

CRANFIELD UNIVERSITY

NDIDIAMAKA ADIUKU

OPTIMISED OBSTACLE DETECTION AND AVOIDANCE MODEL
FOR AUTONOMOUS VEHICLE NAVIGATION

SCHOOL OF AEROSPACE, TRANSPORT AND
MANUFACTURING

Integrated Vehicle Health Management Centre

Doctor of Philosophy

Academic Year: 2021 – 2024

Supervisor: Nico Avdelidis
Associate Supervisor: Gilbert Tang
Feb. 2024

CRANFIELD UNIVERSITY

SCHOOL OF AEROSPACE, TRANSPORT, AND
MANUFACTURING.

Doctor of Philosophy

Academic Year: 2021- 2024

NDIDIAMAKA ADIUKU

Optimised Obstacle Detection and Avoidance Model For
Autonomous Vehicle Navigation

Supervisor: Nico Avdelidis
Associate Supervisor: Gilbert Tang
Feb 2024

This thesis is submitted in partial fulfilment of the requirements for
the degree of Doctor of Philosophy at Cranfield University.

© Cranfield University 2024 . All rights reserved. No part of this
publication may be reproduced without the written permission of the
copyright owner.

ABSTRACT

Driven by cutting-edge research in AI vision, sensor fusion and autonomous systems, intelligent robotics is poised to revolutionise Aviation hangars, shaping the "hangars of the future" by reducing inspection time and improving defect detection accuracy. Many hangar environments, especially in maintenance, repair and overhaul (MRO) operations, rely on manual processes and algorithms that need to be optimised for the increasing complexity of these settings. These include varied obstacle structures, often low-light conditions, and frequent changes in the scene.

The application of mobile robot solutions demands enhanced perception, accurate obstacle avoidance, and efficient path planning, essential for effective navigation in the busy hangar environment and aircraft inspections. The application of ROS navigation stack has been at the center of most solutions and is mostly efficient in static settings while limited in complex environments. These systems are often computationally intensive and require pre-configuration of environmental parameters, making them less efficient in changing environments with real-time demand. Deep learning models and ROS integration have shown promising improvements, leveraging experiential learning and large datasets. However, accurately detecting obstacles of different shapes and sizes, especially in varying lighting conditions, poses a significant challenge and affects safe navigation.

To overcome these challenges in complex environments, this research proposes a novel solution for enhanced obstacle detection, avoidance and path planning. Our system leverages LiDAR and camera data fusion with a real-time and accurate YOLOv7/YOLOv5 object detection model for robust identification of diverse obstacles. Additionally, we proposed a combination with ROS planners, including Dijkstra, RRT and DWA, for path planning optimisation to enable collision-free navigation. The system was validated with ROS Gazebo and real-Turtlebot3 robot. It achieved zero collisions with YOLOv7 and RRT integration, a 2.7% increase in obstacle detection accuracy, and an estimated 2.4% faster navigation speed than the baseline methods.

Keywords:

Autonomous navigation, object detection, obstacle avoidance, mobile robot, deep learning, computer vision, ROS navigation

I am dedicating this work to God Almighty and to my best friend and husband Engr. Nwabueze Adiuku for his immense support and belief in me.

Also to my children Mandy, Freddie and Chimzy.

ACKNOWLEDGEMENTS

Embarking on this PhD journey has been one of the most transformative experiences of my life. While challenging, it has been rewarding, especially now with the finish line in sight, thanks to the support of many remarkable people who came my way.

First, I am deeply grateful to my supervisors, Professor Nico Avdelidis and Dr Gilbert Tang, for providing the platform and their unwavering support. I appreciate their insightful guidance and willingness to always share their wealth of knowledge and experiences, and this helped me shape my research and as well push me to strive for excellence. I extend my sincere appreciation to my first supervisors, Professor Suresh Perinpanayagam and Dr Joni Pelham, for giving me the platform to start, with great support and encouragement, especially at the beginning of the program.

Special thanks go to Angelos Plastropoulos, whose friendship and academic support have been invaluable. He continuously provides opportunities for learning and development, enriching my academic journey.

The journey wouldn't have been successful without the support of my friends and colleagues. Desmond Bisandu and Maren immensely helped me to kickstart my research, while Abiola and Angela Okeke were always there, always encouraging me during the challenging times. I am profoundly grateful to them all. To the "When Women Pray" prayer group and everyone who played a part in this achievement. Thank you for your collective prayers and support.

My deepest gratitude goes to my family, especially my mother and my children, Mandy, Freddie and Chimzy. Their endless love, patience, understanding and prayers fuelled my spirit throughout this endeavour.

My heartfelt thanks go to my husband, Engr. Nwabueze Adiuku. He has been my inspiration and the cornerstone of my academic journey. His sacrifices – of his career, time and finances, have made this achievement possible. Words cannot

express my appreciation for his love, care and generosity, which has been the bedrock of my success.

Finally, I offer my sincere gratitude to God Almighty for providing me with the strength, wisdom and blessings that sustained me through this three-year journey. I am forever grateful for your guidance and grace.

TABLE OF CONTENTS

ABSTRACT	i
ACKNOWLEDGEMENTS.....	iv
TABLE OF CONTENTS	vi
LIST OF FIGURES.....	ix
LIST OF TABLES	xi
LIST OF ABBREVIATIONS.....	xii
LIST OF EQUATIONS.....	xiv
1 Background	17
1.1 Motivation	18
1.1.1 The aircraft inspection problem:	18
1.1.2 The environment perception problem.....	19
1.1.3 The Obstacle Detection Problem	19
1.1.4 The obstacle avoidance and path planning problem	19
1.2 Aim, Gaps and Objectives	20
1.2.1 Research Gaps	20
1.2.2 Objectives	21
1.3 Methodology	22
1.3.1 Phase 1: Literature review.....	22
1.3.2 Phase 2: Simulation, Data capturing and Pre-processing.	22
1.3.3 Phase 3: Model Development & Integration.	24
1.3.4 Phase 4: Model transfer, validation, and Evaluation	25
1.3.5 Phase 5: Conclusion and thesis completion.....	26
1.4 Thesis Structure.....	27
CNN-fusion architecture with visual and thermal images for object detection	28
Part 1: LITERATURE REVIEW	29
2 Advancements in Learning-Based Navigation Systems for Robotic Applications in MRO Hangar: Review	30
2.1 Introduction	31
2.2 Related Work	34
2.2.1 Object Detection Models applicable to robot navigation.....	35
2.2.2 Reinforcement learning-based Models used in robot navigation.	36
2.2.3 Navigation with Hybrid Models	37
2.3 Concept and Background	38
2.3.1 MRO Hangar in Aviation.....	38
2.3.2 Intelligent Robotic in MRO Hangar	41
2.3.3 Robotic Navigation	42
2.3.4 Vision Sensors	43
2.3.5 Obstacle detection	45

2.3.6 Obstacle Avoidance	45
2.3.7 Path Planning	46
2.4 Learning-based navigation techniques	47
2.4.1 Deep Reinforcement Learning (DRL)	48
2.4.2 Object detection model.....	50
2.4.3 Hybridization with Neural Network	53
2.5 Drawbacks and Future Work	55
2.6 Conclusion	58
PART 2: DEVELOPMENT OF DEEP LEARNING BASED OBSTACLE DETECTION MODEL	59
3 CNN-FUSION ARCHITECTURE WITH VISUAL AND THERMOGRAPHIC IMAGES FOR OBJECT DETECTION	60
3.1 INTRODUCTION	61
3.2 Related Work	63
3.3 IMAGE ENHANCEMENT FOR ROBOTS OBJECT DETECTION	65
3.3.1 Overview Of Yolov5 Algorithm	65
3.3.2 Proposed FUSE-YOLO Object Detection Model	66
3.4 FUSE-YOLO ARCHITECTURE	74
3.4.1 Data Preparation	75
3.4.2 Relevance of the datasets.....	76
3.5 EXPERIMENT AND RESULT	77
3.5.1 Model Training	77
3.5.2 Evaluation Metrics.....	79
3.5.3 Results	80
3.6 DISCUSSION	81
PART 3: INTEGRATION OF YOLO MODEL INTO ROS NAVIGATION STACK	83
4 Mobile Robot Obstacle Detection and Avoidance with NAV-YOLO.....	84
4.1 Introduction	84
4.2 Related Work	88
4.3 Methodology	89
4.3.1 YOLOV5.....	90
4.3.2 Ros Navigation Stack	91
4.3.3 The Proposed Method.....	93
4.3.4 Obstacle Avoidance	96
4.4 Experiment.....	97
4.5 Result and Discussion	100
4.5.1 Result.....	100
4.5.2 Discussion.....	102
4.5.3 Conclusion	103
5 Improved hybrid model for obstacle detection and Avoidance in ROS framework (RRT and DWA).....	105

5.1 Introduction	106
5.2 Related Work	108
5.3 Methodology	110
5.3.1 CNN-based YOLOv7 Model	110
5.3.2 Sensor Fusion	113
5.3.3 System Architecture	114
5.4 Experiment.....	121
5.4.1 Data Collection	123
5.4.2 Result	124
5.4.3 Obstacle Detection Performance	126
5.4.4 Path Planning Evaluation	128
5.4.5 Transfer to physical Turtlebot3 waffle_pi.....	128
5.4.6 Discussion.....	130
5.5 Conclusion	131
6 Conclusion & Future Work.....	132
6.1 Discussion	132
6.2 Addressing the research aim and objectives.	132
6.3 Contributions to knowledge.....	136
6.4 Future Work	137
6.4.1 Focus on Lightweight model structure.....	137
6.4.2 Generating High Fidelity data from sensor perception	138
6.4.3 Real-Time update.....	138
6.4.4 Sensor Fusion Enhancement.....	138
6.4.5 Enhancement of the path planning algorithms (ROS Navigation Update)	139
6.4.6 Hardware considerations.....	139
7 Appendices.....	141
References	147

LIST OF FIGURES

Figure 1-1: Unstructured busy hangar.....	16
Figure 1-2: Low-light condition in busy hangar.....	16
Figure 1-3: Developed ROS-Gazebo simulation environment.....	21
Figure 1-4: Data generation process.....	21
Figure 1-5: Custom Datasets.....	22
Figure 1-6: LLVIP datasets.....	22
Figure 1-7: Methodology flow diagram.....	24
Figure 2-1: Overview of the learning-based Architecture in this survey.....	32
Figure 2-2: Simulated Cranfield University MRO hangar.....	36
Figure 2-3: Industry 4.0 technologies	38
Figure 2-4: Structure of Robotic Intelligence.....	39
Figure 2-5: Learning-based Path planning framework.....	41
Figure 2-6: Overview of interaction between agent and environment.....	45
Figure 3-1. The Fuse-YOLO architecture.....	63
Figure 3-2: Synthetic Datasets	
Figure 3-3: RGB images from TUI hangar	
Figure 3-4: RGB and Thermal images Pairs.	
Figure 3-5: LLVIP dataset	
Figure 3-6. Fused RGB and thermal images with DenseFuse	
Figure 3-7. The structure of CBAM.....	64
Figure 3-8. Performance comparison of the yolov5 models and Fuse-YOLO....	68
Figure 3-9. Object detection results on the model test using RGB Thermal image, thermal and RGB image.....	68
Figure 4-1: Integrated YOLO and ROS navigation framework	74
Figure 4-2: YOLO object detection to scan data conversion workflow.....	80

Figure 4-3: Robot obstacle detection in a hangar environment.....	83
Figure 4-4: Real-world object detection test result.....	83
Figure 4-5: Algorithms in Proposed and Baseline solution	
Figure 4-6: NAV-YOLO Trajectory.....	85
Figure 4-7: ROS NAV Trajectory.....	86
Figure 5-1: NAV-YOLO-RRT system architecture.....	97
Figure 5-2: Detected objects transformation workflow.....	98
Figure 5-3: Object detection and transformation to laser scan workflow.....	100
Figure 5-4: Navigation operation of integrated ROS path planner and YOLO object detection.....	101
Figure 5-5: Compared NAV-YOLO methods	103
Figure 5-6: Comparison analysis of NAV-YOLO and NAV-YOLO-RRT trajectory with baseline.....	105
Figure 5-7: Observed lag with NAV-YOLO in obstacle detection.....	102
Figure 5-8: Improved Obstacle detection with NAV-YOLO-RRT.....	103
Figure 5-9: Detected and estimated obstacle position.....	108
Figure 5-10: NAV-YOLO-RRT detection and path planning in real-world robot.....	109

LIST OF TABLES

Table 1-1 List of publications.....	Error! Bookmark not defined.	6
Table 2-1Comparing traditional and intelligent robotics applications in MRO hangar		37
Table 2-2 Comparison of papers on DRL-based robot navigation systems.....		47
Table 2-3 Recent solution with integrated ROS and Object detection models...		51
Table 2-4Strengths and limitations of hybrid systems solutions in obstacle avoidance and path planning.....		52
Table 3-1 Result Comparison between YOLOv5 models and fuse-YOLO on RGB_thermal		66
Table 3-2 Comparative performance of YOLOv5m and Fuse-YOLO on four datasets based on (mAP) _{0.5}		67
TABLE 4-1 Comparing Time and Distance Covered by NAV-YOLO and ROS DWA.....		84
TABLE 4-2 Comparing Obstacles Avoided by NAV-YOLO and ROS DWA in Different Environments.....		84
TABLE 4-3 Comparing the Accuracy of NAV-YOLO with YOLOV5 Variants.....		84
Table 5-1 Performance evaluation of developed solutions with baseline.....		105
Table 5-2 Performance results of YOLOv7 and YOLOv5.....		106
Table 6-1 Defined checkpoints for robot navigation.....		116

LIST OF ABBREVIATIONS

A*	A Star
AE	Auto Encoder
AMCL	Adaptive Monte Carlo localisation
AMR	Autonomous Mobile Robot
AP	Average precision
AV	Autonomous Vehicle
CAM	Channel Attention Module
CBAM	Convolutional Block Attention Module
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CSP	Cross-Stage-Partial-Connections
CSV	Comma Separated Value
CUDA	Compute Unified Device Architecture
CVAE	Conditional Variational Auto-Encoder
DARTeC	Digital Aviation Research and Technology Centre
DL	Deep Learning
DNN	Deep Neural Network
DRL	Deep Reinforcement Learning
DWA	Dynamic Windows Approach
DQN	Deep Q-Network
DDPG	Deep Deterministic Policy Gradient
DPPO	Deterministic Policy Gradient Algorithms
FLOP	Floating Point Operations
FN	False Negative
FOV	Field of View
FP	False Positive
FPS	Frame per Second
GPU	Graphics Processing Unit
ICCMA	International Conference on Control, Mechatronics and Automation.
IJMERR	International Journal of Mechanical Engineering and Robotics Research
IORT	Internet of Robotic Things
IOT	Internet of Things
KITTI	Karlsruhe Institute of Technology and Toyota Technological Institute

LiDAR	Light detection and ranging
LLVIP	Low-light Visible-infrared Paired
LSTM	Long Short-Term Memory
ML	Machine Learning
MLP	Multi-Layer Perceptron
MPC	Model predictive control.
MRO	Maintenance, Overhaul, Repair
MS	Microsoft
NN	Neural Network
PANet	Path Aggregation Network
PP	Pure pursuit
PPO	Proximal Policy Optimization
R-CNN	Region convolutional neural network
RRT	Rapidly exploring Random Tree
RGB	Red, Green, Blue
RGB-D	Red, Green, Blue and Depth
ROI	Region Of Interest
RL	Reinforcement learning
ROS	Robot Operating System
RPN	Region Proposal Network
RVIZ	ROS Visualization
SAM	Spatial Attention Module
SLAM	Simultaneous Localization and Mapping
SPIE	Society of Photo-Optical Instrumentation Engineers
SSD	Single Shot MultiBox Detector
SVM	Support Vector Machine
TED	Time Elastic Band
TP	True Positive
TRPO	Trust Region Policy Optimization Central Processing Unit
UV	Unmanned Vehicles
YAML	Yet Another Markup Language
YOLO	You Look Only Once

LIST OF EQUATIONS

(3-1)..... 66

(3-2).....66

(3-3).....66

(3-4).....66

(3-5).....67

(4-1).....80

(4-2).....80

(4-3).....81

(5-1).....98

(5-2).....98

(5-3).....98

(5-4).....99

(5-5).....99

(5-6).....99

(5-7).....99

(5-8).....100

1 Background

The MRO hangar is an essential domain in the aviation sector with diverse processes that ensure the safety and reliability of the aircraft MRO operations. Aircraft visual inspection is among the core tasks involving mostly manual navigation through different checkpoints to detect and evaluate any defects around the exterior path of the aircraft. Integrating advanced technologies including robotics and artificial intelligence has improved operational efficiency with automated and autonomous solutions [1]. With this, Intelligent mobile robots can autonomously navigate and interact with such busy hangar environments to perform collision-free aircraft inspection, leading to more effective and efficient maintenance practices. AMR-safe navigation is a critical factor in recent research due to the complexity and dynamism of real-world structures.

This thesis focuses on techniques that improve a robot's environment perception, obstacle detection, avoidance and path planning for safe navigation in complex environments. The mobile robot must be equipped with advanced sensors for accurate interpretation and response to their surroundings [2]. LiDAR and camera sensors were employed, adopting a fusion strategy to enhance and expand robots' perception of their environments, overcoming the limitations of individual sensors [3]. The ROS framework is central to most robotic applications and has a structural platform that integrates and fuse sensor information with relevant algorithms to facilitate effective navigation in different environments [4]. The path planning algorithms (RRT and DWA) are leveraged with sensor streams to optimise obstacle avoidance and path planning mechanisms. Looking at the complex structure of the MRO hangars with dynamic and diverse obstacle types, accurate obstacle detection and classification can be challenging, limiting safe navigation. This facilitates the need for expanded functionalities leveraging accurate obstacle detection capabilities of the YOLO model. The combined system is a hybrid solution that integrates conventional ROS RRT and DWA path planners, multisensory data, and a YOLO object detection model for advancing the capabilities of autonomous mobile robots required for autonomous aircraft inspection.

1.1 Motivation

Despite the increasing focus on digitalisation in hangar MRO operations to circumvent complex and dynamic system challenges, most aircraft hangars still need to engage in manual processes and outdated solutions. The hangar environment depicted in Figures 1-1 and 1-2 illustrates the complexity that can arise in busy seasons. The environments are often cluttered with diverse tools and equipment, which vary in shape and colour, adding to these settings' navigational and operational challenges. The low-light condition shown in Figure 1-2 can be very challenging when accessing occluded or inaccessible areas. While there has been a notable rise to address significant challenges in these environments, the embedded functionalities in these systems still need to be improved. These gaps in performance and capability form the foundational motivation for this research project, aiming to enhance the existing robotic systems to address the complexities and dynamic nature of these environments.



Figure 1-1: Unstructured busy hangar.



Figure 1-2: Low-light condition in hangar

1.1.1 The aircraft inspection problem:

Aircraft inspection involves a systematic process where inspectors move from one checkpoint to another within the aircraft to identify defects, diagnose issues, and certify that the aircraft is fit. While widely used, manual inspection is prone to errors and can be time-consuming and costly. Additionally, environmental factors contribute to more complexities and uncertainties in the scenario, further complicating the inspection task and environmental safety [5].

1.1.2 The environment perception problem

While there is a growing trend in developing robotic platforms to automate the inspection process in busy aircraft hangars, many of these robots need to be equipped with more sensor capabilities. Some rely on individual sensors like cameras that can generate rich environmental information but have a limited field of view [6]. LiDAR sensors, on the other hand, offer more comprehensive coverage but may be less effective in perceiving detailed environmental features. These gaps in sensor functionalities are of critical concern, as they directly impact robot's perception of their surroundings, increasing safety issues during navigation for an inspection task.

1.1.3 The Obstacle Detection Problem

In aircraft hangars, mobile robots are tasked with identifying and interacting with various obstacles and navigating through complex aircraft geometries to conduct inspection procedures. There are situations of low light conditions which affect the robot's ability to detect and navigate around dynamic structures [7]. The diverse nature of these obstacles in terms of different shapes, sizes, colours, and intensities, as shown in Figures 1-1 and 1-2, requires robust and versatile methodologies to accurately detect and adapt to the changing structures [8]. Many robotic solutions are based on mathematical computations and preconfigured parameters relative to different environmental states [9]. This approach can be limiting in dynamic and unknown settings, making it difficult for the robot to detect and respond to unpredictable obstacles not anticipated in their initial program.

1.1.4 The obstacle avoidance and path planning problem

The mobile robot systems are designed and programmed to execute diverse tasks, including safe navigation necessary for aircraft inspection. MRO hangars' dynamic nature and increasing complexities challenge conventional path-planning algorithms. The robot is required to respond swiftly to moving obstacles and unforeseen changes, making prompt control to avoid the barriers and generate a feasible path to the target location. Conventional path planning

algorithms like the ROS Dijkstra, RRT global, and DWA local planners are limited in intelligence [10] and struggle to dynamically avoid and adjust planned routes in complex real-time scenarios [11]. The reliability and adaptability of such robotic solutions in real-world busy hangars are minimal.

1.2 Aim, Gaps and Objectives

Aim

To develop an improved vision-based obstacle detection and avoidance model using a combined custom YOLOv5 and ROS navigation stack for mobile robot safe navigation in unstructured and changing environments.

1.2.1 Research Gaps

This comprehensive review identified several research gaps, which serve as the foundation for this research.

Research Gap 1: The increasingly complex and constant changes in the busy hangar environment challenge the efficiency of existing robotics navigation solutions. This impacts the robot's ability to make real-time control decisions required for safe path planning during an inspection task.

Research Gap 2: Identifying obstacles of diverse structures under different lighting conditions remains challenging with existing algorithms. This limits the ability of the robots to accurately recognise obstacles and respond promptly for avoidance, which is crucial for a safe navigation experience in dynamic environments like aircraft hangars.

Research Gap 3: Many robotic systems rely on sensors like LiDARs and cameras to perceive and interact more naturally with the world. While LiDAR and similar sensors offer 360-degree field of view, they have limitations in acquiring detailed environmental features such as colour and texture, relevant for safe and efficient navigation. On the other hand, cameras can provide rich visual environmental information about the environment but are constrained to perceive objects within a certain range and angle. The use of individual sensor can limit

the robot's navigation safety and efficiency and not robust for real-world applications.

1.2.2 Objectives

The objective is generally to review existing work and find gaps that can be improved to establish an optimised mobile robot solution suitable for aircraft inspection in busy hangar environments. This is detailed as follows.

Objective 1:

- A literature review that examined studies that detailed mobile robots' use in real-world complex environments focused on aircraft visual inspection in aviation hangars.
- Identified the main challenges faced in this context, such as identifying obstacles of varied nature, varying lighting conditions, and manoeuvring environmental factors to achieve safe navigation for an inspection task.
- Explored gaps in ROS obstacle avoidance path planning algorithms, deep learning models specific to complex and dynamic environments, and available improvement techniques to address these gaps.

Objective 2

- Set up ROS and Gazebo simulation with turtlebot3 robot package integrating vision-based sensors (RGB-D and LiDAR) with incremental complexity for testing robotic application, as widely applied in robotic research.
- Create a realistic DARTeC hangar model in a Gazebo simulation environment to generate data and validate performance, replicating real-world operations.
- Combine and Preprocessing datasets according to deep learning standards and suitable YOLO format, enhancing detection accuracy.

Objective 3

- Developed a customized YOLOv5 object detection framework incorporating DenseFuse and CBAM to achieve 70-80% improved detection accuracy compared to baseline model.

Objective 4

- Develop an improved NAV-YOLO solution with YOLOv5 object detection model and ROS navigation stack to enhance real-time obstacle detection and avoidance, 10-20% collision reduction in complex settings.

Objective 5

- Evaluate and compare NAV-YOLO performance with baseline ROS navigation stack planners. Target 5-20% performance improvement.
- Establish simulation to real world transfer to physical robot.

1.3 Methodology

This research work was achieved in 5 phases, as illustrated in Figure 1-7.

1.3.1 Phase 1: Literature review.

The literature review in this study focused on exploring the enabling technologies of Industry 4.0, particularly robotics and Artificial Intelligence (AI) algorithms that are pertinent to the safe navigation of mobile robots in complex environments.

1.3.2 Phase 2: Simulation, Data capturing and Pre-processing.

Simulation

The simulation environment was set up with ROS noetic, Gazebo 11, PyTorch, OpenCV libraries, and a Turtlebot3 waffle-pi robot package equipped with LiDAR and OAK-D Camera packages on Ubuntu 20.04. A hangar-model environment was developed and integrated, as shown in Figures 2-2, with relevant ROS nodes connected to improve obstacle detection and avoidance experience. An overview of the hangar launch file was shown in Appendix E.

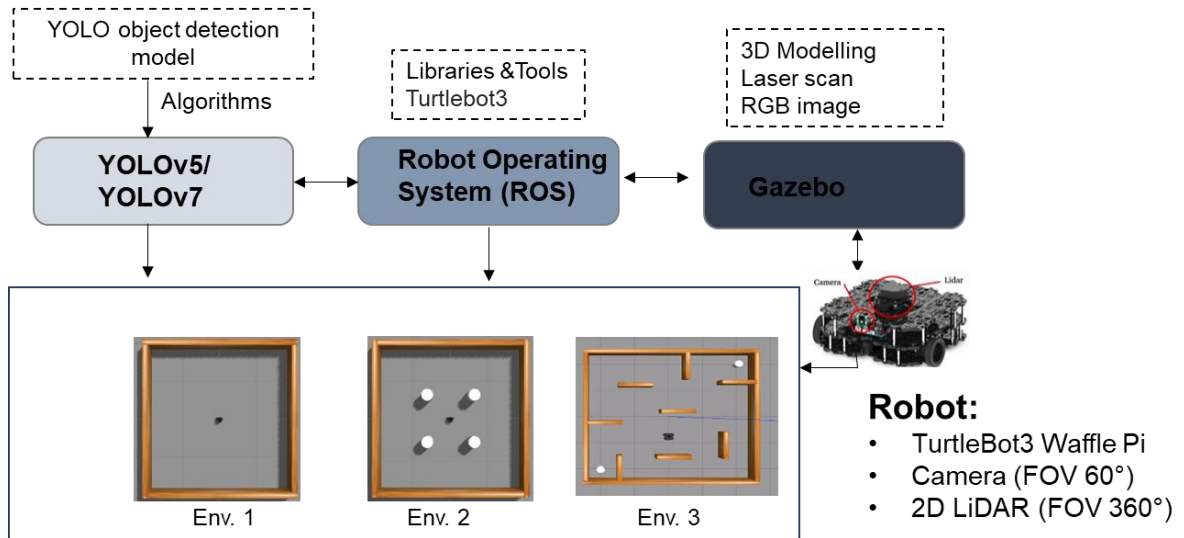


Figure 1-3: Developed ROS-Gazebo simulation environment.

Data Capturing

The Image and video data were captured using camera and OpenCV libraries from the simulation environment as described in Figure 1-4, TUI hangar using RGB and thermal cameras, and integration of LLVIP public datasets to expand the

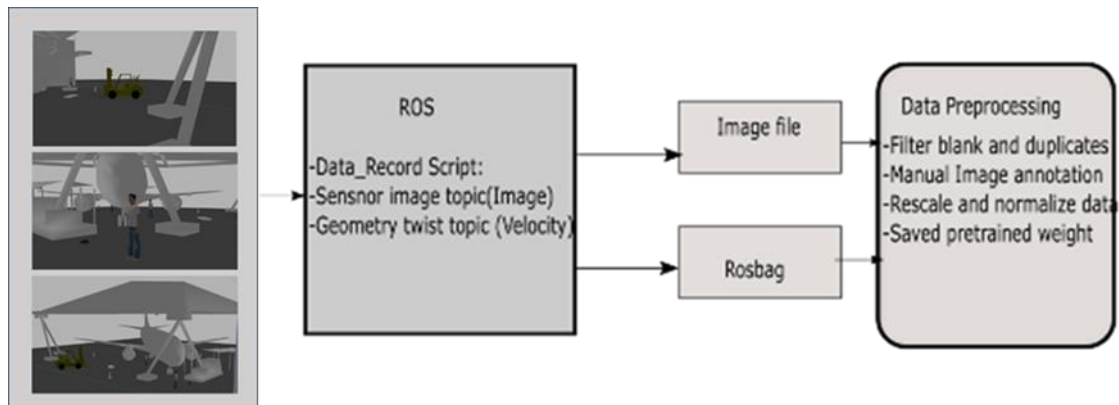


Figure 1-4: Data generation process.

Model capability. More experimental Data was generated using ROSBAG for performance evaluations. The dataset image representation is shown in Figure 1-5 and 1-6, and the description is as follows.

- MS COCO – Default pre-train YOLOv5 (328,000 images)

- LLVIP Generated fro Dataset – public datasets (15,488) [12]
- TUI hangar dataset – Generated from TUI hangar (365 image pairs) [13]
- Synthetic datasets (generated from simulation) – 280 images.

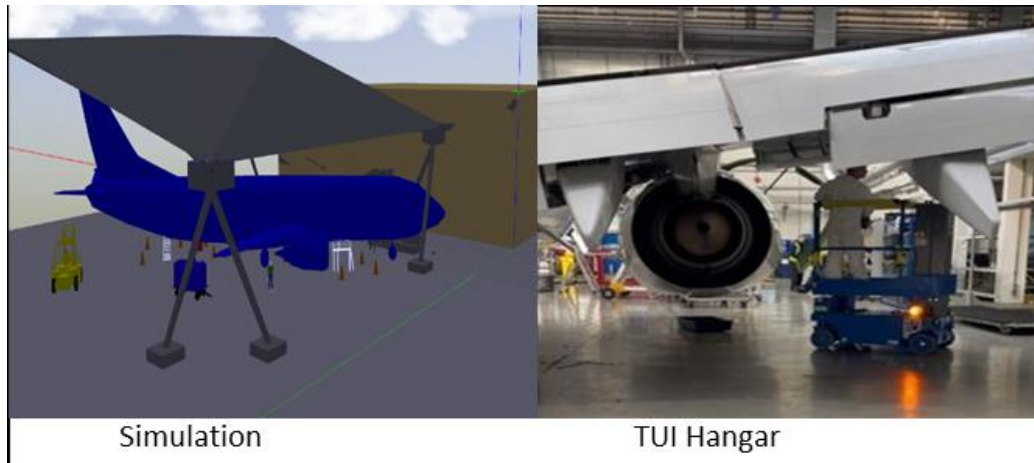


Figure 1-5: Custom Datasets



Figure 1-6: LLVIP datasets

Data pre-processing

The data was filtered to remove duplicate and blank data to maintain consistency. The YOLO data labelling format was adopted, with each image associated with a corresponding data file that contains bounding box annotations for each object within the image.

1.3.3 Phase 3: Model Development & Integration.

The NAV-YOLO framework combines the capability of the YOLOv5 real-time object detection algorithm with the ROS navigation stack integrated with fuse

sensor data to enable the robots to detect obstacles in complex environments and use this information to improve navigation experience in an inspection task.

Object detection module.

This addresses research gap two and part 3, where many datasets from relevant environments and low-light conditions were combined to customise and train the YOLO obstacle detection model. This was developed in the obstacle detection module, FUSE-YOLO [14] with YOLOv5 model. The classic YOLOv5 was pre-trained with COCO datasets using the PyTorch platform. The trained YOLO file consists of labelled datasets, YOLOv5 weights and a YAML file with parameters and settings that can be tuned to adapt the model to perform well on the navigation task. YOLOv7 was also integrated for more performance checks, as shown in Table 5.

Object Avoidance module

The module focused on research gaps 1 and 3 by developing NAV-YOLO and improving NAV-YOLO models to address inconsistencies in obstacle avoidance and path planning functionality in recent robot navigation applications. This module uses the transformed information that the obstacle detection module provided to decide how to avoid obstacles from the starting to the target location. Considering the use case, which is aircraft visual inspection, 5 different checkpoints were defined in the environment configuration and the received transformed information from the obstacle detection module is used to improve ROS planners ability to generate a feasible and safe route. Figures 4-1 and 5-1 present the NAV-YOLO framework structure. It employs the ROS algorithm for path planning, including Dijkstra, RRT global planners, and DWA local planners, to navigate free spaces and establish safe navigation operations in complex settings.

1.3.4 Phase 4: Model transfer, validation, and Evaluation

In this study phase, the improved NAV-YOLO solution undergoes evaluation to validate its proof of concept. For quality assurance, data from the TUI hangar was used and enhanced with other datasets. The proposed solution was validated in

simulation and physical turtlebot3 robot. The Turtlebot3 robot was setup with OAK-D camera shown in Appendix A. The solution is iteratively evaluated and compared with the capabilities of classic Dijkstra, RRT, DWA planners, and YOLOv5/YOLOv7 models.

1.3.5 Phase 5: Conclusion and thesis completion

In this concluding phase of the study, areas for future research were identified and highlighted, providing direction for future exploration. The conclusion was drawn from the findings presented in journals and the thesis, laying the groundwork for ongoing advancements in this area of robotics.

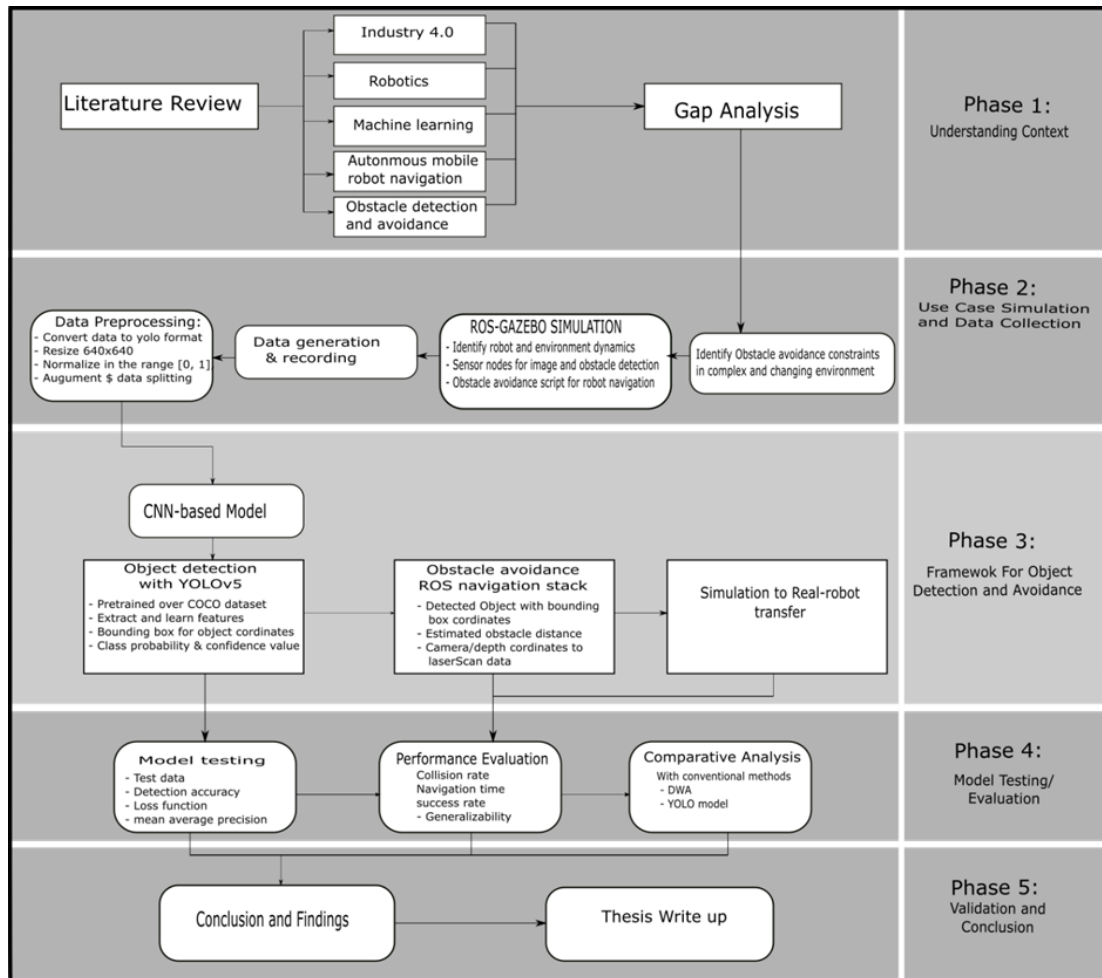


Figure 1-7: Methodology flow diagram

1.4 Thesis Structure

This thesis adopts a paper-based format and consists of six chapters. It is structured in three parts, part 1 covers the literature review, in part two, the details of deep learning model centred in chapter three was outlined. Part three involves chapter four and five. Looking at the chapter structures, chapter one covers the project's background and describes critical objectives and contributions. Chapter Two detailed the literature review, emphasising the notable impact of improved obstacle avoidance and path planning systems in hangar environment applications based on related work and applications in similar environments. Chapter three discussed the YOLOv5 model improvement with custom and public datasets and enhancement with an attention mechanism. A combined YOLOv5 and default ROS navigation stack model was developed in Chapter 4 to facilitate accurate obstacle detection in complex scenarios for a navigation task. Chapter 5 explained improvement strategies integrating YOLOv7 and the RRT path planning algorithm. The final part of the thesis provides a general discussion of the different phases of the research work and alignment with the research objectives, drawing conclusions and discussing potential research direction. The chapters represent distinct phases in the research methodology, as Figures 1-7 demonstrate, aligning with the objectives and ensuring a structural flow of information throughout the papers. The papers in alignment with the objectives are listed in Table 1-1.

Table 1-1: List of publications.

Chapters	Paper Title	Publications	Status
Chapter 2	Advancements in Learning-Based Navigation Systems for Robotic Applications in MRO Hangar: Review"	MDPI Journal.	Published
Chapter 3	CNN-fusion architecture with visual and thermal images for object detection	Presented In SPIE ThermoSense Conference.	Published.
Chapter 4	Mobile Robot Obstacle Detection and Avoidance with NAV-YOLO	Presented in ICCMA Conference. Published in IJMERR Journal.	Published.
Chapter 5	Improved Hybrid Model for Obstacle Detection and Avoidance in Robot Operating System Framework (Rapidly Exploring Random Tree and Dynamic Windows Approach)	MDPI Journal	Published.

Part 1: LITERATURE REVIEW

Robot navigation and real world complexities and uncertainties were considered in finding the research gaps. The use case in this project is to enhance aircraft visual inspection in complex MRO hangars.

Key considerations include:

- Unstructured and changing environments.
- Obstacles of varying features.
- Difficulty in low light situations.
- Difficulty in obstacle detection and avoidance.
- Safe navigation.
- Existing ROS planners and Deep learning capabilities.
- Real-time and adaptability demand.

2 Advancements in Learning-Based Navigation Systems for Robotic Applications in MRO Hangar: Review

This research paper focused on the growing interest in learning-based navigation for mobile robots, particularly in the context of visual aircraft inspections in MRO hangars. It highlighted the need for efficient perception and obstacle avoidance in these dynamic environments. The paper reviewed conventional methods that use mathematical computations for local and global planning, noting their limitations in complex, dynamic settings. It emphasised advancements in machine learning and intense learning for enhancing navigational capabilities. The review also covered learning-based strategies for environment perception, obstacle detection, avoidance and path planning using fused vision-based sensors (LiDAR and cameras). The paper concludes by discussing current challenges and future directions in this field of learning-based navigation for mobile robots, which is crucial for understanding the evolving landscape of robotic navigation and the potential pathways for enhancements in the domain.

Paper published on: <https://www.mdpi.com/1424-8220/24/5/1377#:~:text=The%20review%20categorised%20algorithms%20based,applications%20in%20real%2Dworld%20scenarios>.

