor. A custom user interface (UI) is developed to extract, sub-sample, overlay, and perform thermal & radiometric analysis, making the inspection task easier for the end user. Experimental studies show that the proposed system and solutions can effectively identify cracks and corrosion and has the potential to detect other infrastructural defects, e.g., hot-spot, leaks, or subsurface anomalies.

7.2 Future Work

The future goal of this dissertation includes combining the defect detection projects to devise an integrated advanced inspection system onboard a UAV platform for real-time end-to-end adaptation. Swapping the single board computer (SBC) with a higher computational-capable one and modular integration can make real-time inspection feasible. An obstacle detection and avoidance system can be added to the offline trajectory generation method to avoid unforeseen obstructions, resulting in a situationally aware, more robust navigation system. Further, the corrosion detection model performance can be improved, and simultaneous crack and corrosion detection model can be devised by appropriately increasing and enriching the data set in future. Additionally, advanced image processing techniques or machine learning algorithms can be investigated to improve the quality of multispectral overlaying operations. Collected images can be used to train a multispectral-based DL algorithm to develop advanced inspection techniques. An active thermography approach using external excitation can be explored in the future.

References

[1] Young-Jin Cha, Wooram Choi, and Oral Buyukozturk. Deep learning-based crack damage detection using convolutional neural networks. *Computer-Aided Civil and Infrastructure Engineering*, 32:361–378, 03 2017.

[2] Qipei Mei, Mustafa Gül, and Md Riasat Azim. Densely connected deep neural network considering connectivity of pixels for automatic crack detection. *Automation in Construction*, 110:103018, 11 2019.

[3] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection, 2020.

[4] Lutz Roeder. Netron, Visualizer for neural network, deep learning, and machine learning models, 12 2017. date-released: 2017-12-04, cited: 2022-08-07.

[5] Moulay A Akhloufi and Abdelhakim Bendada. Fusion of active and passive infrared images for face recognition. In *Thermosense: Thermal Infrared Applications XXXV*, volume 8705, pages 84–93. SPIE, 2013.

[6] n.d. NIR vs LWIR thermal camera side-by-side comparison why thermal is best for target detection. Retrieved from https://www.infinitioptics.com/video/nir-vs-thermal-comparison. Accessed: 2022-10-19.

[7] DNV (2021). Energy transition outlook 2021. Retrieved from https://eto.dnv.com/2021/highlights/energy-transition-outlook. Accessed: 2022-10-09.

[8] Gerhardus Koch, Michiel Brongers, N. Thompson, Y. Virmani, and Joe Payer. Corrosion cost and preventive strategies in the united states. *United States. Federal Highway Administration*, No. FHWA-RD-01-156, R315-01:1–16, 03 2002.

[9] Gerhardus Koch. Cost of corrosion. In A.M. El-Sherik, editor, *Trends in Oil and Gas Corrosion Research and Technologies*, Woodhead Publishing Series in Energy, pages 3–30. Woodhead Publishing, Boston, 2017.

[10] Katsumi Kubo, Shigeru Kanemoto, and Shimada Hideo. Image processing technologies in nuclear power plant monitoring. *Toshiba Rebyu*, 50(8):619–622, 1995.

[11] Gerald Seet, Song Huat Yeo, W. C. Law, Burhan, Choon Yue Wong, S. Sapari, and K. K. Liau. Design of tunnel inspection robot for large diameter sewers. *Procedia Computer Science*, 133:984–990, 2018.

[12] Cao Dung and Anh Le Duc. Autonomous concrete crack detection using deep fully convolutional neural network. *Automation in Construction*, 99:52–58, 12 2018.

[13] Janosch Nikolic, Michael Burri, Joern Rehder, Stefan Leutenegger, Christoph Huerzeler, and Roland Y. Siegwart. A uav system for inspection of industrial facilities. *2013 IEEE Aerospace Conference*, pages 1–8, 2013.

[14] Waseem Rawat and Zenghui Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation*, 29:2352–2449, 06 2017.

[15] Kasthurirangan Gopalakrishnan. Deep learning in data-driven pavement image analysis and automated distress detection: A review. *Data*, 3:28, 07 2018.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[17] n.d. Appendix l: Interface requirements document outline. Retrieved from https://www.nasa.gov/seh/appendix-l-interface-requirements-document-outline/. Accessed: 2022-10-09.

