

CRANFIELD UNIVERSITY

JIAN LI

ENABLING CONTEXT-AWARE COLLISION
AVOIDANCE FOR AUTONOMOUS RAILWAY
MAINTENANCE VEHICLE

FACULTY OF ENGINEERING AND APPLIED SCIENCES
PhD in Manufacturing

PhD
Academic Year: 2019–2024

Supervisors: Dr. Isidro Durazo Cardenas
Dr. Cristobal Ruiz Carcel
Prof. Andrew Starr
December 2024

CRANFIELD UNIVERSITY

FACULTY OF ENGINEERING AND APPLIED SCIENCES
PhD in Manufacturing

PhD

Academic Year: 2019–2024

JIAN LI

Enabling Context-aware Collision Avoidance for
Autonomous Railway Maintenance Vehicle

Supervisors: Dr. Isidro Durazo Cardenas
Dr. Cristobal Ruiz Carcel
Prof. Andrew Starr
December 2024

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

Abstract

Compared to the recent advancements in self-driving cars, There is a significant gap for autonomous railway vehicles, which still rely on a basic “start-stop” approach without being aware of their surrounding environment. This research developed a context-aware collision avoidance system for autonomous railway vehicles. It addressed the gap by enabling the vehicle to make appropriate decisions by distinguishing obstacles’ types and locations. A key challenge was obtaining an annotated dataset for railway object detection, alongside balancing detection accuracy with precise distance estimation using cameras and LiDAR. The model utilised Transfer Learning by pre-training models on large datasets like COCO and RailSem19. It then leveraged this knowledge by fine-tuning it with a custom dataset tailored to railway maintenance scenarios. A sensor fusion approach was implemented to combine the strengths of both sensors. This multimodal approach extended 2D object tracking results into 3D by aligning the camera and LiDAR’s coordinate systems. The collision avoidance system was integrated into a robotic framework built on ROS2, which supported real-time processing and communication between each sensor. The validation experiments included both controlled indoor and outdoor scenarios. Indoor validation evaluated the system’s performance under various lighting conditions; outdoor validation was conducted on a test track to ensure a high Technology Readiness Level. The indoor validation identified the optimal operating range for illumination and provided guidelines for handling extreme conditions, such as tunnels or night-time environments. The outdoor validation achieved a high success rate of 80% by making correct actuation decisions based on the detected obstacles’ types and positions.

Keywords

3D object tracking; Autonomous railway maintenance vehicle; Context-aware collision avoidance; Machine learning; Multi-sensor fusion; Railway collision avoidance system; ROS2 robotic framework; Transfer learning

Contents

Abstract	iii
Contents	v
List of Figures	ix
List of Tables	xii
List of Abbreviations	xiv
Acknowledgements	xviii
1 Introduction	1
1.1 Background and motivations	1
1.1.1 Evolution of task automation in railway maintenance	2
1.1.2 From manual to autonomous: collision avoidance in railways . . .	4
1.1.3 Limitations of existing railway collision avoidance solutions . . .	6
1.1.4 Motivations of this study	7
1.2 Research scope, aim and objectives	8
1.2.1 Research scope	8
1.2.2 Research aim and objectives	9
1.3 Research gaps and novelties	10
1.3.1 Research gaps	10
1.3.2 Research novelties	12
1.4 Contributions to knowledge	14
1.4.1 Created a specialised dataset for railway maintenance scenarios .	14
1.4.2 Enhanced railway object detection with transfer learning	15
1.4.3 Developed a multimodal 3D object tracking system	16
1.4.4 Implementation and validation of collision avoidance system . . .	17
1.4.5 Broader implications	19
1.5 Thesis layout	19
2 Literature Review	21
2.1 Introduction	21
2.2 Collision avoidance in other fields	24
2.2.1 Common challenges in collision avoidance systems	25
2.3 Railway scenarios analysis	26

2.3.1	Railway track possession	26
2.3.2	Railway system components	28
2.3.3	Railway scenarios study	32
2.3.4	Summary and domain-specific considerations	34
2.4	Learning-based object detection	36
2.4.1	Introduction	36
2.4.2	Two-stage (region-based) detection algorithms	38
2.4.3	Conclusion on two-stage detection algorithms	45
2.4.4	One-stage (unified) detection algorithms	47
2.4.5	Conclusion on one-stage detection methods	52
2.4.6	Discussion on learning-based object detection	54
2.5	Object detection with transfer learning	59
2.5.1	Introduction	59
2.5.2	Transfer learning problem classification	60
2.5.3	Transfer learning approach classification	63
2.5.4	Negative transfer	70
2.5.5	Discussion on transfer learning	72
2.6	Object tracking	75
2.6.1	Introduction	75
2.6.2	Tracking challenges	77
2.6.3	Tracking framework	82
2.6.4	Appearance model	83
2.6.5	Motion model	87
2.6.6	Search strategy	89
2.6.7	Tracking methods	95
2.6.8	Discussion on object tracking	97
2.7	Depth estimation for tracked objects	104
2.7.1	Monocular vision-based methods	105
2.7.2	Stereo vision-based methods	106
2.7.3	LiDAR/radar-based methods	107
2.7.4	Machine learning methods	109
2.7.5	Discussion on depth estimation	110
3	Methodology	113
3.1	Introduction	113
3.2	Robotic framework design and system integration	115
3.2.1	Overview of ROS2	117
3.2.2	System architecture	118
3.2.3	Hardware integration and vehicle actuation	119
3.2.4	Sensor list	120
3.2.5	Evaluation of sensor selection	121
3.2.6	Collision avoidance architecture overview	122
3.2.7	Sensor reliability and safety mechanisms	123
3.2.8	Modularity and scalability	124
3.3	Railway object detection	126
3.3.1	Detection framework selections	126

3.3.2	Problem definition	127
3.3.3	Experiment overview	130
3.4	3D object tracking	132
3.4.1	Methodological approach overview	133
3.4.2	Validation plan	135
4	Object Detection with Transfer Learning	137
4.1	Introduction	137
4.1.1	Experiment overview	137
4.1.2	Objective and hypotheses	139
4.1.3	Structure of the chapter	140
4.2	Dataset preparation	140
4.2.1	Overview of dataset used in this experiment	140
4.2.2	Custom dataset construction	142
4.2.3	RailSem19 dataset modification	149
4.3	Experiment setup	153
4.3.1	Experiment design	153
4.3.2	Hardware and software specifications	157
4.3.3	Evaluation metrics	159
4.4	Results and discussion	160
4.4.1	Introduction	160
4.4.2	Baseline experiment results	161
4.4.3	Analysis between one-stage & two-stage pre-training	163
4.4.4	Analysis on impact of layer freezing techniques	167
4.4.5	Analysis on the impact of different source datasets	173
4.4.6	Analysis on the impact of RailSem19 variations	176
4.4.7	Key findings and discussions	190
4.4.8	Future work	196
4.5	Chapter conclusion	196
4.5.1	Key contributions	196
4.5.2	Summary of key findings	197
4.5.3	Research impact	198
5	3D Object Tracking and Collision Avoidance	199
5.1	Introduction	199
5.2	Methodology	201
5.2.1	Methodology approach	201
5.2.2	Camera-LiDAR calibration	202
5.2.3	Camera to LiDAR transformation	209
5.2.4	3D frustums construction for object filtering	213
5.2.5	Distance measurement	215
5.3	Experiment design	216
5.3.1	Experiment setup	217
5.3.2	System robustness validation	218
5.3.3	Collision avoidance system validation	223
5.4	Results and discussion	224
5.4.1	Introduction	224

5.4.2	Discussion on robustness validation results	225
5.4.3	Discussion on collision avoidance validation results	231
5.4.4	Future works	234
5.5	Chapter conclusion	235
6	Conclusion	237
6.1	Research overview	237
6.1.1	Research gaps, objectives and novelties	238
6.2	Summary of contributions	240
6.2.1	Railway object detection with transfer learning	240
6.2.2	Multi-modal 3D object tracking	243
6.2.3	Collision avoidance system and real-world validation	247
6.3	Research impact	248
6.4	Challenges and limitations	249
6.5	Edge cases and ethical considerations	250
6.6	Future works	252
	References	254
A	Appendix	288
A.1	RailSem19 variation results	289
A.1.1	RailSem19 instance table	289
A.1.2	Results: one-stage pre-trained with RailSem19	290
A.1.3	Results: two-stage pre-trained with RailSem19	292
A.1.4	Best performing dataset for each class	294

List of Figures

1.1	Autonomous railway maintenance systems.	3
1.2	Railway Inspection and Maintenance Vehicle (RIMV) is a lightweight, adaptable, autonomous platform designed for railway maintenance tasks. Its modular design allows for integration with various equipment to suit different maintenance requirements. This image showcases the vehicle equipped with ultrasonic testing technology for rail defect detection on an active railway track under possession.	14
1.3	Flowchart for camera LiDAR fusion	17
1.4	Indoor robustness validation with on-vehicle lighting	18
1.5	Outdoor collision avoidance validation with worker and trespasser	18
2.1	Illustration of railway possession [1]	27
2.2	Railway system components [2]	29
2.3	Exemplary railway scenarios, several tracks and different connection types, such as single turnouts and a diamond crossing, are shown. Each track is built up of two parallel rails with a fixed distance [3].	31
2.4	Different learning processes between (a) traditional machine learning and (b) Transfer Learning (TL)	60
2.5	Categorisation of TL, adapted from [4].	61
2.6	Illustration of negative transfer. The two oval dashed lines represent the distribution shape in the source and target domains. TL facilitates the training of a fine-tuned classifier in the target domain even with small samples, but distribution discrepancy may cause negative transfer, leading to poor generalisation in target tasks [5].	71
2.7	Illustration of imperfect sensor input and appearance change [6]	78
2.8	Model of trackers with five most important components [7]	83
2.9	Target region representations used in tracking. From left to right, top to bottom: target bounding box, target contour, target blob, patch-based, sparse set of salient features, parts, and multiple bounding boxes [7]. . . .	84
2.10	An image comparing the linear motion model (a) with the non-linear motion model (b) [8]	88
2.11	An illustration of online (left) and offline (right) tracking. (For interpretation of the references to colour in his figure, the reader is referred to the web version of this article.) [9].	89
2.12	Sample degeneracy and impoverishment illustrated in 1D state space, where green circle indicates particles with significant weight and red circle indicates particles with small weight [10].	92