2.1 Introduction

In recent years, the aviation sector has made significant strides in the periodic inspection and maintenance of aircraft, aiming to keep pace with the increasing global air traffic demand. This focus is driven by a commitment to safety and the goal to reduce operational costs, which currently represent 10 -15% of airlines' operating costs and are projected to rise from \$67.6 billion in 2016 to \$100.6 billion in 2026 [15]. This has heightened interest in automated visual aircraft inspection to reduce traditional manual check and by technicians, which are often time-intensive and susceptible to transcriptional error, especially when accessing areas within the aircraft that are hidden and hazardous [16]. To address these challenges and provide efficiency in the aircraft visual inspection process, the aviation industry is constantly adopting digitalization, including the application of autonomous robotic systems, including mobile robots and drones. Here, the robot is expected to interact with its surroundings and navigate effectively around obstacles without collision. Having a good understanding of dynamic and changing environmental structures like in MRO hangars, with the capability for accuracy and real-time detection and avoidance of obstacles [17]. The hangar environment is unpredictable, with diverse object irregularities, including light variations contributing to environmental uncertainties and navigational difficulties. Consequently, autonomous vehicles need to be equipped with reliable obstacle detection and avoidance mechanisms to improve their ability to navigate the surrounding environment safely.

Traditionally, mobile robots have utilized technologies such as Radar, GPS, and other sensors for navigation. However, in comparison to these sensors, RGB-D (Red, Green, Blue – Depth) cameras and LiDARs (light detection and ranging) systems, although more expensive, offer significantly broader ranges and higher resolutions. These advanced sensors enable the capture of a more detailed representation of the environment. RGB-D cameras provide a rich visual and depth perception, while LiDAR systems offer more precise environmental mapping, making them superior for complex navigation tasks [18]. The data generated by these vision-based sensors passes through processing networks to develop environment models that facilitate application of obstacle avoidance

solutions. The use of mobile robots to perceive, detect, navigate through environments, and enhance inspection processes has gained considerable attention in this field [19]. However, the principal challenge extensively investigated is accomplishing a navigational task that ensures an optimal, collision-free, and shortest path to the designated target. This challenge is amplified by the inherently complex and unstructured nature of the changing environments, and this affects the real-time decision-making process as the robot's navigation efficiency in the scene. Consequently, the robot's navigation and obstacle avoidance ability is challenged.

Previous studies have demonstrated various methods of autonomous navigation, with significant attention given to conventional approaches that combine local and global path planners. These methods, including the dynamic windows approach (DWA) [20], Rapidly exploring random tree (RRT) [21], and Dijkstra [22], have shown notable results, especially in static environments and can successfully navigate robots from one point to another with a reasonable level of confidence that they will avoid collisions with obstacles. However, these approaches frequently rest on a set of presumptions that are unlikely to be true in practice [23], involve a significant computing burden [24], and manual tuning of system parameters [19]. Also, it requires extensive engineering effort to develop and adapt the system to different environments, especially with dynamic obstacles of varying shapes and sizes.

Recent studies have demonstrated an increasing interest in learning-based techniques, underpinned by advancements in deep learning and computer vision, for their efficacy in self-learning, optimisation and adapting to variable conditions. This transforms raw sensory input into an adaptive understanding of the environmental features, facilitating obstacle detection, avoidance, and path planning in unstructured and complex scenes. This equips mobile robots with the capability to tackle the intricacies of dynamic real-world environments effectively. The work in [25] is applied to learn navigation tasks and plan a safe path efficiently. Nagabandi *et al.* [26] address adaptability and generalizability in dynamic environments, while Koh *et al.* [27] improve real-time capabilities for

practical navigation experience in aircraft inspections. While numerous mature algorithms for obstacle avoidance exist, the development of highly robust obstacle avoidance algorithms, particularly those enabling robots to operate effectively in unstructured environments, remains an area meriting further investigation.

This research paper comprehensively analyses learning-based approaches to autonomous ground robot navigation within complex environments. It aims to facilitate obstacle detection, obstacle avoidance and path planning within MRO hangar environments' challenging and confined spaces, striving for safe and intelligent navigational outcomes. The application of deep learning-based detection techniques, including deep reinforcement learning DRL [25], Fast R-CNN (Fast region convolutional neural network) [28], and YOLO (you only look once) [29], have been extensively studied. These models can detect obstacles with varied and unpredictable form dimensions and even under challenging lighting conditions. Each technique employs unique methods; DRL utilises policy, value, and reward systems, while YOLO & FAST RCNN uses bounding boxes and class labels to provide comprehensive information. In combination with sensor fusion, visibility in low-light situations and improved detection capability can be attained, consequently enhancing obstacle-free path planning and adaptation to a wide range of scenarios. Multiple sensor information creates a more robust and accurate representation of the environment, which is used to train and enhance the applicability and reliability of the model [14]. This study presents numerous learning-based mobile robot navigation algorithms proposed to address navigation difficulties, enhance motion decisions, and generalise in changing and complex hangar scenes. These methodologies are organised into three distinct categories: deep learning, object detection and hybrid approach, all focusing on improvements in environmental perception, adaptability and safe navigation capabilities in handling complexities and uncertainties inherent in a changing and dynamic environment. The core components of learning-based methods discussed in this review work are shown in Figure 2-1. The significant contributions are summed up as follows:

- This study examines learning-based navigation strategies, emphasising efficiency, safety and adaptability for use in complex settings such as MRO hangars.
- The review categorised algorithms based on deep learning, obstacle detection, and hybrid techniques, contributing to the generation of optimal and safe paths in a given environment.
- It elucidates these algorithms' challenges and potential directions for their applications in real-world scenarios.

The rest of the sections are structured as follows: The subsection of section I highlights some related literature. Section II presents the concept of machine learning and robotics as industry 4.0 (I4.0) technologies with their significance in autonomous mobile robot (AMR) systems for the digitalisation of aircraft inspection and maintenance operations. Section III presents the learning-based frameworks and potential use cases in aircraft maintenance and repair operations. Section IV discussed associated difficulties, future trends, and opportunities, critically looking at simulation to real-world transfer. Finally, the conclusion and future direction are discussed in section V.

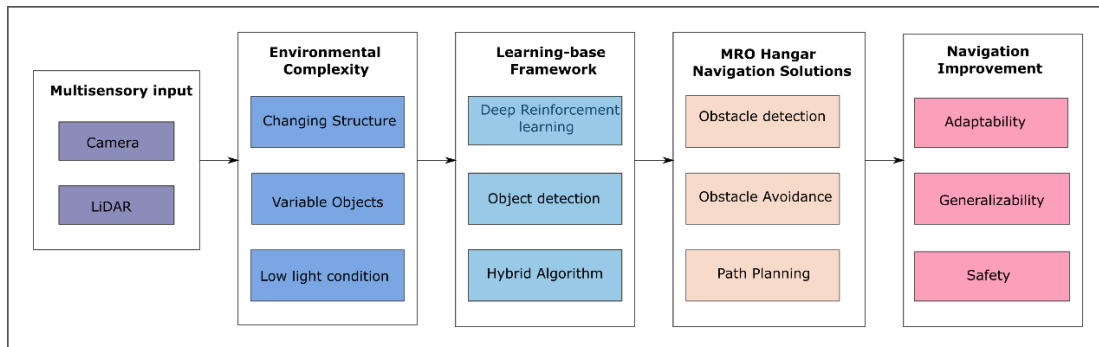


Figure 2-1: Overview of the learning-based Architecture in this survey

2.2 Related Work

In recent years, there has been remarkable advancement in obstacle detection, avoidance, and path planning in different domains. This advancement plays a significant role in realising the vision of the “Hangar of the future,” which envisions a highly automated, efficient, and safe environment for aircraft maintenance and

inspection. Recent studies engaged AMR navigation and interaction with its environment to automate aircraft inspection in a hangar environment [30]. Thanavin *et al.* [31] investigated robot navigation mechanisms for aircraft inspection in complex hangar environments. In [32], the authors proposed an automated approach to aircraft inspection using a depth camera-based mobile robot by following a predefined path to the target location. Another work applied obstacle avoidance control with the mobile robot while navigating around an aircraft for visual inspection [33]. However, much research has not been done specifically on hangar environments, but similar challenges have been addressed in recent research studies. This harnesses the power of deep learning techniques and the capabilities of various visual sensors such as cameras, LiDAR, and depth cameras to improve the navigational abilities of mobile robots. This primarily focuses on overcoming the limitations of traditional methods by enhancing environmental perception and adaptability mechanisms, thereby effectively addressing unforeseen complexities and uncertainties of environmental factors [25]. Many extensive reviews have been conducted to explore the application of learning techniques in mobile robot navigation, encompassing aspects like obstacle detection, obstacle avoidance, and path planning.

2.2.1 Object Detection Models applicable to robot navigation.

Deep learning is a promising technique for solving object detection problems and challenges. Sampo *et al.* [34] conducted a comprehensive survey highlighting the multifaceted capabilities of deep learning. These include managing multimodal sensor data, extracting features, learning complex and high-dimensional states, and addressing advanced object recognition challenges. In [35], the authors reviewed recent and successful object detection methods that have significantly advanced the field of autonomous vehicles. Furthermore, the study in [36] provides an overview of object detection and segmentation systems specific to autonomous vehicles, focusing on the detection methods, sensors, and fusion capabilities to achieve results and extend to related challenges in other application domains. The research paper in [37] delved into the range of object detection and tracking techniques, emphasising their generalizability in complex

settings. The works by Simegneu *et al.* [35] discuss the capabilities of object detection models by evaluating object detection metrics from sensor input. The implementation of the YOLOv5 object detection model was introduced in [38] which demonstrates reduced computational demand and improved system accuracy when tested on KITTI datasets. However, a limitation noted in the review study [39] is its ineffectiveness in detecting particular environmental objects. The application of the Faster RCNN two-stage object detector model, as discussed in [40], addresses detection issues by providing efficient learning-based object detection and tracking solutions for autonomous vehicles. This model, however, faces challenges due to its structural complexity and high computational demands.

2.2.2 Reinforcement learning-based Models used in robot navigation.

Applying deep reinforcement learning represents another facet of deep learning methods, specifically in addressing challenges related to robot obstacle avoidance and path planning [41]. Diverse modalities have been analysed, including learning from scratch in model-based and model-free states and learning from experience. The authors in [38] reviewed the transition from classic obstacle avoidance and motion planning to the capability of the learning-based approaches, yielding notable results. However, this review focused on applying RL only in mobile robot autonomous navigation. Significant limitations of DRL in real-world service robots were elaborated in [42], and to address these challenges, Zhao *et al.* [40] explored DRL methods, which represent a synergistic combination of RL and DL. They highlighted the significant contributions of DRL in achieving optimal path planning and efficient navigation, particularly in the context of constraint factors. Jahanzaib *et al.* [43] reviewed the capabilities of deep learning through environmental perception and modelling for an efficient navigation experience.

Most models integrate Q-learning techniques to solve navigation challenges through path planning and obstacle avoidance using discrete actions [44][45]. The author in [46] used a reward function and continuous action space to achieve

safe navigation tasks. Imitation learning, the act of behaviour cloning, has also been adopted recently by many scholars with outstanding success. Mustafa *et al.* [47] demonstrated using experience solutions to learn and solve new problems using machine learning concepts. Whereas the one by [48] reviews imitation and reinforcement learning methods that build essential obstacle perception and control mechanisms from environmental data for the fully autonomous and intelligent robot navigation system.

In another study, the application of behaviour-cloning techniques using the fuzzy controller [49] and from human demonstration [50] [51] demonstrated optimisation of the robot's navigation experience. The concepts of intelligent robot navigation have been significantly enhanced with learning-based systems, including supervised learning that focuses on feature extraction and adaptability in changing environments leveraging vast amounts of data [52]. Some of these have been integrated into the conventional algorithms to enhance their suitability in real-time applications and real-world environments.

2.2.3 Navigation with Hybrid Models

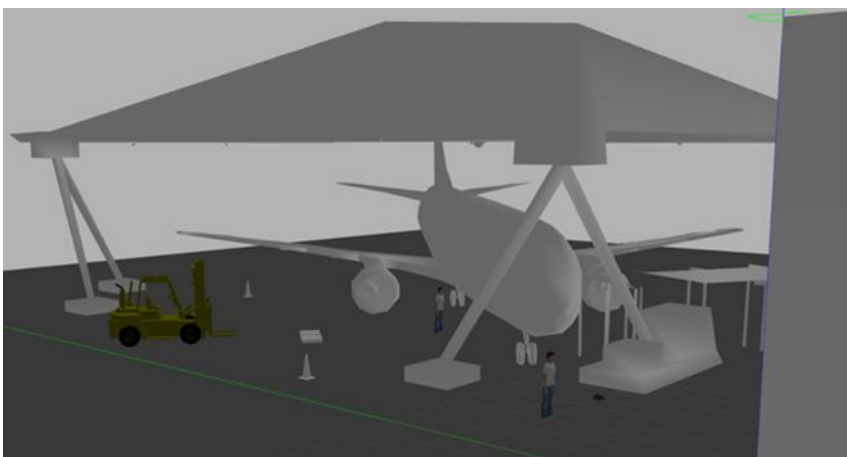
The combination of classical path-planning and learning techniques, presumed to offer more excellent practical stability, has garnered considerable attention. A comprehensive review of navigation conditions in static and dynamic environments was conducted by Patel *et al.* [53], exploring potential hybrid mobile robot navigation techniques and the suitability of their performance across various environments. In their work, Shrinivas *et al.* [54] in their work explored the advantages of hybrid methods with reference to different areas of application. Salim *et al.* [55] reviewed integrating neural network-based solutions, focusing on input, output and environment state and their ability to address major obstacle avoidance and path planning constraints. The work in [56] comparatively analysed heuristic neural network algorithms for path planning and obstacle avoidance. Wenzheng *et al.* [57] further explored the application of real-time neural network algorithms for generating collision-free routes to the destination, aiming to improve robot systems' motion control and obstacle detection accuracy. Overall, this paper leveraged many learning-based navigation methods studies,

providing a broader understanding and recommendation on the application and suitability of recent models based on their functionalities in complex environments.

2.3 Concept and Background

2.3.1 MRO Hangar in Aviation

The aviation industry has continually revolutionized to improve the safety and reliability of aircraft usage in air travel. The traditional aircraft maintenance and inspection approach involves semi-automated systems with human control to execute tasks. Sensing and navigation systems are usually pre-programmed to follow predefined inspection paths and do not adapt to unexpected conditions or obstacles. These factors are time-consuming and increase the overall operation cost. There is a growing need for more advanced and automated systems, potentially reducing cost and enhancing safety. The aviation industry has adopted the application of robotics solutions to enhance the MRO processes of aircraft towards the “Hangar of the Future” initiative. The MRO hangar represented in a simulation model shown in Figure 2-2 is a major part of the Aviation sector where industry 4.0(I4.0) technologies have gained wide adoption and are instrumental in improving safety and operational efficiency in the environment [30], [58]. Robotics and artificial intelligence are some of the critical enablers of I4.0, as



illustrated in

Figure 2-2: Simulated Cranfield University MRO hangar

Figure 2-3. These technologies have been effectively harnessed using unmanned vehicles, including intelligent ground robots for autonomous navigation in a busy and changing hangar environment, particularly for inspection, maintenance and repair tasks. A comparative description of the smart application of robotics over the conventional method is shown in Table 2-1. The use of mobile robots have emerged as a promising technology that enables efficient and precise operations in many tasks implementation, including assembly, drilling, painting, and inspections. Intelligent Robots use machines built and programmed to perform a specific task, combined with artificial intelligence techniques that instil and optimise intelligence through automated, data-driven learning capabilities. The integration of these technologies has spurred a digitalisation drive within the sector, promoting the concept of “Hangar of the future”. This is where intelligent robots play major roles by improving aircraft inspection efficiency and reducing aircraft-on-ground (AOG) time and overall operation cost.

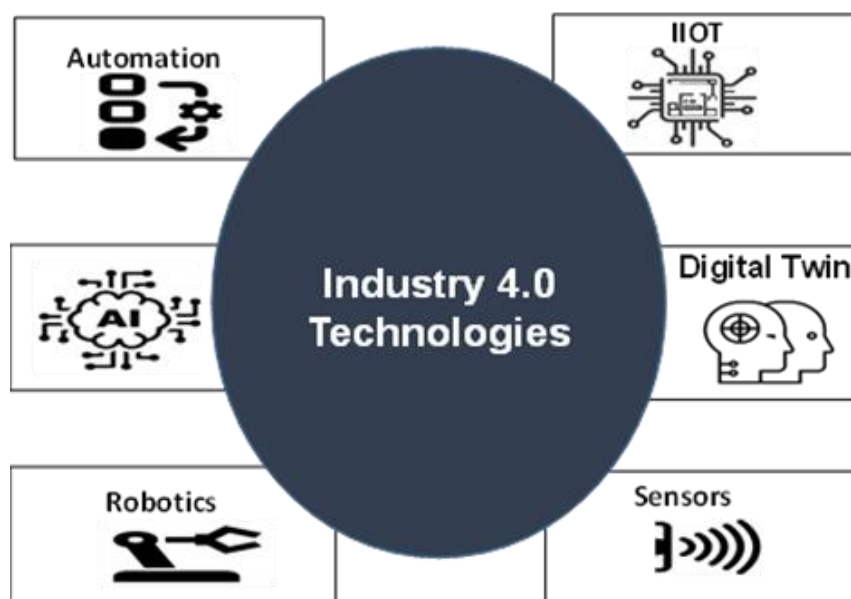
A typical hangar environment is characterised by highly complex configuration space due to unstructured and dynamic objects that vary in shape, size, and colour. Low-light conditions in such environments can also impact visibility in certain areas. These factors pose a challenge for robots, as their ability to navigate from the starting point to the target location is constrained by objects and changing environmental structures [59]. Mobile robots must proactively engage with their surroundings, interacting with and exploring the aircraft environment to ensure efficient navigation experiences [60]. In this process, they generate valuable information using.

Table 2-1: Comparing MRO hangar's traditional and intelligent robotics applications.

Tasks	Traditional approach	Intelligent robotics application
Inspection accuracy	Dependent on program quality and human interaction	Enhanced accuracy facilitated by learning and data analysis.

Automation level	Manual -semi-automated with human oversight	It is fully automated with limited human input.
Algorithms	Predefined algorithms combined with essential sensor input	Sensor fusion, advanced navigation algorithms and machine learning models.
Obstacle detection and navigation	Basic, simple path planning algorithms	I am using real-time and advanced deep-learning models to enhance path planning.
Task performance	Best suited for repetitive and defined tasks	Able to manage varied and complex tasks.
Cost	Higher cost for longer maintenance time and error management	Lower maintenance engagements and cost.

Various sensors facilitate the detection of environmental features, including positions of obstacles, which are essential for environment modelling and safe



navigation to their destinations [61]. Different machine learning-based functionalities have been developed, leveraging environmental information for robotic applications [41]. These have been demonstrated through various robotic platforms, such as the human-like robots from Boston Dynamics and the Crawling inspection robot by Cranfield University [62]. These robot systems follow standard robot architecture comprising sensory data acquisition, environmental perception, decision-making process, and execution of actions. This architecture is embedded within the robot's hardware framework to effectively learn the robot's orientation relative to a set of state space variables for optimal navigation in complex and dynamic environments.

Figure 2-3: Industry 4.0 technologies

2.3.2 Intelligent Robotic in MRO Hangar

Mobile robots encompass comprehensive system structures that work together through perception, detection, motion planning and control, as illustrated in Figure 2-4, to perform a series of navigation tasks. Robotic scientists in this field have proposed many intelligent technologies integrated to form an Internet of Robotic Things (IORT) that can interact with the environment and learn from sensor information or real-time observation without human intervention [63]. This technology empowers robots to operate more independently and make decisions based on the information they gather from their environments. Machine Learning (ML) is an Artificial Intelligence approach at the core of these enabling intelligent technologies with widespread adoption [64] become an essential component in accomplishing many intelligent tasks in robotics. The ML techniques incorporate sensor information fusion, object detection [65], collision avoidance mechanism [66], pathfinding [67], and control systems [34] robot's autonomous navigation problems [68] Diverse arrays of sensors, including Laser scanners, cameras, LiDAR and others, are leveraged for information gathering, mapping, obstacle detection, and robot positions and velocities. The fusion of information from these multiple sensors has brought a paradigm shift in developing more robust and accurate models of robotic systems. The multisensory fusion augments the capabilities of each sensor, thereby enhancing the overall system's visual

perception and its efficacy in obstacle detection and avoidance under various operational conditions [69]. ML methods have revolutionized robot navigation, especially in unstructured and complex environments, by offering highly accurate and robust capabilities [70], from properly recognizing and categorizing objects from photos or video streams in real time to training models by learning from data to adapting to various obstacles encountered during navigation.

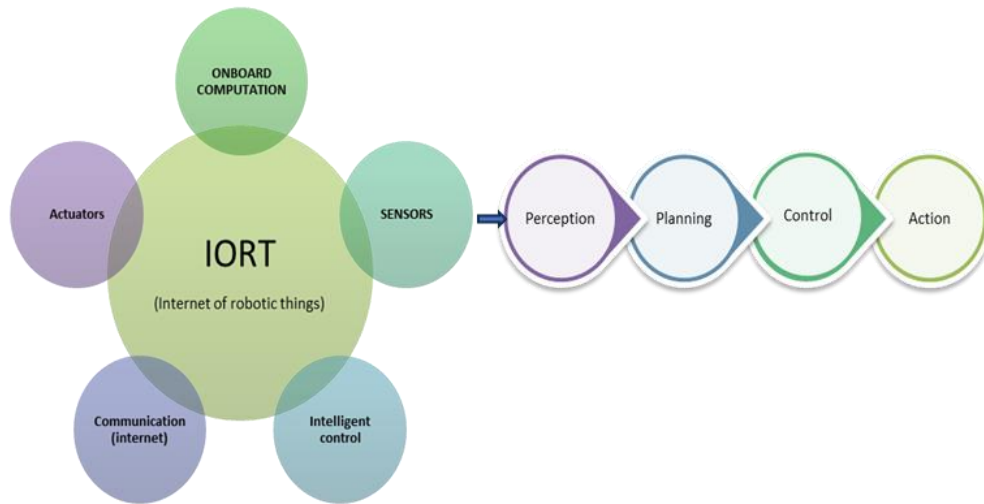


Figure 2-4: Structure of Robotic Intelligence.

2.3.3 Robotic Navigation

Autonomous robot navigation entails mapping, localisation, obstacle detection, avoidance, and achieving an optimal path efficiently from a starting point to a predefined target location [64]. The nature of obstacles encountered can be static or dynamic, depending on the environment structure. Navigating such an environment can be challenging due to the reliance on the sensory and real-time capability of analysing the vast amount of environmental data. Some of the robot navigation problems include the need to accurately perceive, identify, and respond to the environment's geometry, the robot's shape, obstacle types, and obstacle position using a suitable model. However, improper navigation processes often result in inaccuracies in perception, the development of flawed models of the environment, and emergencies of learning complexities, significantly limiting the robot from achieving its navigational goal. Applying advanced computational techniques like parallel processing and deep neural

network (DNN) algorithms has significantly improved the navigation experience. In the context of a neural network-enabled approach for obstacle avoidance and path planning, the architecture encompasses several interconnected modules, each contributing uniquely to overall system efficiency. As illustrated in Figure 2-5, the modules collaboratively contribute to achieving optimal path planning, adapting to changing scenarios, and minimising obstacle collisions in complex environments.

For many applications, various researchers have added specialist knowledge and undertaken studies to improve these modules to solve navigation problems. In most cases, the agents learn from data or through trial and error to master navigational skills and facilitate the generalisation of learned skills in similar settings in simulation environments, which is valuable for reducing training time and real-world difficulties. The virtual platform helps manage environmental factors and task structure that can influence these models' efficiency, adaptability, and reusability before transferring to the real world. In the MRO hangar environment context, the robotic systems are subject to complexities and uncertainties due to the unstructured nature of the settings, variability in object types and sensor capacity. This demands robust solutions capable of perceiving, responding and adapting to real-time changes.

2.3.4 Vision Sensors

Mobile robots must thoroughly understand their environment to perform robotics tasks effectively. To achieve this, robots are equipped with sensors to perceive and gather relevant information from the surroundings. Vision-based sensors, including LiDAR and Cameras, have become the most used equipment for unmanned vehicles (UV) detection and navigation tasks [71]. LiDAR is extensively used in the detection and tracking of

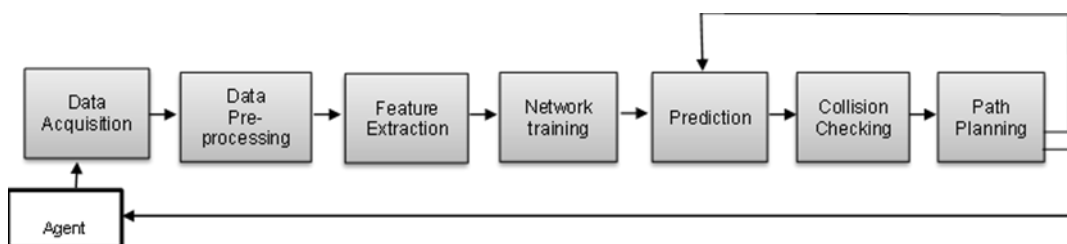


Figure 2-5: Learning-based Path planning framework.

AMR, even though it may be more costly than some alternatives. The sensor can obtain reliable information, including basic shape, precise distance measurements, and position of the obstacle and is more efficient in different weather and lighting conditions [72]. However, it is limited in capturing objects with rich information that is required to achieve precision for obstacle detection compared to RGBD sensors [73] [74]. This limitation can result in safety issues when navigating in a busy and dynamic environment with real-time demand. RGBD cameras have also shown excellent capabilities, including high resolution and generating rich and detailed environment information, though within a limited range, but remarkably efficient in object position estimation using depth information [75] [76]. However, the performance is highly susceptible to lighting conditions, which can be associated with specific areas in the hangar environments. The hangar environment significantly influences the choice of appropriate perception sensors for operational use within the space. Obstacle detection sensors are designed to interact with the environment and generate environmental data through sensor devices. They then use algorithms based on computer vision and object recognition for obstacle detection, tracking, and avoidance in a navigation system. To complement the capabilities of the RGB camera, depth sensing was combined in [77] to provide an accurate distance between obstacles and the robot position based on operational range and resolutions. The author in [78] employs depth camera information to estimate robots' poses for an efficient navigation experience. Depth cameras like Microsoft Kinect, Intel RealSense, and OAK-D offer valuable 3D spatial data that can precisely enhance robots' understanding of their environment. The integration facilitates obstacle sensing and state estimation for robust obstacle avoidance and path planning. Like the RGB cameras, variable lighting conditions and environmental factors can affect the accuracy of the perceived obstacles and position. This perception constraint is part of obstacle avoidance and path-planning challenges in complex settings.

Recent research has significantly contributed to intelligent obstacle detection and avoidance solutions based on sensor usage and algorithm improvement.

2.3.5 Obstacle detection

Obstacle perception and identification for robot navigation involves locating potential obstacles that could influence a robot's ability to navigate its surroundings. The mobile robot utilises its sensory systems, including LiDARs or cameras, to perceive and understand its environment, enabling it to plan a safe and collision-free path to its intended destination. Deep Learning has gained wide adoption in research and industry, leading to the development of numerous navigation models that leverage different object detection models and sensor inputs for robot obstacle detection and avoidance systems. Most recent object detection methods are based on convolutional neural networks (CNN) like YOLO [79], Faster R-CNN [28], and single-shot multi-box detectors [80]. Faster R-CNN is renowned for its high detection accuracy and employs a two-stage deep learning framework. This network structure impacts the computational efficiency and speed, which are crucial for real-time applications [81]. On the other hand, the YOLO model is a one-stage object section approach known for significant speed and real-time performance. This makes it well-suited for autonomous mobile robot navigation, where prompt decision-making is vital for obstacle avoidance and motion control.

2.3.6 Obstacle Avoidance

Ensuring the safety of the working environment is a primary priority when deploying mobile robots for navigation tasks in complex environments. The safety solution should be able to perceive the environment and take proactive actions using reliable sensors to avoid obstacle collisions [66]. The mobile robot should be able to identify a safe and efficient path to navigate within its operational environment, which may contain static and dynamic obstacles to the target destination. Different learning-based obstacle avoidance algorithms have been developed to enable robots to effectively and precisely complete intended tasks. Some are integrated with local and global planners to efficiently adjust the direction and speed of robot motion in response to detected obstacles within static and dynamic environments to generate an improved path to reach the target location [82]. Recent review studies, learning-based models in robotic

navigation, have demonstrated notable success by learning and generating obstacle data from the environment sensors. These models extract obstacle features from images and video streams, allowing them to classify and locate different obstacles within the given environment. Integrating these models into Robot operating system (ROS) based planners has improved robotic navigation performance. Planning algorithms like the Dynamic Window Approach (DWA) [20] have good capabilities in a dynamic and complex environment and have been widely combined with learning algorithms for more excellent capability, efficiency, and intelligent path planning [83].

2.3.7 Path Planning

Autonomous learning in path planning has made significant progress recently, where technologies such as CNN and deep reinforcement learning have been increasingly adopted. Path planning entails a sequence of configurations based on robot types and environment models that enable robots to navigate from a starting point to a target location [84]. The environment can be mapped to represent geometric information about the environment and connectivity between nodes or maps. The map-based method enables the robotic solutions to compute the robot's dynamics and environment representation for an optimal global path planning to the goal [85]. Local path planning relies on real-time sensory information to safely navigate static and dynamic obstacles. Another aspect of path planning configuration is the maps model, which requires no predefined map of the environment but capitalises on frameworks like deep learning models to learn and enhance optimal navigation strategies. Path planning in an MRO hangar can be challenging as the environment is often changing and complex with a high density of obstacles. To ensure a robust obstacle-free path, ongoing research is focusing on path tracking [86], advanced deep learning [87] and hybrid approaches for more autonomous and intelligent robot path planning to target locations [83].

3.8 Path tracking

Safe and efficient robot navigation requires a path-tracking system that guides mobile robots along the planned trajectory to a target location, managing and

minimising deviation from the scheduled route. This involves continuously monitoring and updating the planned route based on sensor feedback and the changing environment. The work in [86] reviewed path-tracking algorithms relative to high and low speeds. For high-speed applications, the reaction time available for the robots to perceive, process and respond to obstacles is significantly reduced at high velocities, making it harder to execute quick and sharp manoeuvres without compromising stability or safety. Looking at low-speed use cases, the application of robotic systems in MRO hangars involves low-speed movement and the requirement for precise path tracking in complex settings. The low-speed movement of these robots can lead to path-tracking errors, especially when dealing with sharp turns and frequent changes in direction. Accurate modelling of low-speed dynamics is essential to adjust the robot's behaviour for optimal path tracking. The combination of adaptive control systems, sensor technologies [88] capability for robot navigation in such scenarios. From the study in [89] variety of scenarios [90].

2.4 Learning-based navigation techniques

The dynamic and unforeseeable state of edge cases in the real world makes applying the navigation task challenging. Ensuring that the systems can detect and respond effectively to changing and unstructured scenarios is essential for safe and reliable navigation [91]. The conventional approach, which works best in a static environment, is known to be computationally intensive and must be adjusted to varying environment states and motion dynamics. The learning-based approach has been instrumental in addressing the limitations of the traditional methods. These methods utilise complex neural network architecture to process sensory data and extract relevant features, allowing them to adapt to a wide range of environmental structures and make more informed decisions [48]. The primary focus of this paper is on integrating deep learning methods that specifically address challenges associated with obstacle detection, avoidance, and path planning in dynamic and complex environments.

2.4.1 Deep Reinforcement Learning (DRL)

This represents a transformative autonomous navigation approach that combines the perception ability of deep learning and the decision-making potential of reinforcement learning to effectively map sensory input to navigation actions,

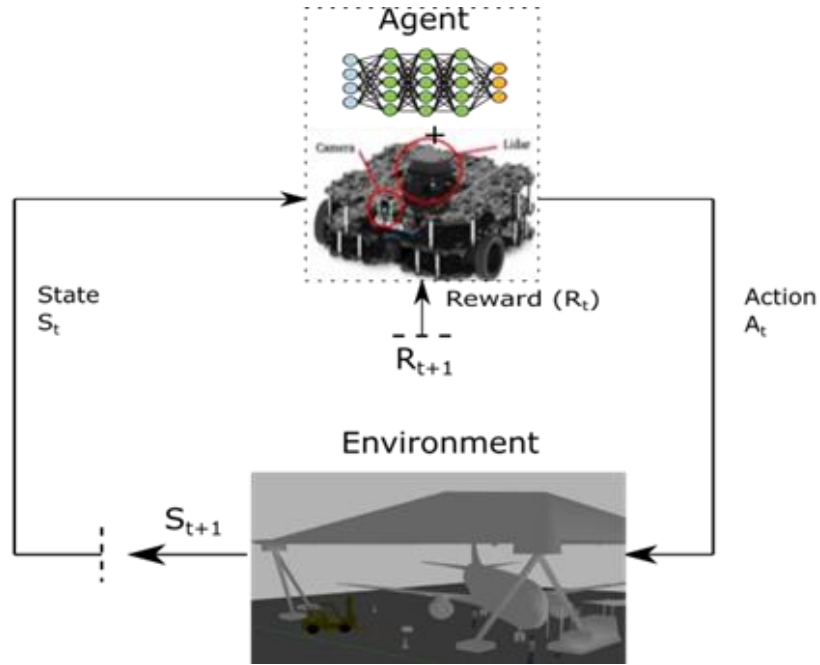


Figure 2-6: Overview of interaction between agent and environment

leading to an improved end-to-end navigation process. The DRL-based navigation model description, shown in Figure 6, has demonstrated a great capacity to achieve safety, adaptability, and efficiency by learning about the workspace with less reliance on the accuracy of sensor information. The Deep neural network functionalities of the DRL, like CNN and Autoencoder (AE), can automatically extract varying features from highly complex environments.

Researchers have adopted this paradigm and have developed different approaches that address autonomous navigation challenges by leveraging DRL algorithms like Proximal Policy Optimization Algorithms (PPO) [92], Deep Deterministic Policy Gradient Algorithms (PPO) [92], Deep Deterministic Policy Gradient Algorithms (DDPG) [93], Trust Region Policy Optimization (TRPO) [94] and others. Most of these algorithms are constrained by sparse rewards [95] which impacts model training and convergence rate. Some proposed methods have been shown to discretise the action space. However, these have achieved

limited success in addressing the complexity of specific settings. On the other hand, the Policy gradient approach, utilising gradient descent, has been applied to solve continuous space problems by calculating the policy parameters to maximise the expected reward [96]. In achieving optimal navigation in complex environments, value-based RL algorithms have shown enhancement for applications involving discrete action spaces by using an argmax to choose the action with the highest Q-value [97]. Further improvement is achieved by adequately tuning system hyperparameters and incorporating deep learning functions. This includes multiple hidden layers, extensive datasets, ReLU activation functions, and Adam optimiser. Such configurations have been demonstrated to achieve efficiency in obstacle detection, avoidance and path planning in autonomous navigation systems.

Table 2-2. Comparison of papers on DRL-based robot navigation systems.

Ref	Principle.	Method	Model	Strength	Limitations
[98]	Avoid Obstacles and Navigation.	Deep Q network (DQN) & Duelling Double Deep Q network (D3QN)	Model-free	Can learn from very depth noise	Applicable in less complex settings.
[99]	Path planning in continuous space.	DDPG	Model-free	Optimal path generation with less environmental information	Predefined tracks can obstruct efficient navigation in high-dynamic scenes.
[93]	Generalization ability of path planning.	LSTM + DDPG	Model-free	100% success rate, 18.8% better training time and 21. % shorter distance covered.	Only considered static obstacles.
[100]	Collision-free navigation.	DDPG with a separate experience.	Model-free	An improved replay mechanism was adopted for training and improved network performance.	Network parameters are randomly set, and limited in real-time situations.
[101]	Object recognition and robot navigation	Deep neural network (DNN)	Map-based	Average recognition accuracy of 80%.	Require a more complex environment to validate real-world applicability.
[102]	Object detection and tracking for navigation	CNN	Model-based.	Can detect and track multiple obstacles.	The default ROS algorithms used are limited in complex and real-time demanding environments.

2.4.2 Object detection model.

Effective obstacle perception and detection are fundamental to ensuring robots can plan collision-free movements. The mobile robot utilises sensing systems, including LiDARs or cameras, to perceive environmental features that could interfere with navigation tasks. Various methodologies are adopted to improve

these capabilities. Some are designed for static settings, detailed environment maps, sensor fusion, and dynamic scenarios, potentially enhancing navigation ability. The integration of machine learning has proven great efficiency and accuracy in object detection applications. This learning-based method drives learning from data to perform MR navigation tasks through domain knowledge [56]. In the MRO hangar context, the changing environmental structure with varying object types, occlusion and low-light conditions limits the capability of the learning model. It requires more training costs to scale to new environment variables. These are among the major difficulties of obstacle detection in real-world mobile robot navigation that have resulted in many environment-related accidents. Object detection algorithms use the explicit definition of environment variables like obstacle size, shape, depth and object distance range in the environment [64], [72] to predict obstacles, improve obstacle avoidance and plan a smooth route to the destination. Most algorithms employ pre-trained object detection networks to develop models relevant to obstacle avoidance and path-finding optimization. Some that are widely applicable in robotics system include CNN, Faster R-CNN, and YOLO.

2.4.2.1 CNN

CNN is a type of deep learning framework that is characterised by numerous convolutional layers instrumental in processing and analysing input data. Structured with potentials to identify and extract relevant data features from images automatically as required for detection and classification in real-time settings [64]. Recent studies have shown different approaches to developing object detection and collision avoidance models. *Qi et al.* [103] used Dropout and DropConnect techniques to improve CNN trained on Pascal VOC public datasets to detect and classify obstacles in difficult settings, improving object detection accuracy. Additionally, *Mechal et al.* [104] presented a CNN model trained with different types of images, such as RGBD, RGB and HSD, enabling the classification of obstacle avoidance actions. [103], [105] used CNN to estimate the depth of objects in an image to consequently steer commands for a quadrotor. This study provides evidence demonstrating the superior performances of CNNs compared to traditional methods. Similarly, *Liu et al.* [106] proposed a CNN

vision-based improved model for obstacle avoidance, aiming to generate steering commands for a mobile robot while reducing the need for complex and time-consuming hand-engineering of features. The ability of CNN to learn feature representations directly from raw image data without the need for manual feature extraction marks a significant advancement in the field of object detection and has led to widespread adoption in various applications, including autonomous vehicles and robotics.

2.4.2.2 Faster R-CNN

Faster R-CNN is an extension of CNN based on the region proposal network (RPN), which is classified as a two-stage convolutional neural network. This sophisticated approach is intended to create bounding boxes for each proposed region and then extract important features from these regions, making it easier to classify the objects contained in each region later [81]. Faster R-CNN is mentioned in *Lee et al.* [107] paper where it was used to assist unmanned vehicles to avoid collision. Mahendrakar *et al* [81]. employed this technique to ease robot navigation while showing the excellent accuracy of the Fast-R CNN. In the study conducted by Hakim *et al.* [108], Faster R-CNN was explained to operate in two phases. Initially, it generates bounding boxes around several objects within an image, and subsequently, in the second phase, the proposed regions undergo classification to accurately detect objects. Compared to other object detection models like YOLO, Faster R-CNN is more precise in the object detection process with a more comprehensive and multiple-layer structure [103] and this makes it an attractive option for real-time applications. However, the gain in accuracy comes at the expense of real-time identification speed, as the algorithms exhibit slower performance in this regard. In addition, significant time is required to collect training data in the case of multiple-layer CNN models [109]. Real-world applications require obstacle detection systems with high accuracy and real-time capabilities to efficiently respond to the increasingly complex demands of different domains. Faster R-CNN is potentially more computationally intensive and of low inference.