[18] n.d. Appendix e: Creating the validation plan with a validation requirements matrix. Retrieved from https://www.nasa.gov/seh/appendix-e-creating-the-validation-plan. Accessed: 2022-10-10.

[19] n.d. Raspberry pi documentation - camera. Retrieved from https://www.raspberrypi.com/documentation/accessories/camera.html. Accessed: 2022-10-09.

[20] n.d. Lepton® 3.5 by teledyne flir — groupgets. Retrieved from https://groupgets.com/manufacturers/flir/products/lepton-3-5. Accessed: 2022-10-09.

[21] n.d. Data sheet: Nvidia jetson nano system-on-module. Retrieved from https://developer.nvidia.com/embedded/dlc/jetson-nano-system-module-datasheet. Accessed: 2022-10-10.

[22] n.d. Purethermal 2 - flir lepton smart i/o module — groupgets. Retrieved from https://groupgets.com/manufacturers/getlab/products/purethermal-2-flir-lepton-smart-i-o-module. Accessed: 2022-10-19.

[23] Harry T Roman. Robotic applications in pse&g's nuclear and fossil power plants. *IEEE Transactions on Energy Conversion*, 8(3):584–592, 1993.

[24] G Seet, SH Yeo, WC Law, CY Wong, S Sapari, KK Liau, et al. Design of tunnel inspection robot for large diameter sewers. *Procedia computer science*, 133:984–990, 2018.

[25] Naoto Kawauchi, Shigetoshi Shiotani, Hiroyuki Kanazawa, Taku Sasaki, and Hiroshi Tsuji. A plant maintenance humanoid robot system. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 3, pages 2973–2978. IEEE, 2003.

[26] Michael Burri, Janosch Nikolic, Christoph Hürzeler, Gilles Caprari, and Roland Siegwart. Aerial service robots for visual inspection of thermal power plant boiler systems. In *2012 2nd international conference on applied robotics for the power industry (CARPI)*, pages 70–75. IEEE, 2012.

[27] Mo Shan, Fei Wang, Feng Lin, Zhi Gao, Ya Z Tang, and Ben M Chen. Google map aided visual navigation for uavs in gps-denied environment. In *2015 IEEE international conference on robotics and biomimetics (ROBIO)*, pages 114–119. IEEE, 2015.

[28] Hang Li, Andrey V Savkin, and Branka Vucetic. Autonomous area exploration and mapping in underground mine environments by unmanned aerial vehicles. *Robotica*, 38(3):442–456, 2020.

[29] Brodie Chan, Hong Guan, Jun Jo, and Michael Blumenstein. Towards uav-based bridge inspection systems: A review and an application perspective. *Structural Monitoring and Maintenance*, 2(3):283–300, 2015.

[30] Alberto Ortiz, Francisco Bonnin-Pascual, and Emilio Garcia-Fidalgo. Vessel inspection: A micro-aerial vehicle-based approach. *Journal of Intelligent & Robotic Systems*, 76(1):151–167, 2014.

[31] Jan Quenzel, Matthias Nieuwenhuisen, David Droeschel, Marius Beul, Sebastian Houben, and Sven Behnke. Autonomous mav-based indoor chimney inspection with 3d laser localization and textured surface reconstruction. *Journal of Intelligent & Robotic Systems*, 93(1):317–335, 2019.

[32] Enhui Zheng and Jiaping Chen. A method for corrosion image acquisition of chimney inner. In *2019 Chinese Control And Decision Conference (CCDC)*, pages 6411–6415. IEEE, 2019.

[33] Neshat Bolourian and Amin Hammad. Lidar-equipped uav path planning considering potential locations of defects for bridge inspection. *Automation in Construction*, 117:103250, 2020.

[34] Yueh-Jaw Lin, Rahul Mahabaleshwarkar, and Elena Massina. Cad-based cmm dimensional inspection path planning–a generic algorithm. *Robotica*, 19(2):137–148, 2001.

[35] E Cerit and I Lazoglu. A cam-based path generation method for rapid prototyping applications. *The International Journal of Advanced Manufacturing Technology*, 56(1):319–327, 2011.