3.1	Overview of the methodological framework	114
3.2	Original bance rail trolley with hand controller	115
3.3	Converted RIMV with autonomous capability	115
3.4	System architecture and simulation framework	119
3.5	System architecture and dataflow for RIMV	120
3.6	Context-aware decision logic for collision avoidance.	122
3.7	Experimental design for object detection with TL	131
3.8	3D tracking methodology illustration	134
4.1	Dataset annotation with LabelImg	143
4.2	Example of custom dataset with annotation	145
4.3	Comparison between polygon and bounding box annotation	149
4.4	Comparison of polyline and bounding box annotation (where the teal line is the polyline representation, and the red line is the bounding box representation, better seen in colour).	150
4.5	Overview of TL experiment design (better to view in colour), “1pt_r0” means one-stage pre-training with RailSem19 v0, “2pt_r0” means two-stage pre-training with COCO and RailSem19 v0.	154
4.6	Baseline Experiment Box Loss for Training and Validation	163
4.7	All-class results comparison between one-stage and two-stage pre-training with different RailSem19 versions	164
4.8	All-class results comparison between one-stage and two-stage pre-training with different RailSem19 versions with f0135	166
4.9	All-class F1 comparison for different freezing scenarios	169
4.10	All-class mAP50 comparison for different freezing scenarios	169
4.11	All-class results comparison between different RailSem19 variations with f0135	171
4.12	Class-specific F1 comparison between different RailSem19 variations with f0135	173
4.13	Class-specific mAP50 comparison between different RailSem19 variations with f0135	173
4.14	All-class comparison for one-stage pre-training with different RailSem19 versions	178
4.15	Class-specific F1 comparison for one-stage pre-training with different RailSem19 versions	179
4.16	Class-specific mAP50 comparison for one-stage pre-training with different RailSem19 versions	179
4.17	All-class comparison for two-stage pre-training with different RailSem19 versions	182
4.18	Class-specific F1 comparison for two-stage pre-training with different RailSem19 versions	183
4.19	Class-specific mAP50 comparison for two-stage pre-training with different RailSem19 versions	183
4.20	All-class comparison for one-stage pre-training with different RailSem19 versions with f0135	185

4.21	Class-specific F1 comparison for one-stage pre-training with different RailSem19 versions with f0135	187
4.22	Class-specific mAP50 comparison for one-stage pre-training with different RailSem19 versions with f0135	187
4.23	All-class comparison for two-stage pre-training with different RailSem19 versions with f0135	188
4.24	Class-specific F1 comparison for two-stage pre-training with different RailSem19 versions with f0135	189
4.25	Class-specific mAP50 comparison for two-stage pre-training with different RailSem19 versions with f0135	190
5.1	Example of 2D tracking result, the “warthog” vehicle refers to another type of autonomous railway robot used for maintenance tasks.	201
5.2	Camera and Light Detection and Ranging (LiDAR) coordinate comparison	203
5.3	Chessboard used for camera LiDAR calibration	204
5.4	Visualisation of calibration process with ROS Visualisation tool (RVIZ)2	208
5.5	Depth estimation with stereo camera	210
5.6	Visualisation of the camera to LiDAR transformation	213
5.7	Multimodal 3D tracking pipeline using stereo and LiDAR fusion	214
5.8	Visualisation of 3D frustum	215
5.9	RITA contains a ballastless track with a standard track gauge of 1.435 metres and a total length of 18 metres, used for testing and validating autonomous railway systems.	217
5.10	Light setup	218
5.11	Object setup	218
5.12	On-vehicle light setup	221
5.13	Light point to camera	221
5.14	Light not point to camera	221
5.15	External light setup scenarios	221
5.16	Collision avoidance test scenarios	224
5.17	Environmental light impact on camera	226
5.18	On-vehicle light impact on camera	227
5.19	The confidence rate decreases with higher brightness at 1286 Lux	228
5.20	When dynamic light out camera’s Field of View (FoV), and point towards the camera	229
5.21	When dynamic light within camera’s FoV, and point towards the camera .	230
5.22	Example of how glare impacts camera’s detection	231
5.23	Collision avoidance validation experiment setup	232
5.24	Vehicle’s perspective and control console during testing	233

List of Tables

2.1	Different approaches for different settings [11]	61
2.2	Generative and discriminative methods comparison, adapted from [9] . .	98
2.3	Online and offline methods comparison, adapted from [9]	100
3.1	Sensors and equipment used in RIMV	120
3.2	Source dataset comparison	132
4.1	Overview of datasets used in the experiment	142
4.2	Object classes in the custom dataset	145
4.3	Distribution of object classes in the training and validation datasets	147
4.4	Comparison of object classes in different RailSem19 versions	153
4.5	Hardware and software specifications for the experiment	157
4.6	Baseline results for one-stage pre-training with COCO	162
4.7	Class-specific F1 comparison for one-stage and two-stage pre-training across RS19 versions.	165
4.8	Class-specific mAP50 comparison between one-stage and two-stage pre- training across RailSem19 versions	165
4.9	Class-specific F1 comparison for one-stage between two-stage pre-training across RailSem19 versions with f0135	166
4.10	Class-specific mAP50 comparison for one-stage between two-stage pre- training across RailSem19 versions with f0135	166
4.11	Different freezing scenarios tested	167
4.12	Results comparison between 1s-coco and rs_1sv2 without freezing	174
4.13	Results comparison between 1s-coco (f0to5) and 1sv1 (f01357) with the best freezing scenario	175
4.14	Comparison of different RailSem19 datasets	177
4.15	Class-specific results comparison across different RailSem19 versions (one- stage pre-training), with best-performing versions highlighted.	180
4.16	Class-specific results comparison across different RailSem19 versions (two- stage pre-training), with best-performing versions highlighted.	184
4.17	Class-specific results comparison across different RailSem19 versions (one- stage pre-training with f0135), with best-performing versions highlighted. .	186
4.18	Class-specific results comparison across different RailSem19 versions (two- stage pre-training with f0135), with best-performing versions highlighted. .	188
5.1	Camera and LiDAR impact at different measurement locations	220
5.2	Controlled variables in robustness experiments	220

5.3	Metric details, measurement methods, and expected outputs	223
5.4	Expected actuation for each test scenario	224
5.5	Lighting conditions scenarios tested	225
6.1	Mapping research gaps to objectives and novelties.	238
A.1	Comparison of instances in different versions of RailSem19 dataset . . .	289
A.2	Results: RS_v0 one-stage pre-trained	290
A.3	Results: RS_v1 one-stage pre-trained	290
A.4	Results: RS_v2 one-stage pre-trained	291
A.5	Results: RS_v3 one-stage pre-trained	291
A.6	Results: RS_v4 one-stage pre-trained	291
A.7	Results: RS_v0 two-stage pre-trained	292
A.8	Results: RS_v1 two-stage pre-trained	292
A.9	Results: RS_v2 two-stage pre-trained	293
A.10	Results: RS_v3 two-stage pre-trained	293
A.11	Results: RS_v4 two-stage pre-trained	293
A.12	Best overall performance metrics and corresponding dataset versions . . .	294

List of Abbreviations

ACM Active Contour Models

ADAS Advanced Driver-Assistance Systems

ADS-B Automatic Dependent Surveillance-Broadcast

AI Artificial Intelligence

ATC Automatic Train Control

AIS Automatic Identification Systems

ATO Automatic Train Operation

BBX Bounding Box

CAMshift Continuously Adaptive Meanshift

CF Correlation Filter

CNN Convolutional Neural Network

CPU Central Processing Unit

CSP Cross-Stage-Partial

DBSCAN Density-Based Spatial Clustering of Applications with Noise

DLR Docklands Light Railway

DNN Deep Neural Networks

DPM Deformable Parts Model

DTL Deep Transfer Learning

DDS Data Distribution Service

EKF Extended Kalman Filter

ERTMS European Rail Traffic Management System

ETCS European Train Control System

FC Fully Connected

FCN Fully Convolutional Networks

FPN Feature Pyramid Networks

FSD Full Self Driving

FoV Field of View

FPS Frames Per Second

GNSS Global Navigation Satellite System

COLREGs International Regulations for Preventing Collisions at Sea, AKA Collision Regulations

GPS Global Positioning System

GPU Graphics Processing Unit

GSM-R Global System for Mobile Communications – Railway

IMU Inertial Measurement Unit

IoT Internet of Things

IoU Intersection over Union

KITTI Karlsruhe Institute of Technology and Toyota Technological Institute

KF Kalman Filter

KLIEP Kullback-Leibler Importance Estimation Procedure

KMM Kernel Mean Matching

LiDAR Light Detection and Ranging

ML Machine Learning

MLN Markov Logic Networks

MQTT MQ Telemetry Transport

MOT Multiple Object Tracking

NMS Non-Maximum Suppression

OHEM Online Hard Example Mining

PF Particle Filter

RAM Random-Access Memory

RANSAC Random Sample Consensus

RCAS Railway Collision Avoidance System

RF Radio Frequency

RIMV Railway Inspection and Maintenance Vehicle

RITA Railway Innovation Test Area

RPN Region Proposal Network

RIRS Robotic Inspection and Repair System

RoI Region of Interest

RON Reverse Connection With Objectness Prior Networks

ROS2 Robot Operating System 2

RVIZ ROS Visualisation tool

SAE Society of Automotive Engineers

SOT Single Object Tracking

SPP Spatial Pyramid Pooling

SSD Single Shot MultiBox Detector

SVM Support Vector Machines

TCAS Traffic Collision Avoidance System

TCEWS Train Collision Early Warning System

TMCAS Train Monitoring and Collision Avoidance System

TL Transfer Learning

TRL Technology Readiness Levels

UAV Unmanned Aerial Vehicle

UKF Unscented Kalman Filter

V2V Vehicle to Vehicle

VOQ Variability of Quality

YOLO You Only Look Once

Acknowledgements

I would like first to thank my supervisor, Dr. Isidro Durazo Cardenas, for all the support, guidance and continuous encouragement throughout my doctoral journey. His patience, approachability, and commitment to encouraging my growth as a researcher have made an impact on both my academic journey and personal development. I have learned not only how to conduct research but, more importantly, the skills needed to identify and solve problems on my own.