2.4.2.3 YOLO

YOLO, on the other hand, utilizes a single-stage kind of object detection with bounding box coordinates created around detected objects in the grid-based network. The technology programs images in grid cells with bounding box coordinates, confidence scores and object type information for object detection [81]. Emmi *et al.* [110] compared YOLOv3 with Retina Net-Resnet to address map-building challenges and selected YOLO based on its performance. Cao *et al.* The author in [111] based its solution on YOLOv3 to achieve high accuracy in object detection and system and processing speed to enable efficient underwater robot navigation. Also, Mahendrakar *et al.* [81] proposed improved YOLOv5 in autonomous navigation to demonstrate the effectiveness of YOLO in real-time situations. This is a significant factor in real-time object detection, as that practicality of object detection models lies on their processing speed considering real-world uncertainties. the speed of the process holds significant importance for practical obstacle avoidance. Comparative analysis between YOLO, CNN and Faster R-CNN [112], highlights YOLO's superior adaptability in real-time object detection scenarios. While YOLO's speed and adaptability make it an attractive option for real-time applications, its dependency on extensive training data underscores a potential limitation, especially in an environment with data scarcity. Therefore, the choice between the object detection models should be guided by specific application requirements to balance speed and accuracy.

2.4.3 Hybridization with Neural Network

Hybridisation of algorithms has become commonplace as it integrates the strength of learning and non-learning methods to achieve more effective, accurate and reliable obstacle detection and avoidance solutions. This section investigates notable publications on classic ROS-based and machine-learning methods. The concept of Robot operating system (ROS) navigation planners provides a detailed framework that can integrate different types of algorithms and sensors to develop complex robotics applications [4]. Emerging solutions have utilised neural networks (NN) to optimise traditional path planning and obstacle avoidance techniques in recent times[55], [113]. A neural network is a layered

framework of inter-connected nodes that takes input based on a designed task and produces network prediction or classification as output. The author [113] applied a long Short-Term memory (LSTM) neural network using a robot pose and agent to obstacle distance as input to solve end-to-end path planning while avoiding a dynamic obstacle. Another work in [114] used a padding mean neural dynamic model to address the challenges of traditional neural dynamics by enhancing the completeness and optimality properties of path planning in static and dynamic environments. However, integrating multiple techniques can increase computational demand and the system's complexity and affect real-time performance. Other factors, like environment dependency, may affect generalizability to different platforms.

Table 2-3: Recent solution with integrated ROS and Object detection models

Ref	Sensor	Environment	Learning Algorithm	Strength	Limitations
[115]	Vision sensor	Dynamic inSim/real world	YOLOv4 and ROS DWA	Increased detection speed of 15 FPS higher	3% Reduced detection accuracy
[116]	Vision sensors	Static & Dynamic in Simulation	Yolov3 & ROS DWA	Improved obstacle avoidance with about 82%	Default ROS planners used require improvement.

				detection accuracy of	
[117]	Monocular camera	Complex in simulation	Fully Connected Network (FCN) & A*	Reduced path length and optimised trajectory	The model cannot adapt to changes due to limited data coverage.
[118]	RGB-D cameras	Static in simulation	CNN & ROS	Improved autonomous navigation with camera and IMU sensors.	Limited in low-light situations
[119]	Lidar and camera	Complex in simulation	ROS & YOLOV3	Improve obstacle detection and navigation performance.	Limited to a static environment.
[120]	RGBD image	Static in Real world	Yolo3 & odometry program	Successful collision-free trajectory	Require trade-off between speed and obstacle detection accuracy.
[121]	LiDAR	Static in Real world	YOLO+ Simultaneous localization and mapping (SLAM)	70% less time to compute inference which improves obstacle detection and navigation.	Used a sensor with limited functionality and in a less complex environment.
[122]	Monocular camera	Dynamic Real-world	MASK R CNN (Region Convolutional Neural Network)	Improved obstacle detection in low-light situations.	Obstacle detection accuracy is limited when the robot to obstacle distance is less than 1.5m.
[123]	LIDAR	Static in Real world	Yolov3 +ROS global planner	Improved obstacle avoidance in a static environment.	Robots lose the path to the target location when two obstacles are close to each other.

2.5 Drawbacks and Future Work

Most obstacle detection, avoidance, and path planning solutions are unstable and require robustness to be applicable in real-world scenarios. These systems rely

on various parameters, and finding suitable configurations is complex and results in unpredictable performance [113]. This can be exacerbated when dealing with limited training data. Generating data and experience from

Table 2-4: Strengths and limitations of hybrid systems solutions in obstacle avoidance and path planning.

Ref.	Model composition	Strength	Limitations
[124]	Multilayer perceptron (MLP)	Avoid obstacles and find the shortest path significantly.	Inability to integrate with other path planners
[113]	Long Short-Term Memory (LSTM) synchronised with ROS	Completeness and optimality of path planning	The experiment is limited to the simulation environment.
[125]	CNN-based ROS with ROS-cafe	Improved obstacle avoidance and motion control	Limited in dynamic and complex environments.
[117]	FCN and A*	Obstacle avoidance and path planning	Use of only a camera which limits the field of view (FOV)
[118]	CNN & ROS	Obstacle detection and avoidance	Limited in a dynamic setting
[126]	Sampling-based planner (RRT*) and conditional variational Autoencoder (CVAE).	Improved path planning	Scaling to complex problems can be difficult.

The robot's environment through perception and interaction can be challenging owing to environmental factors, system functionalities and even the robot's dynamics. These can impact the functionality and capability of the algorithm, especially in cases where sufficient and relevant data is required for practical robot training. Conversely, recent research has demonstrated the significance of extensive and versatile data and experience to improve end-to-end robot navigation experience. Robots tend to generalise better in this case and can

apply learned skills to novel situations, making them reliable in real-world applications.

The effectiveness of learning-based methods also depends on the quality of sensor information. Poor sensor data can affect robotics solutions' stability, reliability and performance. They can be subject to high inaccuracies and noise in sensor data that breed uncertainties and deviations in performance, affecting safe navigation [127]. Calibrating and simulating sensors to match actual sensor specifications are essential to bridge the gap. Also, augmenting simulated sensor data to replicate the noise, uncertainty, and limitations of real-world sensor properties should be able to fine-tune the robot's ability accordingly. Combining data from multiple sensors with sensor fusion techniques has been proven to bridge the reality gap.

Another primary research concern is transferring the trained model from simulation to real-world robot platforms. The simulation environment accommodates high-level collisions and reduces training difficulties compared to training in a physical environment, which can be dangerous and cause damage to the robot [124]. For the learning-based algorithm to be helpful for robot navigation, it must be swift in adapting to the new environment and generalising to related tasks without the cost of additional training. Tai *et al.* [128] demonstrated a maples motion planner built on observations from a low-dimensional range laser sensor and asynchronous deep-RL technique to generalise to a real differential robot platform without further retraining. Achieving efficient transfer without performance degradation for obstacle avoidance and route planning solutions requires developing a high-fidelity simulation environment that closely mimics the real-world environment [106]. However, not all tasks can leverage knowledge from one another, and some necessitate specialised models tailored to specific challenges. For example, in an MRO hangar, the mobile robot domain experience might be specific to obstacles unique to aviation maintenance settings, and this complexity requires a customised model. However, integrating advanced Neural networks and deep learning

techniques has marked a significant improvement in intelligent robotics, paving the way for more efficient and reliable solutions.

2.6 Conclusion

This research review provides a comprehensive overview of the current landscape of learning-based object detection and avoidance in emerging intelligent and autonomous vehicles. The complex and dynamic MRO hangar environment requires robust and intelligent robot navigation systems to manage tasks with high demand efficiently. An optimal path planning model is necessary to generate an obstacle-free and shortest route to the target point through efficient obstacle detection and avoidance solutions. Recent work in this area showed improved safety and reliability and significantly contributed to broader AI-driven robotics in dynamic real-world applications. We critically examined the inherent functionalities and challenges of applying these models in AV environments, mainly focusing on the trade-offs between detection accuracy and avoidance efficiency. Our exploration of various learning-based techniques has underscored their potential to utilise extensive datasets significantly and experience learning to enhance adaptability and task generalisation, factors vital for real-time robot navigation in a changing MRO hangar environment. Choosing models for safe and efficient robot navigation is an operational environment-specific requirement. Relative to the MRO hangar scenario, the combination of deep learning architecture, advanced path planning, and obstacle avoidance strategies from our recent work using LiDAR and camera data fusion have enhanced reliability and adaptability in changing structures.

The findings emphasise the need for continued research in this domain, particularly in developing algorithms that balance performance with computational demands. These advancements promise to make robotics systems more adaptive, intelligent, and efficient in the complex environment.

PART 2: DEVELOPMENT OF DEEP LEARNING BASED OBSTACLE DETECTION MODEL

The literature review established the gaps in conventional navigation systems that are majorly based on ROS framework and key limitations in managing real-world complexities related to busy hangar environment. It also explored the capabilities of deep learning models and hybrid solutions that have been studied in recent time. This facilitated the development of YOLO deep learning-based object detection model to improve the accuracy and real time adaptability of the navigation system in complex environment. This chapter developed improved YOLOv5 object detection model and the overview of this project include;

- Data capturing from TUI hangar and simulation environment.
- Public dataset integration and processing.
- Dataset Integration into the YOLO model.
- Customization with CBAM for improved YOLO object detection.
- Performance Evaluation
- Comparison with baseline model

Paper published in: <https://www.spiedigitallibrary.org/conference-proceedings-of-spie/12536/125360S/CNN-fusion-architecture-with-visual-and-thermographic-images-for-object/10.1117/12.2665984.full>.

3 CNN-FUSION ARCHITECTURE WITH VISUAL AND THERMOGRAPHIC IMAGES FOR OBJECT DETECTION

This research paper aims to develop a novel image fusion-based CNN architecture for enhancing autonomous robotic systems in aircraft visual inspection within MRO operations. The focus is improving obstacle detection and avoidance capabilities necessary for safe navigation in complex environments with diverse obstacles. The crowded and varied nature of obstacles in hangar environments poses a significant challenge for the current neural network-based

systems, especially in low conditions. The research developed an improved YOLOv5 object detection model, specifically tailored using custom fused RGB, thermal images, and deep learning attention mechanism. The model development detailed how data was generated, processed, and fused with data from other sources, including public datasets, to improve the model's capabilities. This customisation demonstrated an improved model's accuracy and efficiency in detecting obstacles in challenging situations compared to the conventional YOLOv5 object detection model. This development significantly improves the YOLOv5 model's suitability for enhancing robot navigation and preparedness for integration into the ROS navigation stack.

3.1 INTRODUCTION

The increasing potential for autonomous mobile robots (AMR) to achieve intelligent operations has attracted continuous research interest in domains like the aviation industry. One of the major application areas is visual aircraft inspection for capturing aircraft surface defects during maintenance, repairs and Overhaul (MRO) operations in a hangar environment. The busy hangar environment is inconsistent in structure, with objects of varying shapes, sizes, colours, and intensities contributing to difficulties in the robot's real-time motion control while detecting and avoiding obstacles in a navigation task. AMR requires high detection accuracy in distinguishing solid and low-density moving objects for safe navigation in different lighting or environmental conditions. Significant improvement has been achieved by enhancing the robot's perception [129] which is heavily dependent on sensing devices and background information which can be ambiguous. Thermal and RGB cameras are vision-based sensors used for object detection in autonomous mobile robot navigation tasks and have different capabilities based on light variations. RGB camera images provide more detailed information in good light conditions, but shadows and illumination intensity impair performance.

On the other hand, thermal images are good at spotting objects of even small shapes in visually degraded and high-contrast conditions. Still, image information usually lacks detailed features like colour, which compromises application in

different scenes and situations. The fusion of visible and thermal images [130] has improved the perception of object features in complex environments. It has increased the robustness of the information in object detection and avoidance tasks. Image fusion-based methods are critical concepts that have been widely developed in the literature to improve the visual quality of the fused image [131] and enhance object detection in different applications with more accurate information and clarity. This includes using traditional methods like deep neural network-based approach [132], fuzzy logic [133], and multiscale and sparse representation. These algorithms often rely on handcrafted feature extraction method to fuse image information. Traditional methods are usually tuned for specific task or datasets and might not generalise well to new scenarios or object types. Also designing an effective traditional technique requires significant domain knowledge which can be time consuming. This can be very challenging especially for shallow objects [134] and results in poor-quality image classification. Deep learning methods have demonstrated improved performance by automatically learning relevant features from the input data.

This research aims to develop an efficient and more robust object detection technique for mobile robots to respond timely to obstacles and navigate safely in unstructured and dynamic scenarios using deep learning on fused image data. Image-based fusion for object detection has been widely studied using deep learning and applied in different use cases with a reasonable success rate [135] Convolutional neural networks (CNN) are one of the variants of deep learning models and are powerful for feature extraction from image data. Recent improvements with CNN architecture have been proposed to extract relevant object features and classify them accordingly to enhance object detection, like ResNet, Alex Net, FusionGAN [136], NestFuse [137], and DenseFuse with encoding and decoding networks [138] These CNN networks utilise dense blocks, which compromises computational requirements and limits the real-time application of the models.

To address this drawback, we propose an adaptive YOLOv5 model [29] based on fused RGB and thermal images using a pre-processing module and attention

mechanism to improve object detection accuracy. In addition, the model will incorporate custom augmentation techniques and a convolutional block Attention module (CBAM) to take advantage of the image sources' feature dissimilarity to improve object recognition and avoidance. The goal is to create more informative images that increase object visibility and improve scene understanding for accurate object detection and avoidance in challenging cases.

The main contribution of this work can be summarised as follows:

- Multisensory data preparation and processing from simulation, real-world hangar environment and public datasets.
- The DenseFuse network is utilized to fuse information from multiple sources such as RGB and Thermal images. To enhance the visibility and clarity of the input images captured in low-light or dark areas.
- The CBAM is integrated into the YOLOv5 model architecture to enhance attention mechanism and focus on relevant features in the input images to improve detection accuracy.
- Comparison of the performance improvement of YOLOv5 with the addition of DENSEFUSE and CBAM using traditional YOLOv5m as a baseline.

3.2 Related Work

Recent research on object detection with multi-source images has shown improvement and robust environmental adaptation capability. Traditional and deep-learning detection methods are two critical categories of object detection that have been widely studied. Conventional methods like the sliding window approach [139], support vector machine (SVM) [140] and template matching-based algorithm [141] involve manually designed feature extraction procedures for object detection and classification in an image. This method is challenging, computationally inefficient and poorly performed with small and diverse objects. Therefore, the automatic and hierarchical approach to feature extraction in convolutional neural network (CNN) algorithms [142] has garnered more research attention than traditional methods, especially in solving the problems mentioned earlier.

The most popular CNN architectures for object detection are based on the single-stage and two-stage detectors network. Faster R-CNN [143] is among the two-stage variants that were improved by using region of interest (ROI) and region proposal network (RPN) pooling. The study in [144] generates sparse related bounding boxes for target objects using non-maximal suppression technique to select the region of interest. However, this still is limited in achieving the real-time improved inference speed required for application in autonomous vehicles. On the other hand, Yolov5 is among the one-stage detection methods that use a grid framework to predict the position and classification of multiple objects in the frame for better detection accuracy. Different improvements have been made in yolov5 in [145]. The model elements' structural adjustment method was used to improve small object detection and inference time, and [146] authors expanded the cross-stage-partial-connections (CSP) module to enhance the use of shallow features.

Many more researchers have proposed better models using fused visible and thermal. Yunfan et al. [147] use a multi-layered CNN architecture that combines RGB and thermal images for improved pedestrian detection. Wagner et al. in [148] based detection model to train the KAIST dataset described for pedestrian detection using merged FIR and RGB images. Osman et al. [149] trained the YOLOv3 model with a merged image spectrum but required more images to optimise the network's performance. In [150], a multispectral model based on Yolov4 with combined RGB and thermal images was used to demonstrate high detection accuracy and adaptation capability in changing scene scenarios. More improvement was seen in [151] where the authors used illumination-aware deep neural networks with visual and thermal images to enhance object detection performance. Early fusion enhancement was used in [152] by using a CNN-based fusion module to extract useful features from the RGB and thermal images towards better object performance. Based on these studies, we have proposed a fuse-YOLO that leverages thermal and RGB features and a CBAM module for the yolov5n network to improve object detection accuracy. This will enhance relevant object feature extraction, lightweight capability, and computational suitability for mobile robot navigation tasks.

3.3 IMAGE ENHANCEMENT FOR ROBOTS OBJECT DETECTION

In this context, we focus on improving the performance of an existing deep neural network capable of recognising different types of objects under varied lighting situations by supplementing RGB images with thermal images through a fusion network. The RGB images are known to have high resolution and can capture detailed information representing environment features. But suffer in low light scenes where image qualities are impaired, making it challenging to extract relevant information. In contrast, thermal images are robust to different lightning conditions, can detect the temperature differences of objects in complex scenes but are associated with low spatial resolution. This results to less detailed images and can affect application in daylight situation where colour information is essential. To address the limitations of each modality, studies have explored fusion techniques that can combine RGB and thermal data to leverage their individual strength, provide richer information and enhance image quality [153].

The aim of this project is to develop a pretrained model that can learn from diverse datasets covering relevant environments and low light scenarios. This model will be integrated into a robotic platform to improve the navigation system within hangar environments characterised by varying lightning conditions. This work will enhance navigation capabilities of autonomous mobile robots, especially in detecting obstacles that may not be apparent or well captured by RGB cameras, which are commonly employed in robot navigation. Additionally, it will improve obstacle recognition and avoidance algorithms necessary for robot application in aircraft inspection operations.

3.3.1 Overview Of Yolov5 Algorithm

YOLOv5 is a popular CNN-based object detection framework that uses a single-stage architecture and grid of cells to train the detector to operate precisely and with more visibility, enabling the system to identify any object in complex environments. It is easy to deploy and train. The architecture comprises the backbone based on the cross-age partial connection (CSP-Darknet-53) model [154] that connects convolutional layers to extract object features from input images at a reduced computational cost. The neck structure is in the middle

phase and uses a Path Aggregation Network (PANet) [155] to up-sample, down-sample, fuse feature maps, and extract the multiscale object. This is done at different levels of the backbone region and helps the model to generalise on unknown data. It improves the detection and localisation accuracy of objects of interest in an image. The YOLOv5 detection head uses a dynamic anchor assignment approach to cluster the ground-truth objects by adjusting the sizes and shapes from input images. This will generate multi-dimensional bounding box information, including probabilities of the object class, confidence and box coordinates.

The YOLOv5 model is configured with a yaml file that is used to define the parameters that make up the model architecture [145] and has a variety of pre-trained models represented as YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l and YOLOv5x where the difference between them lies on inference time and size of each model in terms of the depth and width of the network [156]. The general YOLOv5 model [29] consist of series of convolutional layers with different kernel sizes, batch normalisation [157] and ReLU activation functions [158] to optimise feature extraction and detection accuracy. The YOLOv5m model performed better with our custom dataset and was used as the baseline model in this context.

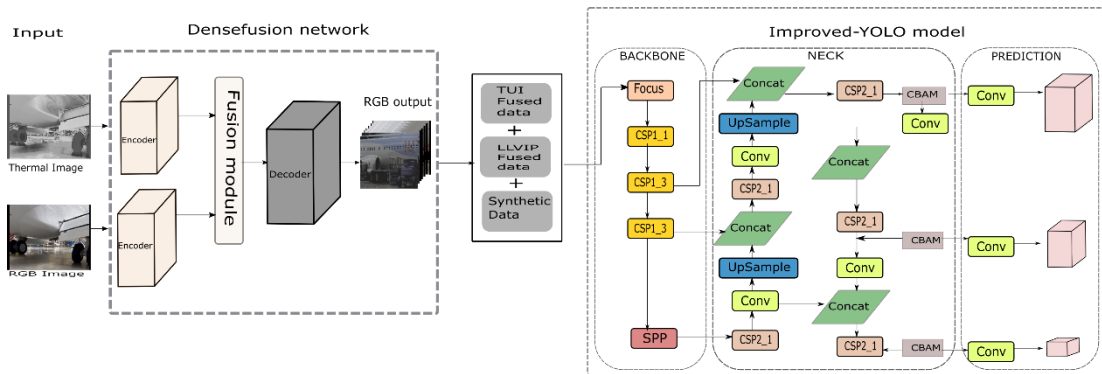


Figure 3-1. The Fuse-YOLO architecture.

3.3.2 Proposed FUSE-YOLO Object Detection Model

The proposed solution combines Densefuse network, YOLOv5m object detection model and CBAM network, to develop FUSE-YOLO model. This approach

leveraged the capabilities of each component to enhance image visibility and model performance in accurately detecting varying obstacle in changing and challenging environment even in dark areas. Additionally, the solution incorporates diverse datasets including Synthetic data, real-world TUI hangar environment data, and LLVIP datasets, to improve the robustness and reliability of the model.

3.3.2.1 Synthetic Images from the DARTeC simulation model.

An existing DARTeC hangar model was adopted in this project for simulating robot obstacle avoidance and navigation task relevant for aircraft visual inspection. The hangar model was integrated into the ROS-Gazebo simulation environment used in this study. A default turtlebot3 package was implemented and turtlebot3 simulated robot configured to navigate and capture data from the environment using camera sensor. A total of 280 images were generated and annotated using roboflow. Figure 3-2 is a representation of images captured in the simulation environment.

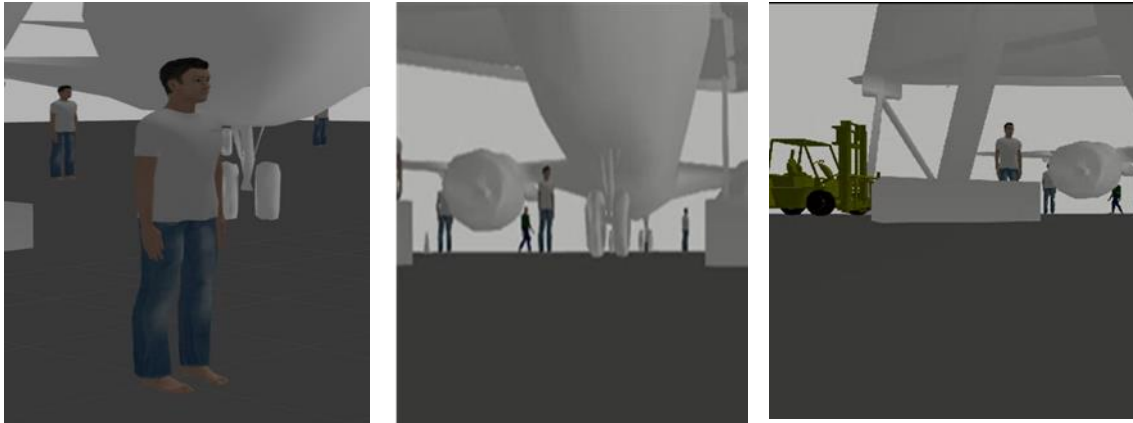
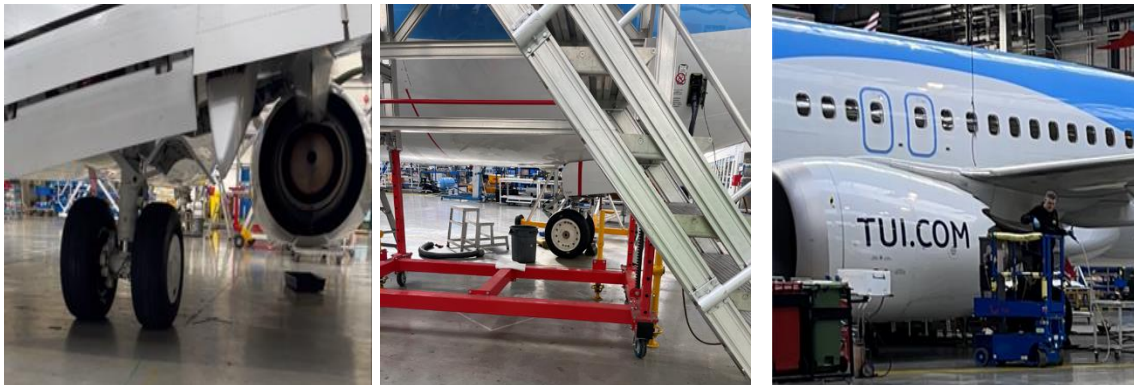


Figure 3-2: Synthetic Datasets.

3.3.2.2 RGB Images from TUI hangar

The RGB images are captured from TUI hangar using an RGB camera. TUI hangar is busy environment with changing and unstructured environment features. The generated images represent colourful features of the cluttered

environment as illustrated in figure. Additionally, some part of the images captured dark areas containing obstacles that are obscure from the visible light



as depicted in the image pairs in figure. The data collected from low light areas can pose challenges for a deep learning model especially for object detection task due to poor visibility and image quality. To overcome this, data fusion approach has shown good improvement [159]. By combining data from multiple sensors like thermal images and rgb images, visibility and feature detection can be enhanced in low light conditions. In this use case, the proposed approach involves preparing the model on large datasets including low-light specific datasets. By leveraging these datasets, the model can bootstrap its learning process and adapt more effectively to the changing hangar environment. The captured images were 365 images, and they were resized, cropped and annotated using Roboflow platform. Figure 3-3 below is an illustration of TUI hangar images.

Figure 3-3: RGB images from TUI hangar

3.3.2.3 Thermal Images from TUI hangar

The use of thermal cameras to capture thermal images offer unique advantage for obstacle detection in complex robot environments. The FLIR T560 thermal cameras was used to capture thermal images from TUI hangar [13] which includes diverse scenes, occluded areas and different types of obstacles relevant to robot's navigation tasks. They detect infrared radiations emitted by objects in the environment which makes them effective in low-light conditions but offer lower resolution images when compared to RGB cameras making it difficult to capture fine details of environment objects. The RGB and thermal image pairs in figure

3-4 are representation of thermal image complementary capability to enhance detection in images captured from dark areas. Thermal images have grayscale intensity values, different temperature distributions and provide information beyond just shape and colour [160]. The YOLOv5 model used in this study, leverage thermal images to compliment RGB images in improving obstacle detection in cluttered hangar environment.

The thermal image datasets are labelled with bounding boxes using roboflow annotation platform [161] around obstacles that the robots need to detect and navigate around. The pre-processing include image resizing, data augmentation and converting the annotated bounding box coordinates the format compatible with the input dimensions of the YOLO network. However, the traditional YOLO model is designed to process RGB images with three color channels, but thermal images use single intensity channel. Here the proposed solution introduces DenseFuse [138] network to effectively fuse the images and reconstruct it to RGB image format compatible with YOLO input format. The model is trained with these datasets to learn to detect obstacles that are in the dark or hidden parts as well as in visible parts of the environment. This enables the robot through transfer learning to adapt to the changing environment and enhances obstacle identification and avoidance.



Figure 3-4: RGB and Thermal images Pairs.

3.3.2.4 LLVIP DATASET (Thermal & RGB)

The LLVIP datasets [12] are collection of images captured in low light scenarios from RGB & thermal cameras, used to develop and evaluate computer vision algorithms aimed at enhancing images captured in dark environment where visibility and image clarity is often compromised. This publicly available dataset comprises images captured from a wide range of dark scenes and environments using RGB and thermal cameras, shown in Figure 3-5. These images showcase various real-world scenarios where low-light conditions present challenges. The LLVIP datasets was pre-processed, tailored for training and evaluating YOLO object detection model. In the proposed solution, the LLVIP datasets were fused with DenseFuse and integrated to augment the capabilities of the YOLO model in detecting objects in low-light images. By leveraging this dataset during training, the YOLO model can learn to accurately detect objects in challenging lightning conditions, thereby improving its performance and application in robot navigation in this context.



Figure 3-5: LLVIP dataset.

3.3.2.5 DATA FUSION WITH DENSEFUSE

YOLO networks are traditionally designed to process RGB images with three colour channels. Modifying YOLO to accept input from both thermal and RGB modalities with separate branches for feature extraction and fusion at different

stages of the network can be complex and computationally demanding. The use of DENSEFUSE to fuse and reconstruct image with three RGB channels was adopted for efficient YOLO model training. DenseFuse is a deep learning framework that employs a feature fusion approach that combines information from thermal and RGB images to enhance the visibility of objects [130] for applications like robot navigation. The network layer consists of multiple layers and dense connections to facilitate feature extraction, fusion and reconstruction to generate the fused image while preserving spatial details. The feature extraction is done by the encoder module where meaningful feature representations of the input images are captured. The network is configured by taking a weighted average of the RGB, green and blue channel to create a single grayscale channel. The grayscale is used as the input to the encoder layer which operates on a single channel and extract features relevant to the fusion task. DenseFuse maintains a separate pathway for RGB and thermal data at the encoder layer to process each type of data independently and extract features.

This is transferred to the fusion module, which integrates the thermal and RGB features using adaptive mechanism to capture precise details and adjust the contribution of features from each source. At the decoder layer which consists of deconvolutional layers, the fused features are upsampled by learning to generate colour channels that are consistent with the original RGB training data. The final output is presented in RGB format, incorporating features from multiple sources. The network is trained using default parameter tuning which include learning rate of 1×10^{-4} , batch size of 2 and epoch set to 4. The model adopts minimized loss functions to guide the optimization of the network parameters and to preserve feature details in the fused image. Figure 3-6 shows a resulting fused image from thermal and RGB images, which is the input to the proposed YOLOv5 object detection algorithm.

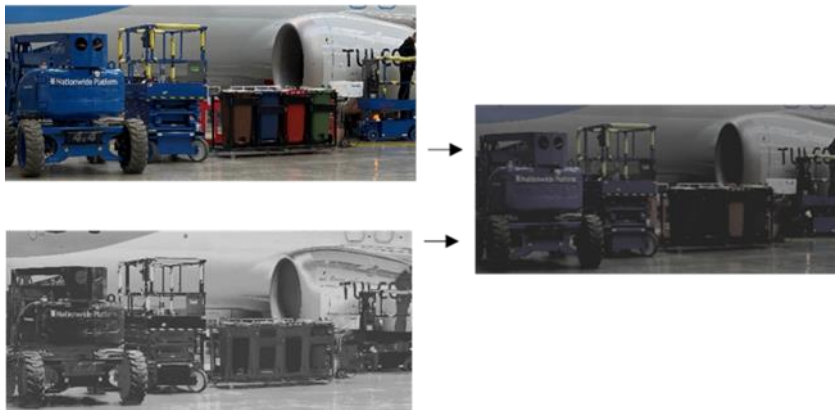


Figure 3-6. Representation of Fused RGB and thermal images with DenseFuse

The use of DenseFuse has contributed to the advancement in image fusion, deep learning and computer vision generally. Recent studies have applied the DenseFuse network to obtain high resolution fused images in data fusion process [162]. The work in [146] also used DenseFuse model to enhance data based on combined thermal and RGB images for YOLO object detection. Another work [146] applied DenseFuse model to improve the visibility of defect features and detection accuracy using YOLOv5. The same approach is applied to seamlessly fuse multiple image sources (TUI thermal/RGB images and LLVIP thermal/RGB images) to generate enhanced visible images important for YOLO object detection model. This will improve YOLO network's ability to perform object detection in cluttered and changing environments when integrated into robot platform for real-time navigation tasks.

3.3.2.6 IMPROVED YOLOv5 WITH CONVOLUTIONAL BLOCK ATTENTION MODULE (CBAM)

The introduction of CBAM was to enhance the capabilities of YOLO objection detection model, and its ability to detect objects accurately by providing it with more robust feature representation. CBAM is a module that enhances features representation in convolution neural networks [163]. It focuses on critical information by considering both the Channel attention module and Spatial attention module as illustrated in figure 3-7. The channel attention module (CAM) captures inter-channel dependencies in the feature map, assigns weight to each channel, focusing on informative channels and suppressing the less relevant

ones. The module computes attention weights for each channel of the feature maps based on its global average and global maximum pooling representation as shown in equation 1, computed with equation 2 representing modulation of the input feature map by the channel attention score. The spatial attention module (SAM) at the other hand focuses on distribution of feature within a channel, assigns weight to different spatial locations, learn to highlight important object regions while suppressing less important ones. The module combines mean and maximum pooling operation shown in equation 3 to produce attention maps [164] that is focused on extracting useful spatial information for the target object.

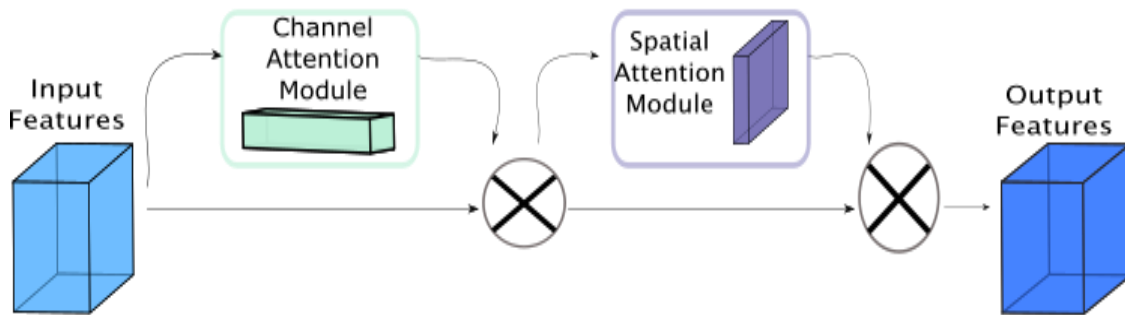


Figure 3-7: The structure of CBAM

The network structure was redesigned by assigning weights to the feature maps to improve fewer effective features and this can significantly improve detection accuracy in complex and changing environment. Studies have shown significant improvement in integrating CBAM into YOLO for accurate object detection. The authors in [164] integrated CBAM to improve YOLOv5 model detection accuracy of small objects. This mechanism was also integrated into YOLO model in another study, to improve detection accuracy apple leaf disease by extracting useful information [165] In the same order, this study illustrates YOLO model enhancement with CBAM by extracting relevant information of the target features improving the object detection accuracy.

3.4 FUSE-YOLO ARCHITECTURE

The YOLOv5 is employed in this solution as the object detection model, leveraging its real-time capabilities and accuracy in detecting objects in images. The FUSE-YOLO model integrates multiple components including DenseFuse network that generates fused images, the YOLOv5 model and the CBAM technique to focus on relevant features and improve object detection accuracy. The solution incorporates diverse datasets including synthetic data that help to augment the training dataset, real-world TUI hangar environment data and LLVIP datasets that provide realistic and challenging scenarios encountered in actual operational environment. The FUSE-YOLO leverages these datasets to learn robust representations and adapt to different environment conditions, ensuring reliable detection of varying obstacles even in the dark areas.

The FUSE-YOLO network illustrated in figure 3-1 has two modules, first is the Image fusion module using Densefuse [138] with details shown in Appendix B. This network uses encoder-fusion-decoder architecture with convolutional layers that extract rough features and dense blocks that extract high-level depth features from the input to the fusion layer. The fused feature maps are fed through a decoding network, which reconstructs the fused image using up-sampling and convolutional layers to improve visual quality. The LLVIP thermal and RGB images are fused and reconstructed as enhanced RGB output through DenseFuse, as well as the TUI thermal and RGB images. The fused TUI, LLVIP and simulation RGB images were merged and fed into the backbone region of YOLOv5 model for feature extractions. The FUSE-YOLO was Initialise with default YOLO model trained weight which is pre-trained on MS COCO [166] datasets. A redefined data augmentation structure was introduced using image cropping, flipping, rotation, and contrast adjustment for fused and simulation images. Also, CBAM attention mechanism was incorporated into the neck region of YOLO to enhance the distinctiveness of relevant features. (CBAM) [163]] as shown in Figure 3-3 and detailed in Appendix C consist of the channel attention module (CAM) and the spatial attention module (SAM) that facilitate useful features extraction for the target object. The model head region outputs detected feature maps using a fully connected SoftMax layer, applying anchor boxes with

object class probability and confidence scores. The fuse-YOLO employs iterative hyperparameters tuning in the experiment to train and optimised detection accuracy. This is saved as a pretrained YOLO weight that can be beneficial in bootstrapping object detection from camera feed in ROS and robot's environment for improved obstacle detection and avoidance.

3.4.1 Data Preparation

The yolov5 model is trained on diverse datasets including simulation environment data, real-world TUI hangar environment data, and LLVIP datasets, to ensure robustness and adaptability to different environmental conditions and scenarios. The use case dataset was generated from TUI hangar [13] with standard RGB and FLIR T560 thermal cameras. The datasets consist of 365 pairs of thermal and RGB images and 365 fused images and was manually annotated based on 13 categories of objects. This includes crowded images with instances of objects found in maintenance, repair and overhaul (MRO) hangar environments such as Chairs, airstairs, persons, waste-packs, cables, materials, stands, toolboxes, aeroplane-body, ground-vehicles, trolleys, electricals and tires. We use customised data augmentation techniques to increase the diversity of the training data, which was relatively small and introduce the LLVIP public dataset [12] obtained for object detection in low light conditions. The LLVIP datasets contains 30976 images with 15488 pairs by default, but 1500 pairs were used in this study. This results to 1500 fused images which was annotated with four object classes including person, car, bicycle, and motorcycle and pre-processed to YOLO model training format. It was added to increase the number of training samples, assist the model in learning to recognise a broader range of objects and handle various scenarios. The synthetic RGB images include 7 objects classes including ground-vehicle, cone, stand, tire, wall, person, and airstairs. All the images were labelled and pre-processed with Roboflow annotation platform. The images were resized to 640x640, and uniform format of preprocessing was utilized to maintain consistency in the network.

Datasets:

- LLVIP Dataset – Low-light object features (1500 fused images, 4 object classes)
- TUI hangar dataset – custom (365 fused image, 13 object classes)
- Synthetic datasets (simulation) – 280 RGB images, 7 object classes.

Preprocessing:

The RGB images were configured for DENSEFUSE network by taking a weighted average of the RGB, green and blue channel to create a single grayscale channel and images were reconstructed to RGB format after fusion. For YOLOv5 model training;

- Resizing - All the images were resized to 640x640 dimensions to maintain consistency.
- Augmentation –Image cropping, flipping, rotation, and contrast adjustment for all the images.

The datasets were first added together for training with FUSE-YOLO including 1500 fused LLVIP, 365 fused TUI images and 280 RGB simulation images to give 2145 images. The 2145 images were divided into training, validation, and testing sets at a ratio of 80:10:10. The custom training dataset contained 1717 images, and the test and validation datasets contained 214 images, respectively. The next training was done using TUI RGB and Thermal images separately. The 365 TUI RGB images as well as the 365 thermal images were divided into 293 for training, 36 images for validation and 36 for testing.

3.4.2 Relevance of the datasets

The TUI hangar datasets and the simulation environment data represents the robot operational environment in real-world and simulation environment respectively. The TUI data provides accurate representation of the challenges, variations and complexities that might not be able to replicate in the simulation environment. On the other hand, the simulation data are generated from the DARTeC hangar simulated environment with different objects to compliment and

expand real-world data. This data is used to improve the accuracy and efficiency of the YOLO models, before deploying it in real-world applications. The LLVIP dataset includes a wide range of real-world scenarios captured in low-light conditions. Training the YOLOv5 model on such realistic data allows the model to learn robust representation of objects in varying lightning situations and this improves model performance.

For all the thermal images used, the relevance is to detect obstacles in low-light situations. The thermal images give the model an understanding of the obstacle structures in dark situations, improving object detection in dark areas of the hangar environment. The diversity and volume of images used in training the FUSE-YOLO model ensures that the model learns to detect objects accurately in different environment and help the model to adapt to different environmental factors when deployed in real-world.

3.5 EXPERIMENT AND RESULT

To train our models and guide our studies, we used a dataset of bounding box annotated objects from the perspective of an autonomous mobile robot navigation for aircraft visual inspection in a hangar environment. The software configuration of the proposed model includes PyTorch framework, python 3.9, and CUDA version 11, running on the Windows 10 operating system (OS). The OS runs on Intel(R) Core (TM) i7-10750H CPU working at 2.60GHz with 32G memory. And Jupyter notebook was used to train and evaluate the model. To ensure that our trained model can detect different kinds of objects, we initialised the model trained with a pretrained weight based on MS coco datasets which is the default state of YOLO models [167].