[36] Guoqiang Zeng, Chin-Yin Chen, Dishan Huang, and Yingdan Zhu. Robotic trajectory planning based on cl data. In *2015 IEEE international conference on robotics and biomimetics (ROBIO)*, pages 1723–1728. IEEE, 2015.

[37] Chin-Yin Chen, Junjie Li, Yindan Zhu, Liyan Tu, and Wenwu Weng. Automatic finishing system research for industrial robot. In *2017 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 266–271. IEEE, 2017.

[38] Charles Farrar and Keith Worden. *Structural Health Monitoring: A Machine Learning Perspective*. Wiley, 2012.

[39] Mohammad Sabokrou, Mohsen Fayyaz, Mahmood Fathy, Zahra Moayed, and Reinhard Klette. Deep-anomaly: Fully convolutional neural network for fast anomaly detection in crowded scenes. *Computer Vision and Image Understanding*, 172:88–97, 02 2018.

[40] Alireza Mirrashid and Ali-Asghar Shirazi. Rqcsnet: A deep learning approach to quantized compressed sensing of remote sensing images. *Expert Systems*, 38(8):e12755, 12 2021.

[41] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen van der Laak, Bram Ginneken, and Clara Sánchez. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42:60–88, 02 2017.

[42] Dheyaa Ibrahim, Dilovan Zebari, Hussam Mohammed, and Mazin Mohammed. Effective hybrid deep learning model for covid-19 patterns identification using ct images. *Expert Systems*, page e13010, 05 2022.

[43] Suguru Yokoyama and Takashi Matsumoto. Development of an automatic detector of cracks in concrete using machine learning. *Procedia Engineering*, 171:1250–1255, 12 2017.

[44] Leo Pauly, Harriet Peel, Shan Luo, David Hogg, and Raul Fuentes. Deeper networks for pavement crack detection. In *34th International Symposium in Automation and Robotics in Construction*, page 479–485, 07 2017.

[45] Hoo-chang Shin, Holger Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, and Ronald Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. In *IEEE Transactions on Medical Imaging*, volume 35, page 1285–1298, 02 2016.

[46] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.

[47] Wilson Silva and Diogo Schwerz de Lucena. Concrete cracks detection based on deep learning image classification. In *Proceedings of the 8th International Conference of Experimental Mechanics*, volume 2, page 5387, 06 2018.

[48] Byunghyun Kim and Soojin Cho. Automated vision-based detection of cracks on concrete surfaces using a deep learning technique. *Sensors*, 18:3452, 10 2018.

[49] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25:1097–1105, 01 2012.

[50] Chen Feng, Ming-Yu Liu, Chieh-Chi Kao, and Teng-Yok Lee. Deep active learning for civil infrastructure defect detection and classification. In *ASCE International Workshop on Computing in Civil Engineering 2017*, pages 298–306, 06 2017.

[51] Andrea Vedaldi and Karel Lenc. Matconvnet: convolutional neural networks for matlab. In *Proceedings of the 23rd ACM International Conference on Multimedia*, pages 689–692, 10 2015.

[52] Dongho Kang and Young-Jin Cha. Autonomous uavs for structural health monitoring using deep learning and an ultrasonic beacon system with geo-tagging. *Computer-Aided Civil and Infrastructure Engineering*, 33:885–902, 05 2018.

[53] Markus Eisenbach, Ronny Stricker, Daniel Seichter, Karl Amende, Klaus Debes, Maximilian Sesselmann, Dirk Ebersbach, Ulrike Stoeckert, and Horst-Michael Gross. How to get pavement distress detection ready for deep learning? a systematic approach. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2039–2047, 05 2017.

[54] Baoxian Li, Kelvin Wang, Allen Zhang, Enhui Yang, and Guolong Wang. Automatic classification of pavement crack using deep convolutional neural network. *International Journal of Pavement Engineering*, 21:1–7, 06 2018.

[55] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 01 2022.

[56] Ju Huyan, Wei Li, Susan Tighe, Junzhi Zhai, Zhengchao Xu, and Yao Chen. Detection of sealed and unsealed cracks with complex backgrounds using deep convolutional neural network. *Automation in Construction*, 107:102946, 11 2019.

[57] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *32nd International Conference on Machine Learning*, pages 448–456, 02 2015.

[58] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot MultiBox detector, 2016.

[59] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017.

[60] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017.