I am also very grateful to my associate supervisors, Dr. Cristobal Ruiz Carcel and Prof. Andrew Starr, for their dedicated support and expert guidance. Cristobal has provided valuable feedback and thoughtful suggestions at every stage of my research, ensuring the clarity of my work. Andrew has been an exceptional mentor, offering invaluable leadership, support, and guidance while sharing his extensive experience. Both of which have had a lasting impact on my academic and professional development.

I am also grateful to my good friends, Yousef and Bernadin, for their constant support and encouragement. Despite their own commitments, they have always made time to review my work, provide constructive feedback, and help me understand complex concepts. Their friendship and generosity have been a vital source of motivation throughout this journey.

Chapter 1

Introduction

1.1 Background and motivations

Railways have been a critical component in global transportation infrastructure, supporting economic growth and environmental sustainability by facilitating the movement of passengers and goods. In the UK alone, train travel increased by 89% over the last two decades, reaching a record high of 1.8 billion journeys in 2018–19 [12]. Similarly, European rail passenger traffic grew 81.5% from approximately 2,440 billion passenger-kilometres in 2004 to 4,429 billion passenger-kilometres in 2023, demonstrating significant growth over the period [13]. Despite the decline during the COVID-19 pandemic, both the UK and EU have seen a rapid recovery in railway travel. However, railway infrastructure faces increasing pressures due to ageing assets, with many parts of the British network nearly 200 years old, including geographically dispersed signalling, electrification, and crossing systems [14]. The scale of railway infrastructure highlights the magnitude of maintenance challenges. For instance, Network Rail spent over £1 billion on the maintenance of British railway infrastructure, which includes over 20k miles of track, 30k bridges, and 2500 stations [15]. In Europe, approximately 300k km of track require maintenance and costs exceeding €25 billion annually [16, 17].

The adoption of autonomous system for railway maintenance has emerged as a promis-

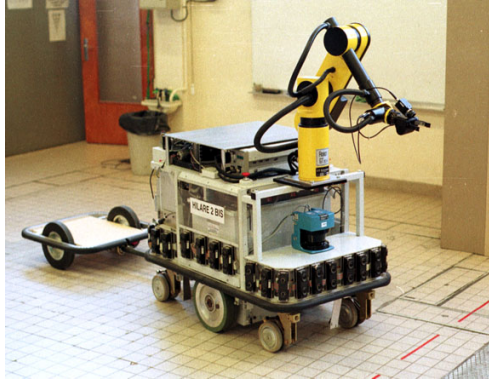
ing direction to address these critical challenges. This system can reduce accidents by minimising human exposure in hazardous environments, such as active tracks or high-voltage areas. Furthermore, their ability to operate continuously improves operational efficiency and significantly reduces costs by automating routine inspections and decreasing the reliance on manual labour. Although autonomous trains already operate in some applications, such as metro systems [18] and airport shuttles [19], their deployment in complex and dynamic mainline environments is still under experimental [20, 21]. Additionally, the adoption of autonomous systems for railway maintenance tasks is still in its infancy. The following subsection will discuss the current developments and challenges in autonomous railway maintenance.

1.1.1 Evolution of task automation in railway maintenance

Automation in railway maintenance tasks has evolved significantly over the past century, driven by the need to enhance safety, efficiency, and cost-effectiveness in railway operations. This subsection explores the historical development of task automation in railway maintenance, focusing on technologies to reduce human intervention in railway maintenance tasks.

1920s–1970s: early adoptions

The first automated systems for railway maintenance were Track Geometry Cars introduced in the 1920s [25]. These vehicles enabled non-disruptive measurement of track parameters such as alignment, curvature, and position, allowing early detection of track irregularities without interrupting regular railway operations. Building on these innovations, the mid-20th century saw notable developments. Companies like Speno International SA, founded in 1960, introduced specialised trains for grinding and re-profiling tracks, enhancing rail longevity and safety [26]. Complementing these advancements, Sperry Rail Service pioneered non-destructive testing technologies in the 1970s, such as ultrasonic and induction testing, to detect internal rail defects with specialised vehicles



(a) Hilare 2bis [22]



(b) RailPod [23]



(c) BOB Sarpsborg [24]



(d) Warthog [17]

Figure 1.1: Autonomous railway maintenance systems.

[27]. These early adoptions built the foundation for modern autonomous railway maintenance, and established a driving force for continued innovation in the field.

1980s–2000s: advancements in mobile systems

The late 20th century and early 21st century witnessed significant advancement in using mobile robotics and autonomous systems for railway maintenance. The development of the “Hilare 2bis” robot in 1992 showcased the feasibility of using robotics for repetitive maintenance tasks [22], such as bolt tightening and track inspection, marking a turning point in railway automation. In the early 2000s, systems like RailPod emerged [23], equipped with sensors for electromagnetic and acoustic defect detection. These technologies automated track inspections, providing detailed data to guide maintenance efforts and reducing reliance on human-led inspections.

2010s–present: advancements in autonomous maintenance systems

The 2010s and beyond marked a new era in autonomous railway maintenance, driven by the integration of Artificial Intelligence (AI), sensor fusion, and the Internet of Things (IoT). Systems like the Autonomous Track Geometry Measurement System (ATGMS) employed non-contact measurement technologies to perform continuous inspections, enabling predictive maintenance strategies [14]. European initiatives, such as the Shift2Rail IN2SMART project, also fostered the adoption of intelligent asset management systems and advanced robotic solutions for railway maintenance [28]. In recent years, autonomous railway maintenance has expanded to include mobile manipulators and lightweight robotic platforms like SINTEF’s BOB (Boots Off Ballast). These systems integrate advanced vision systems, LiDAR, and ultrasonic sensors for defect detection and repair [24]. Additionally, efforts such as Cranfield University’s Robotic Inspection and Repair System (RIRS) have focused on developing prototype frameworks for technology exploration in inspection, repair, and navigation. These systems leverage road-rail conversion and communication interfaces with railway signalling systems but remain at an early stage of development, far from achieving the status of commercial products [17, 29].

1.1.2 From manual to autonomous: collision avoidance in railways

Despite these advancements in autonomous railway maintenance systems, these technologies primarily focus on automating inspection and repair tasks; less attention has been received on the autonomous navigation aspect, and most of these solutions still rely on the remote control or a basic “start-stop” approach without situational awareness ability for the surrounding environment. Railway maintenance tasks often take place in highly dynamic and complex environments with maintenance workers, other machines, and railway infrastructures. It is essential for this system not only to perform maintenance tasks but also to navigate safely by making intelligent decisions based on surrounding obstacles’ behaviour in real-time and step towards fully autonomous railway maintenance. This

section explores the evolution of railway collision avoidance from manual operations to automated solutions.

Currently, railway safety mostly relies on manual operation, where train drivers adhere to signals and speed restrictions to prevent collisions. The European Train Control System (ETCS) was developed to standardise safety protocols in European railways. The current implementation of ETCS level 2 provides communication protocol via Global System for Mobile Communications – Railway (GSM-R), which allows real-time monitoring and control by communicating between the train and track-side equipment. It improves efficiency by removing traditional line-side signals and supporting semi-automated operations. However, it still relies on manual intervention when the GSM-R signal is restricted [30].

The Docklands Light Railway (DLR) in London demonstrated the shift towards automation in urban railways. Since its opening in 1987, the DLR has operated without an onboard driver by integrating collision avoidance strategy through its Automatic Train Control (ATC) and Automatic Train Operation (ATO) systems [18]. These systems monitor the location and speed of trains with track-side and on-train sensors and relay data to a centralised control system to adjust train movement and prevent collisions. By the 2000s, automated metro systems, such as the Automated People Movers systems in Paris airport, integrated onboard radar-based collision detection and avoidance within their Automatic Train Control (ATC) framework, eliminating driver errors and ensuring safe operation [19]. Advancements in the 2010s introduced the Railway Collision Avoidance System (RCAS), which employed onboard train-to-train communication for independent collision detection and prevention, reducing reliance on centralised control systems [31]. This was complemented by the Train Collision Early Warning System (TCEWS), tested on the Hankou-Yichang Railway, which leveraged GPS, prior maps, and wireless communication for real-time monitoring and predictive collision alerts [32]. In recent years, the Train Monitoring and Collision Avoidance System (TMCAS) has demonstrated the ability by integration of IoT technologies, enabling real-time exchange of train data via MQ

Telemetry Transport (MQTT) protocols for onboard and distributed collision avoidance without full dependence on centralised systems [33].

1.1.3 Limitations of existing railway collision avoidance solutions

Despite the advancement in railway collision avoidance systems, there are still limitations. Many of these systems are designed to operate in constructed and predictable environments. These environments present fewer challenges compared to the mainline railway or maintenance environments. For example, airport shuttles operate on a dedicated track with minimal external influences, such as vehicles and trespassers. Although effective within the intended scope, these systems cannot be applied to a larger scale within a more complex environment.