3.5.1 Model Training

To ensure our trained model can detect different kinds of objects, we initialised the model trained with a pretrained weight based on MS coco datasets which is the default state of YOLO models. The pretrained weight combined merged images from the DENSEFUSE output which includes fused LLVIP datasets and fused TUI hangar datasets and in addition, the simulation environment data. The

Fuse-YOLO algorithm and baseline YOLOv5m model were first trained on the merged data with image resolution of 640, batch sized of 16 and 100 epochs. This is to establish a standard weight for FUSE-YOLO model. The image resolution of 640 pixel means that each input images are fed into the algorithm at a consistent width and height of 64 pixels. The batch size accounts for the number of training samples processed at each iteration of the training algorithm. This balances between training efficiency and memory constraints. Training for 100 epochs indicates that the mode iterated over the entire datasets 100 times and this choice ensures that the model has sufficient exposure to the training data to learn meaningful representations. During trainings, I used the Adam optimiser with momentum at 0.9 and learning rate at 0.005 for the 100 epochs.

The second phase of the experiment involves training each of the YOLOv5 variants (Medium, Large and extra-large) and FUSE-YOLO on only fused TUI images and the performance outcome was shown in Table 3-1.

Table 3-1: Result Comparison between YOLOv5 models and FUSE-YOLO on fused TUI RGB_thermal images.

Models	mAP(0.5)%	mAP(0.5-0.9)%	P(%)	R(%)
Yolov5m	65.8	48.8	72.9	64.6
Yolov5l	63.9	48.8	67.5	66.9
Yolov5x	65.1	50.8	72.5	63.4
Fuse-YOLO	68.1	47.9	63.4	74.9

The last phase of training includes training FUSE-YOLO algorithm and baseline YOLOv5m model independently using each of TUI RGB and thermal images and using the same standard parameters of 640 pixels, batch 16 and 100 epochs set for each training stage. The fused LLVIP images were also used to train the FUSE-YOLO and baseline model to evaluate and compare the performance outcome. The best weight from each of the trained models was saved and used to detect the accuracy on test images. The result from these experiments based on detection accuracy is detailed in Table 3-2 below.

Table 3-2: Comparative performance of YOLOv5m and Fuse-YOLO on four datasets based on (mAP)_{0.5}.

Datasets	YOLOv5m (%)	Fuse-YOLO(%)
TUI RGB-Image	58.2	61.8
TUI Thermal-Image	47.4	49.1
Fused TUI RGB_Thermal	62.5	64.2
Fused LLVIP	73.3	75.6

3.5.2 Evaluation Metrics

To evaluate the performance of the proposed fuse-Yolov5 model, we used the custom and public LLVIP datasets to compare the detection accuracy and computational efficiency. We will review some of the evaluation metrics we employed in our experiment that are widely used to evaluate the performance of object localisation. The classification model includes precision and recall, which measure the percentage of accurate positive detections made by the algorithm and the rate of accurate positive detections made by all ground truth objects, respectively. Next is the Intersection over union (IoU) concept that measures the overlap ratio between the predicted boundaries and ground truth bounding boxes to the union region. It calculates the correctness of object localisation and is used as a threshold value for accurate measurement with mean Average Precision (mAP) [168] The mAP represents the standard metric for object detection accuracy evaluation of a detection model. The speed with FPS (Frames per second) is another evaluation metric relevant to real-time applications. See the formulas below representing the evaluation metrics.

$$\text{Precision} = \frac{TP}{TP+FP} \quad (3-1)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3-2)$$

$$mAP = \frac{1}{N} \sum AP \quad (3-3)$$

$$AP = \int_0^1 Pre(Rec) dRec \quad (3-4)$$

$$IoU = \frac{target \cap prediction}{target \cup prediction} \quad (3-5)$$

Where TP = True Positive is the number of correct objects detected, FP = False Positive is the number of falsely detected objects, FN = False Negative is the number of missed objects, and AP = Average precision (average accuracy of each object type).

3.5.3 Results

In each experiment we trained, evaluated and tested the performance of our model. The YOLOv5m models and FUSE-YOLO were trained and tested on four different datasets to validate performance. Also, three variants of YOLOv5(m, l, & x) and FUSE-YOLO were trained on fused TUI images and performance were compared based on their detection accuracy. The results were recorded and evaluated to compared performance with baseline YOLOv5m models. The same evaluation metrics which includes precision, recall and mAP were applied where the mAP IoU threshold was set to 0.5 for each category. In Table 3-1, the FUSE-YOLO showed significant improvement in detection accuracy of about 68.1% against the 65% mAP average accuracy recorded for the YOLO variants. This was graphically represented in figure 3-8 showing improvement with data fusion (DenseFuse) network and CBAM where both enhanced feature representation of images and focus on capturing informative object features, and this improved the overall detection accuracy of the model.

In Table 3-2, Different datasets were used to compare FUSE-YOLO performance with the baseline model. Focusing on the TUI datasets which is the operational environment data, The FUSE-YOLO showed better accuracy on all the datasets when compare with the baseline YOLOv5m. The improved detection accuracy was more significant on the fused TUI dataset with 64.2% accuracy over 62.5% recorded for YOLOv5. This is further illustrated in figure 3-5 showing visual display of detection accuracy with confidence score. The fuse TUI images showed detection of more object at higher accuracy compared to others, and this demonstrates the effectiveness of DenseFuse network in enhancing performance of the YOLO model object detection task. The evaluation with LLVIP was done

as well. However, it showed higher accuracy due to limited complexities in the images. The images have 4 classes and less cluttered structure making object detection task more straightforward, leading to higher accuracy.

3.6 DISCUSSION

The FUSE-YOLO improved mean average precision (mAP) by 1.5% on the fused TUI custom dataset and 2.3% on the LLVIP, and the precision dimension showed slight improvement. The general accuracy was minimal by several per cent in some categories, especially for objects like stands that represent different frames of stands, while other object classes showed significant improvement. The proposed model performed better in accuracy than the YOLOv5 classical models with significant variations. FUSE-YOLO detected varying objects through the fusion network, although more improvement is required in especially in data annotation as this will increase the model performance. The overall experiment results, showed that the proposed FUSE-YOLO can significantly increase the identification and detection of objects in complex settings, proving that the method can improve obstacle avoidance in robot navigation as is in this context.

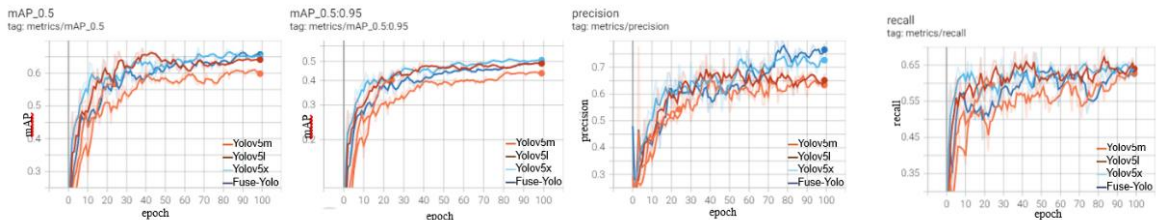


Figure 3-8. Performance comparison of the yolov5 models and Fuse-YOLO based on Fused TUI images.

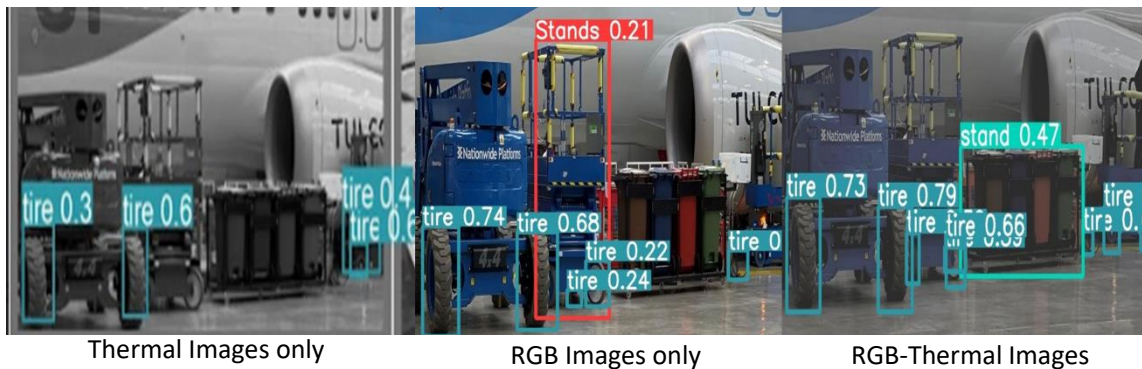


Figure 3-9. Object detection results on the model test using RGB_Thermal image, thermal and RGB image.

3.3. CONCLUSION

The motivation of this work is to develop a system that improves the accurate detection of objects of different structures in light variations, and this is a significant challenge in the design of autonomous mobile robots for aircraft visual inspections. The research proposes a multi-spectral convolutional neural network called fuse-YOLO that combines the features information of RGB and thermal images. The fuse-yolov5 model was developed based on the Yolov5m network and can detect objects varying at different light variations compared to the baseline model. The work significantly improved real-time object detection accuracy by combining RGB and thermal images to help identify features under bright and low light intensity, respectively.

The experiment's result demonstrated the algorithm's efficiency and was verified based on the custom and LLVIP public datasets. Compared with the standard Yolov5 network, mAP's overall value was improved by approximately 2%, and most object classes' AP were slightly improved. Our model shows great potential in identifying objects even in low light states and facilitates application in optimising robot obstacle avoidance and motion control decisions in complex environments. Further work will be done by training the images independently and fusing the output at feature fusion or decision fusion level to improve model performance and generalisation capability.

PART 3: INTEGRATION OF YOLO MODEL INTO ROS NAVIGATION STACK

In this part of the projects, the developed and improved YOLOv5 model was integrated into the ROS ecosystem. The process of developing the model called NAV-YOLO was covered in chapter 4 and 5. Chapter four focused on establishing the concept and working model. However, there were performance issues that are related to detection accuracy, lag in data synchronisation and slow navigation. These were the focus in Chapter five and prompted integrating more recent YOLOv7 model to improve detection accuracy and real-time capability. RRT ROS planner was also used in the system development to enhance path planning. The project highlights include;

- Integration of YOLOv7 into NAV-YOLO
- Integration of RRT into NAV-YOLO
- Develop improved NAV-YOLO-RRT
- Performance evaluation
- Transfer to TurtleBot physical robot
- Comparative analysis.

Paper published in: <https://www.ijmerr.com/2024/IJMERR-V13N2-219.pdf>

4 Mobile Robot Obstacle Detection and Avoidance with NAV-YOLO

This paper proposes the NAV-YOLO system, a vision-based navigation model integrating a pre-trained YOLOv5 detection model with the ROS navigation stack, aiming to optimise obstacle detection and avoidance in MRO hangar environments. This paper used the pre-trained YOLOv5 model developed in Chapter 3, focusing on improving the detection and avoidance of varied obstacles in a hangar environment with the robot navigating and performing an inspection task. The ROS-YOLO integration in this project utilised a camera stream to detect environmental information. The detected data from the integrated YOLO model with bounding box coordinates are transformed to laser scan data format to update the ROS planners with real-time environment information, enhancing the robot's ability to detect and avoid obstacles. This paper demonstrated the model's effectiveness by validating the performance using ROS-Gazebo simulation and Turtlebot3 waffle-pi robot platform. The paper showed improved performance results for obstacle detection and avoidance, enabling the robot to navigate through predefined checkpoints to its target location without collisions in a modelled hangar environment.

4.1 Introduction

The application of mobile robots for visual inspection has been adopted in MRO (Maintenance, Repair, and Overhaul) hangars, where a significant level of autonomy and automation is required to navigate safely environments with irregular object structures and varying complexities. This implies that the mobile agents must autonomously observe, detect, avoid, and make control decisions

from multisensory information to make collision-free navigation in unstructured and dynamic scenes. One of the critical navigation challenges is detecting and avoiding obstacles in complex and changing environments, such as busy hangars, where obstacles of varying shapes and sizes can lead to potential collisions and safety hazards. Efficient mobile robot navigation requires robust perception capability [169], and most recent approaches use vision-based sensors like RGB cameras and LiDAR to gain an understanding of environmental features and improve object detection, avoidance, and navigation techniques [170]. Laser devices are more costly, and conventional algorithms based on this information source are usually complex. On the other hand, RGB cameras provide depth and feature-rich environmental information and are widely used for developing robot navigation models. However, extracting accurate obstacle information from these input devices and modelling it for a reliable and safe navigation experience is still widely studied.

Over the years, conventional vision-based navigation systems based on the ROS framework have been widely used. ROS is an open-source robotic platform that was first released in 2007 [171] to provide classical support tools, packages, and libraries ready-to-use [4], [172] for developing robotic applications. The ROS navigation stack is an aspect of the ROS ecosystem with a toolset and functionalities that equip robots to interact with their environment, avoid obstacles, and plan paths with reasonable intelligence [173]. This system's accuracy, reliability, and environment suitability depend on obstacle detection, collision avoidance and path planning algorithms. The default ROS (Robot Operating System) navigation stack uses A* (A-star) global and Dynamic Windows Approach (DWA) local planning algorithms [174] where the latter employs a local search strategy that evaluates potential trajectories based on robot dynamics, environmental constraints, and safety margins. DWA has been widely used in different domains and has shown promising results in robot local path planning and obstacle avoidance cases. However, it's challenged with scenarios that require complex navigation behaviours, accurate real-world obstacle identification and handling of dynamic obstacles. Also, achieving good performance with DWA requires handcrafted parameter tuning to suit changing

scenarios, limiting its capability in real-world applications. Recent advancements have applied deep learning [175] techniques with Convolutional Neural Networks (CNN) to learn deeper feature representations from environmental data for object classification and detection [176] to enhance intelligent navigation. The YOLOv5 (you look only once) model is among the most widely adopted systems for object detection [14] struggles with detection accuracy in complex and changing environments, mainly where varying object types and lighting conditions are critical factors. Different hybrid models have been proposed but are still limited in performance. The primary focus of this research is to design and develop an advanced object detection and obstacle avoidance system that transforms YOLOv5 detection output into laser scan data suitable for ROS navigation. This will create a robust and accurate real-time system that can effectively detect obstacles of varying shapes and sizes in different lighting conditions to ensure maximum safety and efficiency in the face of uncertainties. In our recent work, we developed a fused CNN [14] that uses image fusion from RGB (red, green, and blue) and infrared data to improve object visibility, detection accuracy, and feature extraction using the YOLOv5 object detection model [14]. The fuse-YOLO solution is integrated into the ROS navigation stack in this part of the project for efficient object recognition and avoidance.

This study presents a vision-based navigation model that employs a pre-trained YOLOv5 model and ROS navigation stack to improve obstacle detection and avoidance for safe navigation in a complex environment. This solution uses an RGB-D camera as the primary perception modality, providing rich environmental data through real-time visual feedback to enhance the robot's situational awareness in busy and cluttered hangar environments. By integrating the YOLOv5 object detection algorithm, the robot can respond proactively and accurately to varying and irregular obstacles. The YOLOv5 object detection model uses deep learning techniques to identify and classify objects [29] in the robot's field of view, enhancing robot perception detection and avoidance capabilities [177] The object detection node is created in the ROS framework to publish detected objects' positions and sizes. It is transformed into a laser scan data representation that interacts with the DWA planner in the ROS navigation

stack. This provides a complete navigation pipeline for the robot to improve path planning and obstacle avoidance using sensor information dynamically. The contribution is to enhance the obstacle detection and avoidance problems in the complex aircraft hangar environment using coordinate transforms from camera frames to map frames to improve the accuracy and reliability of the obstacle detection and avoidance mechanism. The complete research process is shown in Fig. 1. First, the OpenCV AI Kit with Depth (OAK-D) camera [178] is embedded in the turtlebot3 waffle_pi robot to gather environmental information through the detection node created with a pre-trained YOLOv5 model [14].

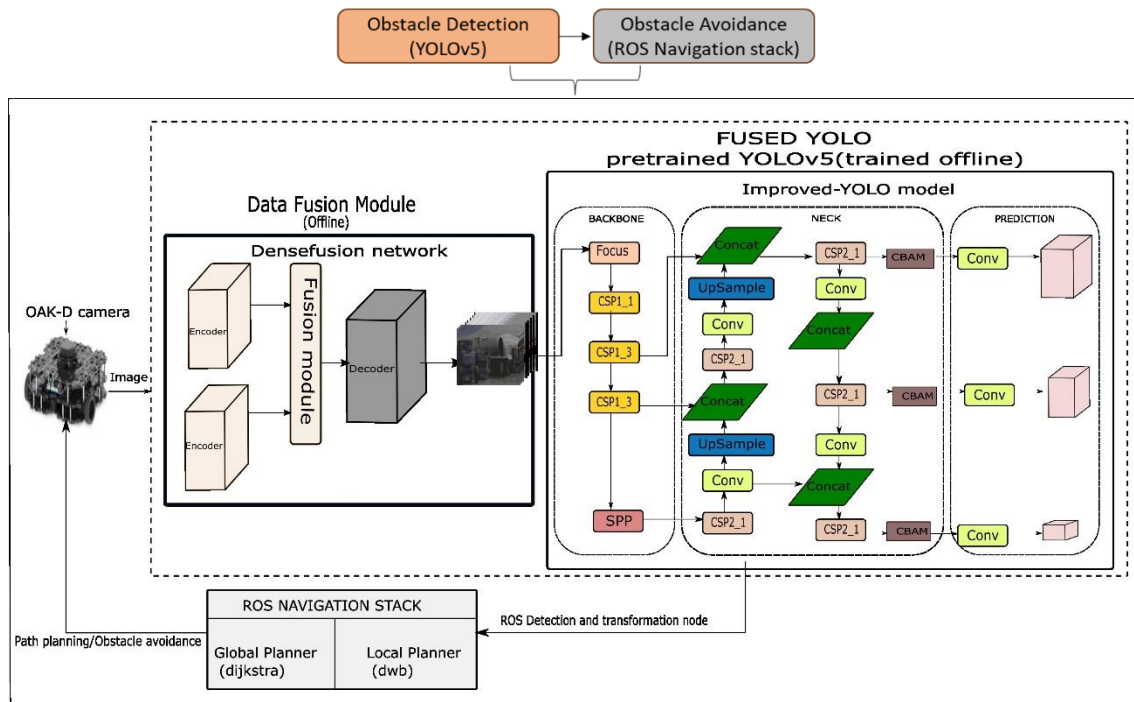


Figure 4-1: Integrated YOLO and ROS navigation framework.

The output with bounding box representing object size, position, and distance estimation is converted to interface with the costmap parameter node in the ROS navigation stack to detect obstacles accurately and plan a smooth obstacle avoidance path for the mobile robot. The detection and avoidance capabilities were benchmarked with YOLOv5 baseline variants and the default ROS navigation stack. The performance is evaluated, analysed, and validated with a real robot in an office environment. The overall contribution includes:

- Integrate the OAK-D stereo camera in the Turtlebot3 waffle_pi robot to obtain environmental information.
- Create a ROS node that generates object detection results using pre-trained YOLOv5 weight.
- Develop an improved navigation algorithm by transforming object detection results to laser scan format for the navigation stack's local planner to improve obstacle detection and avoidance.
- Evaluate and validate the solution using baseline YOLOv5 versions and DWB local planner.

The remainder of this paper is organised as follows: Section II briefly reviews the related works for object detection and the ROS combination approach in mobile robot navigation. Section III introduced the necessary background information on YOLOv5 and the ROS navigation stack, along with a detailed overview related to the proposed methodology. Section IV demonstrates the experimental and evaluation of the effectiveness of the proposed algorithm. This is followed by Section V, which dwells on the result and discusses the achieved result. Finally, section VII presents the summary and future work of the paper.

4.2 Related Work

The requirement for obstacle detection and avoidance for mobile robots to safely navigate in complex scenarios is a crucial area of attention in robotics research. In real-world operations, accurately detecting static and dynamic obstacles is still a bottleneck. Conventional obstacle detection employs mathematical approaches, limiting detection efficiency and extracting detailed obstacle features. Recent profound learning-based obstacle avoidance developments have gained significance in research and application[179]. Image feature representations are learned through stacked convolutional layers to realise the description of environmental-complicated objects [180]. For instance, Madawi *et al.* [170] proposed the use of Faster RCNN, which is a region-based convolutional neural network CNN (R-CNN) and multi-stage object detector model improved to

detect and avoid hazardous objects using a custom image dataset for an indoor mobile robot operational environment. Henke *et al.* [181] proposed a safe navigation system with YOLOv2 that recognises objects and calculates their distances to the robot for a control decision. However, these algorithms did not consider the object's height and size, which are critical for quality detection and safety in a complex setting. To improve the detection capability, Hairol *et al.* [182] proposed an object detection and avoidance algorithm that can give control instructions to a mobile robot but is limited to objects with either square or rectangular shapes. This relates to traditional obstacle avoidance methods based on mapped environments and struggles with environmental changes and dynamic structures. Hybrid solutions have been studied to detect and avoid irregular-shaped objects to address this for real-world applications. Luo *et al.* [183] developed Contextual-YOLOv3 with improved environment perception capability to help identify even small objects and enhance obstacle avoidance techniques for an intelligent inspection task. Aghi *et al.* [184] used the ROS navigation system and CNN techniques to create a resilient navigation model that is power efficient, and Lee *et al.* [185] integrated a pre-trained deep convolutional neural network to detect and classify obstacle types for efficient obstacle detection and avoidance. In general, research on obstacle and avoidance for mobile robot navigation is still widely studied, and the introduction of object detection models to improve performance still requires further improvement. This paper proposes a new robust and accurate real-time object detection and avoidance method based on the YOLOv5 and ROS navigation stack that strikes a balance between accuracy and efficiency, enabling the robot to navigate smoothly and safely in dynamic environments

4.3 Methodology

For a safe visual inspection with mobile robots, accurate perception and an avoidance system are required to navigate autonomously in a complex environment. This paper proposes a combination of object detection and avoidance methods, as shown in Figure 4-1, where a pre-trained and improved YOLOv5 model [14], trained with custom datasets, was integrated into the ROS

framework for real-time obstacle recognition and avoidance towards improved navigation for aircraft visual inspection tasks. The study compares the proposed method against the existing default navigation stack planners: the A* algorithm and the traditional DWA.

4.3.1 YOLOV5

YOLOv5 (*You Only Look Once*) is one of the releases of the YOLO family developed by Redmon *et al.* [79], a single-stage object detection model that can learn image features at different scales. Compared to other detection algorithms, YOLOv5 is widely used for its fast inference time and low computer resource demand, making it efficient for real-time applications [186]. It was built on the PyTorch framework and uses the CSPDarknet53 backbone structure to extract feature maps from the input image. The neck region architecture is designed with a path aggregation network to improve the detection of objects of different sizes. At the same time, the head structure consists of prediction layers that generate bounding box coordinates, class probability and object confidence score [187]. The model employs many techniques to improve detection accuracy, like anchor boxes, post-processing techniques, optimisers, and mish activation functions. It performs significantly well in detecting different object structures and uses non-maximum suppression to suppress multiple detections of the same object. YOLOv5 contains small, medium, large, and extra-large types distinguished by their respective network structures. An improved YOLO5s was applied in this context, which was developed by integrating the DENSEFUSE network [138] and the CBAM-based YOLOv5s technique [164]. Fused environment-specific thermal and RGB images and LLVIP datasets [12] were trained for efficient obstacle detection and avoidance capability for a mobile robot navigating in a cluttered hangar environment with lightning variations. The package generates output from a ROS topic named `/yolov5/yolov5`, disseminating crucial information like bounding box coordinates and object detection probability value from the processed camera feed.

4.3.2 Ros Navigation Stack

The ROS navigation package is a configurable framework that houses modules that manage autonomous navigation system capabilities for mobile robots. This includes the move-base sub-package that provides functionalities for action servers, path planning, obstacle avoidance and motion control modules [188]. Different path-planning algorithms can be implemented, and navigation parameters can be configured based on the application scene. The ROS framework, by default, consists of Dijkstra global and DWA local planners. These are used as the baseline in this context to compare the navigation abilities of mobile robots through obstacle routes without collision. The Dynamic Windows Approach (DWA) is widely used for robot motion planning and obstacle avoidance [189]. It employs a local search strategy that allows the robot to select the best trajectory in real time while avoiding collisions with detected obstacles. Comparing its capability with other ROS local planners, such as potential field [190], DWA uses a velocity-based method to determine a collision-free path towards the objective against the way following the approach of the possible field. The robot can dynamically access and choose viable velocities that steer it towards the intended destination while ensuring obstacle avoidance [76]. The combination with the improved YOLOv5 object detection model enhances the mobile robot's obstacle detection and avoidance strength and overall navigation behaviour, which are critical for safe navigation.

4.3.3 Transfer Learning

Intelligent robots often face resources constraints like limited processing power and sensor capabilities, making it challenging to run large object detection models efficiently. Recent studies have applied transfer learning to address these issues, allowing robots to benefit from pretrained models and adapt them to their specific tasks. Transfer learning is imitating human ability to learn and adapt to different environments regardless of their primary sense like a blind person can navigate a new environment using touch and sound based on prior knowledge. Pretrained models are models developed based on experience and are trained on massive datasets, to learn general features and object categories. In this study, diverse

datasets of images encompassing the objects and scenarios that the robot will encounter in its operational environment including LLVIP public datasets, simulations, and manually captured TUI images, were used. These were used to pretrain YOLOv5 model enabling the model to leverage its experience with different environments to improve its performance in real-world application. The pretrained YOLO model was transfer to the robot platform to reduce real-time complexities of training and adaptation to different sensor perception of environmental features, through knowledge learned from different sources including thermal and RGB images.

The OAK-D camera integrated on the robot platform allows real-time perception and processing of environment data leveraging pretrained YOLO model. Densefuse network was trained with thermal and RGB images and not connected to any sensor. The OAK-D camera was used in the ROS framework via robot package to perceive environment information at real-time and the pretrained model aided fast data processing based on learned experience. This eliminates training from scratch in embedded devices considering computational demand, facilitating perception and detection accuracy. The YOLOv5 model by default was developed and trained with MS COCO datasets [166]] promoting transfer learning and eliminating the complexity of training from scratch. In [191], the study utilized synthetic learning approach to allow the model to learn from datasets establishing learning that is transferred to robot for semantic segmentation and object detection. Another project [192] used transfer learning from model that is pretrained on synthetic data to improve autonomous soccer robot by allowing it use prior knowledge about useful objects. Yazid et al. [193] adopted transfer learning through pretrained YOLO model to improve on detection accuracy and low latency performance for robotic manipulator. Also, in [194], parts and tools recognition using a controlled robot was implemented using a transferred knowledge from a pretrained CNN model and this improve detection accuracy in the operation. Adopting transfer learning in this research work yielded significant improvement including minimized data processing complexity, harnessed system adaptation to changing environment, and obstacle detection and path planning improvement.

4.3.4 The Proposed Method

We aim to develop a real-time object detection and avoidance system to overcome object irregularities and light variations in a complex environment when carrying out an autonomous navigation task. The core idea is that it can pick out where the obstacles are and create a trajectory to avoid them accordingly. This will improve its ability to detect the barriers and augment the trajectories it produces.

4.3.4.1 OAK-D Camera

This paper uses the DepthAI OAK-D camera [178] as the primary vision-sensing device to capture images or video footage of the robot's environment. The camera is specifically designed for conducting depth estimation, object detection, and tracking experiments that effectively promote the obstacle detection and avoidance tasks required in this case study. Cameras are notable for high resolution and detailed information compared to LiDARs, especially when tracking fast-moving objects accurately in real time. The data from the LIDAR is used by default to update the local planner costmap, which is a 2D grid-based representation of the local environment around the Turtlebot in ROS navigation stack. The costmap assigns cost values to each cell in the grid based on the information received from the LaserScan message. Obstacles are assigned higher costs, while free spaces are given lower costs. A mechanism is established to convert the objects detected within the camera's field of view into relevant scan-based information to achieve this capability with a camera device.

From the simulation perspective, the obstacle detection and avoidance system process begin with integrating the OAK-D DepthAI ROS repository developed for ROS into ROS navigation nodes as the data stream source for object detection. This is configured to communicate with the YOLO model through ROS image message using /camera/image_raw topic. The improved YOLOv5 model used in this work was developed to detect varieties of object structures from the camera in changing and light variation environments. These are set up in the ROS

simulation environment [195] to acquire information from the environment as ROS image message using /camera/image_raw topic, which is converted to a suitable YOLOv5 readable format and resized using the OpenCV package. With the pre-trained YOLOv5s package that includes model weights and configuration files, the ROS detector node process generates information to identify and describe objects in the environment. The detection output in the form of a ROS topic named /yolov5/yolov5 contains structural properties like the bounding box coordinates, class probabilities and object detection probability values that are transformed to generate a safe path to the target location [121].

YOLO's Detected Result to Laser Scan Data

The detected object transformation starts by fetching the positions of the objects detected by the YOLOv5 package and injecting them into the local path planning algorithm. The DWA planner, by default, takes laser scan data as input from the LiDAR device to steer the velocity commands [125]. The detected YOLO result is converted to laser scan data and reprogrammed in the move-base package to improve the DWA planner function as displayed in Figure 2 workflow. The /yolov5/yolov5 topic, which is of vision_msgs/Detection2DArray of ROS object detection message type, generates the bounding boxes' pixel coordinates, among other features and computes the centre of the bounding boxes. To increase the robustness of the solution and ensure more accurate laser information, the left and right parts of the pixel coordinates of the obstacles were also computed. This provides precise detection of varying shapes of obstacles within the 10m detection range of the camera device. The pixel coordinates are converted to camera coordinates with correspondence to the depth image captured by the RGB-D camera that provides the distance of each pixel from the camera's optical centre. This conversion is facilitated using camera intrinsic and extrinsic parameters provided by turtlbot3 Waffle Pi sensor_msgs/CameraInfo topic. This contains the focal lengths f_x and f_y and the principal points C_x and C_y

that are used to compute the distance point of the objects as shown below in equations 1 and 2.

$$\text{Camera coordinate in } x = \frac{(\text{Pixel coordinate in } x - C_x)}{f_x} \quad (4-1)$$

$$\text{Camera coordinate in } y = \frac{(\text{Pixel coordinate in } y - C_y)}{f_y} \quad (4-2)$$

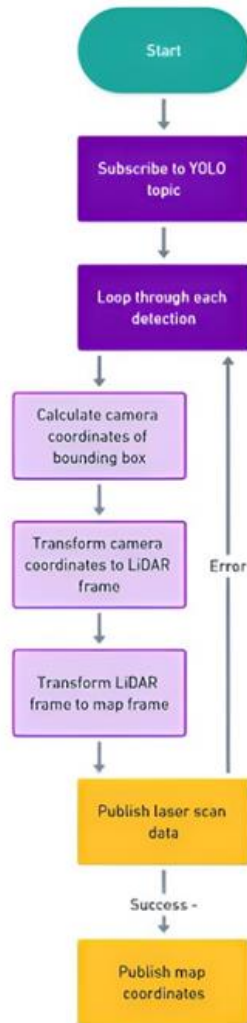


Figure 4-2: YOLO object detection to scan data conversion workflow.

Once the camera coordinates were obtained, the ROS tf2 library, which provides built-in tools to transform the camera data into scan data, was employed to automate data transformation between frames. With the 3D coordinates of the detected objects in the laser scan frame, the system can calculate the Euclidean distance from each object to the robot's centre. The Euclidean distance formula

in equation 4-3 is applied to determine the straight-line distance between the detected object and the robot's centre. The LaserScan message is generated and published on a specific ROS topic to allow interface with other ROS nodes or applications.

$$distance = \sqrt{(X - x_o)^2 + (Y - y_o)^2} \quad (4-3)$$

Where (X,Y) represent the transformed coordinate of the 3D point and (x_o,y_o) represent the origin of the robot's centre which in this case will be (0,0). To be sure of the consistency of the data converted, the LaserScan data corresponding to the position of the bounding boxes were also converted to map coordinates and printed. Then, the robot was placed in front of an obstacle of a known position. The map coordinates printed were slightly different by an avoidable few centimeter from the obstacle coordinates.

4.3.5 Obstacle Avoidance

The process of detecting and transforming the object and generating distance values is repeated for all the pixels in the camera frame, essential for the DWA local planner to understand the immediate obstacles and free spaces surrounding the robot. The transformed /scan node that generates coordinate frames in suitable format for path planning and obstacle avoidance in the navigation stack is configured in the costmap file of the move-base package. The navigation stack capability is enhanced, and the ground robot from the experiment navigates through different checkpoints while avoiding detected obstacles with significant accuracy. The resulting solution is a lightweight system that can detect small to large environmental objects and respond in real-time to achieve a safe autonomous navigation experience. Performance metrics were defined for comparing the models, such as mean average precision (mAP), inference speed, time, distance covered, and number of obstacles detected. Both accuracy and efficiency were considered to strike a balance between performance and real-time capabilities.

4.4 Experiment

Aiming at a solution that will solve the navigation problem of obstacle detection and avoidance, the integrated solution was evaluated and validated through systematic experiments. These experiments were conducted on a computer running an Ubuntu 20.04 64-bit operating system, ROS noetic, and powered by an 8th generation Intel® CPU, with YOLO packages, OAK-D dependencies and ROS requirements adequately configured. To measure the solution's detection accuracy and collision avoidance efficiency, the robot was navigated from its initial position to a designated goal within the hangar setting. The average time taken for the robot to reach its destination was approximately 15 minutes. Throughout this process, the models' performance was observed in correctly identifying obstacles and instances where obstacles were not detected at all. We used common performance indicators for the object detection model and avoidance measures in the ROS framework, including the number of obstacles detected and avoided per distance and time, intersection over Union (IOU), precision, and mean average precision.

We implemented our proposed model in the ROS_Gazebo simulation environment with the Cranfield DARTeC aircraft hangar virtual world model as shown in Figure 4-3, and a total of fifteen obstacles, 12 static and 3 dynamic obstacles, were spawned in the environment at an increasing rate. The obstacles include a moving person and other aircraft hangar-related object types. The turtlebot3 robot navigation experience was tested in three different environments with varying complexities and 5 checkpoints each, to study the behaviour of the robot at different instances. The Adaptive Monte Carlo Localization (AMCL) pose of the robot was recorded using the rosbag tool. The AMCL is a localisation algorithm used in ROS to obtain more accurate information about the robot's pose in its environment. The robot poses were plotted to compare the path taken for the two methods and the total distance covered by the turtlebot3 was computed

using the Euclidean distance formula shown in equation 3 to determine the shortest path.

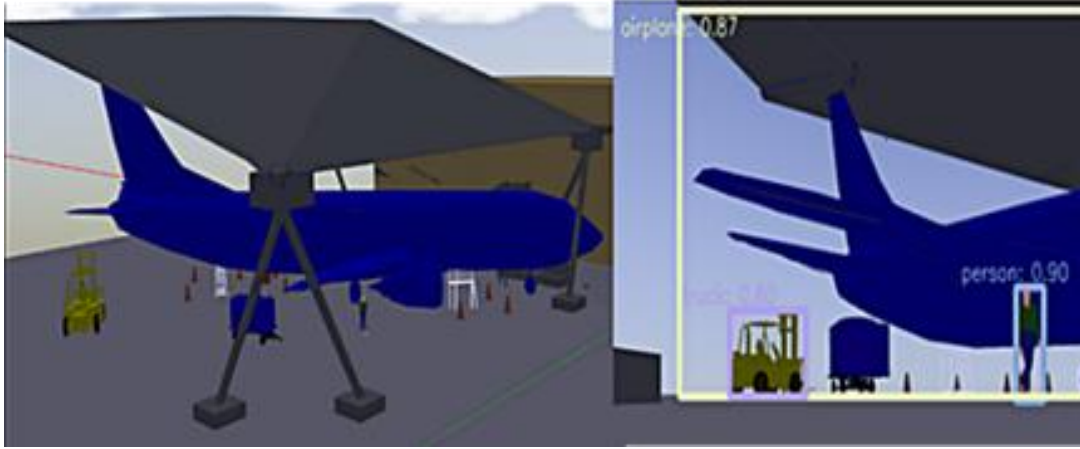


Figure 4-3: Robot obstacle detection in a hangar environment.

Finally, the total navigation time for each of the methods and the number of collisions were recorded. The model was transferred to the real-world Turtlebot3 Waffle_pi robot, which showed good detection avoidance accuracy when used in an office environment, as shown in Figure 4-4.

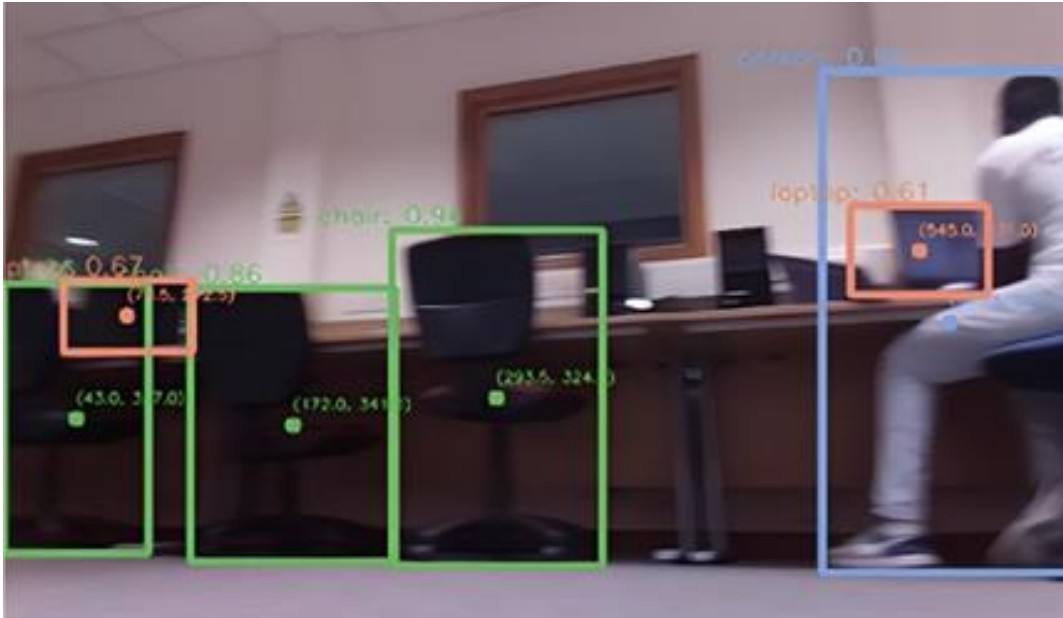


Figure 4-4: Real-world object detection test result.

We compared the NAV-YOLO framework with the default ROS navigation stack DWA planner based on the number of obstacles detected and avoided per distance, distance covered and time to the target location. The detection accuracy

was also evaluated based on YOLOv5 small and medium versions to observe the model's performance and generalizability to increased environmental complexities. To ensure robustness and reliability in the evaluation process, each experiment type is repeated five times. This repetition serves to mitigate any potential anomalies and to derive a more accurate and representative understanding of the system's behaviour across multiple instances. The performance evaluation results are detailed in Tables I, II, and III with visual representations in Figures 4-5 and 4-6 below.

TABLE 4-1: Comparing Time and Distance Covered by NAV-YOLO and ROS NAV-stack

Models	Time(mins)	Distance Covered (m)	Number of Collisions
ROS NAV-Stack	16.37	278.9	2
NAV-YOLO	14.43	262.7	0

TABLE 4-2: Comparing Obstacles Avoided by NAV-YOLO and ROS NAV-Stack in Different Environments.

Models	Environ 1 (5 Obstacles)	Environ 2 (10 obstacles)	Environ 3 (15 obstacles)
ROS Navigation Stack	5	7	11
NAV-YOLO	5	10	14

TABLE 4-3: Comparing the Accuracy of NAV-YOLO with YOLOV5 Variants.