[61] Hiroya Maeda, Yoshihide Sekimoto, Toshikazu Seto, Takehiro Kashiyama, and Hiroshi Omata. Road damage detection and classification using deep neural networks with smartphone images: Road damage detection and classification. *Computer-Aided Civil and Infrastructure Engineering*, 33:1127–1141, 06 2018.

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

[63] Rohit Ghosh and Omar Smadi. Automated detection and classification of pavement distresses using 3d pavement surface images and deep learning. *Transportation Research Record: Journal of the Transportation Research Board*, 2675:1359–1374, 04 2021.

[64] Naga Siva Pavani Peraka, Krishna Prapoorna Biligiri, and Satyanarayana Kalidindi. Development of a multi-distress detection system for asphalt pavements: Transfer learning-based approach. *Transportation Research Record: Journal of the Transportation Research Board*, 2675:538–553, 05 2021.

[65] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, 2016.

[66] Donghan Lee, Jeongho Kim, and Daewoo Lee. Robust concrete crack detection using deep learning-based semantic segmentation. *International Journal of Aeronautical and Space Sciences*, 20:287–299, 01 2019.

[67] Long Truong, Omar Mora, Wen Cheng, Hairui Tang, and Mankirat Singh. Deep learning to detect road distress from unmanned aerial system imagery. *Transportation Research Record: Journal of the Transportation Research Board*, 2675:776–788, 04 2021.

[68] Upesh Nepal and Hossein Eslamiat. Comparing yolov3, yolov4 and yolov5 for autonomous landing spot detection in faulty uavs. *Sensors*, 22:464, 01 2022.

[69] Forrest Iandola, Matt Moskewicz, Sergey Karayev, Ross Girshick, Trevor Darrell, and Kurt Keutzer. Densenet: Implementing efficient convnet descriptor pyramids, 2014.

[70] Zhi Zhang, Tong He, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of freebies for training object detection neural networks, 2019.

[71] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12993–13000, 02 2020.

[72] Nvidia tensorrt, 07 2022. cited 07 Aug, 2022.

[73] J.k. Jung. tensorrt_demos, 2022. cited 07 Aug, 2022.

[74] Mousumi Rizia, Angel Ortega, Julio Reyes Muñoz, Michael McGee, Ahsan R Choudhuri, and Angel Flores-Abad. A cam/am-based trajectory generation method for aerial power plant inspection in gps-denied environments. In *AIAA Scitech 2020 Forum*, page 0858, 2020.

[75] Mousumi Rizia, Julio A Reyes-Munoz, Angel G Ortega, Ahsan Choudhuri, and Angel Flores-Abad. Autonomous aerial flight path inspection using advanced manufacturing techniques. *Robotica*, pages 1–24, 2022.

[76] C. F. Özgenel. Concrete crack images for classification. In *Mendeley Data*, 2019. doi: 10.17632/5y9wdsg2zt.2.

[77] Lei Zhang, Fan Yang, Yimin Zhang, and Ying Zhu. Road crack detection using deep convolutional neural network. In *IEEE International Conference on Image Processing (ICIP 2016)*, pages 3708–3712, 09 2016.

[78] Diganta Misra. Mish: A self regularized non-monotonic activation function, 2019.

[79] MM Reda Taha, Aboelmagd Noureldin, JL Lucero, and TJ Baca. Wavelet transform for structural health monitoring: a compendium of uses and features. *Structural health monitoring*, 5(3):267–295, 2006.

[80] Ramana M Pidaparti. Structural corrosion health assessment using computational intelligence methods. *Structural Health Monitoring*, 6(3):245–259, 2007.

[81] Carlos Fernández-Isla, Pedro J Navarro, and Pedro María Alcover. Automated visual inspection of ship hull surfaces using the wavelet transform. *Mathematical Problems in Engineering*, 2013, 2013.

[82] Francisco Bonnin-Pascual and Alberto Ortiz. Corrosion detection for automated visual inspection. 2014.

[83] Marinalda C Pereira, José WJ Silva, Heloisa A Acciari, Eduardo N Codaro, and Luis RO Hein. Morphology characterization and kinetics evaluation of pitting corrosion of commercially pure aluminium by digital image analysis. 2012.