Another limitation is the reliance on communication. Solutions such as ETCS Level 2 and the DLR employ a centralised approach, where decisions are made remotely based on on-vehicle or track-side sensors [18, 30]. Although the Vehicle to Vehicle (V2V) approach improves autonomy by making onboard decisions, they still require stable and uninterrupted communication between vehicles. These systems face challenges when communication is disrupted or delayed. Additionally, communication-dependent systems are also vulnerable to cyberattacks, which could lead to unsafe operations or system shutdowns.

Furthermore, these systems lack contextual awareness capability. Modern collision avoidance systems on self-driving cars can make intelligent decisions based on the detected objects' type and movement. For example, Tesla's Autopilot system implements advanced machine learning algorithms for object detection and tracking [34, 35], allowing them to anticipate the movement of other vehicles or workers and adjust their speed or change lanes to avoid collisions. Despite a similar opportunity, existing collision avoidance systems on autonomous railway maintenance systems still rely on basic "start-stop" mechanisms without understanding the context of the surrounding environment.

Lastly and perhaps most notably, none of these systems are designed for railway main-

tenance scenarios. Collision avoidance for passenger and freight rail focuses on ensuring safe operation along predefined tracks. In contrast, railway maintenance vehicles typically operate in a more dynamic and complex environment and require the ability to detect and classify various objects, such as tools and other machinery. Additionally, railway maintenance tasks often require close human-robot collaboration, where autonomous vehicles and human workers operate in shared spaces; this necessitates the capability of understanding workers' movement and dynamically adjusting vehicles' movement to avoid disruption or compromising safety.

1.1.4 Motivations of this study

The limitations of current railway collision avoidance systems highlight the need to address the unique challenges of railway maintenance environments. This research bridges the gap by developing a context-aware collision avoidance system for autonomous railway maintenance vehicles.

Compared to existing systems, the proposed system can make intelligent decision-making based on the obstacles' type and behaviour. For instance, by distinguishing worker and other autonomous maintenance vehicles and anticipating their movement, the system can facilitate human-robot or robot-robot collaboration while ensuring safety. On the other hand, if a trespasser is detected, the system can stop immediately and warn nearby workers of the intrusion. Additionally, the ability to understand railway infrastructure allows the system to make informed decisions about navigation and adhering to operational protocols in complex scenarios, such as identifying track-side signals or pedestrian crossing points. Lastly, recognising track switches allows the system to identify high-priority areas for maintenance, improving the efficiency of defect inspections.

The proposed system shows promising potential for autonomous railway maintenance. It prevents accidents and minimises potential damage to railway infrastructure and nearby workers. The context-aware ability improves operational efficiency by reducing unnecessary stops and delays and reduces operational costs through improved efficiency and

optimised resource management. The system is designed with modularity and scalability, which is adaptable for future advancement in railway automation. Furthermore, the foundational technologies extend beyond the railway sector, offering solutions for other industries which require safety-critical automation, such as logistics and mining.

In addition, this research aligns with government policies promoting the adoption of automation and AI to enhance efficiency and competitiveness. The UK's National AI Strategy emphasise the role of AI-driven technologies in improving operational resilience, productivity, and innovation across various sectors, including transportation [36]. By integrating AI-based context-aware collision avoidance systems, the railway industry can benefit from operational improvements, aligning with national objectives to position the UK as a global leader in AI and innovation.

1.2 Research scope, aim and objectives

1.2.1 Research scope

The scope of this research focuses on the development and validation of a context-aware collision avoidance system for autonomous railway maintenance vehicles operating in dynamic and complex railway environments. It addresses key challenges in railway object detection, 3D object tracking, system integration and validation within a modular robotic framework.

While addressing these challenges, this research aims to improve railway object detection with TL techniques instead of developing new machine learning algorithms. It focuses on investigating how TL impacts object detection performance in railway maintenance scenarios in terms of source dataset size, relevance, and diversity, as well as fine-tuning strategies. The objective is not to maximise object detection performance but to gain insights into the optimal conditions for transferring knowledge to railway-specific tasks. For 3D object tracking, the focus is to transform existing 2D tracking algorithms into 3D by integrating stereo camera and LiDAR data through sensor fusion without de-

veloping novel algorithms or addressing broader applications.

These two systems are further integrated into a context-aware collision avoidance framework within a Robot Operating System 2 (ROS2)-based robotic platform. Validation includes indoor tests to assess robustness under varying lighting conditions and outdoor tests conducted on a controlled, real-world railway test facility to demonstrate feasibility and readiness rather than deployment in active live maintenance settings. The following objectives outline the key steps to achieve these research goals:

1.2.2 Research aim and objectives

This research aims to **develop a context-aware collision avoidance system for autonomous railway maintenance vehicles**, with the following objectives:

- **Objective 1:** *conduct critical literature on context-aware collision avoidance for autonomous railway vehicles*, focusing on railway scene study, object detection, object detection and depth estimation. This objective identifies research gaps, state-of-the-art methodologies, and emerging technologies in existing methods. It establishes a foundation for the subsequent objectives and justifies the research direction and methods adopted in this study.
- **Objective 2:** *create a dataset for training object detection in railway maintenance scenarios*. This objective addresses the gap of lacking annotated datasets by focusing on objects frequently appearing in railway maintenance scenarios, such as identifying maintenance workers or other machinery.
- **Objective 3:** *fine-tune railway object detection models with TL by leveraging knowledge learned from large datasets*. Compare the performance of pre-training with various datasets in terms of data quantity and relevance and evaluate various fine-tuning strategies. This objective aims to identify the optimal TL strategy to improve railway object detection with the custom dataset from objective 2.

- **Objective 4:** *implement a multimodal sensor fusion strategy for detecting and tracking objects in 3D.* This approach combines a stereo camera for object detection and LiDAR for distance estimation by aligning the coordinate system between two sensors. This objective extends 2D object tracking into 3D, enabling RIMV's context awareness about surrounding obstacles' type and position.
- **Objective 5:** *implement the collision avoidance system by integrating the object detection and 3D tracking system* within a ROS2-based robotic framework to ensure system modularity, scalability and real-time performance. This objective enables RIMV's context-aware actuation decision based on the type and location of detected obstacles (as identified in objective 3).
- **Objective 6:** *validate the system through extensive indoor and outdoor testing.* Indoor experiments focus on testing the system's robustness under various lighting conditions commonly encountered for railway maintenance tasks. The outdoor experiment assesses the system's performance in a real-world environment, ensuring RIMV achieve a high Technology Readiness Levels (TRL).

1.3 Research gaps and novelties

The development of a context-aware collision avoidance system for autonomous railway maintenance vehicles presents several unique challenges. This section discusses the gaps and introduces novel methodologies and contributions to address these gaps.

1.3.1 Research gaps

Limited railway maintenance datasets

A critical gap in existing research is the lack of large, annotated datasets for railway maintenance scenarios. Object detection models rely on machine learning algorithms trained on large annotated datasets. Most available datasets, such as COCO [37], which

includes over 330k images, with more than 200k labelled images and 1.5 million object instances, focus on general object detection with less relevance to the unique objects and infrastructure in railway environments. While railway-specific datasets like RailSem19 [38] provide annotations for 19 railway object classes with 8,500 annotated images, it is significantly smaller than COCO and primarily focuses on railway infrastructure (such as track, platform, etc.), excluding maintenance-related objects such as maintenance workers and equipment, which limit the development of robust object detection models for railway maintenance tasks. Addressing this gap requires the development of annotated datasets that are frequently encountered in real-world railway maintenance tasks (e.g., maintenance workers or equipment).

Transfer learning opportunities

Given the limited availability of datasets for object detection in railway maintenance environments, TL has emerged as a promising strategy. However, several key research gaps remain unaddressed. First, it is unclear whether pre-training on large, general-purpose datasets (e.g., COCO) or smaller, domain-specific datasets (e.g., RailSem19) provides more accurate object detection results. Second, the effectiveness of one-stage vs. two-stage pre-training strategies is not well understood, particularly for scenarios where the target dataset (e.g., a custom railway maintenance dataset) is small. Lastly, the role of class diversity in the source dataset has not received enough attention. It is unclear how the variety and number of object classes in the source dataset influence the TL process. Addressing these gaps could guide the selection of appropriate source datasets for object detection in railway maintenance tasks.

3D object tracking and system integration

Another research gap is the development of multi-sensor 3D tracking systems and system integration. 3D object tracking combines the process of object detection and distance estimation for each detected object. While the camera excels at object detection tasks, it

cannot make accurate distance measurements, especially in a poor lighting environment. Conversely, LiDAR provides accurate distance estimation, but it is less accurate in object detection. A sensor-fusion approach can bridge this gap by combining the advantages of both sensors. However, implementing such fusion introduces additional challenges, including sensor calibration, synchronisation, and alignment errors.

Additionally, integrating object detection, 3D tracking, and collision avoidance into a unified, real-time robotic framework remains a significant research gap. Existing work often treats these subsystems separately, and the integrated system needs to ensure synchronisation across multiple sensor streams (e.g., cameras and LiDAR). Delays or misalignment in sensor data processing can compromise safety and real-time performance.

Context-aware collision avoidance and system validation

Compared to the "start-stop" collision avoidance approach deployed on existing autonomous railway vehicles, a context-aware system must not only detect objects but also classify their type, track their movement, and make dynamic actuation decisions based on the object's class and relative position. For example, the system should slow down for maintenance workers but stop immediately for trespassers. Additionally, system validation in controlled environments often fails to generalise to outdoor railway environments. For example, lighting variability, such as night-time or tunnels, and vibration from the vehicle's movement on uneven tracks can introduce noise into sensor readings, affecting system performance. These factors highlight the importance of comprehensive validation under varied operational conditions.