Models	mAP_{0.5} (%)	Precision (5%)
YOLOv5m	75	74
YOLOv5s	70	77
NAV-YOLO	79	78

4.5 Result and Discussion

4.5.1 Result

This section delves into the empirical evaluation of the object detection and avoidance models, highlighting their performance in accurately identifying obstacles of varying shapes and sizes for obstacle avoidance. Two variants of YOLOv5 are used: YOLOv5s and YOLOv5m, as benchmarks to compare the obstacle detection performance of NAV-YOLO in real-time. For the obstacle avoidance evaluation, efficiency was compared based on the ROS DWA planner and the new NAV-YOLO model. The summary of the algorithms applied in ROS DWA planner which is the default and NAV-YOLO is illustrated in figure 4-5. After simulating each method, data shown in Table 4-1 were obtained. The NAV-YOLO experiment was the fastest in terms of execution time, covered a reasonably short distance and the path followed seemed smoother without collision as shown in Figure 4-6.

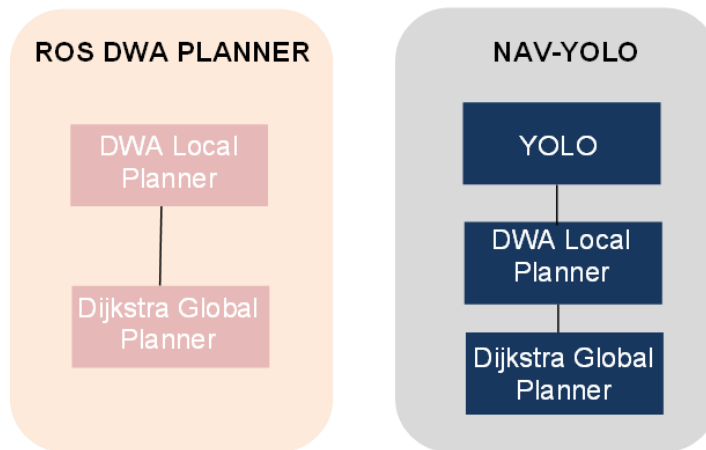


Figure 4-5: Algorithms in Proposed and Baseline solution.

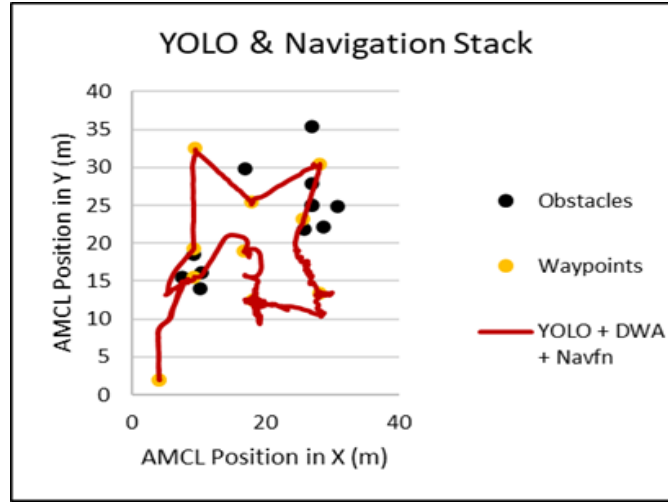


Figure 4-6: NAV-YOLO Trajectory.

The default algorithm experiment encountered an obstacle after reaching the second and fourth checkpoints and Figure 4-7 showed a very rough path to the target location which contributes to the slightly longer average time of 16.37 minutes for 278.9m distance covered. An essential aspect contributing to the extended execution time of the default algorithm lies in the occurrence of robot localization failures during a subset of tests. This phenomenon necessitates a period for the robot to rectify its position before resuming its intended path, consequently influencing the overall navigation time. This gives the proposed solution an edge as it shows better perception ability of its environment which improves its localization capability and overall navigation without collision. In Table 4-2, three different environments were explored to evaluate the performance of the obstacle avoidance system for this work. The first experiment makes use of only static obstacles, in the second experiment, we increase the number of static obstacles plus one dynamic obstacle, and in the final experiment, we introduce three dynamic obstacles into the environment. It is important to note that the start position of the robot in all the experiments is the same. In environment 1, the robot avoided all the static obstacles with both models but showed increased collision rate for the ROS navigation stack in environments 2 and 3. These environments showed that our approach outperformed the baseline method by detecting almost all static and dynamic obstacles at different phases

with augmented scenario complexity. The analysis of the accuracy of the detected objects, as shown in Table III brought out exciting findings. YOLOv5 versions showed a good accuracy of 72.5% on average in detecting obstacles correctly, while NAV-YOLO performed significantly better with an accuracy of 79.2%, as visually presented in Figure 4-3. The investigation is further conducted through real-world experiments on the Turtlebot3 robot, shedding light on its adaptability in navigating changing and complex environments. As shown in Figure 4-4, NAV-YOLO displays an improved level of detection accuracy with a precision rate of 78% in an office environment without prior environmental experience.

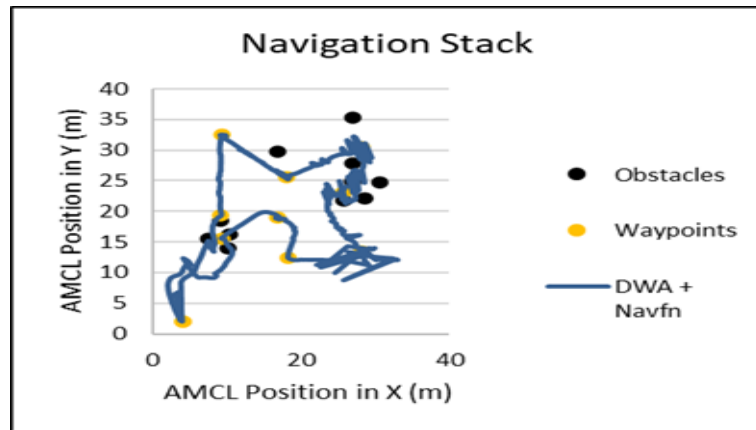


Figure 4-7: ROS NAV Trajectory

4.5.2 Discussion

The successful integration of the ROS local planner and YOLO detector in both simulation and real-world states was proven, highlighting the effectiveness of our approach. The hangar model complexity was changed by increasing the number of obstacles per scene, as shown in Table 2, to observe model efficiency and adaptability in complex settings. The ROS navigation stack with the default DWA planer [196] failed in some instances to avoid obstacles, while the proposed approach showed a significant improvement for both static and dynamic environments in terms of accuracy of detection and avoidance in different environments. The baseline model's uses cosmap concept to represent environment and identify areas that the robot should avoid based on each grid cell values. This mechanism simplifies obstacle representation in the

environment [197] but does not give detailed detected object information like the size, shape, and color of the object, that are critical especially where dynamic obstacles are applicable.

Furthermore, the comparison of YOLOv5 variants and the proposed method shows a disparity in detection accuracy, as displayed in Figure 4-5. The primary factor includes the architectural pattern that increases complexity from small to large versions of YOLOv5 models [29]. An increase in network structure increases detection accuracy and speed but requires more computational resources. The integrated pretrained YOLOv5s model was developed with an improved YOLOv5m structure that balances computational intensity and performance to achieve better detection results. The overall NAV-YOLO-RRT solution [198] shows improved detection avoidance capability for mobile robots compared to the performance of the ROS-based Dynamic Windows Approach planner. One of the notable limitations in this project is that the camera field of view can only capture objects in the forward direction making dynamic obstacle avoidance more challenging in more complex environments. Also representing objects with 2D bounding boxes do not provide accurate information about the distance of objects from the robot. The use of additional stereo cameras or fusion with LiDAR will be explored in the next project to mitigate this problem.

4.5.3 Conclusion

This paper presents a vision-based obstacle detection and avoidance algorithm for robust and efficient navigation in aircraft hangars. The proposed approach successfully combines the YOLO detection model and ROS navigation stack to enhance robots' ability to detect and avoid obstacles accurately. This combination harnesses the strengths of YOLOv5's object detection capabilities and the ROS DWA planner's real-time obstacle avoidance technique facilitating the real-time adaptation of the robot's trajectory to avoid collisions, showcasing the potential for dynamic obstacle avoidance. The ROS DWA planner uses real-time obstacle data from YOLOv5 to adjust and update the robot's planned path and this demonstrates its ability to effectively identify and classify obstacles of varying

structures in different environments. The navigation solution was integrated into a real-world robotic platform (TurtleBot3), and the object detection and avoidance goals were significantly achieved. The proposed solution is a valuable solution for autonomous robot navigation and is applicable in diverse real-world scenarios.

With these promising results, there are still areas for further improvement. The integration of a more advanced planner algorithm can improve obstacle prediction with optimal obstacle avoidance navigation strategy to improve adaptability in unknown environments. For detection enhancement, object detection accuracy, especially in identifying intricate objects, remains a challenge and can be addressed through continuous model refinement and dataset augmentation. More performance metrics will be investigated and optimized to expand the effectiveness of this model in the real world.

5 Improved hybrid model for obstacle detection and Avoidance in ROS framework (RRT and DWA)

This research paper focused on developing an improved NAV-YOLO system that leveraged YOLOv7 and RRT path planning algorithms to address obstacle detection and navigation challenges of state-of-the-art solutions in mobile robot applications. This paper acknowledges the dynamic and unpredictable nature of real-world hangar environments and the need for enhancement with high accuracy for obstacle detection, avoidance, and efficient path planning, to develop the NAV-YOLO-RRT model. The improvement solution in this paper combines an RRT-integrated ROS navigation stack and a pre-trained YOLOv7 object detection model to enhance the capability of the developed work on the NAV-YOLO system. The proposed approach in this study leveraged the high accuracy of the YOLOv7 obstacle detection model and the efficient path-planning capabilities of RRT and DWA to improve the navigation performance of mobile robots in real-world complex and dynamically changing settings using a hangar environment as the use case. The proposed developed solution was extensively tested through simulations and real-world robots, comparing performance with NAV-YOLO and ROS navigation stack as the baseline. The performance results demonstrated a high-level obstacle avoidance capability, ensuring the safety and efficiency of mobile robot navigation operations in hangar environments. However, the speed of the robot navigation might require an improvement for very complex scenarios.

Paper published on: <https://www.mdpi.com/1424-8220/24/7/2262>

5.1 Introduction

The aviation sector is increasingly integrating mobile robot solutions into its operational framework for activities including inspection and maintenance tasks. The robot systems are designed to perform diverse tasks, including inspecting aircraft components for defect identification, collecting data, and transporting tools and equipment. However, safely navigating in a complex aviation environment like the smart hangars is difficult for these robots [199]]. This necessitates the development of enhanced navigation algorithms with sensor technologies crucial for the robot to navigate safely and effectively in such workspace. Recent developments in vision-based navigation systems have seen significant progress, leveraging robot operating system (ROS) navigation stack local and global planners [4]. ROS provides a modular and flexible set of software packages for developing autonomous navigation applications. The ROS navigation stack planners work in conjunction to generate obstacle-free routes from the robot's current position to its goal destination. The local planner includes the time elastic band (TED) [200] and DWA [201] algorithm while the global planners consist of Dijkstra and A*, employed as default components of the ROS navigation stack. The Dijkstra algorithm among the global planners is known for its efficacy in finding the shortest path by exploring the entire space [202]. This can be computationally expensive, time-consuming and impractical in dynamic environment making it more suitable in static environments with predefined obstacles. The increasingly complex and real-world demands in dynamic environment limits the practical application of this algorithm. Looking at the smart hangar environment where the environment is dynamic and requires real-time responsiveness, the changing conditions of environmental factors necessitate navigation solutions that are adaptive and capable of real-time decision-making [203]. Rapidly exploring random tree (RRT) planner is among the global planners that employs sampling strategy, by sampling points by exploring the robots' configuration space and incrementally builds a tree structure based on these samples. This enables the robot to explore diverse areas where the obstacle may appear or move and adapt its path in response to changing conditions. While its capable of adapting to changes in the environment, it may

not react quickly to dynamic obstacle and rapidly changing conditions. Consequently, there is a need for more advanced hybrid approaches that can effectively handle the unpredictable nature of such environments, ensuring safe, efficient, and real-time navigation of mobile robots [204]. The Integration of advanced hybrid solutions, which can learn, process, and adapt to varying environmental conditions, has shown significant improvement for an efficient mobile robot solution in complex settings. Some of these solutions have combined the strength of machine learning [47], [205] sensor fusion [206] or other conventional navigation techniques [207], [208] to create robust systems capable of real-time decision-making and adaptation for robot navigation. Machine learning technique such as YOLO models play a significant role in object detection and real-time capabilities [209], enabling the robot to detect varying obstacles in its environment as it navigates. While YOLO does not directly drive the robots path planning and obstacle avoidance approach, it enhances the robot's situational awareness by continuously identifying obstacles in the scene. This information can be utilized by the robot to make more informed decisions and update its RRT-based path to navigate round obstacles in real-time.

This paper proposes an improved hybrid framework that integrates the capability of the RRT-based ROS navigation stack with the YOLOv7 object detection model, which harnesses sensor information from LiDAR and camera systems to enhance the adaptability and navigation functionalities of mobile robots in complex and changing environments. This brings improvement to an established NAV-YOLO framework in the ongoing research project that leverages path planning proficiency of RRT and sensor fusion to significantly improve the precision and reliability of autonomous robot navigation in environments characterized by unpredictability and variable obstacles. The camera sensor, while providing high-resolution visual information, is constrained by a limited field of view, and this is complimented by integrating LiDAR input that provides 360-degree environmental coverage. This broadens the scope of obstacle detection and significantly enhances the mobile robot's ability to perceive, navigate and interact with a wide range of environmental features. The object data detected by the pre-trained YOLO node [14] from the camera stream is integrated with laser scan

data from the LiDAR sensor within the cost map file of the ROS ecosystem. This communicates multi-dimensional environmental information to RRT and DWA planners within the navigation system and facilitates the robot's accurate obstacle avoidance decision-making and path-planning capabilities in real time. This framework is simulated and validated in turtlebot3-enabled ROS-Gazebo environment and physical robot. The experiment showed an intelligent obstacle detection and avoidance solution that significantly enhanced robot navigation capabilities for aircraft inspection in a hangar environment. The general contribution includes,

1. RGB camera and LiDAR sensor fusion to broaden the robot's field of view in complex and dynamic settings.
2. Integration of customized pre-trained YOLOv7 model into ROS framework for accurate detection of varying and unstructured environmental obstacles.
3. Enhancement of NAV-YOLO with RRT global planner for effective real-time obstacle avoidance and path planning.
4. Comparative analysis of ROS navigation stack, NAV-YOLO and proposed RRT enables NAV-YOLO to validate improved performance.

The rest of the paper is structured in four parts; section 3 discusses related works that have contributed to obstacle detection with YOLO, sensor fusion capability and robot navigation with ROS framework. Section 4 details the methodology, describing the YOLOv7 version of the object detection model, the structural component of the RRT path planning algorithm and an overview of the development process. In section 5, the experimental analysis was performed, analysing the detection and path planning capability of the proposed model related to NAV-YOLO and the baseline model. Finally, section 6 presents the summary and future direction.

5.2 Related Work

Diverse approaches have been employed in expanding the potential application of robotics, especially around safe navigation in complex environments. The combination of traditional algorithms and cutting-edge machine learning

techniques has shown significant efficiency and reliability in obstacle detection and avoidance for a safe robot navigation experience. However, adaptability in environments densely populated with varying obstacles and the need for an efficient real-time response for optimal obstacle avoidance and path planning have become major challenges in many applications. The study presented in [207], [208] proposed a CNN-based neural RRT model, similar to research work in [126] that utilized conditional variational autoencoder (CVAE) to enhance the path planning process using a nonuniform sampling distribution method, but environmental obstacles information was not given keen consideration in these methods. Canglong et al. [106] applied a combined solution based on ROS and CNN-based end-to-end learning that utilized datasets acquired from a low-level camera. This was applied in a structured environment and will struggle in real-time complex scenarios.

Another work in [210] uses object detection based on a vision system and deep learning to detect obstacles and accelerate path planning. However, obstacle avoidance capability is limited. This was addressed in [211] by developing a solution that learns obstacle representation in complex environments using LiDAR sensors to demonstrate generalization in unknown features and enhance path planning ability. A sensor-based learning system was presented in [212] that uses an integrated sampling-based planner with deep learning architecture to improve obstacle avoidance and path planning. Here, LiDAR's inability to detect rich environmental information affects the accuracy of obstacle detection and avoidance. From another view, the author in [213] employed depth camera information to develop the YOLOv4 object detection model and enhanced the path planning process with RRT while integrating sensor fusion information with deep reinforcement learning. The work in [214] also applied multimodal LiDAR and camera information for the object detection process using the pre-trained CNN technique to improve autonomous vehicle perception.

This paper proposed a new navigation methodology that learns from perceived environmental features for precise obstacle avoidance and path selection decision making, and this facilitates real-time adaptability and responses in

complex scenes. Our proposed integration framework of YOLOv7 and ROS framework techniques presents a significant advancement with sensor fusion in robot navigation capabilities, enabling safe and efficient operation in complex and unpredictable settings, and opening new avenues for autonomous robot deployment in diverse industries.

5.3 Methodology

This project presents the development of a vision-based hybrid model for obstacle avoidance and path planning in mobile robotics, innovatively merging conventional ROS navigation methods with advanced YOLO deep learning techniques. The approach addresses safety needs critical to autonomous robot navigation, capable of detecting a wide range of obstacles. Autonomous robot navigation not only demands high-precision obstacle detection capability but also needs to respond in real time to complex and changing MRO scenes to ensure that the robot can make timely and accurate decisions. The navigation system is centred on integrating RRT with DWA planners to combine the efficiency of the RRT in optimal path planning and DWA in real-time obstacle avoidance. This solution is enhanced by integrating LiDAR /Camera sensor fusion and the YOLO object detection model, which allows mobile robots to navigate around obstacles with precision while optimizing paths in real-time.

5.3.1 CNN-based YOLOv7 Model

YOLO is one of the many CNN-based computer vision models that is based on a single-stage, end-to-end training and detection process [209]. With the capability to perform multiple detections simultaneously, its improved variants, like YOLOv7, noted with superior accuracy and speed of inference, outperform other related object detection models like SSD [80] and Faster RCNN [144] two-stage object detection models. While these models have been instrumental in object detection, they often lag in speed and optimization efficiency, especially in complex environments with varied obstacle structures. Over time, a series of successive iterations of the YOLO framework have been conceived and are presently undergoing further development [215]. YOLOv7 is among the latest versions in the YOLO series of real-time object detection models, built for real-

world challenges and was officially released in 2022 [216]. The concept involves data processing including image resizing, data augmentation and the use of convolution neural network (CNN) as the backbone architecture for feature extraction. Also, it uses anchor boxes to predict bounding boxes for objects ensuring that anchor boxes closely match object sizes and shapes in the data. Recent study [217] compared YOLO variants including YOLOv5 as faster and more lightweight than previous YOLO while YOLOv7 showed improvement in speed and accuracy which are relevant in this manuscript for real-time robot navigation.

Our previous experiment with YOLOv5 showed reasonable speed and accuracy in simulation environment but limited success rate when deployed on the physical turtlebot3 robot platform obviously due to limited hardware capacity. This necessitated the experimentation with an improved YOLO variant for faster speed in frames per second (FPS) faster and enhanced accuracy. Some of the metric used to evaluate the model's performance as explained in our previous work in [14] include mean average precision (mAP) which is the mean of average precision value calculated for each object class. A higher mAP indicates good performance in object detection. The intersection over union (IoU) is another metric that measures the overlap between the predicting bounding box and the ground truth bounding box. A higher IoU indicates better accuracy in detected objects. The FPS measures the speed at which the model can process images and higher FPS indicates faster processing and model efficiency.

5.3.2 ROS Planners

The ROS planners are part of the ROS navigation stack algorithms that provide path planning and navigation functionalities in the ROS framework. The default ROS path planning algorithms include Dijkstra and DWA global and local planners, respectively [218]. Other advanced path-planning algorithms have been developed over the years, like the RRT, which, although not a native part of the ROS navigation stack, can be integrated to optimize path-planning functions. RRT is an incremental sampling-based algorithm [219] with a tree structure that selects a random state x_{rand} from a uniform distribution in every

iteration and then identifies the nearest vertex $x_{nearest}$ in the tree state [220]. Every successful connection establishes a new state, x_{new} , ensuring a feasible transition where new vertexes and edges are formed between $x_{nearest}$ and x_{new} . This iteratively expands the tree structure, exploring the state space and balancing the computational demand until target conditions are met. To address some issues encountered during the experiments, specific parameters were updated within the RRT 'move_base' package. The goal tolerance was changed from the default 0.5 to 1 meter to reduce computational issues that made most of the integrated package struggle to function and improve the robot's ability to smoothly align with the target location. The number of iterations was also reduced from 4000 to 1000 to make the path search faster. The branch length was realized to be short and affected the robot's ability to align with the generated path. This was increased from 0.2 to 0.6 to address this problem. The RRT allows exploring the wide space of the hangar. For this study, the RRT is integrated to enhance the functionalities of the Dynamic Window Approach (DWA) local planner.

The DWA local planner is specifically designed for local trajectory planning and obstacle avoidance. It takes into account the robot's current velocity, acceleration limits, and the immediate environment to calculate the safe and feasible path towards the goal. DWA operates by creating a dynamic window of possible commands (linear and angular velocities) that the robots can execute, considering their dynamics and the feasibility of the trajectory [201]. The planner utilizes the local costmap, which represents the environment around the robots and the detected obstacles to evaluate the potential collision of simulated trajectories [221]. DWA is highly configurable with parameters that can be fine-tuned for specific navigation tasks, such as velocity ranges, acceleration limits, etc. While DWA is adept at avoiding obstacles in less complex environments [203], its performance in highly cluttered or constrained environments, like the MRO hangars, may be suboptimal. In such environments, DWA benefits from being paired with complimentary algorithms to achieve optimal performance, as it may require additional strategies to navigate effectively through dense and unpredictable spaces.

5.3.3 Sensor Fusion

One major research interest in intelligent robotics is the integration of multisensory capabilities to expand and enhance a robot's perception of its environment and navigation operations. This provides a more comprehensive understanding of the surrounding robot's environment and enables it to make more accurate and informed decisions, adapting to varying environmental structures and conditions compared to relying on an individual sensor [222]. This work integrates LiDAR and camera sensor devices, LiDAR's provide geometry of the surroundings and a 360-degree field of view [223], while transformed camera frames from YOLO detectors add rich definitions to environment variables. The data from LiDAR and YOLO-detected camera frames are registered in the robot's costmap. The costmap is a 2D grid-based representation of the environment around the robots. The fused data stream continuously updates the costmap, providing the planners with a dynamic and accurate representation of the environment [224]. This information is then used in conjunction with the planners to enhance the robot's perception capabilities and make better-informed decisions for obstacle detection, avoidance, and path planning.

The Lidar data is published using `sensor_msgs/LaserScan`, while YOLO-detected object information is integrated with `sensor_msgs/Image` ROS package. The coordinates of the detected objects are transformed to be in the same reference as the robot's navigation system. This information is updated in the ROS navigation stack `costmap_common_params.yaml` file, where the obstacle layer that represents obstacles detected by lidar is configured in `local_costmap_params.yaml` and YOLO-detected object layer created as an extension in the costmap for YOLO information. The obstacle layers from LiDAR and YOLO layers are included in the `move_base_params.yaml` file. The combined information improves the robot's field of view in the environment. This integration strives to capitalize on the strengths of both sensor types [222], overcoming the limitations posed by LiDAR's temporal resolution while leveraging its wide field of view for robust obstacle detection and avoidance.

5.3.4 System Architecture

The solution was first established by combining pretrained YOLOv5 [14] and the default ROS navigation stack to develop a system called NAV-YOLO which showed need for improvement after evaluation [225]. The NAV-YOLO approach interfaced YOLO and default ROS planners which include Dijkstra and DWA planners. The YOLO identifies and localises object in the robot environment based on camera feed and publishes the detected objects information using bounding boxes and class label on a ROS topic. This is configured in the costmap which act as a bridge between YOLO and the ROS planners. The ROS planners then utilize the updated costmap which represents the interface between the robot's environment and ROS planners, to generate safe and efficient navigation paths. The proposed solution in this manuscript followed the same approach but incorporated pretrained YOLOv7 in tune with our previous work [14] and RRT algorithm to enhance the performance of the NAV-YOLO. This development is structured in two modules, the obstacle detection and obstacle avoidance system as shown in Figure 5-1:

5.3.4.1 Obstacle detection module

This system focuses on the accurate perception of environmental features using camera sensors to generate custom datasets for YOLOv7 accurate object detection. The YOLOv7 tiny variant was developed and pretrained using successful techniques developed in our previous work [14]. From the previous work perspective, the contribution is to enhance object detection model using improved YOLO model. This approach includes capturing a wide range of obstacles found in MRO hangars, including tools, vehicles, and structural elements in the real world and in simulation, and using these datasets to train the model. LLVIP [12] public datasets were also integrated to expand the capability of recognising objects in diverse scenarios considering low-light conditions, which are common in certain areas of the MRO hangar or during specific times of the day. For data fusion, DenseFuse network was first used to fuse RGB and thermal images from Real-world and LLVIP datasets to expand the model capability to accurately detect obstacles in low-light conditions. The architecture of the YOLO

model was slightly adjusted to make it suitable for real-time application. This involves the integration of convolution block attention mechanism module (CBAM) which improves performance by attention focusing on relevant features and suppressing of background noise to priorities important objects details relevant for real-time detection [226]. The datasets are merged and pre-processed to suit YOLOv7 input requirements. The pre-processing process includes data annotation, data augmentation with flipping and cropping techniques and image resizing to 640x640 dimension, which were done using Roboflow annotation tool [161]. The training process was fine-tuned using default hyperparameter, batch size of 20, and 640x640 resolution through iterations. The YOLOv7 was trained offline and evaluated to attend accuracy level of about 68-70% which indicates that the model can identify obstacles accurately within the input image even in low light conditions. This also ensures that the model maintains required performance capable of running on robot's onboard hardware with limited computer resources. The model is trained and validated to ensure that the model's inference is fast and reliable to allow real-time processing in ROS. The customised YOLOv7 model and weights file was saved and transferred to a dedicated ROS package with custom nodes that load the model and weights.

The YOLO node subscribes to the /image topic to perform object detection on the received image. The detected result is published and transformed to laser scan format for more informed navigation decisions. This is configured in the ROS navigation costmap which interfaces with the ROS planners. The paper leverage YOLOv7 object detection capabilities to enhance the robot's ability to better perceive their environment, enabling it to make more informed decision-making in navigation task. The RRT can generate feasible paths from the robot's current position to the goal based YOLO's object detection results (detected objects), and can dynamically update the environment map, avoiding collisions with detected obstacles during path planning. YOLO also provides real-time updates on the positions and trajectories of dynamic obstacles, allowing DWA to adjust the robot's velocity and steering commands accordingly to avoid collisions and ensure safe and efficient navigation.

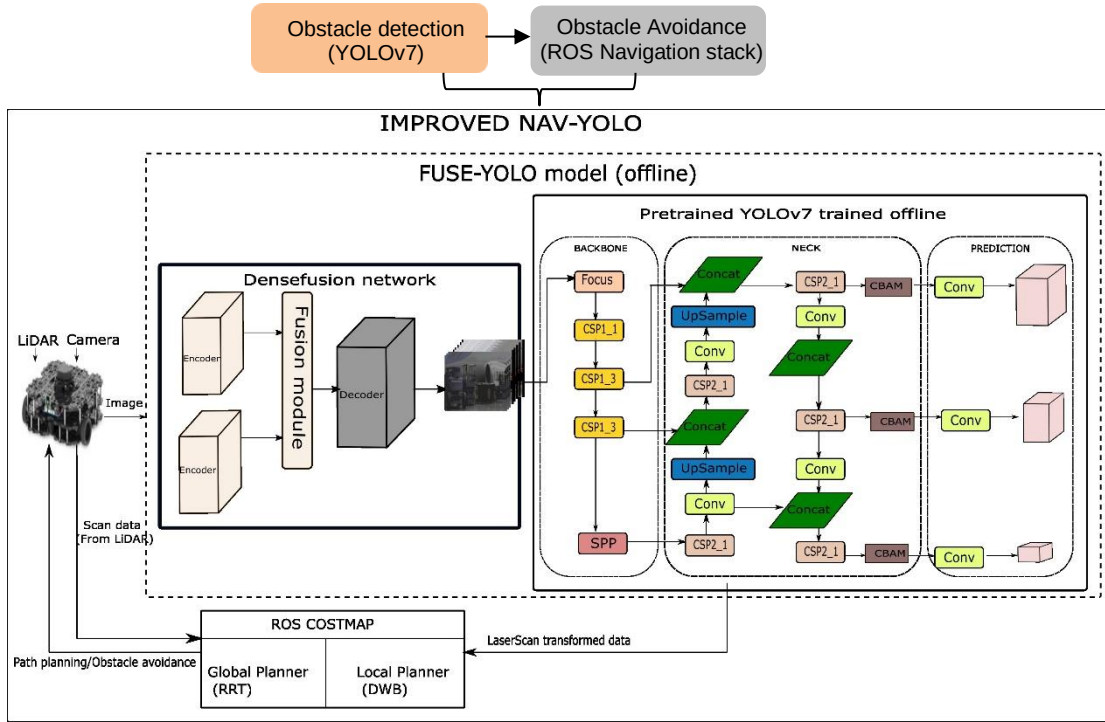


Figure 5-1: Improved NAV-YOLO system architecture.

5.3.4.2 Obstacle avoidance module

To implement the obstacle detection module in the ROS ecosystem, YOLO detection and depth data nodes are created to load a pretrained YOLO model. The camera image stream is subscribed to and processed to detect obstacles published with bounding box information. The bounding box coordinates are generated with classification type and confidence scores for each detected object. With depth map and camera properly calibrated and aligned to the same resolution and dimensions, the bounding box centre is mapped to depth image, to extract depth value at the centre. This typically is the value of the distance from the sensor to the object at that point. The centre of the bounding box can be calculated using the equation below.

$$\text{bbox} = x, y, w, h \text{ (From YOLO detection)} \quad (5-1)$$

$$cx = x + \frac{w}{2} \quad (5-2)$$

$$cy = y + \frac{h}{2} \quad (5-3)$$

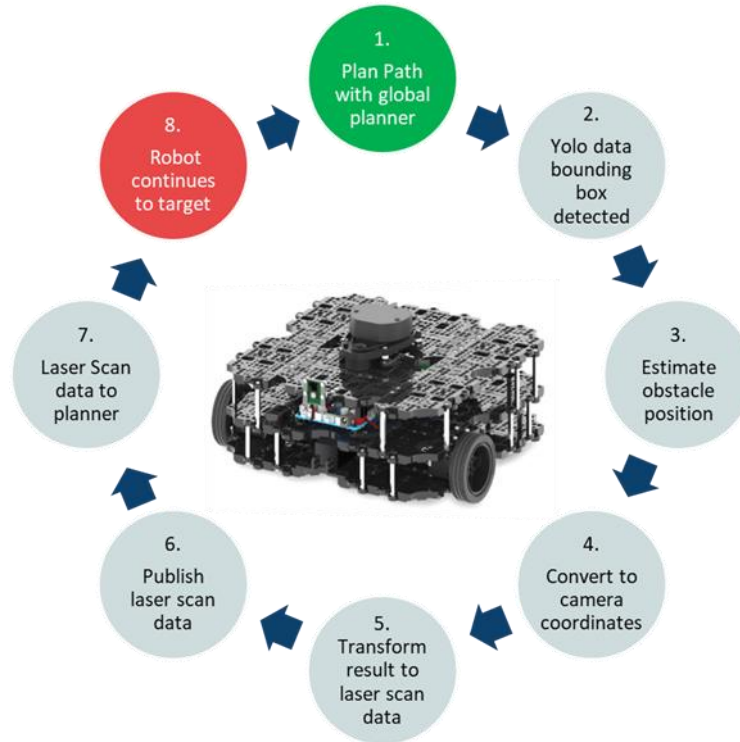


Figure 5-2 : Detected objects transformation workflow

where bbox is the bounding box defined by its top-left corner (x, y) and its dimensions (width w and height h). And cx, cy represents the centre coordinates. However, the process of deriving laser scan data from the centre coordinates assumes that the detected object is solid, without any gaps, as is the case with standing humans or chairs. This was addressed by generating laser scan information from the entire width of the image at the centre of the object's height instead of at the centre of the camera. This adaptation will allow for a more comprehensive representation of the object's spatial extent, accommodating scenarios where there might be gaps or spaces within the object's outline. This adjustment will enhance the fidelity of the generated laser information, contributing to more accurate and reliable obstacle detection and avoidance, especially in situations involving objects with varying shapes and configurations and will be expanded in our next project phase.

5.3.4.3 Converting pixel coordinates to camera coordinates.

The key parameters for this transformation include the 2D coordinate image from the YOLO detector and camera intrinsic matrix with key parameters including focal length (f_x , f_y) and principal point (c_x , c_y) of the camera shown in the equation 1 and 2. The 2D coordinates from the image frame (u , v) are transformed to 3D coordinates in the camera's coordinate (X , Y , Z) relative to the camera's position and orientation to understand objects spatial positioning in real world from their positions in the image. The shown in the equation 1 and 2.

$$X = (x_{obj} - c_x) \times \frac{z_{obj}}{f_x} \quad (5-4)$$

$$Y = (y_{obj} - c_y) \times \frac{z_{obj}}{f_y} \quad (5-5)$$

$$Z = z_{obj} = 0 \quad (5-6)$$

Where x_{obj} , y_{obj} represents detected object coordinates, and z_{obj} represents depth estimation (assuming ground plane projection $Z=0$)

After obtaining the 3D points in the camera coordinates system, the points are transformed to robot's coordinate using extrinsic parameters (rotational matrix R and translation vector t) that describe camera's orientation and position relative to the robot's coordinate. The equation is shown below as;

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = R * \begin{bmatrix} x_{obj} \\ y_{obj} \\ z_{obj} \end{bmatrix} + t \quad (5-7)$$

With the robot's centre coordinate transformed in laser scan frame, considering its geometry and current pose, the Euclidean distance from each detected object to the robot's centre can be calculated [227]. That is the distance between two points in the space and is computed using the below formula:

$$\text{Distance} = \sqrt{(x_{obj} - X)^2 + (y_{obj} - Y)^2 + (z_{obj} - Z)^2} \quad (5-8)$$

This distance is employed to identify potential collisions and plan safe paths around the obstacles. The overall process of detection to transformation is represented in the workflow shown in Figure 5-2.

5.3.4.4 Data Fusion

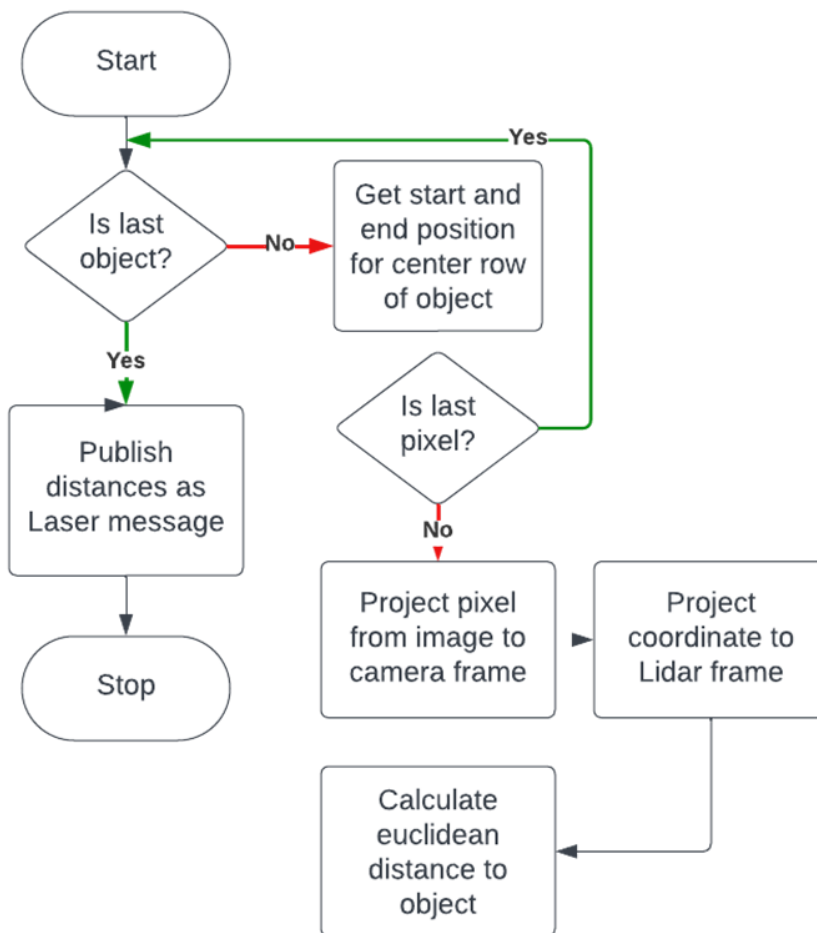
Having YOLO detection transformed, the detected objects are projected onto the LiDAR laser scan by associating it with corresponding position in the LiDAR scan. The LiDAR sensor provides 360-degree scans of the environment, and each scan point represents distance and angle information relative to the robot's position. The system utilizes `sensor_msgs/LaserScan` ROS packages to subscribe to LiDAR data and the coordinates are transformed with ROS `tf` package using robots base frame as the reference. This ensures that camera images and LiDAR scans are represented in the same coordinate system. The transformed LiDAR data and YOLO detected objects nodes are configured in the costmap with ROS observation sources layer configured to subscribe to `scan_1` and `scan_2` which represent Lidar scan as `/trans_scan` and Yolo object detected node as `/scan_yolo` respectively. The transformed data are fused together and publish as an integrated representation of the environment.

5.3.4.5 Costmap configuration

The resulted laser scan data is configured in the costmap to update costmap cells [221]]. The costmap is a grid-based representation of the robot's environment where each cell has a cost associated with traversing through it. The cells that correspond with the position of obstacles are assigned higher costs, while free cells are given lower costs [228]. Also, the LiDAR data is integrated into the `costmap_common_params.yaml` file to specifically expand the robot's field of view. The costmap dynamic layer parameters like the `obstacle_range`, update frequency, and resolution are set to continuously update by setting `static_map` parameter to false and maintaining other default parameters value. The continuous costmap update helps to accurately reflect the current environment. The ROS navigation stack path planners, including RRT and DWA, access the

costmap to generate collision-free paths where it considers the cost values of cells to avoid high-cost areas representing obstacles and determine traversable routes. The RVIZ visualization tool was employed to observe updates with necessary adjustments to enhance path planning algorithm control decisions in planning a safe route around the obstacle while navigating to the target location. Figure 5-4 gives an overview of the navigation work of the integrated ROS Path planner and YOLO object detection solution. Figure 3 gives an overview of the

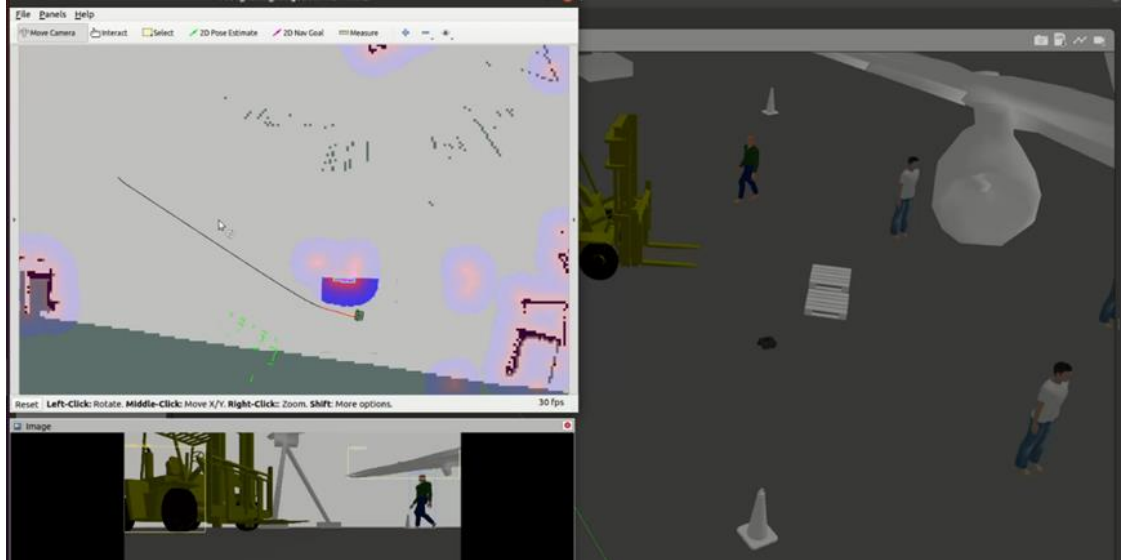
Figure 5-3: Object detection and transformation to laser scan workflow



navigation work of the integrated ROS Path planner and YOLO object detection solution. While the robot is navigating through its environment, figure 3 showed the replanning capability as well as object detected as it navigates around obstacles.