[84] Nhat-Duc Hoang and Tran Duc. Image processing based detection of pipe corrosion using texture analysis and metaheuristic-optimized machine learning approach. *Computational Intelligence and Neuroscience*, In Press, 07 2019.

[85] Margarita R Gamarra Acosta, Juan C Vélez Díaz, and Norelli Schettini Castro. An innovative image-processing model for rust detection using perlin noise to simulate oxide textures. *Corrosion Science*, 88:141–151, 2014.

[86] MR Jahanshahi and SF Masri. Effect of color space, color channels, and sub-image block size on the performance of wavelet-based texture analysis algorithms: An application to corrosion detection on steel structures. In *Computing in Civil Engineering (2013)*, pages 685–692. 2013.

[87] Utkarsh Mahadeo Khaire and R Dhanalakshmi. Stability of feature selection algorithm: A review. *Journal of King Saud University-Computer and Information Sciences*, 2019.

[88] Deegan J Atha and Mohammad R Jahanshahi. Evaluation of deep learning approaches based on convolutional neural networks for corrosion detection. *Structural Health Monitoring*, 17(5):1110–1128, 2018.

[89] Young-Jin Cha, Wooram Choi, Gahyun Suh, Sadegh Mahmoudkhani, and Oral Büyüköztürk. Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types. *Computer-Aided Civil and Infrastructure Engineering*, 33(9):731–747, 2018.

[90] Vedhus Hoskere, Yasutaka Narazaki, Tu Hoang, and BillieF Spencer Jr. Vision-based structural inspection using multiscale deep convolutional neural networks. *arXiv preprint arXiv:1805.01055*, 2018.

[91] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern recognition*, 77:354–377, 2018.

[92] Prem Prakash Jayaraman, Abdur Rahim Mohammad Forkan, Ahsan Morshed, Pari Delir Haghighi, and Yong-Bin Kang. Healthcare 4.0: A review of frontiers in digital health. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(2):e1350, 2020.

[93] Abdur Rahim Mohammad Forkan, Yong-Bin Kang, Prem Prakash Jayaraman, Kewen Liao, Rohit Kaul, Graham Morgan, Rajiv Ranjan, and Samir Sinha. Corrdetector: A framework

for structural corrosion detection from drone images using ensemble deep learning. *Expert Systems with Applications*, 193:116461, 2022.

[94] Leijian Yu, Erfu Yang, Cai Luo, and Peng Ren. Amcd: an accurate deep learning-based metallic corrosion detector for mav-based real-time visual inspection. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–12, 2021.

[95] Samin Khan, Buu Phan, Rick Salay, and Krzysztof Czarnecki. Procsy: Procedural synthetic dataset generation towards influence factor studies of semantic segmentation networks. In *CVPR workshops*, pages 88–96, 2019.

[96] Magnus Wrenninge and Jonas Unger. Synscapes: A photorealistic synthetic dataset for street scene parsing. *arXiv preprint arXiv:1810.08705*, 2018.

[97] Farzan Erlik Nowruzi, Prince Kapoor, Dhanvin Kolhatkar, Fahed Al Hassanat, Robert Laganiere, and Julien Rebut. How much real data do we actually need: Analyzing object detection performance using synthetic and real data. *arXiv preprint arXiv:1907.07061*, 2019.

[98] Noshin Habib. *Synthetic Data Generation for Intelligent Inspection of Structural Components*. PhD thesis, The University of Texas at El Paso, 2022.

[99] Mst Mousumi Rizia, Julio A. Reyes-Munoz, Angel G. Ortega, Ahsan Choudhuri, and Angel Flores-Abad. Autonomous aerial flight path inspection using advanced manufacturing techniques. *Robotica*, 40(7):2128–2151, 2022.

[100] Eric Bianchi and Matthew Hebdon. COCO-Bridge 2021+ Dataset. 10 2021.

[101] Sardar Kashif Ur Rehman, Zainah Ibrahim, Shazim Ali Memon, and Mohammed Jameel. Nondestructive test methods for concrete bridges: A review. *Construction and building materials*, 107:58–86, 2016.

[102] GE Rehkugler and JA Throop. Apple sorting with machine vision. *Transactions of the ASAE*, 29(5):1388–1397, 1986.

[103] BK Miller and MJ Delwiche. Peach defect detection with machine vision. *Transactions of the ASAE*, 34(6):2588–2597, 1991.