1.3.2 Research novelties

This research contributes to the field of autonomous railway maintenance by addressing critical gaps in railway object detection, 3D object tracking, and system integration for collision avoidance. The following subsections describe the research novelties introduced to overcome these challenges and their broader implications.

Limited railway datasets:

- Development of a custom dataset for object detection featuring object classes tailored for railway maintenance scenarios.

Object detection with transfer learning:

- Optimisation of object detection performance using TL with the custom dataset by leveraging knowledge learned from larger datasets.
- Investigation into the impact of dataset size, diversity, and class alignment on TL performance.

3D object tracking:

- Calibration and synchronisation to ensure spatial consistency between the stereo camera and LiDAR.
- Design and implementation of a sensor fusion-based 3D tracking system that combines a stereo camera and LiDAR data through a dynamic frustum-based fusion pipeline. The stereo camera provides preliminary depth estimation, while LiDAR refines object localisation.

Context-aware collision avoidance and system validation:

- Development of context-aware collision avoidance framework by integrating detection and 3D tracking within a modular ROS2-based robotic framework.
- Validation on the proposed system by conducting controlled indoor testing and real-world outdoor evaluations using the RIMV [39] as a test platform (as shown in Figure 1.2).



Figure 1.2: RIMV is a lightweight, adaptable, autonomous platform designed for railway maintenance tasks. Its modular design allows for integration with various equipment to suit different maintenance requirements. This image showcases the vehicle equipped with ultrasonic testing technology for rail defect detection on an active railway track under possession.

1.4 Contributions to knowledge

This research makes a significant contribution to the field of autonomous railway navigation by addressing critical gaps in the development of a context-aware collision avoidance system. Compared to existing systems, which rely on a basic “start-stop” mechanism, this system integrates object detection and 3D tracking with a sensor-fusion approach to enable responsive and context-aware decision-making based on the obstacle’s type and position.

1.4.1 Created a specialised dataset for railway maintenance scenarios

A key contribution of this work is the development of an object detection system for railway maintenance scenarios. A custom dataset was created to address the dataset scarcity challenge; the dataset was collected at the Northampton & Lamport Railway in the UK, a

heritage railway serving tourist purposes but maintaining the same standards and features as a passenger railway. It contains 103 annotated images for training learning-based object detection algorithms. Six object classes were selected: pedestrian, worker, warthog (another type of railway robot), signal, track-switch and pedestrian crossing.

These objects were chosen to ensure the vehicle's understanding of railway infrastructure and the dynamic aspect of the maintenance environment. Maintenance workers and warthogs represent scenarios requiring human-robot and robot-robot collaboration, ensuring safe coexistence during railway maintenance tasks. Pedestrians can be treated as trespassers and used to trigger emergency stops to prevent accidents. Track switches and signals are vital railway infrastructure components that guide traffic rules and indicate maintenance zones.

This dataset addresses the scarcity of railway maintenance datasets by providing annotated images focusing on frequent objects frequently encountered in such environments. It also serves as a foundation for TL experiments and future advancements in railway-specific object detection.

1.4.2 Enhanced railway object detection with transfer learning

TL is implemented to improve railway object detection performance with the relatively small custom dataset. The experiments focus on evaluating TL with various datasets and fine-tuning with the custom dataset, as well as different fine-tuning strategies to optimise model performance:

- **TL with COCO:** COCO is a large-scale, general-purpose dataset containing over 330k images and 80 object classes. However, COCO lacks railway-specific objects. This experiment pre-trained the model with COCO and then fine-tuned it with the customer dataset.
- **TL with RailSem19:** RailSem19 contains 19 object classes and focuses on railway objects such as tracks, switches, platforms, etc. However, it is relatively small in

size (8500 annotated images) compared to COCO and lacks maintenance-related object classes like workers. This experiment pre-trained the model with RailSem19 and then fine-tuned it with the customer dataset.

- TL with focused classes: this experiment modified the RailSem19 dataset from its original 19 classes, progressively narrowing it down to 4, focusing on objects that only appear in the custom dataset: signals, switch, pedestrian-crossing and pedestrian. This experiment focuses on investigating the impact of dataset diversity and class alignment on TL outcomes.
- Fine-tuning strategies: in YOLOv8 architecture, early layers capture low-level features such as edges and textures. In contrast, later layers capture high-level features such as patterns and representations. This study investigated which feature should be preserved during TL for different source datasets.

The results of this experiment addressed the critical question: “Is bigger always better?” by comparing the trade-offs between dataset quantity and relevance in TL. Additionally, the results evaluate the impact of dataset diversity and highlights the importance of aligning the source dataset with the target dataset to optimise model performance. Lastly, the fine-tuning experiment identifies the optimal strategy of feature preservation for specific tasks. Beyond improving railway object detection, these findings establish a comprehensive framework to enhance object detection with a limited dataset.

1.4.3 Developed a multimodal 3D object tracking system

Compared to 2D object tracking, which identifies an object’s position with an image, 3D object tracking facilitates context awareness actuation decisions based on the understanding of detected objects’ movement relative to the vehicle in real-time. The 3D tracking system is built on the YOLOv8 framework combined with ByteTrack 2D object tracking with the stereo camera; it provides the object’s type and 2D position within the image. The

integration of LiDAR data extends this capability into 3D, enabling accurate localisation of objects relative to the vehicle.

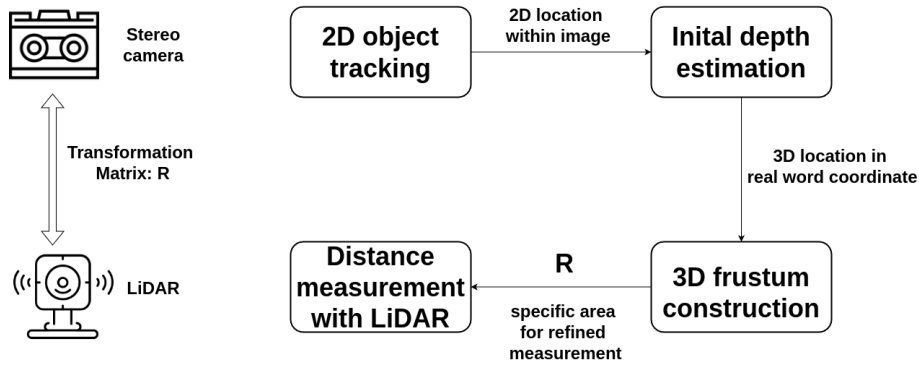


Figure 1.3: Flowchart for camera LiDAR fusion

Figure 1.3 illustrates the fusion pipeline, it begins with the calibration between the stereo camera and LiDAR to identify the transformation matrix between two sensors; this matrix enables the projection of the camera's 2D bounding box into LiDAR's 3D coordinate, ensuring spatial consistency across the two sensors. Subsequently, a stereo camera provides a preliminary depth estimation by calculating disparities between its two lenses. The depth estimation is used to extend the 2D bounding box into 3D frustum space, where the size of the frustum is dynamically adapted based on the detected object type. Finally, a clustering algorithm is implemented to the LiDAR pointcloud data to refine the depth estimation of detected objects. This pipeline enables accurate depth estimation for the detected object by combining the strength of a stereo camera and LiDAR; this system enhances the vehicle's context awareness about the surrounding environment and context-aware collision avoidance for autonomous railway operations.

1.4.4 Implementation and validation of collision avoidance system

The context-aware collision avoidance is built on the foundation of a 3D object tracking system within a ROS2-based robotic framework, ensuring modularity, scalability and real-time operation [39]. Beyond the situational awareness provided by the stereo camera and LiDAR, which supply object type and locations, the system integrates localisation inputs

from Global Positioning System (GPS), Inertial Measurement Unit (IMU), and encoders, providing position awareness about the vehicle's movement and orientation relative to the obstacles. The system outputs include vehicle actuation, controlling motors for movement and LED lights/alarms for safety alerts. Additionally, it supports peripheral connection, allowing integration with specialised inspection and maintenance equipment for versatile applications.

This study conducted both indoor and outdoor validation to evaluate the system's robustness and practicality. The indoor validation focuses on assessing the system's robustness under various lighting scenarios, including environmental lighting, vehicle's headlight and external light from other vehicles, simulating real-world challenges such as tunnels and night-time operations. The outdoor validation was conducted in a real-world track to assess its ability to make context-aware actuation based on the surrounding obstacles' type and position. The results demonstrated the system's effectiveness but also identified gaps for improvement, offering valuable insights and standards for deploying such systems in real-world railway applications. Figure 1.4 and 1.5 illustrate the validation setup for both indoor and outdoor. These evaluations demonstrate the system's potential for practical deployment in autonomous railway maintenance while providing a framework for addressing identified limitations.



Figure 1.4: Indoor robustness validation with on-vehicle lighting



Figure 1.5: Outdoor collision avoidance validation with worker and trespasser

1.4.5 Broader implications

The developed context-aware collision avoidance system represents a significant improvement over the existing “start-stop” approach, which cannot consider an object’s type or movement. This system can make context-aware decisions by understanding the type and movement of obstacles. This advancement not only reduces the risk for human operators and railway infrastructures but also increases efficiency by eliminating unnecessary stops caused by less-threatening scenarios, such as surrounding workers that do not introduce the vehicle’s path. Moreover, the system’s ability to classify and track obstacles enables intelligent path planning and enhances human-robot collaboration.

Beyond railway maintenance tasks, this work has broader implications for the design and deployment of autonomous systems in other safety-critical environments. The methodologies developed, such as sensor fusion and object detection with TL, have cross-domain applicability, extending to industries like logistics, mining, and urban mobility.