5.4 Experiment

The improved NAV-YOLO model was developed and evaluated in the



ROS_Gazebo simulation environment. Gazebo is a robot simulation framework integrated into ROS that provides a safe and controlled environment for accurate modelling of real-world scenarios. The Cranfield University hangar environment was replicated in the Gazebo simulation for this experiment as shown in Figure 1-4. The experiment for this project was conducted on a computer equipped with an Ubuntu 20.04 64-bit operating system with ROS noetic version, powered by an Intel(R) Core (TM) i7-10750H CPU at 2.60GHz. For deep learning integration, the YOLOv7 tiny object detection model was used, and the RRT global path planning algorithm was also integrated, each at different interfacing nodes within the ROS costmap package. The setup utilized the Turtlebot3 waffle_pi robot package, which comes equipped with embedded LiDAR and camera sensors. For system validation, three sets of experiments were conducted to compare their performance and adaptability in an increasingly complex environment. The first approach evaluated the obstacle avoidance and path planning performance of the default ROS navigation stack comprising Dijkstra and DWA planner. The second approach combined the YOLOv7 object detection model with default ROS global and local planners (NAV-YOLO). The last method integrated the YOLOv7 model and RRT global planner into the ROS navigation stack (NAV-YOLO-RRT). Figure 5-5 shows an overview of the compared three solutions.

Figure 5-4: Robot navigation showing ROS path planning and YOLO object detection.

During the baseline experiment using the default ROS navigation stack, the robot's movement and interactions with obstacles, as well as sensor data, were systematically recorded with conditions that are consistent with other experiments, including starting point and goal location. The NAV-YOLO and NAV-YOLO-RRT experiments followed the same procedure but involved an object detection model that generates transformed laser scan data from the RGB-D images used for the evaluation of obstacle detection performance. In the experiment environment, five distinct checkpoints were defined, representing inspection points around the aircraft. The environment is generally populated with 20 different and strategically positioned obstacles comprising 10 statics and 10 dynamic obstacles. These obstacles were incrementally added, creating three different environments to assess the robot's behaviour in environments of increasing complexity. The three environments (1, 2 & 3) were built with 10 static obstacles each and 3, 6 and 10 dynamic obstacles respectively. The experimental setup is structured to guide the robot through predefined checkpoints and then to the target location which is also the starting point. This is to evaluate the robot's path planning and obstacle avoidance capability through the generated route. The use of "rosbag" plays a crucial role in this process and was used to capture relevant data that was used for the evaluation. This includes measuring metrics like the time taken to reach the target location, distance covered and collision rate, which are necessary for assessing the efficiency of the robot's navigation system in real-time hangar scenarios.

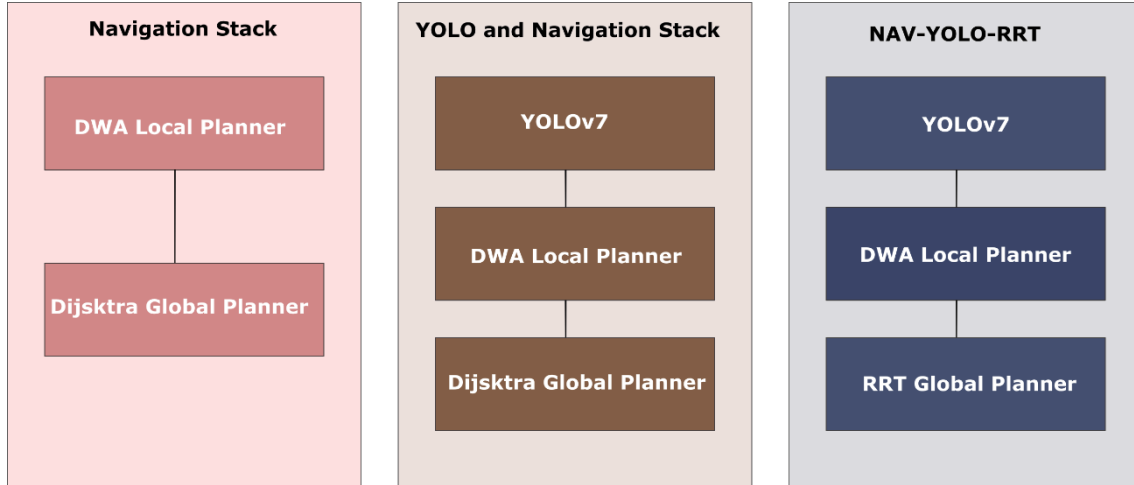


Figure 5-5: Compared NAV-YOLO methods

5.4.1 Data Collection

The data was collected to comprehensively analyse the robot's behaviour, navigation performance, and the accuracy of the object detection system. The Adaptive Monte Carlo Localization (AMCL) pose of the robot was recorded using the 'rosbag' as and converted to csv file as shown in Appendix D, to determine the robot's position and orientation in the environment. AMCL is a robotic algorithm that uses a particle filter to estimate a robot's position in its environment [229]]. In each environment, the 'rosbag' records key metrics including sensor data, robot pose, goal status, and time to reach goals for effective performance evaluation. The AMCL pose of the robot was used to plot and compare the path taken for each of the compared methods. The total distance over time was computed using the Euclidean distance formula shown in Equation 5-5. A motion script was developed that initiates a timer at the beginning of the navigation task to its destination, to measure the total navigation time that is used to evaluate the fastest approach. Another script was also used to log each collision event, incrementing a counter every time a collision is detected. The data collection experiments of this project are structured to ensure that the results are consistent and reliable and are conducted multiple times under the same initial conditions. Upon completion of the experiments, the gathered data undergoes rigorous analysis to derive meaningful conclusions.

5.4.2 Result

To assess the efficacy of the proposed solutions, comparative analyses were conducted, using the ROS navigation stack as the benchmark. The choice was as a result of an established standard and framework for ROS navigation stack which is widely adopted. Leveraging it as a baseline will give researchers practical insight and understanding of the improvement and efficiency offered by the proposed solution. Three different simulated environments with a different number of obstacles were used. The evaluation was based on three critical metrics: the time to reach the target location, the total distance covered during navigation and the number of collisions encountered. The start position of the robots was consistently set at the coordinates of (6.0, 7.0, 0.0) in each experiment, ensuring accurate comparison of the system's performance across different environments including Environment 1, 2, and 3 used in this project experiment. In Environment 1 and 2, which has 10 static and 3 dynamic and 10 static and 6 dynamic obstacles respectively, the default ROS navigation stack encountered collision at two different points and along with notable increase in time and distance covered. This signifies its limitation in obstacle detection, avoidance and efficient path planning. Conversely, the other developed solutions guided the robot through the environments within a reasonable timeframes and distances without colliding with obstacles. In environment 3, the complexity increased with 10 static and 10 dynamic moving persons as obstacles. Both the ROS navigation stack and NAV-YOLO solutions encountered collisions in environment 3, surpassing the collisions occurrence observed in Environment 1 and 2. Furthermore, they require more time and distance to navigate to target location as environmental complexity increases when compared to the navigation experience in the earlier environments.

The NAV-YOLO-RRT proposed solution effectively navigated to the target location with no collision encountered in environment 3 as demonstrated in Table 1. Further comparison in Figure 5 a, b and c showed the smooth trajectory generated and followed by the proposed solution while the other solutions showed very rough trajectories indicating inconsistency in replanning route and avoiding obstacles. This shows the efficiency of the NAV-YOLO-RRT solution in

safely navigating complex environments with improved obstacle avoidance mechanism. However, the proposed solution exhibited a longer time and distance covered, spanning 18 minutes and 627.8 meters, respectively, in Environment 3 when compared with the default ROS navigation and NAV-YOLO solutions which covered distances of 619.5 and 615.8 meters, respectively in less time. This result outcome suggests that while the proposed solution achieved collision-free navigation, the complexity of the environment influenced the path planning strategies and requires more time to reroute to target location. This can be improved to enhance real-time capability of the system.

Table 5-1: Performance evaluation of developed solutions with baseline.

	Time(min)	Distance Covered(m)	Number of collisions
ROS Navigation stack	17.57	619.5	3
NAV-YOLO	16.37	615.8	2
NAV-YOLO-RRT	18.0	627.8	0

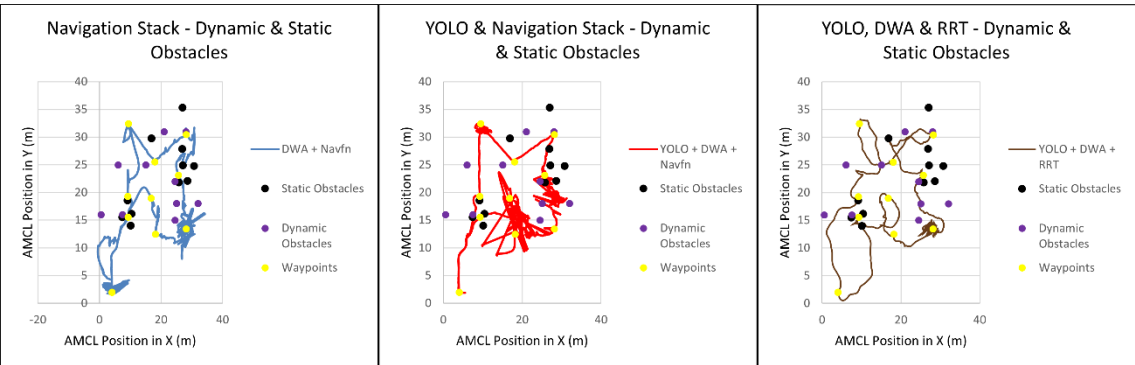


Figure 5-6: Comparison analysis of NAV-YOLO and NAV-YOLO-RRT trajectory with baseline.

5.4.3 Obstacle Detection Performance

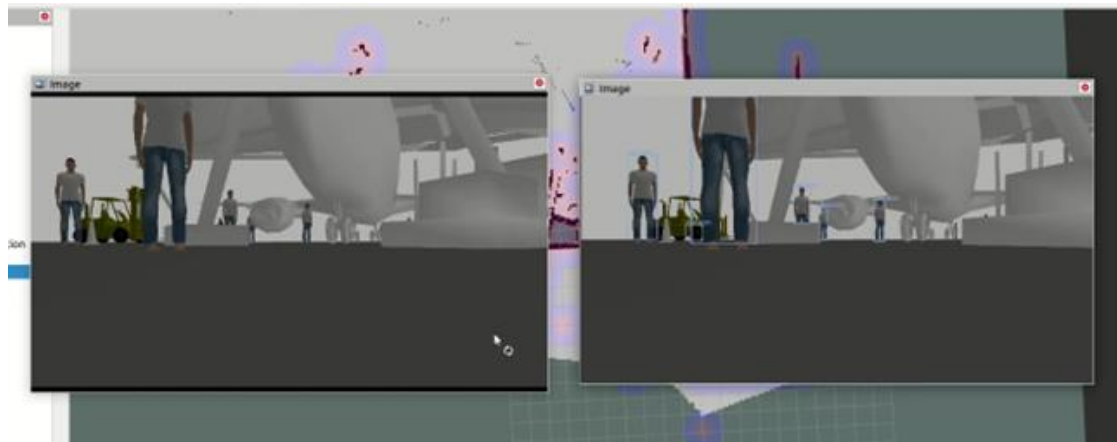
We further compared the obstacle detection and data synchronisation capability of NAV-YOLO and NAV-YOLO-RRT, as shown in Figures 5-7 and 5-8: to ensure a robust evaluation of the detection performance. For the reliability and validity of the experimental results, the experiments were conducted multiple times under the same initial conditions. This provides a robust understanding of any variation in the environment and the robot's behaviour under different scenarios. We conducted the experiment integrating YOLOv3, YOLOv5 and YOLOv7 versions of the YOLO object detection model. The comparison between YOLOv3 and YOLOv5 in terms of inference time per frame highlights a significant difference in processing speed. YOLOv3 had an average inference time of 0.582 seconds per frame, while YOLOv5s demonstrated a faster average inference time of 0.225 seconds. Comparing the proposed NAV-YOLO-RRT solution integrated with YOLOv5 and YOLOv7, it was observed that the average time for the robot to reach its destination was approximately 17minutes, during this period, the performance of the models in accurately identifying obstacles was closely monitored including tracking metrics like frames processed per second (fps), number of obstacles avoided, and number of collisions using a python script as shown in Table 5-2. YOLOv5 showed approximately 66% accurate detection with a faster inference rate of 168 on Turtlebot3, while in YOLOv7, a lower inference rate of 134 and approximately 70% improved accuracy was observed.

Table 5-2: Performance result of YOLOv7 and YOLOv5

Models	Frames rate	Obstacles detected and avoided	Collision with obstacles
YOLOv5s	168	15	5
YOLOv7-tiny	134	17	3

Further comparison between NAV-YOLO and NAV-YOLO-RRT showed that there were lags in obstacle detection in the robot's view as it navigates the environment with NAV-YOLO solution. Figure 6a showed the camera view while

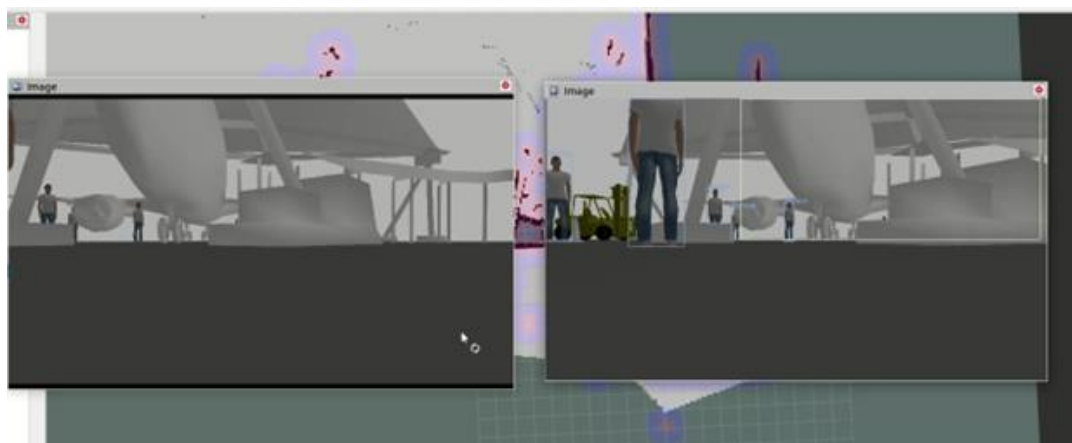
figure 6b showed the YOLO detected obstacles in real-time. The object view in the two images do not correspond due to synchronisation issue between the



sensor input and the transformed detected data. This implies that the processing of data from the point of robot capturing images with camera, to YOLO processing, distance estimation, transformation from camera coordinates to laser scan data and to navigation decisions posed some delays. For NAV-YOLO experiment, hardware limitations, YOLO model and environmental complexity were significant contributing factors..

Figure 5-7: Observed lag with NAV-YOLO in obstacle detection

The experiment with NAV-YOLO-RRT as demonstrated in Figure 7 showed improved data synchronisation and object detection in real-time. The environment perception from camera view and YOLO object detection perspective



tend to align reasonably in real-time.

Figure 5-8: Improved Obstacle detection with NAV-YOLO-RRT

5.4.4 Path Planning Evaluation

The experiment also provided valuable insight into the robot's path generation capability and trajectory, which were analysed using the RVIZ visualization tool while moving through five predefined checkpoints towards its target location. For the evaluation, three distinct environments with increasing levels of complexity were created, ensuring a thorough assessment across a range of scenarios. The start position of the robots was consistently set at the coordinates of (6.0, 7.0, 0.0) in each experiment, ensuring accurate comparison of the system's performance across different environments. Each experiment was repeated five times to minimize deviations and obtain a more accurate understanding of the system behaviour at different instances. The key metrics that were used in this evaluation include average path length, time taken, and observed deviation from the planned trajectory. The comparative result illustrated in Table 5-1 showed three collisions with the Default ROS navigation stack and two with NAV-YOLO in the most complex environment with dynamic and static obstacles. Although the NAV-YOLO system showed better performance in terms of completing the task in a shorter distance and time compared to other evaluated solutions, NAV-YOLO-RRT showed no collision. The proposed NAV-YOLO-RRT solution, on the other hand, displayed a slightly longer average completion time of 18 minutes compared to the NAV-YOLO and baseline system, which averaged approximately 16 and 17 minutes, respectively. The obvious contributing factor was that the solution accumulated additional processing time while replanning the path to accurately avoid obstacles.

5.4.5 Transfer to physical Turtlebot3 waffle_pi

The NAV-YOLO and NAV-YOLO- RRT solutions were transferred to the turtlebot3 robot at different instances for practical application and performance validation. Appendix A showed the Turtlebots robot used with OAK-D camera setup. The DepthAI Oak-D camera was used in the setup to enhance the robot's vision capabilities. The NAV-YOLO- RRT demonstrated proficiency in detecting and estimating obstacle positions, aligning with the camera data stream, as shown in Figure 5-9. The system computes the pose of the detected obstacle

within the laser scan frame by leveraging the bounding box centre of the detected object as illustrated in equations 5-8. This helps the robot to navigate and avoid obstacles efficiently, even in unknown office environments, as shown in Figure 5-10.

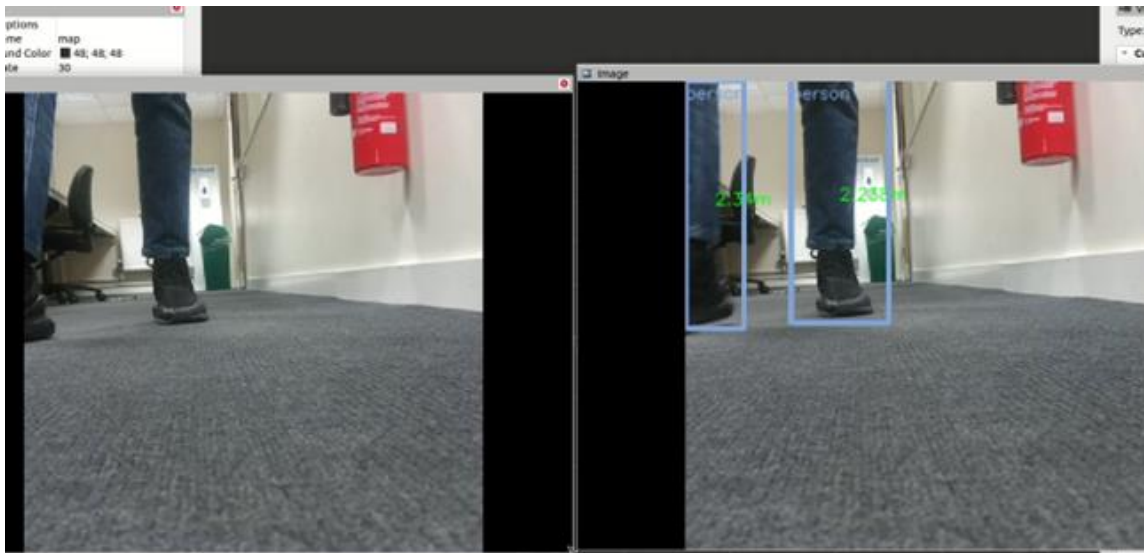


Figure 5-9: Detected and estimated obstacle position

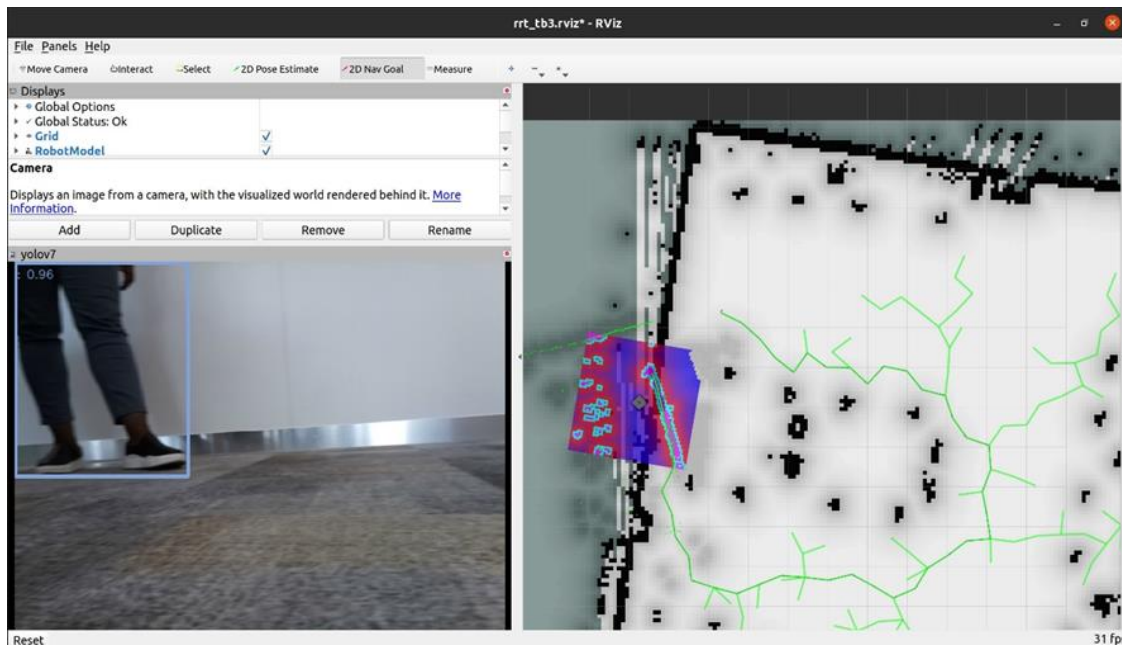


Figure 5-10: NAV-YOLO-RRT detection and path planning in the real-world robot.

5.4.6 Discussion

Analysing the performance of the developed solutions relative to the baseline set by the default ROS navigation stack experiments offers valuable insight. This provides a comprehensive understanding of its effectiveness and the challenges faced in complex navigation scenarios. The proposed system operates at approximately 0.3Hz frequency, which translates to about one frame every 3 to 4 seconds. This rate indicates the frequency at which the system responds to environmental data and was observed to achieve smooth navigation with less path replanning through to the target point. However, the 3 to 4-second delay introduced noticeable latency in its reaction, which could potentially impact obstacle detection and avoidance. This could be problematic considering the real-time responsiveness required in a dynamic hangar environment. In the next phase of the project, there will be a focus on utilising improved prediction algorithms and an upgraded robot platform. From the integrated YOLO models perspective, there was observed disparity in inference time between YOLOv3 and YOLOv5, which can be attributed in part to the architectural improvement in YOLOv5 with CSP-Darknet53 as the backbone model [29], [81]. YOLOv5 also demonstrated higher computational efficiency compared to YOLOv3, requiring 15.9GB@416 FLOPs (Floating Point Operations) against 154.9GB@416 at the same FLOPs counts required.

While YOLOv5 offers a higher potential speed of 170 fps, suitable for real-time processing demands, YOLOv7, despite a lower frame rate of 144 lower frame rate, provides better accuracy in obstacle detection and is more suited for resources-constrained Turtlebot3 robot platform due to its lower computational requirements. However, there would be a need for trade-offs and potential optimization for the integrated NAV-YOLO-RRT system to be more efficient in the hangar environment. A focus will be on improving its average completion time and distance covered compared to other solutions. While the system offers superior obstacle avoidance ability that reduces the risk of collisions, enhancing safety and adaptability which is important for real-world hangar applications, and improving navigation speed will be beneficial as well.

5.5 Conclusion

In conclusion, the YOLO model was trained with custom datasets and integrated into the ROS framework to develop an improved obstacle detection, avoidance, and path planning system tailored for mobile robots in complex and dynamic hangar environments. This is to address the issue of the increasing complexity of real-world busy hangar environments where detecting and avoiding obstacles of diverse structures and dynamics is quite challenging. Firstly, we replaced the customised YOLOv5 in an established NAV-YOLO solution with a customised YOLOv7 object detection model to improve the obstacle detection capability of the default ROS navigation stack. Second, fused LIDAR input with the transformed YOLO detected objects to expand robot's field of view. Then, the proposed NAV-YOLO-RRT was developed, effectively leveraging the RRT planner to adapt to varying environmental conditions and dynamically replan trajectories to update navigation path based on real-time sensor input. The experimental results demonstrated that the integration of the YOLOv7 with mobile robot navigation system significantly improved robot's ability to accurately detect and avoid obstacles. The solution can enable the robot to respond promptly to changes in the environment, minimizing the risk of collision and improving overall navigation performance in the changing and dynamic hangar environments.

The mobile robot can autonomously navigate through predefined checkpoints, detecting and avoiding static and dynamic obstacles accurately, generating a safe path to the target location. This achievement marks a significant contribution to the field of robotics, particularly in enhancing autonomous navigation capabilities in challenging scenarios. In future work, the focus will be on improving the NAV-YOLO-RRT navigation speed, considering a balance with detection accuracy to enhance safety as well as operation efficiency in the hangar environment. Also, addressing latency in data processing, which impacts the robot's ability to replan paths promptly in response to sudden obstacles, is another critical area. In addition, sensor fusion extension and hardware consideration will be part of focus areas for future work.

6 Conclusion & Future Work

6.1 Discussion

The research kick starts from exploring the enabling technologies of Industrial 4.0 to narrow down the best approach to implementing automation and autonomy in Aviation hangar environment. Aircraft visual inspection was realised to be one of the areas affect by the complexities and uncertainties of busy hangar environments. Focusing on improving mobile robot navigation in dynamic MRO hangar, the study identified significant gaps in obstacle avoidance and path planning efficiency. Key objectives were outlined to address these limitations, including real-time obstacle detection and adaptive path planning strategies. As a result, the study proposed NAV-YOLO, a deep learning powered navigation system that utilizes sensor fusion, obstacle detection model and dynamic path planning to overcome these challenges and navigate MRO hangars safely and efficiently. The overall aim is to develop an improved vision-based system that combines YOLO model and ROS navigation planners for mobile robot safe & intelligent navigation in unstructured and changing environments. These goals were achieved by fulfilling the outline objectives as illustrated in the next sub-sections.

6.2 Addressing the research aim and objectives.

Comprehensive Research: This study delves into different critical aspects of obstacle avoidance, detection and path planning to unlock valuable insight for developing an intelligent navigation system. These technologies were comprehensive detailed and discussed in Chapter 2 of this thesis. The leveraged cutting-edge frameworks like YOLO deep learning for obstacle detection, multi-sensory information for robust environment understanding and ROS navigation stack for obstacle avoidance and path planning. This is to bridge a gap between existing algorithms and develop an intelligent framework that could improve the capability of robotic systems to interact and navigate complex, dynamic, and changing environments.

Simulation and data capturing: The development and implementation of the model demands the use of simulation to mitigate the risk associated with experimenting and testing on real-world robot platforms. This is to reduce potential damages and cost implications during the initial phase of model development, iteration and refinement. The high-fidelity CAD model of the hangar environment in ROS-Gazebo simulation replicates real-world features and aids in generating a large number of relevant datasets. Real-world data was captured from the TUI hangar, and additional public datasets were integrated to improve the robustness and adaptability of the system to the changing real-world environment. Chapter 3 provides detailed information on how the datasets were prepared and utilized to train the YOLO object detection model.

Improved YOLO object detection model: A key improvement to the identified research gap is to develop a robotic system capable of detecting obstacles of varied shapes and sizes in different lighting situations. In Chapter 3, the proposed solution leveraged PyTorch deep learning-based YOLO algorithm to enhance obstacle detection capabilities in mobile robots. The combined datasets were processed and used to train and enhance the detection capability of the YOLO model. Another improvement with CBAM was integrated into the YOLO model to improve the speed of accurate detection and reduction in network complexity. The improved YOLOv5 model called FUSE-YOLO showed 2.7% improvement when evaluated and compared with the baseline YOLOv5 model. These are integrated into the ROS navigation stack and contribute to a more lightweight and resource-efficient navigation system.

Muti-sensor data fusion: In Chapter 4, NAV-YOLO was developed by transforming the camera stream through the YOLO model in the ROS ecosystem. However, the robot's limited field of view from the camera input became noticeable from detected collisions. These were more obvious when environmental complexities increased, practically when the number of environmental obstacles rose from 7 to about 15 incrementally.

Leveraging the ROS costmap configuration package, an improvement was done in Chapter 5, integrating data from both camera and LiDAR sensors to expand

the robot's field of view and perception capabilities. The integration of these sensory inputs allows the robot to perceive its environment with greater clarity and depth. This enriches the robot's situational awareness and refines its navigation strategies, enabling it to make more accurate and informed decisions. The applied sensor fusion techniques facilitate intelligent robotic systems capable of navigating and operating autonomously in complex environments.

NAV-YOLO development (Dijkstra+DWA+YOLOv5): The NAV-YOLO model development in Chapter 4 integrates the default ROS navigation package, including Dijkstra global and DWA local planner with YOLOv5 object detection following the same process of model development demonstrated in chapter 2. This demonstrates improvement in accurately navigating complex obstacles, enabling inspection tasks from start to end. The planners are updated with real-time interaction and adaptive potentials from transforming YOLO-detected data. As much as this model showed a 6% improvement in distance covered and reduced collision compared to the default navigation planners, there was a significant lag in data synchronisation. The Turtlebot3 robot's field of view was limited to about 62 degrees as well, which contributed to collisions in more complex and dynamic environments. This prompted more improvements and stability but set this system as a new benchmark for performance in autonomous robotic navigation.

Improved NAV-YOLO (RRT+DWA+YOLOv7): The improved version of the NAV-YOLO system was developed in Chapter 5, addressing the challenges of NAV-YOLO. This incorporates a more enhanced global path planning algorithm, the RRT, and integrated the YOLOv7 object detection model to expand the model's capability in real-time obstacle avoidance and general navigation experience. The RRT showcases path planning effectiveness in complex and dynamically changing environments, while YOLOv7 is known for its improved accuracy and speed in obstacle detection. The combined techniques utilised multi-sensory information from LiDAR and the camera to expand the robot's field of view. The YOLOv7-tiny version was used to make the system more lightweight for embedded systems. With expanded perception capability, the combined

solutions enhance the mobile robots' precision and autonomy. This enables it to efficiently navigate unknown and low lighting spaces, planning paths that maximise safety and efficiency in diverse environments. NAV-YOLO with YOLOv7 was also developed in this chapter for comparative analysis.

Transfer To Real-world Robot (Turtlebot3 waffle_pi): In chapters 4 and 5, the developed NAV-YOLO solution was implemented on the real-world Turtlebot3 robot. This robot platform faced constraint due to its limited hardware functionalities like its raspberry pi computing unit have insufficient processing power, which impacted the developed model's performance. Iterative improvements were made to the model, including optimising the YOLO algorithm with a lighter version and enhancing real-time data processing. The project's next phase intends to implement the solution on a higher-capacity hardware platform to accommodate better and leverage the developed model's capabilities.

Evaluation and Validation (ROS-Gazebo and Real Robot): The effectiveness of the developed methods was evaluated and validated using both ROS-Gazebo simulations and real-world robot implementations. The environment is composed of different checkpoints defined in Table 6-1. Chapter 2 showed the performance evaluation of the proposed FUSE-YOLO with 64% accuracy compared with approximately 63% accuracy of the YOLOv5m model trained with combined thermal and RGB images.

Comparing the integrated solutions with the default ROS navigation stack planners baseline model, the NAV-YOLO developed in Chapter 4 and measured through key evaluation metrics showed an improved path completion time of 10%, no collision at that level of complexity and high success rates from start to target locations. However, the system showed synchronisation issues between sensor data and detected data, which was critical for real-time obstacle avoidance. The NAV-YOLO was significantly improved in Chapter 5 by integrating YOLOv7 and developing the NAV-YOLO-RRT model, which demonstrated superior performance from a significantly limited number of collisions to no collision upon enhancement. The evaluation metrics show an improved path completion time of 10% and high success rates from start to target

locations. Although when the dynamic obstacles were increased to 5, the proposed solution took longer, and more distance was covered to get to the target location, but no collision was recorded. Generally, the developed model offers robust and real-time adaptability techniques for autonomous obstacle avoidance and path planning in a variety of complex settings. The successful application in simulations and real-world conditions shows its reliability and potential applicability in a wide range of autonomous robotics applications and positions it as a significant contribution to autonomous robotics.

Table 6-1: Defined checkpoints for robot navigation

S/N	Name	Goal position (x, y)
1	A	(2.080396175, 2.966490269)
2	B	(4.931311131, 15.1021328)
3	C	(10.3697052, 28.9610939)
4	D	(9.882028026, 14.74886698)
5	E	(5.00, 7.00)

In summary, the project significantly contributes to safe robot navigation in complex settings by bridging the performance gap in obstacle avoidance and path planning. Leveraging YOLO deep learning techniques within the ROS framework and integrating multi-sensory data, we significantly reduced collision rates and increased path completion and success rate. The combined algorithmic techniques development represents an advancement for safe and efficient robot operation in MRO hangars and other industries, paving the way for a future where robots seamlessly integrate into our world.

6.3 Contributions to knowledge

The contributions include:

- The proposed NAV-YOLO model is a significant advancement in the broader field of robotics and autonomous systems. It showed excellent capability to detect and avoid obstacles in complex and ever-changing environments. This contributes to safety and efficiency, which are

paramount in MRO hangars and other commercial and industrial applications.

- The integration of deep learning techniques allows the NAV-YOLO system to learn and adapt to new environments and obstacles over time. This permits the model to handle complex scenarios, especially where traditional sensor-based systems might struggle, such as environments with varying lighting conditions, diverse obstacle types and unstructured terrain.
- By integrating with the ROS Navigation Stack, the NAV-YOLO model can enhance path planning by providing more accurate and timely information about obstacles, enabling the robot to navigate moving and static obstacles safely. The ROS framework also provides a robust and well-supported framework, ensuring compatibility with a wide range of robotics platforms.
- The NAV-YOLO system, with its improved capabilities in accurate obstacle detection and navigation in real-time, can be applied in numerous domains where safety, efficiency and adaptability are crucial. This includes industrial robotics, public space service robots, and drones for inspections and surveillance.
- This development opens avenues for further research, particularly in integrating deep learning with robotics and optimising the systems for more real-time operations. NAV-YOLO can be expanded to work effectively with collaborative robots alongside human workers, ensuring precise interaction and less risk.

6.4 Future Work

The development and implementation of this model encountered numerous challenges and areas that require enhancement to solve the robotics problem.

6.4.1 Focus on Lightweight model structure.

The NAV-YOLO performance is generally impacted by the computationally intensive nature of its models. The YOLO model needs to process image data in

real-time to provide timely information for navigation. The smaller Turtletbots3 platform has limited onboard computational power that strains the system resources and affects performance. Although lighter versions of YOLOv5 and YOLOv7 were used to reduce computational demand, the developed model still lagged in efficiency and real-time capability, experiencing a delay of 0.2 milliseconds per frame. Fine-tuning specific model parameters might yield better performance.

6.4.2 Generating High Fidelity data from sensor perception

The YOLO detected data from the camera's field of view provides bounding boxes around detected obstacles, including coordinates for the corners of the box. The approach applied in this solution involves using the centre of the bounding box to estimate the object's position. This assumes the object is solid and uniform, which is not always the case. For example, a person might have a gap between their legs or an object with openings. In this case, using just the centre point can lead to the wrong data representation of these non-uniform objects. A more effective approach might be to consider the entire width of the box at a specific height. This adjustment will improve the fidelity of the generated laser scan data, especially in scenarios where the obstacles vary in shape and configuration.

6.4.3 Real-Time update

The real-time operation of converting YOLO's output into the costmap suitable format for dynamic obstacle avoidance showed some inconsistent delays. This is due to the image preprocessing and costmap update mechanism. Some critical parameter adjustments were made, including timestamp alignment using message filters to compensate for delays in data processing. Code optimization, as well as accelerating hardware functionalities, can contribute to more advanced solutions.

6.4.4 Sensor Fusion Enhancement

The LiDAR-camera data fusion in this model is focused on expanding the robot's field of view to give the robot 360 degrees of perception coverage. This sensor

fusion technique can be further expanded by complimenting camera visual data with LiDAR for precise distance measurements. Also, choosing an efficient sensor fusion method that can prioritise relevant information and minimise computational overhead while ensuring accurate obstacle representation will be appropriate for future work.

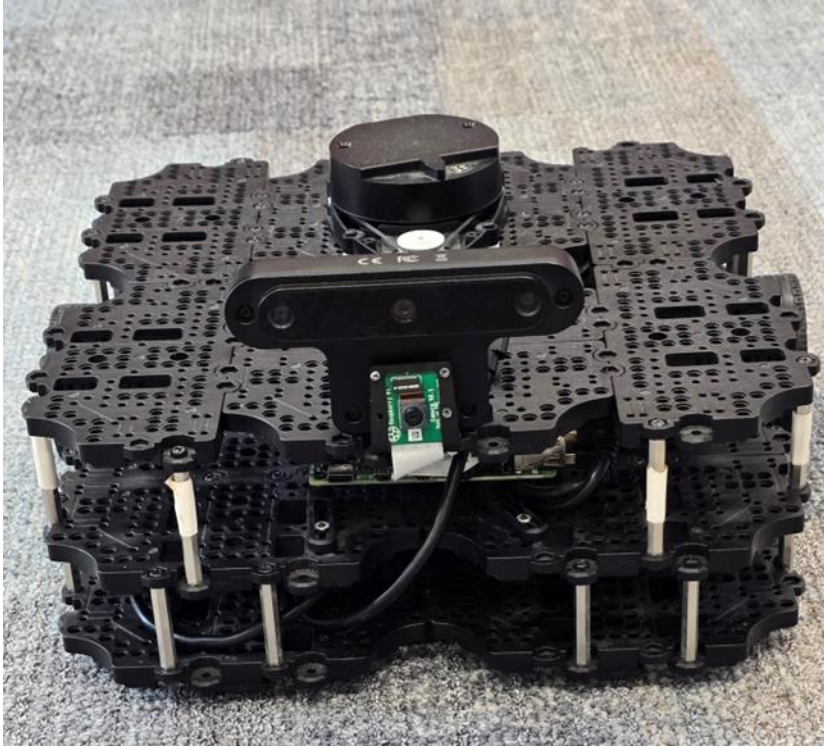
6.4.5 Enhancement of the path planning algorithms (ROS Navigation Update)

The primary aim of improving NAV-YOLO is to significantly reduce lags and improve the responsiveness of the obstacle detection and avoidance system. The processing of detected data and synchronisation with the LiDAR information still introduced notable delays in the system navigation. Specific parameters like the safety margin were adjusted to maintain a safe distance from the obstacle position. The project's future work might consider implementing an obstacle prediction algorithm to anticipate future obstacles positions and proactively update the cost map dynamic mechanism. Additionally, probabilistic approaches that factor in the likelihood of different movement scenarios will support the robot in responding effectively to dynamic changes.

6.4.6 Hardware considerations

The Turtlebot3 robot platform was problematic due to its limited computational resources. The Turtlebot3 robot is a CPU-based system with limited functionalities to experiment with advanced deep learning model integration. One solution implemented for computational limitations was to process data off-board on another computer. This introduced network latency and made the robot unable to operate in real time. Hardware upgrade should efficiently handle this computational, enhance real-time operations, and minimise safety concerns.