[104] V Alchanatis, K Peleg, and M Ziv. Classification of tissue culture segments by colour machine vision. *Journal of Agricultural Engineering Research*, 55(4):299–311, 1993.

[105] Nuria Aleixos, José Blasco, F Navarron, and Enrique Moltó. Multispectral inspection of citrus in real-time using machine vision and digital signal processors. *Computers and electronics in agriculture*, 33(2):121–137, 2002.

[106] Yuanxun Zheng, Shaoqiang Wang, Peng Zhang, Tongxin Xu, and Jingbo Zhuo. Application of nondestructive testing technology in quality evaluation of plain concrete and rc structures in bridge engineering: A review. *Buildings*, 12(6):843, 2022.

[107] D Mader, R Blaskow, P Westfeld, and C Weller. Potential of uav-based laser scanner and multispectral camera data in building inspection. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 41:1135, 2016.

[108] John M Hart, Esther Resendiz, Benjamin Freid, Steven Sawadisavi, CPL Barkan, and N Ahuja. Machine vision using multi-spectral imaging for undercarriage inspection of railroad equipment. In *Proceedings of the 8th world congress on railway research, Seoul, Korea*, volume 18, 2008.

[109] Tarek Omar and Moncef L Nehdi. Application of passive infrared thermography for the detection of defects in concrete bridge elements. In *Proceedings of the International Conference of the Transporation Association of Canada, Toronto, ON, Canada*, pages 22–28, 2016.

[110] Azusa Watase, Recep Birgul, Shuhei Hiasa, Masato Matsumoto, Koji Mitani, and F Necati Catbas. Practical identification of favorable time windows for infrared thermography for concrete bridge evaluation. *Construction and Building Materials*, 101:1016–1030, 2015.

[111] Eva Barreira, Ricardo MSF Almeida, and Maria L Simões. Emissivity of building materials for infrared measurements. *Sensors*, 21(6):1961, 2021.

[112] Vladimir P Vavilov and Arseny O Chulkov. Detecting corrosion in thick metals by applying active ir thermography. In *Thermosense: Thermal Infrared Applications XXXIV*, volume 8354, pages 38–46. SPIE, 2012.

[113] Siavash Doshvarpassand, Changzhi Wu, and Xiangyu Wang. An overview of corrosion defect characterization using active infrared thermography. *Infrared physics & technology*, 96:366–389, 2019.

[114] n.d. What calibration terms are applied in the camera. Retrieved from https://www.flir.com/support-center/oem/what-calibration-terms-are-applied-in-the-camera-there-is-the-ffc-and-also-the-gain-calibration.-are-there-others-can-i-do-my-own-calibration. Accessed: 2022-10-19.

Curriculum Vitae

Mst Mousumi Rizia received her Bachelor's and Master of Science in Mechanical Engineering at the Military Institute of Science and Technology (MIST), Bangladesh. She was awarded the 'Institute Medal' for academic recognition. She is currently a Ph.D. student in the department of Aerospace and Mechanical Engineering at The University of Texas at El Paso (UTEP).

Mousumi is a retired Squadron Leader of the Bangladesh Air Force (BAF), where she served as an Aviation Maintenance and Operations Engineer for over a decade. During her military career, she was in charge of the maintenance and operations of different military aviation units. She supported the BAF 'Air Support Units' for UN missions in DR Congo, Mali, and Haiti. She is the recipient of the coveted 'Chief of Air Staff's Trophy'. She performed instructional duties at the BAF Academy from 2013-2014, and later, she worked as an Assistant Professor and Program Coordinator for the Mechanical Engineering Department at MIST.

At UTEP, Mousumi worked as a Research Associate at the NASA MIRO Aerospace Center (cSETR) and independently taught courses at the Aerospace and Mechanical Engineering department. As an RA, she has worked on a project for autonomous aerial inspection in GPS-Denied Environments supported by the National Energy Technology Laboratory (NETL) of the U.S. Department of Energy (DOE). She has also worked extensively in developing novel infrastructure inspection methods leveraging Advanced Manufacturing Techniques and Deep Neural Networks (DNNs) of Artificial Intelligence (AI). She has authored several journal publications and presented her research work at different international conferences.

Contact information: mrizia@miners.utep.edu, riziamm9@gmail.com