1.5 Thesis layout

The remainder of the thesis is organised into six chapters,

- **Chapter 2: Literature review**, this chapter surveys four critical components of railway collision avoidance systems: railway scene and object analysis, object detection techniques, object tracking methodologies, and depth estimation approaches. This chapter identifies the gaps and limitations by analysing this interrelated topic. It sets the scope for developing a domain-specific approach to enhancing the safety and reliability of autonomous railway systems.
- **Chapter 3: Methodology**, this chapter presents the technical approaches for the proposed collision avoidance system. It includes details on enhancing railway object detection with TL, the sensor fusion strategy for 3D object tracking, and the integration of these components into a ROS2-based robotic framework. Finally, it

discusses the indoor and outdoor validation plan to assess the system's robustness and effectiveness.

- **Chapter 4: Railway object detection with TL**, this chapter details the dataset preparation, experiment design, and evaluation of railway object detection with TL. The results compare the performance of TL from a large general and smaller domain-relevant dataset, as well as the object class diversity, and discuss various fine-tuning techniques.
- **Chapter 5: 3D object tracking and collision avoidance system**, this chapter details the design, implementation, and validation of a sensor-fusion-based 3D object tracking system, alongside the development and validation of a collision avoidance system. Indoor validation assesses the robustness of 3D object tracking under various lighting scenarios, and outdoor validation validates the collision avoidance system in a real-world environment.
- **Chapter 6: Conclusion**, this chapter summarises the contribution of knowledge of this project, discusses the research impact and current limitations of this project, and identifies future works for improvement.

Chapter 2

Literature Review

2.1 Introduction

The purpose of this literature review is to investigate and explore advanced techniques for context-aware and adaptive collision avoidance for autonomous railway maintenance robots. The goal is to enable autonomous railway vehicles to make context-aware decisions based on the type and motion of detected objects, moving beyond the current simplistic start-stop solutions [29]. Specifically, this review focuses on three essential tasks: railway object detection, object tracking and depth (distance) estimation for the tracked objects. By reviewing the state-of-the-art methods and technologies in this field, this literature review aims to identify key challenges, propose potential solutions, and outline future research directions to improve collision avoidance systems for autonomous railway maintenance vehicles, ultimately to enhance the robot's ability to navigate complex railway environments autonomously and efficiently. Additionally, due to the unique challenges that distinguish the railway environment and other settings for autonomous vehicle operation and the limited dataset to train railway object detection algorithms, the literature review also includes a railway scene analysis and TL techniques.

Review methodology

The relevant papers analysed in the upcoming sections were primarily from the Scopus, IEEE, and Google Scholar databases. Additionally, the review further includes technical reports, government documents, and industry standards. These sources offer guidance for regulatory requirements and industry practices.

To ensure a systematic search process, two groups of keywords were used. The first group focuses on high-level concepts that define the broader scope of the research: **autonomous railway vehicle, railway maintenance robot, collision avoidance system, railway environment analysis, LiDAR-based object detection, and railway-specific object detection**. The second group focuses on more technical and methodological aspects directly relevant to the research's objectives: **(learning-based) object detection, TL, object detection/classification datasets, object detection benchmark, railway datasets, 2D/3D object tracking, multi-object tracking, object tracking benchmark, depth estimation, and sensor fusion**.

When the two sets of keywords were combined, each of the selected databases returned an extensive number of results, often exceeding 10k records. Several filtering criteria were applied to refine the search results. First, the scope of the search was restricted to subject areas directly relevant to this research, which are: **robotics, computer vision, artificial intelligence, machine learning, autonomous systems**. Additionally, the results were narrowed to include only peer-reviewed journal articles, conference proceedings, and more recent publications after 2010.

In the initial selection, paper titles were used to identify relevance to the research objectives of this thesis. For titles that suggested relevance, the abstracts were analysed to assess the significance and direct contribution, such as object detection, 3D tracking, and multi-sensor fusion for autonomous railway robots. The final selection of articles was informed by their relevance to the core objectives of the thesis. The most relevant works were reviewed in full, with particular attention given to state-of-the-art approaches to sensor fusion, TL for object detection, multi-object tracking and highly cited review

papers.

Chapter structure

The rest of this chapter is structured as follows:

- **Section 2.2:** Collision avoidance in other fields and common challenges
- **Section 2.3:** Railway scene & object analysis
- **Section 2.4:** Object detection
- **Section 2.5:** TL
- **Section 2.6:** Object tracking
- **Section 2.7:** Depth estimation for tracked object

The surveyed topics are interrelated and collectively support the development of context-aware collision avoidance systems for autonomous railway maintenance vehicles. **Section 2.2: Collision avoidance in other fields** explored the state-of-the-art collision avoidance system used in other fields and identified the common challenges. **Section 2.3: Railway scene & analysis** provide essential context by discussing the railway infrastructures and the unique challenge specific to the railway maintenance environment.

Object detection ensures that the collision avoidance system can distinguish between various objects, such as maintenance workers and trespassers, to make informed decisions about the vehicle's actuation. **Section 2.4: Object detection** surveys the recent advancement in learning-based object detection algorithms. **Section 2.5: TL** explores TL techniques, which can address the dataset scarcity problem in railway maintenance scenarios, improving detection performance even with limited railway-specific training data.

Object tracking enables the collision avoidance system to monitor the motion of detected objects, predict their trajectories, and assess potential collision risks. **Section 2.6:**

Object tracking explores popular tracking methods. Lastly, **Section 2.7: Depth estimation** investigates techniques to accurately measure the distance and spatial relationship of tracked objects to the vehicle. These surveyed topics collectively advance context-aware collision avoidance systems, addressing object detection, TL, object tracking and depth estimation to overcome unique challenges in autonomous railway maintenance.

2.2 Collision avoidance in other fields

Collision avoidance systems have been extensively explored across various industries, such as aviation, maritime and automobile industries. These systems typically leverage multiple sensor systems to monitor obstacles' position and prevent collision. Recent advancements in data fusion and AI have further improved the functionality and reliability. This section briefly discusses how collision avoidance is implemented in another field and the common challenges.

In aviation, systems such as the Traffic Collision Avoidance System (TCAS) are critical for maintaining safe air travel. TCAS operates by communicating with nearby aircraft about their flight status, such as range and altitude. It identifies potential collisions by creating a 3D map, alerts pilots about nearby aircraft and provides specific instructions to maintain safe separation [40]. Emerging technologies are enhancing traditional TCAS capabilities. For example, the integration of Automatic Dependent Surveillance-Broadcast (ADS-B) systems provides enhanced situational awareness by broadcasting aircraft positions to nearby systems [41]. Additionally, autonomous collision avoidance algorithms are being developed for unmanned aerial vehicles (UAVs), where sensor fusion of radar, LiDAR, and cameras ensures safe navigation in increasingly crowded airspaces [42].

Maritime collision avoidance systems increasingly rely on radar, Automatic Identification Systems (AIS), and GPS for safe navigation. By fusing AIS data with techniques such as Gaussian Mixture Models, systems can predict vessel trajectories and improve

navigational safety [43]. These systems are typically designed to comply with international regulations such as International Regulations for Preventing Collisions at Sea, AKA Collision Regulations (COLREGs), ensuring safe navigation and adherence to legal standards even in congested waters [44].

In the automotive industry, emerging AI-driven solutions are reshaping the capabilities of collision avoidance systems. For example, Tesla's Full Self Driving (FSD) implemented a vision-based system with advanced neural networks, and it trains vast datasets from its onboard camera for object detection and distance estimation [34, 35]. This approach achieves real-time situational awareness and decision-making in complex driving scenarios. Additionally, many other solutions integrate AI with LiDAR and cameras to detect obstacles, predict collisions, and execute corrective actions such as automated braking or steering. Research has shown Advanced Driver-Assistance Systems (ADAS) to be highly effective in reducing rear-end collisions and mitigating human errors [45].

2.2.1 Common challenges in collision avoidance systems

Traditional approaches employed in maritime and passenger aviation often rely on centralised or communication-based systems to maintain safety. For example, TCAS and AIS depend on radar and transponder communication to monitor nearby vessels/aircraft positions to prevent collision. These systems are highly effective in structured and less complex environments, but they face challenges in dynamic, unstructured scenarios or when communication infrastructure is compromised. Furthermore, these systems typically assume cooperative participants, limiting their effectiveness in scenarios involving non-cooperative or unexpected obstacles. On the other hand, more advanced solutions, such as those developed for Unmanned Aerial Vehicle (UAV) and self-driving cars, focus on real-time object detection, obstacle trajectory prediction, and decision-making.

Railway collision avoidance systems face similar challenges, and they rely on multiple sensors such as LiDAR, cameras, and radar for localisation and obstacle detection, employing sensor fusion to combine data from Global Navigation Satellite System (GNSS),

IMU, and vision-based systems for enhanced accuracy. Additionally, similar to TCAS and AIS, communication in railway systems must integrate with centralised traffic management frameworks like European Rail Traffic Management System (ERTMS) and ETCS, contrasting with the decentralised communication protocols commonly used in road and air systems. However, unlike road vehicles, which operate in open terrains with lateral flexibility, or aviation systems operating in a 3D space, trains are restricted to fixed tracks, requiring precise longitudinal obstacle detection and stopping capabilities. The high mass and speed of trains necessitate longer braking distances. Furthermore, railway maintenance vehicles often operate with environmental constraints, such as track-specific obstacles, fallen debris, and adverse weather conditions. These differences highlight the need for tailored approaches to railway collision avoidance systems.

2.3 Railway scenarios analysis

The development of RIMV offers a flexible platform for railway maintenance tasks. These vehicles can be equipped with additional kits, such as ultrasonic sensors for defect detection or robotic arms for maintenance and repair. Unlike urban or suburban environments, railway scenes involve specific infrastructure and dynamic elements. This section focuses on the analysis of the unique characteristics of railway scenes and objects.