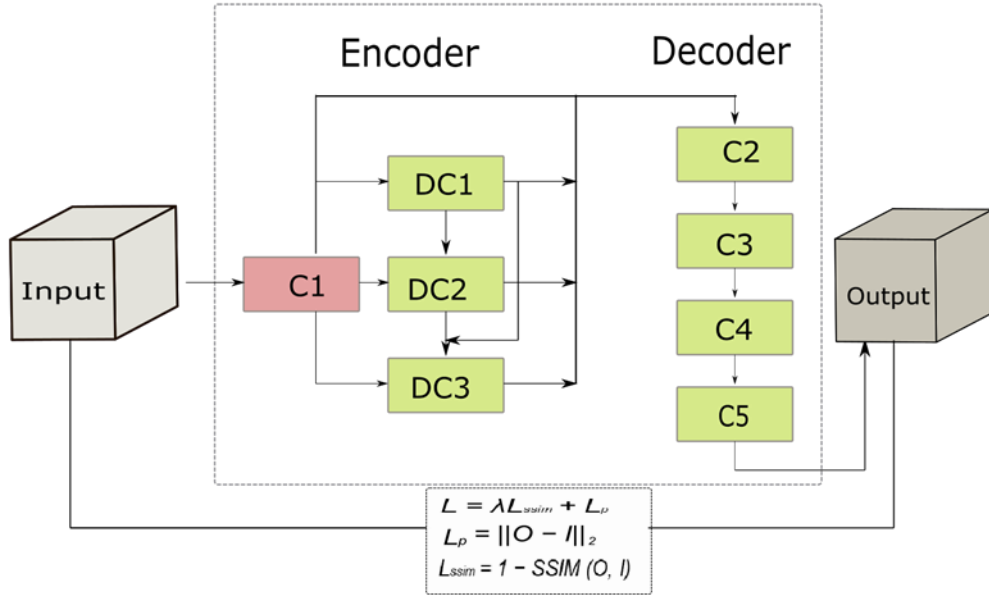
7 Appendices



Appendix A: Turtlebot3 robot with OAK_D camera.

Appendix B: DENSEFUSE

DENSEFUSE



$$f^m(x, y) = \sum_{i=1}^k \phi_i^m(x, y) \quad \dots\dots\dots \text{Additional fusion strategy}$$

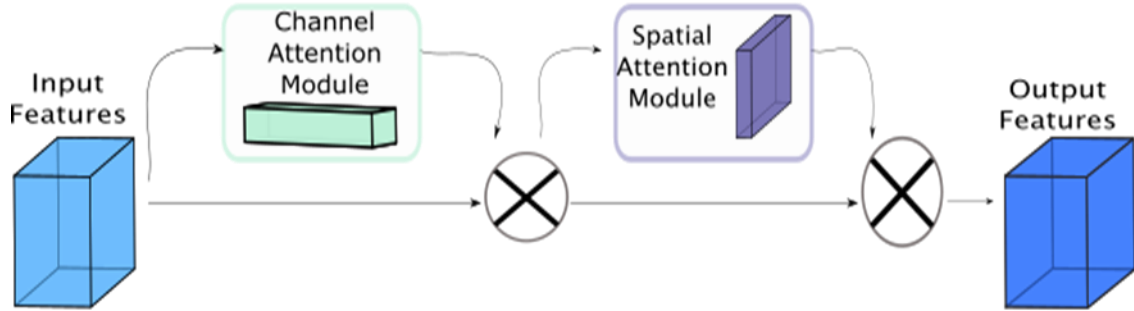
$$f^m(x, y) = w_1(x, y) \times \phi_1^m(x, y) + w_2(x, y) \times \phi_2^m(x, y) \dots\dots\dots l\text{-norm fusion strategy}$$

Other key features:

- Encoder with dense block to retain salient features
- Regularisation *l-norm* and softmax operation to avoid overfitting

Appendix C: Convolutional Block Attention Mechanism

CBAM (Channel Attention Mechanism)



$$M_c(F) = \sigma \left(MLP(AvgPool(F)) + MLP(MaxPool(F)) \right)$$

$$F' = M_c(F) \otimes F$$

$$M_s(F') = \sigma \left(f^{7 \times 7}([A; vgPool(F')MaxPool(F')]) \right)$$

$$F'' = M_s(F') \otimes F'$$

$\otimes$ = element-wise multiplication, F = the input feature map, $M_c(F)$ = channel attention map output, $M_s(F')$ = spatial attention map output, F'' = feature map output by the CBAM.

Key features:

- Channel attention to focus on the object features
- Spatial attention module focuses on target location.

Appendix D: AMCL- pose data from robot navigation.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	time	header.se	header.sti	header.fri	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose	pose.pose
2	1970/01/0	0	0	5.39E+08 map	4.020573	1.937011	0	0	0	0	0.707459	0.706754	(0.23786483687700866,	-0.002285474390902209,	0.0, 0.0, 0.0, 0.0,	-0.002285474390902209,	0.11				
3	1970/01/0	1	31	3.39E+08 map	4.02268	2.123924	0	0	0	0	0.70681	0.707404	(0.23783415243632788,	0.0033255223373487297,	0.0, 0.0, 0.0, 0.0,	0.003325522337350506,	0.05				
4	1970/01/0	2	32	1.39E+08 map	4.030948	2.326819	0	0	0	0	0.706492	0.707721	(0.22739871465781647,	-0.0013190666487439273,	0.0, 0.0, 0.0, 0.0,	-0.0013190666487439273,	0.				
5	1970/01/0	3	32	9.39E+08 map	4.037536	2.530218	0	0	0	0	0.703512	0.710684	(0.23448990909288625,	-0.0026331003399082675,	0.0, 0.0, 0.0, 0.0,	-0.0026331003399082675,	0.				
6	1970/01/0	4	33	7.39E+08 map	4.017867	2.741082	0	0	0	0	0.713296	0.700863	(0.23151916570921216,	-0.0006095607414007276,	0.0, 0.0, 0.0, 0.0,	-0.0006095607414007276,	0.				
7	1970/01/0	5	34	5.39E+08 map	4.002919	2.946182	0	0	0	0	0.70946	0.704745	(0.2373876293965722,	-0.0020424376867929084,	0.0, 0.0, 0.0, 0.0,	-0.0020424376867946847,	0.0				
8	1970/01/0	6	35	3.39E+08 map	4.039162	3.152509	0	0	0	0	0.704592	0.709612	(0.25142874776749835,	-0.002453881974881611,	0.0, 0.0, 0.0, 0.0,	-0.002453881974881611,	0.01				
9	1970/01/0	7	36	1.39E+08 map	4.03777	3.361913	0	0	0	0	0.717618	0.696437	(0.24548321304463983,	-0.00540259879720395,	0.0, 0.0, 0.0, 0.0,	-0.00540259879720395,	0.0096				
10	1970/01/0	8	36	9.39E+08 map	4.02413	3.572076	0	0	0	0	0.718866	0.695148	(0.2450934849521147,	-0.004153354524666142,	0.0, 0.0, 0.0, 0.0,	-0.004153354524666142,	0.009				
11	1970/01/0	9	37	7.39E+08 map	4.031315	3.780125	0	0	0	0	0.717134	0.696935	(0.25925058275367974,	-0.0049554924131349765,	0.0, 0.0, 0.0, 0.0,	-0.0049554924131349765,	0.				
12	1970/01/0	10	38	5.39E+08 map	4.009931	3.985691	0	0	0	0	0.727529	0.686077	(0.25568813861576345,	-0.003101347395238818,	0.0, 0.0, 0.0, 0.0,	-0.003101347395238818,	0.00				
13	1970/01/0	11	39	3.39E+08 map	4.005323	4.194079	0	0	0	0	0.728298	0.685261	(0.25467174651940283,	-0.0019357169629330429,	0.0, 0.0, 0.0, 0.0,	-0.0019357169629330429,	0.				
14	1970/01/0	12	40	1.39E+08 map	4.024529	4.408526	0	0	0	0	0.727472	0.686137	(0.27069350598213404,	-0.0017959648084158175,	0.0, 0.0, 0.0, 0.0,	-0.0017959648084158175,	0.				
15	1970/01/0	13	40	9.39E+08 map	4.006996	4.6114	0	0	0	0	0.731436	0.68191	(0.27046473908492885,	-0.004961521199572161,	0.0, 0.0, 0.0, 0.0,	-0.004961521199572161,	0.00				
16	1970/01/0	14	41	7.39E+08 map	3.972413	4.816738	0	0	0	0	0.729934	0.683518	(0.2764117415431748,	-0.009099363499284863,	0.0, 0.0, 0.0, 0.0,	-0.009099363499284863,	0.010				
17	1970/01/0	15	42	5.39E+08 map	3.956547	5.019357	0	0	0	0	0.727191	0.686435	(0.2698193258191921,	-0.008932796120241449,	0.0, 0.0, 0.0, 0.0,	-0.008932796120241449,	0.010				
18	1970/01/0	16	43	3.39E+08 map	3.89505	5.227886	0	0	0	0	0.722499	0.691372	(0.2803727069178148,	-0.010083070636184743,	0.0, 0.0, 0.0, 0.0,	-0.010083070636184743,	0.010				
19	1970/01/0	17	44	1.39E+08 map	3.904395	5.435733	0	0	0	0	0.708919	0.70529	(0.26744815448120285,	-0.006394099442896817,	0.0, 0.0, 0.0, 0.0,	-0.006394099442896817,	0.01				
20	1970/01/0	18	44	9.39E+08 map	3.860957	5.643347	0	0	0	0	0.708089	0.706123	(0.25906806487799905,	-0.01024700974572923,	0.0, 0.0, 0.0, 0.0,	-0.01024700974572923,	0.0099				
21	1970/01/0	19	45	7.39E+08 map	3.873209	5.853339	0	0	0	0	0.69918	0.714946	(0.2871167589040198,	-0.00957638031922059,	0.0, 0.0, 0.0, 0.0,	-0.00957638031922059,	0.01137				
22	1970/01/0	20	46	5.39E+08 map	3.840389	6.06744	0	0	0	0	0.703118	0.711074	(0.2647774867222399,	-0.0101056370781194,	0.0, 0.0, 0.0, 0.0,	-0.0101056370781194,	0.0101647				
23	1970/01/0	21	47	3.39E+08 map	3.826094	6.272617	0	0	0	0	0.69817	0.715932	(0.2693289031848085,	-0.010249190515292383,	0.0, 0.0, 0.0, 0.0,	-0.010249190515292383,	0.009				
24	1970/01/0	22	48	1.39E+08 map	3.83623	6.480788	0	0	0	0	0.69624	0.717809	(0.27350721299596614,	-0.008625871484230885,	0.0, 0.0, 0.0, 0.0,	-0.008625871484230885,	0.00				
25	1970/01/0	23	48	9.39E+08 map	3.873654	6.686779	0	0	0	0	0.693279	0.720669	(0.27486597484628206,	-0.007541982891833499,	0.0, 0.0, 0.0, 0.0,	-0.007541982891833499,	0.00				
26	1970/01/0	24	49	7.39E+08 map	3.878253	6.891796	0	0	0	0	0.708491	0.70572	(0.2902362987969216,	-0.00659844787503161,	0.0, 0.0, 0.0, 0.0,	-0.00659844787503161,	0.00940				
27	1970/01/0	25	50	5.39E+08 map	3.870815	7.105614	0	0	0	0	0.71994	0.694037	(0.31470705649526387,	-0.0016826902206794614,	0.0, 0.0, 0.0, 0.0,	-0.0016826902206794614,	0.				
28	1970/01/0	26	51	3.39E+08 map	3.893993	7.313459	0	0	0	0	0.726631	0.687028	(0.310755555784306,	-0.002281103408964924,	0.0, 0.0, 0.0, 0.0,	-0.002281103408964924,	0.008				
29	1970/01/0	27	52	1.39E+08 map	3.88793	7.520308	0	0	0	0	0.735479	0.677548	(0.3103014561044226,	-0.004125863048809464,	0.0, 0.0, 0.0, 0.0,	-0.004125863048809464,	0.008				
30	1970/01/0	28	52	9.39E+08 map	3.857009	7.725973	0	0	0	0	0.734812	0.678271	(0.24772567562768621,	-6.967918547928775e-06,	0.0, 0.0, 0.0, 0.0,	-6.967918544376062e-06,	0.0				

Appendix E: Hangar model launch file

File Edit Format View Help

```
<node
  name="spawn_model"
  pkg="gazebo_ros"
  type="spawn_model"
  args="-file $(find hangar_model)/urdf/hangar_model.urdf -urdf -model hangar_model"
  output="screen" />

<node
  name="fake_joint_calibration"
  pkg="rostopic"
  type="rostopic"
  args="pub /calibrated std_msgs/Bool true" />

<!-- Spawn robot -->

<arg name="model" default="waffle_pi"/>
<arg name="x_pos" default="4"/>
<arg name="y_pos" default="2"/>
<arg name="z_pos" default="0"/>
<arg name="yaw_pos" default="1.57125"/>
<param name="robot_description" command="$(find xacro)/xacro --inorder '$(find turtlebot3_description)/urdf/turtlebot3_$(arg model).urdf.xacro'" />
<node name="spawn_robot" pkg="gazebo_ros" type="spawn_model" args="-urdf -model turtlebot3_$(arg model) -x $(arg x_pos) -y $(arg y_pos) -z $(arg z_pos) -Y $(arg yaw_pos) -param
  $(arg robot_description)" />

<!-- Load the turtlebot3_navigation stack -->
<include file="$(find turtlebot3_navigation)/launch/turtlebot3_navigation.launch">
  <arg name="map_file" value="$(find hangar_model)/map/hangar_map_v1.yaml"/>
  <arg name="initial_pose_x" value="$(arg x_pos)"/>
  <arg name="initial_pose_y" value="$(arg y_pos)"/>
  <arg name="initial_pose_a" value="$(arg yaw_pos)"/>
  <param name="robot_description" command="$(find xacro)/xacro --inorder '$(find turtlebot3_description)/urdf/turtlebot3_$(arg model).urdf.xacro'" />
</include>

<!-- YOLOv7 launch file -->
<include file="$(find hangar_model)/launch/yolov7.launch" />

<!-- rqt_image_view -->
<node pkg="rqt_image_view" type="rqt_image_view" name="rqt_image_view" args="/yolov7/yolov7/visualization" respawn="false" output="screen" />

<!-- Load obstacles -->
<include file="$(find hangar_model)/launch/obstacles.launch" />

</launch>
```

Appendix F: RRT_DWB planner launch file.

File Edit Format View Help

```
<launch>
  <!-- Arguments -->
  <arg name="model" default="$(env TURTLEBOT3_MODEL)" doc="model type [burger, waffle, waffle_pi]"/>
  <arg name="map_file" default="$(find turtlebot3_navigation)/maps/map.yaml"/>
  <arg name="move_forward_only" default="false"/>
  <arg name="cmd_vel_topic" default="/cmd_vel" />
  <arg name="odom_topic" default="odom" />

  <!-- Gazebo -->
  <include file="$(find turtlebot3_gazebo)/launch/turtlebot3_world.launch">
    <arg name="gui" value="false"/>
    <arg name="world_name" value="$(find turtlebot3_gazebo)/worlds/turtlebot3_world.world"/>
  </include>

  <!-- Turtlebot3 -->
  <include file="$(find turtlebot3_bringup)/launch/turtlebot3_remote.launch">
    <arg name="model" value="$(arg model)" />
  </include>

  <!-- Map server -->
  <node pkg="map_server" name="map_server" type="map_server" args="$(arg map_file)"/>

  <!-- AMCL -->
  <include file="$(find turtlebot3_navigation)/launch/amcl.launch"/>

  <!-- move_base -->
  <node pkg="move_base" type="move_base" respawn="false" name="move_base" output="screen">
    <param name="base_local_planner" value="dwa_local_planner/DWAPLannerROS" />
    <param name="base_global_planner" value="global_planner/RTTGlobalPlanner"/>
    <rosparam file="$(find turtlebot3_navigation)/param/costmap_common_params_$(arg model).yaml" command="load" />
    <rosparam file="$(find turtlebot3_navigation)/param/costmap_common_params_$(arg model).yaml" command="load" />
    <rosparam file="$(find turtlebot3_navigation)/param/local_costmap_params.yaml" command="load" />
    <rosparam file="$(find turtlebot3_navigation)/param/global_costmap_params.yaml" command="load" />
    <rosparam file="$(find turtlebot3_navigation)/param/move_base_params.yaml" command="load" />
    <rosparam file="$(find turtlebot3_navigation)/param/dwa_local_planner_params_$(arg model).yaml" command="load" />
    <rosparam file="$(find rrt-global-planner)/params/rrt_global_planner.yaml" command="load" />
    <remap from="cmd_vel" to="$(arg cmd_vel_topic)"/>
    <remap from="odom" to="$(arg odom_topic)"/>
    <param name="DWAPLannerROS/min_vel_x" value="0.0" if="$(arg move_forward_only)" />
  </node>

  <!-- rviz -->
  <node pkg="rviz" type="rviz" name="rviz" required="true">
```

Appendix G: YOLOv7 launch file

yolov7.launch - Notepad

File Edit Format View Help

launch>

```
<arg name="img_topic" default="/yolov4_publisher/color/image"/>
<arg name="depth_image_topic" default="/yolov4_publisher/stereo/depth"/>
<arg name="depth_info_topic" default="/yolov4_publisher/color/camera_info"/>

<node pkg="yolov7_ros" type="detect_ros.py" name="detect" output="screen"
ns="yolov7">
  <!-- Download the official weights from the original repo -->
  <param name="weights_path" type="str"
value="$(find yolov7_ros)/config/weights/yolov7.pt"/>
  <!-- Path to a class_labels.txt file containing your desired class labels. The i-th entry correspond
  <param name="classes_path" type="str" value="$(find yolov7_ros)/config/class_labels/coco.txt" />

  <!-- topic name to subscribe to -->
  <param name="img_topic" type="str" value="$(arg img_topic)" />
  <param name="depth_image_topic" type="str" value="$(arg depth_image_topic)" />
  <param name="depth_info_topic" type="str" value="$(arg depth_info_topic)" />

  <!-- topic name for the detection output -->
  <param name="out_topic" type="str" value="yolov7" />
  <!-- confidence threshold -->
  <param name="conf_thresh" type="double" value="0.35" />
  <!-- intersection over union threshold -->
  <param name="iou_thresh" type="double" value="0.45" />
  <!-- queue size for publishing -->
  <param name="queue_size" type="int" value="1" />
  <!-- image size to which to resize each input image before feeding into the
  network (the final output is rescaled to the original image size) -->
  <param name="img_size" type="int" value="640" />
  <!-- flag whether to also publish image with the visualized detections -->
  <param name="visualize" type="bool" value="true" />
  <!-- 'cuda' or 'cpu' -->
  <param name="device" type="str" value="cpu" />
</node>
/launch>
```

References

- [1] M. Soori, B. Arezoo, and R. Dastres, "Artificial intelligence, machine learning and deep learning in advanced robotics, a review," *Cognitive Robotics*, vol. 3, pp. 54–70, Jan. 2023, doi: 10.1016/J.COGR.2023.04.001.
- [2] R. N. Albustanji, S. Elmanaseer, and A. A. A. Alkhatib, "Robotics: Five Senses plus One—An Overview," *Robotics 2023, Vol. 12, Page 68*, vol. 12, no. 3, p. 68, May 2023, doi: 10.3390/ROBOTICS12030068.
- [3] J. Fayyad, M. A. Jaradat, D. Gruyer, and H. Najjaran, "Deep Learning Sensor Fusion for Autonomous Vehicle Perception and Localization: A Review," *Sensors 2020, Vol. 20, Page 4220*, vol. 20, no. 15, p. 4220, Jul. 2020, doi: 10.3390/S20154220.
- [4] D. Chikurtev, "Mobile Robot Simulation and Navigation in ROS and Gazebo," *2020 International Conference Automatics and Informatics, ICAI 2020 - Proceedings*, Oct. 2020, doi: 10.1109/ICAII50593.2020.9311330.
- [5] T. P. Sattar, S. Chen, B. Bridge, and J. Shang, "ROBAIR: MOBILE ROBOTIC SYSTEM TO INSPECT AIRCRAFT WINGS AND FUSELAGE".
- [6] C. Medina Sánchez, M. Zella, J. Capitán, and P. J. Marrón, "From Perception to Navigation in Environments with Persons: An Indoor Evaluation of the State of the Art," *Sensors 2022, Vol. 22, Page 1191*, vol. 22, no. 3, p. 1191, Feb. 2022, doi: 10.3390/S22031191.
- [7] L. Wijayathunga, A. Rassau, and D. Chai, "Challenges and Solutions for Autonomous Ground Robot Scene Understanding and Navigation in Unstructured Outdoor Environments: A Review," *Applied Sciences 2023, Vol. 13, Page 9877*, vol. 13, no. 17, p. 9877, Aug. 2023, doi: 10.3390/APP13179877.
- [8] F. Rubio, F. Valero, and C. Llopis-Albert, "A review of mobile robots: Concepts, methods, theoretical framework, and applications," 2019, doi: 10.1177/1729881419839596.

- [9] H. Qin, S. Shao, T. Wang, X. Yu, Y. Jiang, and Z. Cao, "Review of Autonomous Path Planning Algorithms for Mobile Robots," *Drones* 2023, Vol. 7, Page 211, vol. 7, no. 3, p. 211, Mar. 2023, doi: 10.3390/DRONES7030211.
- [10] M. Reda, A. Onsy, A. Ghanbari, and A. Y. Haikal, "Path planning algorithms in the autonomous driving system: A comprehensive review," *Rob Auton Syst*, p. 104630, Jan. 2024, doi: 10.1016/J.ROBOT.2024.104630.
- [11] H. Hu, Y. Wang, W. Tong, J. Zhao, and Y. Gu, "Path Planning for Autonomous Vehicles in Unknown Dynamic Environment Based on Deep Reinforcement Learning," *Applied Sciences* 2023, Vol. 13, Page 10056, vol. 13, no. 18, p. 10056, Sep. 2023, doi: 10.3390/APP131810056.
- [12] X. Jia, C. Zhu, M. Li, W. Tang, and W. Zhou, "LLVIP: A Visible-infrared Paired Dataset for Low-light Vision", Accessed: Mar. 30, 2023. [Online]. Available: <https://bupt-ai-cz.github.io/>
- [13] "TUI Airlines | Hangar 61 | SDC." Accessed: Mar. 30, 2024. [Online]. Available: <https://www.sdc.co.uk/portfolio/hangar-61/>
- [14] N. Adiuku, N. P. Avdelidis, G. Tang, A. Plastropoulos, and S. Perinpanayagam, "CNN-fusion architecture with visual and thermographic images for object detection," <https://doi.org/10.1117/12.2665984>, vol. 12536, no. 12, pp. 180–189, Jun. 2023, doi: 10.1117/12.2665984.
- [15] J. P. Sprong, X. Jiang, and H. Polinder, "Deployment of Prognostics to Optimize Aircraft Maintenance – A Literature Review," *JOURNAL OF INTERNATIONAL BUSINESS RESEARCH AND MARKETING*, vol. 5, no. 4, pp. 26–37, 2020, doi: 10.18775/jibrm.1849-8558.2015.54.3004.
- [16] M. Kaur Dhoot, I.-S. Fan, and Z. Skaf, "Review of Robotic Systems for Aircraft Inspection," *SSRN Electronic Journal*, Oct. 2020, doi: 10.2139/SSRN.3718054.
- [17] M. Lakrouf, S. Larnier, M. Devy, and N. Achour, "Moving obstacles detection and camera pointing for mobile robot applications," *ACM*

International Conference Proceeding Series, vol. Part F128050, pp. 57–62, Feb. 2017, doi: 10.1145/3068796.3068816.

- [18] N. V H and P. V Venkatesh, “Comparison Review on LiDAR vs Camera in Autonomous Vehicle,” *International Research Journal of Engineering and Technology*, 2020, Accessed: Nov. 06, 2023. [Online]. Available: www.irjet.net
- [19] U. Papa and S. Ponte, “Preliminary Design of an Unmanned Aircraft System for Aircraft General Visual Inspection,” *Electronics 2018, Vol. 7, Page 435*, vol. 7, no. 12, p. 435, Dec. 2018, doi: 10.3390/ELECTRONICS7120435.
- [20] D. Fox, W. Burgard, and S. Thrun, “The dynamic window approach to collision avoidance,” *IEEE Robot Autom Mag*, vol. 4, no. 1, pp. 23–33, Mar. 1997, doi: 10.1109/100.580977.
- [21] Z. Wu, Z. Meng, W. Zhao, and Z. Wu, “Fast-RRT: A RRT-based optimal path finding method,” *Applied Sciences (Switzerland)*, vol. 11, no. 24, Dec. 2021, doi: 10.3390/app112411777.
- [22] F. Ahmad, “Application of Dijkstra Algorithm for Robot’s Obstacle Avoidance System in A Simulated Environment Using ROS (Robot Operating System) and Gazebo Simulator”.
- [23] W. Zhang, S. Wei, Y. Teng, J. Zhang, X. Wang, and Z. Yan, “Dynamic Obstacle Avoidance for Unmanned Underwater Vehicles Based on an Improved Velocity Obstacle Method”, doi: 10.3390/s17122742.
- [24] D. Zhou, Z. Wang, S. Bandyopadhyay, and M. Schwager, “Fast, On-line Collision Avoidance for Dynamic Vehicles Using Buffered Voronoi Cells,” *IEEE Robot Autom Lett*, vol. 2, no. 2, pp. 1047–1054, Apr. 2017, doi: 10.1109/LRA.2017.2656241.
- [25] K. Zhu and T. Zhang, “Deep Reinforcement Learning Based Mobile Robot Navigation: A Review,” 2021.

- [26] A. Nagabandi, I. Clavera, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, C. Finn, "Learning to Adapt In Dynamic, Real-World Environments Through Meta-Reinforcement Learning", International Conference on Learning Representations (ICLR), Feb. 2018, doi: 10.48550/arXiv.1803.11347.
- [27] S. Koh, B. Zhou, H. Fang, P. Yang, Z. Yang, Q. Yang, L. Guan, and Z. Ji, "Real-time deep reinforcement learning based vehicle navigation," *Applied Soft Computing*, vol. 96, p. 106694, Nov. 2020, doi: 10.1016/J.ASOC.2020.106694.
- [28] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Trans Pattern Anal Mach Intell*, vol. 39, no. 06, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.
- [29] H. Marko, J. Ljudevit, G. Gordan "A comparative study of YOLOv5 models performance for image localization and classification", 2022.
- [30] I. Jovančević, J.-J. Orteu, T. Sentenac, R. Gilblas Igor Jovančević, R. Gilblas, and I. Jovančević, "Automated visual inspection of an airplane exterior," <https://doi.org/10.1117/12.2182811>, vol. 9534, pp. 247–255, Apr. 2015, doi: 10.1117/12.2182811.
- [31] M. Thanavin, I. Fan and G. Tang "Navigation for a mobile robot to inspect aircraft," July 2023, doi: 10.1109/YEF-ECE58420.2023.10209290.
- [32] Y. Sun and O. Ma, "Automating Aircraft Scanning for Inspection or 3D Model Creation with a UAV and Optimal Path Planning," *Drones 2022, Vol. 6, Page 87*, vol. 6, no. 4, p. 87, Mar. 2022, doi: 10.3390/DRONES6040087.
- [33] D. Leca, V. Cadenat, T. Sentenac, A. Durand-Petiteville, F. Gouaisbaut, and E. Le Flecher, "Sensor-based obstacles avoidance using spiral controllers for an aircraft maintenance inspection robot," *2019 18th European Control Conference, ECC 2019*, pp. 2083–2089, Jun. 2019, doi: 10.23919/ECC.2019.8795882.

- [34] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, "A Survey of Deep Learning Applications to Autonomous Vehicle Control".
- [35] A. Gupta, A. Anpalagan, L. Guan, and A. S. Khwaja, "Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues," *Array*, vol. 10, p. 100057, Jul. 2021, doi: 10.1016/J.ARRAY.2021.100057.
- [36] D. Feng *et al.*, "Deep Multi-Modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1341–1360, Mar. 2021, doi: 10.1109/TITS.2020.2972974.
- [37] S. A. Khan, H. J. Lee, and H. Lim, "Enhancing Object Detection in Self-Driving Cars Using a Hybrid Approach," *Electronics 2023, Vol. 12, Page 2768*, vol. 12, no. 13, p. 2768, Jun. 2023, doi: 10.3390/ELECTRONICS12132768.
- [38] H. Meng and H. Zhang, "Mobile Robot Path Planning Method Based on Deep Reinforcement Learning Algorithm," <https://doi.org/10.1142/S0218126622502589>, Jun. 2022, doi: 10.1142/S0218126622502589.
- [39] S. Y. Alaba and J. E. Ball, "Deep Learning-Based Image 3-D Object Detection for Autonomous Driving: Review," *IEEE Sens J*, vol. 23, no. 4, pp. 3378–3394, Feb. 2023, doi: 10.1109/JSEN.2023.3235830.
- [40] T. Zhang and H. Mo "Reinforcement learning for robot research_ A comprehensive review and open issues", June 2021, doi:10.1177/17298814211007305
- [41] M. W. Otte, "A Survey of Machine Learning Approaches to Robotic Path-Planning."
- [42] G. Dulac-Arnold, D. Mankowitz, and T. Hester, "Challenges of Real-World Reinforcement Learning," Apr. 2019, Accessed: Oct. 19, 2023. [Online]. Available: <https://arxiv.org/abs/1904.12901v1>

- [43] J. Shabbir, and T. Answer, "A Survey of Deep Learning Techniques for Mobile Robot Applications", vol.14, Mar. 2018, Doi: 10.48550/arXiv.1803.07608
- [44] K. Almazrouei, I. Kamel, and T. Rabie, "Dynamic Obstacle Avoidance and Path Planning through Reinforcement Learning," *Applied Sciences* 2023, vol. 13, no. 14, p. 8174, Jul. 2023, doi: 10.3390/APP13148174.
- [45] P. Gao, Z. Liu, Z. Wu, and D. Wang, "A global path planning algorithm for robots using reinforcement learning," in *IEEE International Conference on Robotics and Biomimetics, ROBIO 2019*, Institute of Electrical and Electronics Engineers Inc., Dec. 2019, pp. 1693–1698. doi: 10.1109/ROBIO49542.2019.8961753.
- [46] O. Bouhamed, H. Ghazzai, H. Besbes and Y. Massoud, "Autonomous UAV Navigation: A DDPG-Based Deep Reinforcement Learning Approach," 2020 IEEE International Symposium on Circuits and Systems (ISCAS), 2020, pp. 1-5, doi: 10.1109/ISCAS45731.2020.9181245.
- [47] M. F. Abdelwahed, A. E. Mohamed, and M. A. Saleh "Solving the motion planning problem using learning experience through case-based reasoning and machine learning algorithms", *Ain Shams Engineering Journal*, vol. 11, p. 133-142, March 2020.
- [48] J. Hua, L. Zeng, G. Li, and Z. Ju, "Learning for a Robot: Deep Reinforcement Learning, Imitation Learning, Transfer Learning," *Sensors (Basel)*, vol. 21, no. 4, pp. 1–21, Feb. 2021, doi: 10.3390/S21041278.
- [49] T. Thanh Van Nguyen, M. Duong Phung, and Q. Vinh Tran, "Behavior-based Navigation of Mobile Robot in Unknown Environments Using Fuzzy Logic and Multi-Objective Optimization."
- [50] B. Cèsar-Tondreau, G. Warnell, E. Stump, K. Kochersberger, and N. R. Waytowich, "Improving Autonomous Robotic Navigation Using Imitation Learning," *Front Robot AI*, vol. 8, Jun. 2021, doi: 10.3389/frobt.2021.627730.

- [51] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *International Journal of Robotics Research*, vol. 35, no. 11, pp. 1352–1370, Sep. 2016, doi: 10.1177/0278364915619772.
- [52] A. Lookingbill, J. Rogers, D. Lieb, J. Curry, and S. Thrun, "Reverse optical flow for self-supervised adaptive autonomous robot navigation," *Int J Comput Vis*, vol. 74, no. 3, pp. 287–302, Sep. 2007, doi: 10.1007/s11263-006-0024-x.
- [53] B. K. Patle, G. Babu L, A. Pandey, D. R. K. Parhi, and A. Jagadeesh, "A review: On path planning strategies for navigation of mobile robot," *Defence Technology*, vol. 15, no. 4, pp. 582–606, Aug. 2019, doi: 10.1016/J.DT.2019.04.011.
- [54] S. Deshpande, A. K. Kashyap, and B. K. Patle, "A review on path planning ai techniques for mobile robots," *Robotic Systems and Applications*, vol. 3, no. 1, pp. 27–46, Jun. 2023, doi: 10.21595/RSA.2023.23090.
- [55] S. Janji and A. Kliks, "Neural Networks for Path Planning," Jul. 2022, [Online]. Available: <http://arxiv.org/abs/2207.00874>
- [56] H. Bharadwaj and E. Vinodh Kumar, "Comparative study of neural networks in path planning for catering robots," in *Procedia Computer Science*, Elsevier B.V., 2018, pp. 417–423. doi: 10.1016/j.procs.2018.07.051.
- [57] W. Du, Q. Zhang, Z. He and X. Wang "Real Time Neural Network Path Planning Algorithm for Robot," *International Journal of Frontiers in Engineering Technology*, vol. 3, no. 5, 2021, doi: 10.25236/ijfet.2021.030507.
- [58] A. Galiza, M.V. Alves, L. K. Carvalho, J. C. Basilio, "Supervisory Control-Based Navigation Architecture_ A New Framework for Autonomous Robots in Industry 4.0 Environments", Jan. 2019, DOI:10.17648/sbai-2019-111525
- [59] K. Kurzer, "Path Planning in Unstructured Environments: A Real-time Hybrid A* Implementation for Fast and Deterministic Path Generation for

the KTH Research Concept Vehicle Situation assessment and semantic maneuver planning under consideration of uncertainties for cooperative vehicles View project Path Planning in Unstructured Environments A Real-time Hybrid A* Implementation for Fast and Deterministic Path Generation for the KTH Research Concept Vehicle,” 2016, doi: 10.13140/RG.2.2.10091.49444.

- [60] I. Jovancevic, J.-J. Orteu, T. Sentenac, I. J. Jovančević, and R. Gilblas, “Automated visual inspection of an airplane exterior FASTE 2 View project Real Time True Temperature Measurement View project Automated visual inspection of an airplane exterior,” 2015.
- [61] C. Sevastopoulos and S. Konstantopoulos, “A SURVEY OF TRAVERSABILITY ESTIMATION FOR MOBILE ROBOTS”.
- [62] M. K. Dhoot, I.-S. Fan, and Z. Skaf, “Review of Robotic Systems for Aircraft Inspection”, Accessed: Dec. 04, 2023. [Online]. Available: <https://ssrn.com/abstract=3718054>
- [63] R. S. Batth, A. Nayyar, and A. Nagpal, “Internet of Robotic Things: Driving Intelligent Robotics of Future - Concept, Architecture, Applications and Technologies,” *Proceedings - 4th International Conference on Computing Sciences, ICCS 2018*, pp. 151–160, Jul. 2019, doi: 10.1109/ICCS.2018.00033.
- [64] T. Li, W. Xu, W. Wang, and X. Zhang, “Obstacle detection in a field environment based on a convolutional neural network security,” *Enterp Inf Syst*, vol. 16, no. 3, pp. 472–493, Mar. 2022, doi: 10.1080/17517575.2020.1797180.
- [65] F. Valenti, D. Giaquinto, L. Musto, A. Zinelli, M. Bertozzi, and A. Broggi, “Enabling Computer Vision-Based Autonomous Navigation for Unmanned Aerial Vehicles in Cluttered GPS-Denied Environments,” *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, vol. 2018-Novem. pp. 3886–3891, 2018. doi: 10.1109/ITSC.2018.8569695.

- [66] R. A. Rill and K. B. Faragó, "Collision Avoidance Using Deep Learning-Based Monocular Vision," *SN Comput Sci*, vol. 2, no. 5, pp. 1–10, Sep. 2021, doi: 10.1007/S42979-021-00759-6/TABLES/2.
- [67] C. Zhou, B. Huang, and P. Fränti, "A review of motion planning algorithms for intelligent robots," *Journal of Intelligent Manufacturing*, vol. 33, no. 2. Springer, pp. 387–424, Feb. 01, 2022. doi: 10.1007/s10845-021-01867-z.
- [68] X. Song, H. Fang, X. Jiao, and Y. Wang, "Autonomous mobile robot navigation using machine learning," in *ICIAFS 2012 - Proceedings: 2012 IEEE 6th International Conference on Information and Automation for Sustainability*, 2012, pp. 135–140. doi: 10.1109/ICIAFS.2012.6419894.
- [69] O. Vermesan, R. Bahr, M. Ottella, M. Serrano, T. Karlsen, T. Wahlstrøm, H. E. Sand, M. Ashwathnarayan and M. T. Gamba, "Internet of Robotic Things Intelligent Connectivity and Platforms," *Frontiers in Robotics and AI*, vol. 7. 2020. doi: 10.3389/frobt.2020.00104.
- [70] K. L. Masita, A. N. Hasan, and T. Shongwe, *Deep Learning in Object Detection: a Review*. 2020.
- [71] M. Nowakowski and J. Kurylo, "Usability of Perception Sensors to Determine the Obstacles of Unmanned Ground Vehicles Operating in Off-Road Environments," *Applied Sciences 2023, Vol. 13, Page 4892*, vol. 13, no. 8, p. 4892, Apr. 2023, doi: 10.3390/APP13084892.
- [72] A. Ennajar, N. Khouja, R. Boutteau, and F. Tlili, "Deep Multi-modal Object Detection for Autonomous Driving," p. 10, 2021, doi: 10.1109/SSD52085.2021.9429355.
- [73] X. Dang, Z. Rong, and X. Liang, "Sensor Fusion-Based Approach to Eliminating Moving Objects for SLAM in Dynamic Environments," *Sensors (Basel)*, vol. 21, no. 1, pp. 1–24, Jan. 2021, doi: 10.3390/S21010230.
- [74] Z. Feng, L. Jing, P. Yin, Y. Tian, and B. Li, "Advancing Self-supervised Monocular Depth Learning with Sparse LiDAR," Sep. 2021, Accessed: Oct. 15, 2023. [Online]. Available: <https://arxiv.org/abs/2109.09628v4>

- [75] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "CenterNet++ for Object Detection," Apr. 2022, Accessed: Jun. 06, 2023. [Online]. Available: <http://arxiv.org/abs/2204.08394>
- [76] P. Huang, P. Huang, Z. Wang, X. Wu, J. Liu, and L. Zhu, "Deep-Learning-Based Trunk Perception with Depth Estimation and DWA for Robust Navigation of Robotics in Orchards," *Agronomy* 2023, Vol. 13, Page 1084, vol. 13, no. 4, p. 1084, Apr. 2023, doi: 10.3390/AGRONOMY13041084.
- [77] C. Y. Chen, S. Y. Chiang, and C. Te Wu, "Path planning and obstacle avoidance for omni-directional mobile robot based on Kinect depth sensor," *Int J Embed Syst*, vol. 8, no. 4, pp. 343–351, 2016, doi: 10.1504/IJES.2016.077793.
- [78] D. Maier, A. Hornung, and M. Bennewitz, "Real-Time Navigation in 3D Environments Based on Depth Camera Data," 2012.
- [79] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection", Accessed: Aug. 30, 2023. [Online]. Available: <http://pjreddie.com/yolo/>
- [80] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu and A. C. Berg, "SSD: Single Shot MultiBox Detector," *Lecture Notes in Computer Science*, vol. 9905, pp. 21–37, Sept. 2016, doi: 10.1007/978-3-319-46448-0_2.
- [81] T. Mahendrakar *et al.*, "Performance Study of YOLOv5 and Faster R-CNN for Autonomous Navigation around Non-Cooperative Targets," *IEEE Aerospace Conference Proceedings*, vol. 2022-March, 2022, doi: 10.1109/AERO53065.2022.9843537.
- [82] A.-T. Nguyen and C.-T. Vu, "Obstacle Avoidance for Autonomous Mobile Robots Based on Mapping Method".
- [83] N. Mebarki Pausti, R. Ndeda Jkuat, N. Ouadah Cdta, And C. O. Adika, "Multi-Object Detection For Autonomous Motion Planning Based On Convolutional Neural Networks", Accessed: Nov. 19, 2023. [Online]. Available: <Http://Www.Irphouse.Com>

- [84] A. D. Sabiha, M. A. Kamel, E. Said, and W. M. Hussein, "Real-time path planning for autonomous vehicle based on teaching–learning-based optimization," *Intell Serv Robot*, vol. 15, no. 3, pp. 381–398, Jul. 2022, doi: 10.1007/S11370-022-00429-3/FIGURES/19.
- [85] J. R. Sánchez-Ibáñez, C. J. Pérez-Del-pulgar, and A. García-Cerezo, "Path Planning for Autonomous Mobile Robots: A Review," *Sensors 2021*, Vol. 21, Page 7898, vol. 21, no. 23, p. 7898, Nov. 2021, doi: 10.3390/S21237898.
- [86] Y. Chen and Y. Zheng, "A Review of Autonomous Vehicle Path Tracking Algorithm Research," 2022, doi: 10.22541/au.166990184.45015188/v1.
- [87] M. Quiñones-Ramírez, J. Ríos-Martínez, and V. Uc-Cetina, "Robot path planning using deep reinforcement learning," 2023, Accessed: Nov. 19, 2023. [Online]. Available: <https://gazebosim.org/home>
- [88] K. Geng and S. Liu, "Robust Path Tracking Control for Autonomous Vehicle Based on a Novel Fault Tolerant Adaptive Model Predictive Control Algorithm," *Applied Sciences 2020*, Vol. 10, Page 6249, vol. 10, no. 18, p. 6249, Sep. 2020, doi: 10.3390/APP10186249.
- [89] M. Rokonuzzaman, N. Mohajer, S. Nahavandi, and S. Mohamed, "Review and performance evaluation of path tracking controllers of autonomous vehicles," *IET Intelligent Transport Systems*, vol. 15, no. 5, pp. 646–670, May 2021, doi: 10.1049/ITR2.12051.
- [90] A. B. Martinsen, A. M. Lekkas, and S. Gros, "Reinforcement learning-based NMPC for tracking control of ASVs: Theory and experiments," *Control Eng Pract*, vol. 120, p. 105024, Mar. 2022, doi: 10.1016/J.CONENGPRAC.2021.105024.
- [91] K. Almazrouei, I. Kamel, and T. Rabie, "Dynamic Obstacle Avoidance and Path Planning through Reinforcement Learning," *Applied Sciences 2023*, Vol. 13, Page 8174, vol. 13, no. 14, p. 8174, Jul. 2023, doi: 10.3390/APP13148174.