2.3.1 Railway track possession

Another unique challenge in railway collision avoidance is the need to comply with “railway possessions.” Possession refers to the temporary planned closure of certain railway tracks to regular train traffic, allowing maintenance work to be conducted safely. To ensure this safety, possession management is governed by rules such as the GERT8000 [1]. These regulations provide a framework for coordinating signalling, train movements, and worker safety, highlighting the structured and controlled nature of possession operations. Figure 2.1 illustrates a typical railway possession; apart from the intended work site, a

possession includes an additional buffer zone of 400-metre on each side of the work site and safety mechanisms such as detonator protection and junction clearance. The buffer zones provide a safety margin to prevent unauthorised access and allow trains to stop safely before entering the work area. Detonator protection serves as a physical fail-safe, alerting train drivers through audible signals in case of signalling failures. Additionally, the junction clearance avoids operational disruption on unaffected routes and prevents signalling confusion.

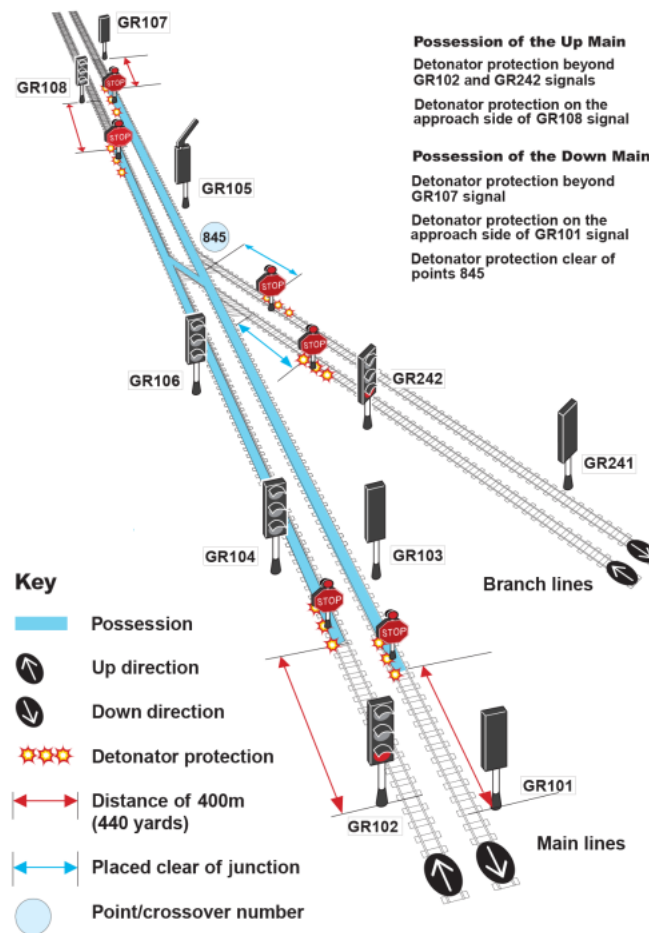


Figure 2.1: Illustration of railway possession [1]

During possessions, railway maintenance vehicles are expected to collaborate closely with human workers and add another layer of complexity. Human workers typically perform tasks such as equipment operation, material handling and vegetation management. Autonomous maintenance vehicles assist by transporting equipment, conducting auto-

mated inspections, or performing specific maintenance tasks, which require advanced navigation capability, detecting and classifying obstacles, and predicting the movements of workers and equipment.

Furthermore, as the railway industry moves towards greater automation, these autonomous systems are also expected to be deployed outside of possession periods. They must interact not only with surrounding workers and machinery but also with other trains in active rail environments. This demands the integration of communication and collaboration mechanisms with centralised traffic management systems such as ERTMS and ETCS to ensure real-time synchronisation with train schedules, signalling, and infrastructure conditions.

2.3.2 Railway system components

When modelling a complex railway environment, accurate geospatial information about the railway infrastructures and surroundings is an essential prerequisite for autonomous railway robots. Compared to other types of scenes, railway infrastructure geometric relations remain roughly unchanged within established regions and have similarities between them [46]. Figure 2.2 includes an overview of typical railway infrastructure; key elements of the railway system comprise rail tracks, overhead cables, masts, poles, etc.

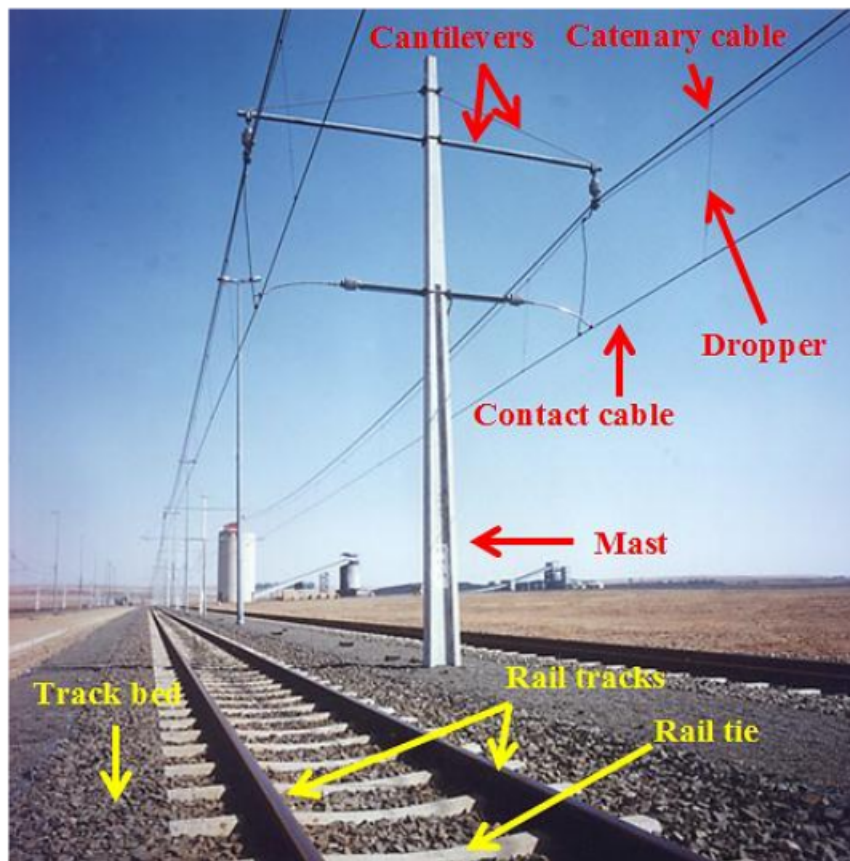


Figure 2.2: Railway system components [2]

Track bed, rail tracks and rail tie

The track bed is the flat surface under the rail track; although the track bed is not a key element of the railway infrastructure, it forms the foundation of the railway infrastructure and facilitates handling of train load, drains well, and plant growth [2, 47]. Additionally, many railway infrastructure (tracks, power cables) are located in the extended area of the track bed. For autonomous railway maintenance vehicles, it is more efficient to recognise the track bed first instead of detecting each component individually [47]. The track bed is designed and constructed to be a surface with the smallest height variation possible to provide a safe and stable platform for trains in motion [2].

Rail ties, also known as sleepers, are horizontal supports placed beneath the rail tracks. They distribute the weight of passing trains across the track bed, ensuring stability and

alignment. Properly maintained rail ties are crucial for safety, as misaligned or damaged ties can lead to operational inefficiencies. Once the track bed has been identified, the rail tracks (also known as gauge) can be detected as two parallel steel rails running along each side of the track bed with a consistent lateral distance. In the UK, the standard gauge is typically 1435 mm, except for some heritage railway lines that use non-standard measurements ¹.

Mast, cantilevers and overhead cable system

Masts are vertical structures that support overhead catenary systems; they are located in regular spatial intervals near rail tracks. Cantilevers are horizontal arms attached to the mast. Both the mast and cantilevers support and ensure the stability and alignment of the overhead catenary cable systems. The catenary cable system provides electric power and is a critical component of electrified railway infrastructure.

The catenary cable system is a critical component of the electrified railway infrastructure, providing the necessary electric power to trains. Contact cables are the lowest in height among all overhead power cables, and they are energised by electricity and transmit the power to trains. Catenary cables are a system of curve-shaped cables suspended between masts. They lie immediately above contact cables and are held in place by drop-pers. Return current cables lie highest among all power cables and are connected to the topmost part of the masts. Similar to catenary cables, return current cables are curvilinear cables that are suspended from the topmost part of the masts [2].

Turnouts

Railway turnouts, also known as switches and crossings, are essential railway components that enable trains to change tracks safely (as shown in Figure 2.3). Turnouts require regular maintenance to handle dynamic interactions and heavy loads without compromising safety. They are subject to high wear rates and failure risks due to the discontinuities in

¹ See: Train Technical Specification at <https://assets.publishing.service.gov.uk>

rail geometry and variability in system stiffness [48].



(a) Two parallel tracks that are connected by two single turnouts.



(b) Several tracks with a diamond crossing in front of the train.

Figure 2.3: Exemplary railway scenarios, several tracks and different connection types, such as single turnouts and a diamond crossing, are shown. Each track is built up of two parallel rails with a fixed distance [3].

Other components in railway maintenance scene

In addition to the aforementioned railway infrastructures, autonomous railway maintenance vehicles must effectively monitor these components to enable safe maintenance operations.

Signal plays a crucial role in regulating train operations, providing visual indicators of speed limits, stops, and track conditions. Although current railway maintenance tasks rely on the “possession” and do not require interaction with this signalling system, future

autonomous maintenance systems will need to detect and interpret signals to synchronise with operational schedules and avoid disrupting regular traffic.

Pedestrian crossing is the safe locations where workers and pedestrians cross railway tracks; these crossings present potential hazards due to the interaction between humans and railway operations. Autonomous maintenance systems need to understand these crossings to ensure safety.

Level crossing, similar to pedestrian crossing points, allows road vehicles to cross railway tracks. Autonomous railway maintenance systems can benefit from understanding these scenes to prevent collisions.