- [92] X. Liang, Y. Ma, Y. Feng, and Z. Liu, "PTR-PPO: Proximal Policy Optimization with Prioritized Trajectory Replay," Dec. 2021, [Online]. Available: <http://arxiv.org/abs/2112.03798>
- [93] H. Gong, P. Wang, C. Ni, and N. Cheng, "Efficient Path Planning for Mobile Robot Based on Deep Deterministic Policy Gradient," *Sensors (Basel)*, vol. 22, no. 9, May 2022, doi: 10.3390/S22093579.
- [94] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, "Trust Region Policy Optimization," Feb. 2015, [Online]. Available: <http://arxiv.org/abs/1502.05477>
- [95] H. Charlesworth and G. Montana, "PlanGAN: Model-based Planning With Sparse Rewards and Multiple Goals."
- [96] D. Silver, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic Policy Gradient Algorithms."
- [97] T. Wang, H. Zhang, W. Q. Toh, H. Zhu, C. Tan, Y. Wu, Y. Liu and W. Jing "Efficient Robotic Task Generalization Using Deep Model Fusion Reinforcement Learning", IEEE International Conference on Robotics and Biomimetics (ROBIO), dec. 2019, doi: 10.48550/arXiv.1912.05205.
- [98] L. Xie, S. Wang, A. Markham, and N. Trigoni, "Towards Monocular Vision based Obstacle Avoidance through Deep Reinforcement Learning".
- [99] T. Yan, Y. Zhang, and B. Wang, "Path Planning for Mobile Robot's Continuous Action Space Based on Deep Reinforcement Learning," *International Conference on Big Data and Artificial Intelligence, BDAI 2018*, pp. 42–46, Nov. 2018, doi: 10.1109/BDAI.2018.8546675.
- [100] X. Gao, L. Yan, G. Wang, T. Wang, N. Du, and C. Gerada, "Toward Obstacle Avoidance for Mobile Robots Using Deep Reinforcement Learning Algorithm," *Proceedings of the 16th IEEE Conference on Industrial Electronics and Applications, ICIEA 2021*, pp. 2136–2139, Aug. 2021, doi: 10.1109/ICIEA51954.2021.9516114.

- [101] A. Ando, and G. Capi, "Deep Learning Based Object Recognition and Robot Navigation in Indoor Environments", Management and Innovation Technology International Conference, dec. 2017.
- [102] D. N. Thang, La. A. Nguyen, P. Trung Dung, T. D. Khoa, N. H. Son, N. T. Hiep, P. V. Nguyen, V. D. Truong, D. H. Toan, N. M. Hung, T. Ngo and X. Truong, "Deep Learning-based Multiple Objects Detection and Tracking System for Socially Aware Mobile Robot Navigation Framework," *NAFOSTED Conference on Information and Computer Science*, pp. 436–441, Jan. 2019, doi: 10.1109/NICS.2018.8606878.
- [103] G. Qi, H. Wang, M. Haner, C. Weng, S. Chen, and Z. Zhu, "Convolutional neural network based detection and judgement of environmental obstacle in vehicle operation," *CAAI Trans Intell Technol*, vol. 4, no. 2, pp. 80–91, Jun. 2019, doi: 10.1049/TRIT.2018.1045.
- [104] C. El Mechal, N. E. A. El Idrissi, and M. Mesbah, "CNN-Based Obstacle Avoidance Using RGB-Depth Image Fusion," *Lecture Notes in Electrical Engineering*, vol. 745, pp. 867–876, 2022, doi: 10.1007/978-981-33-6893-4_78/TABLES/7.
- [105] P. Chakravarty, K. Kelchtermans, T. Roussel, S. Wellens, T. Tuytelaars, and L. Van Eycken, "CNN-based single image obstacle avoidance on a quadrotor," *Proc IEEE Int Conf Robot Autom*, pp. 6369–6374, Jul. 2017, doi: 10.1109/ICRA.2017.7989752.
- [106] C. Liu, B. Zheng, C. Wang, Y. Zhao, S. Fu, and H. Li, "CNN-Based Vision Model for Obstacle Avoidance of Mobile Robot," *MATEC Web of Conferences*, vol. 139, p. 00007, Dec. 2017, doi: 10.1051/MATECCONF/201713900007.
- [107] H. Y. Lee, H. W. Ho, and Y. Zhou, "Deep Learning-based Monocular Obstacle Avoidance for Unmanned Aerial Vehicle Navigation in Tree Plantations: Faster Region-based Convolutional Neural Network Approach," *Journal of Intelligent and Robotic Systems: Theory and*

- Applications*, vol. 101, no. 1, pp. 1–18, Jan. 2021, doi: 10.1007/S10846-020-01284-Z/METRICS.
- [108] H. Hakim and A. Fadhil, “Survey: Convolution Neural networks in Object Detection,” *J Phys Conf Ser*, vol. 1804, no. 1, Mar. 2021, doi: 10.1088/1742-6596/1804/1/012095.
 - [109] S. Yang, S. Konam, C. Ma, S. Rosenthal, M. Veloso, and S. Scherer, “Obstacle Avoidance through Deep Networks based Intermediate Perception,” *arXiv.org*, 2017.
 - [110] L. Emmi, E. Le Flécher, V. Cadenat, and M. Devy, “A hybrid representation of the environment to improve autonomous navigation of mobile robots in agriculture,” *Precis Agric*, vol. 22, no. 2, pp. 524–549, Apr. 2021, doi: 10.1007/S11119-020-09773-9/FIGURES/12.
 - [111] X. Cao, L. Ren, and C. Sun, “Research on Obstacle Detection and Avoidance of Autonomous Underwater Vehicle Based on Forward-Looking Sonar,” *IEEE Trans Neural Netw Learn Syst*, vol. 34, no. 11, pp. 9198–9208, Nov. 2023, doi: 10.1109/TNNLS.2022.3156907.
 - [112] Y. Chen, M. C. Goorden, F. J. Beekman, and R. Cheng, “A survey: Comparison between Convolutional Neural Network and YOLO in image identification,” *J Phys Conf Ser*, vol. 1453, no. 1, p. 012139, Jan. 2020, doi: 10.1088/1742-6596/1453/1/012139.
 - [113] A. Molina-Leal, A. Gómez-Espinosa, J. A. Escobedo Cabello, E. Cuan-Urquizo, and S. R. Cruz-Ramírez, “Trajectory Planning for a Mobile Robot in a Dynamic Environment Using an LSTM Neural Network,” *Applied Sciences 2021, Vol. 11, Page 10689*, vol. 11, no. 22, p. 10689, Nov. 2021, doi: 10.3390/APP112210689.
 - [114] Y. Chen, J. Liang, Y. Wang, Q. Pan, J. Tan, and J. Mao, “Autonomous mobile robot path planning in unknown dynamic environments using neural dynamics,” *Soft comput*, vol. 24, no. 18, pp. 13979–13995, Sep. 2020, doi: 10.1007/s00500-020-04771-5.

- [115] Y. Li and Y. Liu, "Obstacle Avoidance Algorithm for Mobile Robot Based on ROS and Machine Vision," *Communications in Computer and Information Science*, vol. 1449, pp. 619–632, 2021, doi: 10.1007/978-981-16-5188-5_44.
- [116] Y. Li and Y. Liu, "Vision-based Obstacle Avoidance Algorithm for Mobile Robot," *Proceedings - 2020 Chinese Automation Congress, CAC 2020*, pp. 1273–1278, Nov. 2020, doi: 10.1109/CAC51589.2020.9326906.
- [117] S. Lee, I. Skliarova, T.-V. Dang, and N.-T. Bui, "Obstacle Avoidance Strategy for Mobile Robot Based on Monocular Camera," *Electronics 2023, Vol. 12, Page 1932*, vol. 12, no. 8, p. 1932, Apr. 2023, doi: 10.3390/ELECTRONICS12081932.
- [118] K. Inoue, Y. Kaizu, S. Igarashi, K. Furuhashi, and K. Imou, "Autonomous Navigation and Obstacle Avoidance in an Orchard Using Machine Vision Techniques for a Robotic Mower," *Engineering in Agriculture, Environment and Food*, vol. 15, no. 4, pp. 87–99, 2022, doi: 10.37221/EAEF.15.4_87.
- [119] F. He, L. Zhang, F. He, and L. Zhang, "Design of Indoor Security Robot based on Robot Operating System," *Journal of Computer and Communications*, vol. 11, no. 5, pp. 93–107, May 2023, doi: 10.4236/JCC.2023.115008.
- [120] D. Henke, D. Reisa, D. Welferb, M. Antonio, D. Souza, L. Cuadrosc and D. Fernando, T. Gamarra "Mobile Robot Navigation Using an Object Recognition Software with RGBD Images and the YOLO Algorithm," *Applied Artificial Intelligence*, vol. 33, pp. 1290–1305, 2019, doi: 10.1080/08839514.2019.1684778.
- [121] K. J. Singh, D. S. Kapoor, K. Thakur, A. Sharma, and X. Z. Gao, "Computer-Vision Based Object Detection and Recognition for Service Robot in Indoor Environment," *Computers, Materials & Continua*, vol. 72, no. 1, pp. 197–213, Feb. 2022, doi: 10.32604/CMC.2022.022989.

- [122] Z. Machkour, D. Ortiz-Arroyo, and P. Durdevic, "Monocular Based Navigation System for Autonomous Ground Robots Using Multiple Deep Learning Models," *International Journal of Computational Intelligence Systems*, vol. 16, no. 1, pp. 1–18, Dec. 2023, doi: 10.1007/S44196-023-00250-5/FIGURES/12.
- [123] I. Hasan And T. Kara, "Design, Construction And Control Of An Autonomous Mobile Rescue Robot With Visual Feedback," *European Journal Of Science And Technology*, No. 37, Pp. 65–71, Jul. 2022, Doi: 10.31590/Ejosat.1133067.
- [124] I. Sung, B. Choi and P. Nielsen, "On the training of a neural network for online path planning with offline path planning algorithms", vol. 57, April 2021, doi:10.16/j.ijinfomgt.2020.102142.
- [125] C. Liu, B. Zheng, C. Wang, Y. Zhao, S. Fu, and H. Li, "CNN-Based Vision Model for Obstacle Avoidance of Mobile Robot", doi: 10.1051/mateconf/201713900007.
- [126] B. Ichter, J. Harrison, and M. Pavone, "Learning Sampling Distributions for Robot Motion Planning," Sep. 2017, [Online]. Available: <http://arxiv.org/abs/1709.05448>
- [127] Ren and Hongfei, "Path Avoidance System of Intelligent Robot Based on Computer Vision," *JPhCS*, vol. 2493, no. 1, p. 012016, 2023, doi: 10.1088/1742-6596/2493/1/012016.
- [128] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real Deep Reinforcement Learning: Continuous Control of Mobile Robots for Mapless Navigation," Mar. 2017, [Online]. Available: <http://arxiv.org/abs/1703.00420>
- [129] A. Chtourou, P. Merdrignac, and O. Shagdar, "Collective Perception service for Connected Vehicles and Roadside Infrastructure," *IEEE Vehicular Technology Conference*, vol. 2021-April, Apr. 2021, doi: 10.1109/VTC2021-SPRING51267.2021.9448753.

- [130] B. Khalid, A. M. Khan, M. U. Akram, and S. Batool, "Person Detection by Fusion of Visible and Thermal Images Using Convolutional Neural Network," *2019 2nd International Conference on Communication, Computing and Digital Systems, C-CODE 2019*, pp. 143–148, Apr. 2019, doi: 10.1109/C-CODE.2019.8680991.
- [131] H. Li, X.-J. Wu, and J. Kittler, "Infrared and Visible Image Fusion using a Deep Learning Framework," *Proceedings - International Conference on Pattern Recognition*, vol. 2018-August, pp. 2705–2710, Apr. 2018, doi: 10.1109/ICPR.2018.8546006.
- [132] S. Rajkumar and P. V. S. S. R. C. Mouli, "Infrared and Visible Image Fusion Using Entropy and Neuro-Fuzzy Concepts," *Advances in Intelligent Systems and Computing*, vol. 248 VOLUME I, pp. 93–100, 2014, doi: 10.1007/978-3-319-03107-1_11/COVER.
- [133] J. Saeedi and K. Faez, "Infrared and visible image fusion using fuzzy logic and population-based optimization," *Appl Soft Comput*, vol. 12, no. 3, pp. 1041–1054, Mar. 2012, doi: 10.1016/J.ASOC.2011.11.020.
- [134] G. Bogacsovics, J. Toth, A. Hajdu, and B. Harangi, "Enhancing CNNs through the use of hand-crafted features in automated fundus image classification," *Biomed Signal Process Control*, vol. 76, p. 103685, Jul. 2022, doi: 10.1016/J.BSPC.2022.103685.
- [135] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "THIS PAPER HAS BEEN ACCEPTED BY IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS FOR PUBLICATION 1 Object Detection with Deep Learning: A Review".
- [136] J. Ma, W. Yu, P. Liang, C. Li, and J. Jiang, "FusionGAN: A generative adversarial network for infrared and visible image fusion," *Information Fusion*, vol. 48, pp. 11–26, Aug. 2019, doi: 10.1016/J.INFFUS.2018.09.004.

- [137] H. Li, X.-J. Wu, and T. Durrani, "NestFuse: An Infrared and Visible Image Fusion Architecture based on Nest Connection and Spatial/Channel Attention Models," *IEEE Trans Instrum Meas*, vol. 69, no. 12, pp. 9645–9656, Jul. 2020, doi: 10.1109/TIM.2020.3005230.
- [138] H. Li and X.-J. Wu, "DenseFuse: A Fusion Approach to Infrared and Visible Images," *IEEE Transactions on Image Processing*, vol. 28, no. 5, pp. 2614–2623, Apr. 2018, doi: 10.1109/TIP.2018.2887342.
- [139] S. M. Pan and D. H. Madill, "Generalized sliding window algorithm with applications to frame synchronization," *Proceedings - IEEE Military Communications Conference MILCOM*, vol. 3, pp. 796–800, 1996, doi: 10.1109/MILCOM.1996.571384.
- [140] S. Huang, C. A. I. Nianguang, P. Penzuti Pacheco, S. Narandes, Y. Wang, and X. U. Wayne, "Applications of Support Vector Machine (SVM) Learning in Cancer Genomics," *Cancer Genomics Proteomics*, vol. 15, no. 1, pp. 41–51, Jan. 2018, doi: 10.21873/CGP.20063.
- [141] N. S. Hashemi, R. B. Aghdam, A. S. Ghiasi, and P. Fatemi "Template Matching Advances and Applications in Image Analysis". Oct. 2016
- [142] J. Wang, L. Song, Z. Li, H. Sun, J. Sun, and N. Zheng, "End-to-End Object Detection with Fully Convolutional Network," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 15844–15853, Dec. 2020, doi: 10.1109/CVPR46437.2021.01559.
- [143] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *Adv Neural Inf Process Syst*, vol. 28, 2015, Accessed: Mar. 13, 2023. [Online]. Available: <https://github.com/>
- [144] X. Chen and A. Gupta, "An Implementation of Faster RCNN with Study for Region Sampling," Feb. 2017, Accessed: Mar. 20, 2023.

- [145] A. Benjumea, I. Teeti, F. Cuzzolin, and A. Bradley, "YOLO-Z: Improving small object detection in YOLOv5 for autonomous vehicles".
- [146] S. Li, Y. Li, Y. Li, M. Li, and X. Xu, "YOLO-FIRI: Improved YOLOv5 for Infrared Image Object Detection," *IEEE Access*, vol. 9, pp. 141861–141875, 2021, doi: 10.1109/ACCESS.2021.3120870.
- [147] Y. Chen, H. Xie, and H. Shin, "Multi-layer fusion techniques using a CNN for multispectral pedestrian detection," *IET Computer Vision*, vol. 12, no. 8, pp. 1179–1187, Dec. 2018, doi: 10.1049/IET-CVI.2018.5315.
- [148] Y. Hou, Y. Song, X. Hao, Y. Shen, and M. Qian, "Multispectral Pedestrian Detection using Deep Fusion Convolutional Neural Networks." Oct. 2017, doi: 10.1109/icspcc.2017.8242507
- [149] M. O. Gani, S. Kuiry, A. Das, M. Nasipuri, and N. Das, "Multispectral Object Detection with Deep Learning," *Communications in Computer and Information Science*, vol. 1406 CCIS, pp. 105–117, 2021, doi: 10.1007/978-3-030-75529-4_9.
- [150] K. Roszyk, M. R. Nowicki, and P. Skrzypczyński, "Adopting the YOLOv4 Architecture for Low-Latency Multispectral Pedestrian Detection in Autonomous Driving," *Sensors 2022, Vol. 22, Page 1082*, vol. 22, no. 3, p. 1082, Jan. 2022, doi: 10.3390/S22031082.
- [151] D. Guan, Y. Cao, J. Yang, Y. Cao, and M. Y. Yang, "Fusion of Multispectral Data Through Illumination-aware Deep Neural Networks for Pedestrian Detection," *Information Fusion*, vol. 50, pp. 148–157, Feb. 2018, doi: 10.1016/j.inffus.2018.11.017.
- [152] C. Li, D. Song, R. Tong, and M. Tang, "Illumination-aware faster R-CNN for robust multispectral pedestrian detection," *Pattern Recognit*, vol. 85, pp. 161–171, Jan. 2019, doi: 10.1016/J.PATCOG.2018.08.005.
- [153] J. Ma, Y. Ma, and C. Li, "Infrared and visible image fusion methods and applications: A survey," *Information Fusion*, vol. 45, pp. 153–178, Jan. 2019, doi: 10.1016/J.INFFUS.2018.02.004.

- [154] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," Apr. 2020, Accessed: Apr. 02, 2023. [Online]. Available: <https://arxiv.org/abs/2004.10934v1>
- [155] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path Aggregation Network for Instance Segmentation," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 8759–8768, Mar. 2018, doi: 10.1109/CVPR.2018.00913.
- [156] J. Yao, J. Qi, J. Zhang, H. Shao, J. Yang, and X. Li, "A Real-Time Detection Algorithm for Kiwifruit Defects Based on YOLOv5," *Electronics* 2021, Vol. 10, Page 1711, vol. 10, no. 14, p. 1711, Jul. 2021, doi: 10.3390/ELECTRONICS10141711.
- [157] N. Bjorck, C. P. Gomes, B. Selman, and K. Q. Weinberger, "Understanding Batch Normalization," *Adv Neural Inf Process Syst*, vol. 31, 2018.
- [158] A. M. Fred Agarap, "Deep Learning using Rectified Linear Units (ReLU)," Mar. 2018, Accessed: Apr. 03, 2023. [Online]. Available: <https://arxiv.org/abs/1803.08375v2>
- [159] I. Shopovska, L. Jovanov, and W. Philips, "Deep visible and thermal image fusion for enhanced pedestrian visibility," *Sensors (Switzerland)*, vol. 19, no. 17, Sep. 2019, doi: 10.3390/S19173727.
- [160] J. M. Rivera Velázquez *et al.*, "Analysis of Thermal Imaging Performance under Extreme Foggy Conditions: Applications to Autonomous Driving," *J Imaging*, vol. 8, no. 11, Nov. 2022, doi: 10.3390/JIMAGING8110306.
- [161] "Roboflow: Give your software the power to see objects in images and video." Accessed: Mar. 26, 2024. [Online]. Available: <https://roboflow.com/>
- [162] Y. Wang, S. Zeng, X. Wang and J. Wang, "Super-Resolution multi-focus image fusion based on convolutional neural network," *JPhCS*, vol. 1885, no. 2, p. 022011, Apr. 2021, doi: 10.1088/1742-6596/1885/2/022011.

- [163] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional Block Attention Module." pp. 3–19, 2018.
- [164] L. Miao, N. Li, M. Zhou, H. Zhou Lize Miao, and H. Zhou, "CBAM-Yolov5: improved Yolov5 based on attention model for infrared ship detection," <https://doi.org/10.1117/12.2631130>, vol. 12168, pp. 564–571, Mar. 2022, doi: 10.1117/12.2631130.
- [165] M. Lv and W. H. Su, "YOLOV5-CBAM-C3TR: an optimized model based on transformer module and attention mechanism for apple leaf disease detection," *Front Plant Sci*, vol. 14, p. 1323301, Jan. 2023, doi: 10.3389/FPLS.2023.1323301/BIBTEX.
- [166] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick and Piotr Dollár., "Microsoft COCO: Common Objects in Context," *Proceedings - International Conference on Image Processing, ICIP*, pp. 3076–3080, 2014, doi: 10.1109/ICIP46576.2022.9897741.
- [167] P. Kumar, J. A.Thomasson, S. W. Searcy, R. G. Hardin, U. Braga-Neto, S. C. Popescu, D. E. Martin, R. Rodriguez, K. Meza, J. Enciso, J. S. Diaz and T.i Wang "Assessing The Performance of YOLOv5 Algorithm For Detecting Volunteer Cotton Plants in Corn Fields at Three Different Growth Stages", Jul 2022, <https://doi.org/10.48550/arXiv.2208.00519>.
- [168] P. Henderson and V. Ferrari, "End-to-end training of object class detectors for mean average precision," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10115 LNCS, pp. 198–213, Jul. 2016, doi: 10.1007/978-3-319-54193-8_13.

- [169] F. Rubio, F. Valero, and C. Llopis-Albert, "A review of mobile robots: Concepts, methods, theoretical framework, and applications", doi: 10.1177/1729881419839596.
- [170] K. El Madawi, H. Rashed, A. El Sallab, O. Nasr, H. Kamel, and S. Yogamani, "RGB and LiDAR fusion based 3D Semantic Segmentation for Autonomous Driving".
- [171] "Robot Operating Systems (ROS): The Fundamentals of ROS and Its Remarkable." Accessed: Aug. 23, 2023. [Online]. Available: <https://www.ijraset.com/research-paper/robot-operating-systems-the-fundamentals-of-ros-and-its-remarkable>
- [172] G. Chen, H. Yang, and I. M. Mitchell, "ROS-X-Habitat: Bridging the ROS Ecosystem with Embodied AI," *Proceedings - 2022 19th Conference on Robots and Vision, CRV 2022*, pp. 24–31, Sep. 2021, doi: 10.1109/CRV55824.2022.00012.
- [173] K. Zheng September, "ROS Navigation Tuning Guide".
- [174] P. Marin-Plaza, A. Hussein, D. Martin, and A. De La Escalera, "Research Article Global and Local Path Planning Study in a ROS-Based Research," 2018, doi: 10.1155/2018/6392697.
- [175] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," *SN Comput Sci*, vol. 2, no. 6, Nov. 2021, doi: 10.1007/S42979-021-00815-1.
- [176] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "THIS PAPER HAS BEEN ACCEPTED BY IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS FOR PUBLICATION 1 Object Detection with Deep Learning: A Review".
- [177] Z. Xu, X. Zhan, B. Chen, Y. Xiu, C. Yang, and K. Shimada, "A real-time dynamic obstacle tracking and mapping system for UAV navigation and collision avoidance with an RGB-D camera," Sep. 2022, Accessed: Aug. 23, 2023. [Online]. Available: <https://arxiv.org/abs/2209.08258v2>

- [178] “Review: Luxonis Oak-D AI Camera with Computer Vision and Image Depth - Make:” Accessed: Aug. 24, 2023. [Online]. Available: <https://makezine.com/article/maker-news/review-luxonis-oak-d-ai-camera-with-computer-vision-and-image-depth/>
- [179] H. Fujiyoshi, T. Hirakawa, and T. Yamashita, “Deep learning-based image recognition for autonomous driving,” 2019, doi: 10.1016/j.iatssr.2019.11.008.
- [180] R. Sekhar, R. Dheekonda, S. K. Panda, N. Khan, M. Al-Hasan, and S. Anwar, “Object Detection from a Vehicle using Deep Learning Network and Future Integration with Multi-Sensor Fusion Algorithm,” 2017, doi: 10.4271/2017-01-0117.
- [181] D. Henke *et al.*, “Mobile Robot Navigation Using an Object Recognition Software with RGBD Images and the YOLO Algorithm,” *Applied Artificial Intelligence*, vol. 33, pp. 1290–1305, 2019, doi: 10.1080/08839514.2019.1684778.
- [182] Mohd Shah H, ‘Vision Based Obstacle Avoidance for Mobile Robot Using Optical Flow Process’,2019
- [183] H. W. Luo, C. S. Zhang, F. C. Pan, and X. M. Ju, “Contextual-YOLOV3: Implement better small object detection based deep learning,” *Proceedings - 2019 International Conference on Machine Learning, Big Data and Business Intelligence, MLBDI 2019*, pp. 134–141, Nov. 2019, doi: 10.1109/MLBDI48998.2019.00032.
- [184] D. Aghi, V. Mazzia, and M. Chiaberge, “machines Local Motion Planner for Autonomous Navigation in Vineyards with a RGB-D Camera-Based Algorithm and Deep Learning Synergy”, doi: 10.3390/machines8020027.
- [185] T. J. Lee, D. H. Yi, and D. Il Dan Cho, “A monocular vision sensor-based obstacle detection algorithm for autonomous robots,” *Sensors (Switzerland)*, vol. 16, no. 3, Mar. 2016, doi: 10.3390/S16030311.

- [186] M. Štancel, “An Introduction to Image Classification and Object Detection using YOLO Detector”.
- [187] J. Zhang, Z. Ti, X. Lei, R. Arifando, S. Eto, and C. Wada, “Improved YOLOv5-Based Lightweight Object Detection Algorithm for People with Visual Impairment to Detect Buses,” *Applied Sciences* 2023, Vol. 13, Page 5802, vol. 13, no. 9, p. 5802, May 2023, doi: 10.3390/APP13095802.
- [188] Z. Li, B. Xu, D. Wu, K. Zhao, S. Chen, M. Lu and J. Cong, “A YOLO-GGCNN based grasping framework for mobile robots in unknown environments,” *Expert Syst Appl*, vol. 225, p. 119993, Sep. 2023, doi: 10.1016/J.ESWA.2023.119993.
- [189] M. Kobayashi, “Local Path Planning: Dynamic Window Approach With Virtual Manipulators Considering Dynamic Obstacles”, doi: 10.1109/ACCESS.2022.3150036.
- [190] B. Kovács, G. Szayer, F. Tajti, M. Burdelis, and P. Korondi, “A novel potential field method for path planning of mobile robots by adapting animal motion attributes,” *Rob Auton Syst*, vol. 82, pp. 24–34, Aug. 2016, doi: 10.1016/J.ROBOT.2016.04.007.
- [191] M. Szemenyei and V. Estivill-Castro, “Fully neural object detection solutions for robot soccer,” *Neural Comput Appl*, vol. 34, no. 24, pp. 21419–21432, Dec. 2022, doi: 10.1007/S00521-021-05972-1/TABLES/7.
- [192] M. Szemenyei and V. Estivill-Castro, “ROBO: Robust, Fully Neural Object Detection for Robot Soccer”.
- [193] Y. Yazid *et al.*, “Deep Learning Empowered Robot Vision for Efficient Robotic Grasp Detection and Defect Elimination in Industry 4.0.,” *Engineering Proceedings*, vol. 58, no. 1, Nov. 2023, doi: 10.3390/ECSA-10-16079.
- [194] Y. Y. Liao and K. Ryu, “Status Recognition Using Pre-Trained YOLOv5 for Sustainable Human-Robot Collaboration (HRC) System in Mold

- Assembly,” *Sustainability* 2021, Vol. 13, Page 12044, vol. 13, no. 21, p. 12044, Oct. 2021, doi: 10.3390/SU132112044.
- [195] M. Quigley et al., “ROS: an open-source Robot Operating System”, Accessed: Aug. 23, 2023. [Online]. Available: <http://stair.stanford.edu>
- [196] D. A. De Lima and G. A. S. Pereira, “Navigation of an autonomous car using vector fields and the dynamic window approach,” *Journal of Control, Automation and Electrical Systems*, vol. 24, no. 1–2, pp. 106–116, Apr. 2013, doi: 10.1007/s40313-013-0006-5.
- [197] H. Yang and X. Teng, “Mobile Robot Path Planning Based on Enhanced Dynamic Window Approach and Improved A * Algorithm,” 2022, doi: 10.1155/2022/2183229.
- [198] N. Adiuku, N. P. Avdelidis, G. Tang, and A. Plastropoulos, “Improved Hybrid Model for Obstacle Detection and Avoidance in Robot Operating System Framework (Rapidly Exploring Random Tree and Dynamic Windows Approach),” *Sensors* 2024, Vol. 24, Page 2262, vol. 24, no. 7, p. 2262, Apr. 2024, doi: 10.3390/S24072262.
- [199] Yu, L.; Yang E.; Ren P.; Luo C.; Dobie G.; Gu D.; Yan X., ‘Inspection robots in oil and gas industry: A review of current solutions and future trends’, *ICAC 2019 - 2019 25th IEEE International Conference on Automation and Computing*, 2019, doi: 10.23919/ICONAC.2019.8895089.
- [200] J. Wu, X. Ma, T. Peng, and H. Wang, “An Improved Timed Elastic Band (TEB) Algorithm of Autonomous Ground Vehicle (AGV) in Complex Environment,” *Sensors* 2021, Vol. 21, Page 8312, vol. 21, no. 24, p. 8312, Dec. 2021, doi: 10.3390/S21248312.
- [201] Xiquan M.; Di L.; Jian O.; Yongchao L.; ‘An improved dynamic window approach for local trajectory planning in the environment with dense objects’, *J. Phys.: Conf. Ser.* **1884** 012003, 2021, doi: 10.1088/1742-6596/1884/1/012003.

- [202] Li X., 'Path planning of intelligent mobile robot based on Dijkstra algorithm', *J Phys Conf Ser*, vol. 2083, no. 4, p. 042034, Nov. 2021, doi: 10.1088/1742-6596/2083/4/042034.
- [203] Valdes R. A.; and Comendador V. F. A., 'Aviation 4.0: More Safety through Automation and Digitization', *Aircraft Technology*, Mar. 2018, doi: 10.5772/INTECHOPEN.73688.
- [204] A. Loganathan and N. S. Ahmad, "A systematic review on recent advances in autonomous mobile robot navigation," *Engineering Science and Technology, an International Journal*, vol. 40, p. 101343, Apr. 2023, doi: 10.1016/J.JESTCH.2023.101343.
- [205] S. Das and S. K. Mishra, "A Machine Learning approach for collision avoidance and path planning of mobile robot under dense and cluttered environments," *Computers and Electrical Engineering*, vol. 103, p. 108376, Oct. 2022, doi: 10.1016/J.COMPELECENG.2022.108376.
- [206] M. Li, Y. Hu, N. Zhao, and Q. Qian, "One-Stage Multi-Sensor Data Fusion Convolutional Neural Network for 3D Object Detection," *Sensors 2019, Vol. 19, Page 1434*, vol. 19, no. 6, p. 1434, Mar. 2019, doi: 10.3390/S19061434.
- [207] J. Wang, W. Chi, C. Li, C. Wang, and M. Q. H. Meng, "Neural RRT*: Learning-Based Optimal Path Planning," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 4, pp. 1748–1758, Oct. 2020, doi: 10.1109/TASE.2020.2976560.
- [208] H.-T. L. Chiang, J. Hsu, M. Fiser, L. Tapia, and A. Faust, "RL-RRT: Kinodynamic Motion Planning via Learning Reachability Estimators from RL Policies," Jul. 2019, [Online]. Available: <http://arxiv.org/abs/1907.04799>
- [209] E. Reyes, C. Gómez, E. Norambuena, and J. Ruiz-Del-Solar, "Near Real-Time Object Recognition for Pepper based on Deep Neural Networks Running on a Backpack".
- [210] K. Inoue, Y. Kaizu, S. Igarashi, and K. Imou, "The development of autonomous navigation and obstacle avoidance for a robotic mower using

- machine vision technique,” *IFAC-PapersOnLine*, vol. 52, no. 30, pp. 173–177, Jan. 2019, doi: 10.1016/J.IFACOL.2019.12.517.
- [211] R. Strudel Inria, R. Garcia Inria, J. Carpentier Inria, J.-P. Laumond Inria, I. Laptev Inria, and C. Schmid Inria, “Learning Obstacle Representations for Neural Motion Planning”.
- [212] T. Jurgenson and A. Tamar, “Harnessing Reinforcement Learning for Neural Motion Planning”, Accessed: Jan. 01, 2024. [Online]. Available: <https://youtu.be/wHQ4Y4mBRb8>.
- [213] Haoning Z.; Chaoqun W.; Rui G.; Xuewen R.; Jiamin G.; Qixun Y.; Lecheng Y.; Yuliang Z.; Yibin L.; ‘Autonomous live working robot navigation with real-time detection and motion planning system on distribution line’, *High Voltage*, vol. 7, no. 6, p. 1204–1216, 2022, doi: 10.1049/HVE2.12221.
- [214] S. Syntakas, K. Vlachos, and A. Likas, “Object Detection and Navigation of a Mobile Robot by Fusing Laser and Camera Information,” *2022 30th Mediterranean Conference on Control and Automation, MED 2022*, pp. 557–563, 2022, doi: 10.1109/MED54222.2022.9837249.
- [214] M. Hussain, “YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection,” *Machines* 2023, Vol. 11, Page 677, vol. 11, no. 7, p. 677, Jun. 2023, doi: 10.3390/MACHINES11070677.
- [216] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors”.
- [217] O. E. Olorunshola, M. E. Irhebhude, and A. E. Evwiekpaefe, “A Comparative Study of YOLOv5 and YOLOv7 Object Detection Algorithms,” *Journal of Computing and Social Informatics*, vol. 2, no. 1, pp. 1–12, Feb. 2023, doi: 10.33736/JCSI.5070.2023.
- [218] P. Marin-Plaza, A. Hussein, D. Martin, and A. De La Escalera, “Global and Local Path Planning Study in a ROS-Based Research Platform for

- Autonomous Vehicles,” *J Adv Transp*, vol. 2018, 2018, doi: 10.1155/2018/6392697.
- [219] Farzad K.; Amir S.i; Royal A.; Murat U. G.; Hasan B.; M. Ahmed S.; ‘Adapted-RRT: novel hybrid method to solve three-dimensional path planning problem using sampling and metaheuristic-based algorithms’, *Neural Comput Appl*, vol. 33, doi: 10.1007/s00521-021-06179-0.
- [220] L. Wang, X. Yang, Z. Chen, and B. Wang, “Application of the Improved Rapidly Exploring Random Tree Algorithm to an Insect-like Mobile Robot in a Narrow Environment,” *Biomimetics 2023, Vol. 8, Page 374*, vol. 8, no. 4, p. 374, Aug. 2023, doi: 10.3390/BIOMIMETICS8040374.
- [221] K. Zheng, “ROS Navigation Tuning Guide,” *Studies in Computational Intelligence*, vol. 962, pp. 197–226, 2021, doi: 10.1007/978-3-030-75472-3_6/COVER.
- [222] G. A. Kumar, J. H. Lee, J. Hwang, J. Park, S. H. Youn, and S. Kwon, “LiDAR and Camera Fusion Approach for Object Distance Estimation in Self-Driving Vehicles,” *Symmetry 2020, Vol. 12, Page 324*, vol. 12, no. 2, p. 324, Feb. 2020, doi: 10.3390/SYM12020324.
- [223] S. Royo and M. Ballesta-Garcia, “An Overview of Lidar Imaging Systems for Autonomous Vehicles,” *Applied Sciences 2019, Vol. 9, Page 4093*, vol. 9, no. 19, p. 4093, Sep. 2019, doi: 10.3390/APP9194093.
- [224] C. S. Chen, C. J. Lin, C. C. Lai, and S. Y. Lin, “Velocity Estimation and Cost Map Generation for Dynamic Obstacle Avoidance of ROS Based AMR,” *Machines 2022, Vol. 10, Page 501*, vol. 10, no. 7, p. 501, Jun. 2022, doi: 10.3390/MACHINES10070501.
- [225] N. Adiuku, N. P. Avdelidis, G. Tang, A. Plastropoulos, and Y. Diallo, “Mobile Robot Obstacle Detection and Avoidance with NAV-YOLO,” *International Journal of Mechanical Engineering and Robotics Research*, vol. 13, no. 2, pp. 219–226, 2024, doi: 10.18178/IJMERR.13.2.219-226.

- [226] Y. Xia, M. Nguyen, and W. Q. Yan, "A Real-time Kiwifruit Detection Based on Improved YOLOv7".
- [227] S. K. Pathi, A. Kristoffersson, and D. Repsilber, "A Novel Method for Estimating Distances from a Robot to Humans Using Egocentric RGB Camera," *Sensors (Basel)*, vol. 19, no. 14, Jul. 2019, doi: 10.3390/S19143142.
- [228] J. King and M. Likhachev, "Efficient Cost Computation in Cost Map Planning for Non-Circular Robots".
- [229] R. S. Pol, V. N. Aher, S. V. Gaikwad, D. G. Bhalke, A. Y. Borkar, and M. T. Kolte, "Autonomous Differential Drive Mobile Robot Navigation with SLAM, AMCL using ROS," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 5s, pp. 46–53, Nov. 2023, Accessed: Jan. 25, 2024. [Online]. Available: <https://ijisae.org/index.php/IJISAE/article/view/3820>.