Track-side infrastructure, such as utility poles, drainage systems, and fencing, plays an integral role in maintaining the railway environment's stability and functionality. Understanding these enables the maintenance system to perform targeted inspections and maintenance tasks, such as identifying damaged fencing, clearing blocked drainage systems, or inspecting utility poles for wear.

Maintenance workers and other equipments, such as specialised tools and machinery, are dynamic elements commonly present in maintenance scenarios. Autonomous maintenance systems must be capable of detecting them to prevent accidents and facilitate safe human-robot or robot-robot collaboration.

2.3.3 Railway scenarios study

Urban and suburban railway scenarios

The urban railway scene is a complex environment and often contains buildings and other infrastructures, like bridges and tunnels, which usually pose challenges in autonomous navigation. GPS signals frequently suffer from interference and noise, creating error readings that can compromise navigation accuracy [49]. Moreover, the dynamic nature

of these environments with pedestrians and vehicles requires robust detection systems to ensure safe operation [50]. Finally, electromagnetic interference from urban infrastructure can degrade the performance of electronic sensors and interfere with a vehicle's communication component [51].

Suburban is more spacious compared to the urban scene, which facilitates clearer GPS signal reception and reduces issues such as multi-path interference. This environment typically leads to more straightforward navigation and operational protocols due to less frequent interactions with dynamic obstacles such as pedestrians and vehicles. This simplifies the integration of automated systems into railway operations and allows for smoother implementation of standard navigation technologies. However, vehicles face the challenges of higher operational speeds, necessitating robust communication to maintain safety and efficiency. Additionally, suburban environments may encounter tunnels, trenches, and vegetation, which introduce further challenges for perception and navigation in autonomous railway maintenance systems.

Rural/outdoor railway scenarios

The rural (outdoor) railway setting presents unique challenges and advantages for autonomous navigation. These environments typically feature wide-open spaces that can facilitate reliable GPS signal reception with fewer obstructions. On the other hand, the lack of infrastructure, such as communication towers, can lead to connectivity issues, which are crucial for tasks which require network accessibility, such as remote monitoring, and even the vehicle can achieve full autonomy. Natural obstructions such as forests and vegetation can further complicate signal propagation and navigation. For instance, research highlights the impact of forest stand characteristics and topography on GNSS, Radio Frequency (RF) transponder accuracy and radio signal propagation, emphasising the need for sophisticated equipment to handle these environmental variables [52].

Additionally, rural railways require advanced detection systems to safely manage wildlife crossings, ensuring both wildlife conservation and vehicle safety. Vegetated

terrain introduces another degree of complexity: what is considered an obstacle from a purely geometric point of view may not represent a danger for the vehicle if it is composed of compressible vegetation, such as a tuft of tall grass or a small bush [53].

Tunnel scenarios

Navigating railway tunnels presents distinct challenges for communication and sensor technology due to the enclosed, often signal-restrictive environment. Alternative localisation methods, such as inertial measurement units, are required due to the lack of a GPS signal. The vehicle must be robust enough to handle the high levels of noise due to the unique acoustic properties of tunnels. Electromagnetic interference and reflective surfaces within tunnels can disrupt standard radio and signal transmissions, complicating train control and monitoring processes [54]. Furthermore, Adequate lighting is crucial for ensuring comfort and safety in road tunnels. High-efficiency, energy-saving lighting systems are necessary to provide a reliable visual environment [55], both for operator monitoring and object detection algorithms.

2.3.4 Summary and domain-specific considerations

The majority of the literature reviewed in this chapter originates from the automotive domain and focuses on foundational components of perception systems — such as object detection, multi-object tracking, and depth estimation. However, many of these methods are not directly applicable to railway operations. Self-driving cars benefit from high manoeuvrability, allowing lane changes or steering-based avoidance. They are also supported by large-scale public datasets and rich semantic context, including road signs, lane markings, and traffic lights. Additionally, their deployment often assumes prior access to high-definition maps, which assist with localisation and path planning.

In contrast, railway vehicles are restricted to one-dimensional motion with no capacity for lane-changing or re-routing. As a result, collision avoidance relies entirely on high-confidence early detection and requires significantly longer stopping distances. Fur-

thermore, the surrounding railway infrastructure — such as overhead cables and gantries — can interfere with sensor systems not only by obstructing the field of view but also through electromagnetic interference. Unlike structured urban environments, many railway settings lack reliable infrastructure for GPS or high-bandwidth internet connectivity, complicating localisation and cloud-based processing. The scarcity of annotated, railway-specific datasets also presents challenges for training and evaluating machine learning-based perception algorithms. Finally, physical vibrations from the track can degrade sensor stability and introduce noise into perception pipelines.

Railway maintenance environments introduce additional complexity due to their crowded and dynamic nature. These scenes often involve multiple track workers and mobile machinery operating in proximity, and the collaborative interaction between human workers and autonomous systems further increases situational complexity. Maintenance work is frequently conducted under low-visibility conditions — including night-time, tunnels, fog, or rain — where reliable perception becomes more difficult and lighting varies unpredictably.

Despite the domain-specific differences mentioned above, the core perception tasks remain consistent across railway, automotive, and other collision avoidance systems. These systems must identify objects of interest, estimate their position relative to the vehicle, and update this information to support safe navigation and decision-making. To achieve this, they rely on key components such as object detection, multi-object tracking, and depth estimation — all of which must operate under real-time constraints and in environments affected by variable lighting, occlusions, or motion-induced sensor noise. Moreover, the growing use of deep learning, TL, and multimodal sensor fusion is common across domains, particularly in response to limited annotated data and the need for generalisation. These shared challenges justify the inclusion of related literature from the automotive and robotics fields in this review, as many of the underlying methods and architectures provide valuable technical foundations for the development of railway maintenance systems.

2.4 Learning-based object detection

2.4.1 Introduction

Generic object detection aims at determining whether or not there are instances of objects from predefined categories and estimates the 2-dimensional locations of each object instance within the image. Object detection can be grouped into two types [56, 57]: detection of broad categories and detection of specific instances. The first type detects (usually previously unseen) instances of some predefined object categories (for example, humans, cars, bicycles, and dogs). The second type can be treated as a matching problem; it aims to detect instances of a particular object (such as a specific car model). For context-aware collision avoidance in railway maintenance vehicles, object detection focuses on the first type to detect multiple objects, therefore enabling the system to respond intelligently, such as an emergency stop for detected trespassers trespasser and slow down for detected maintenance workers. If an object does not belong to the predefined categories, the system relies on a secondary mechanism by employing traditional start-stop collision avoidance without context awareness ability.

Object detection has undergone significant transformations, from earlier techniques based on hand-crafted features to sophisticated deep learning models that leverage large datasets and powerful computational resources [58, 59]. Initially, object detection relied on methods such as template matching and feature-based approaches, where the detection was performed using simple geometric and texture features. These methods were generally limited to detecting objects under constrained environments with minimal variation in appearance and pose.

With the advent of machine learning, particularly the development of support vector machines and decision trees, object detection strategies became more robust, allowing for some degree of variability in object appearances [60]. However, these approaches still need to work on scalability and require careful feature engineering to be effective [61, 6].

The breakthrough came with the introduction of deep learning, particularly Convolutional

Neural Network (CNN), which transformed the landscape of object detection [62]. Deep learning models, trained end-to-end, learn to extract features automatically from vast amounts of data, significantly improving the detection performance across diverse scenarios [59, 63]. This shift increased detection accuracy and enhanced the ability of systems to generalise across different object categories and environments [61, 9].

In the deep learning era, object detection algorithms can be classified based on the number of stages: two-stage (region-based, see **Section 2.4.2**) and one-stage (unified, see **Section 2.4.4**) framework. As the name suggests, the two-stage approach is implemented in a two-step manner. The first stage generates a set of candidate regions (or object proposals) that should contain all objects while filtering out most negative locations; the second stage determines the object region, classifying and regressing them into object categories or backgrounds with Deep Neural Networks (DNN). The two-step process matches the attentional mechanism of the human brain to some extent, which gives a scan of the whole scenario first and then focuses on Region of Interest (RoI) [58]. The region proposal-based methods mainly include R-CNN, SPP-Net, Fast R-CNN, Faster R-CNN, R-Fully Convolutional Networks (FCN), Mask R-CNN and Feature Pyramid Networks (FPN). Some of these are correlated with each other (e.g., SPP-net modifies R-CNN with an Spatial Pyramid Pooling (SPP) layer).

In contrast, a one-stage detector directly predicts class probabilities and bounding box offsets from full images with a single feed-forward CNN that does not involve region proposal generation, post-classification, or feature re-sampling, encapsulating all computation in a single network and trains end-to-end directly. Since the whole pipeline is a single network, it can be optimised end-to-end directly on detection performance [64]. One-stage detectors are applied over a regular, dense sampling of object locations, scales, and aspect ratios. Due to the high efficiency, the one-stage approach has attracted much more attention in recent years [65, 66], Such as You Only Look Once (YOLO), Single Shot MultiBox Detector (SSD), OVERFeat, MultiBox, CornerNet and RetinaNet.

2.4.2 Two-stage (region-based) detection algorithms

R-CNN

R-CNN [67] were among the first to explore CNN for generic object detection. R-CNN uses region proposals instead of sliding windows to find objects in images by integrating AlexNet [68]. It first generates region proposals (bounding boxes) by **Selective Search** [69, 70]. Then, each proposal is rescaled to a fixed size for deep feature extraction and classification. Next, class-specific Support Vector Machines (SVM) classifiers are trained on features extracted from the CNN, replacing the softmax classifier learned by fine-tuning. Lastly, post-processing is used to refine the bounding boxes, eliminate duplicate detections, and re-score the boxes based on other objects in the scene.

R-CNN pros: R-CNN offer several advantages for object detection compared to CNN. The use of selective region proposals allows for better object localisation and segmentation, while shared convolutional layers reduce computational overhead, leading to faster and better detection accuracy [64, 71, 72]. Additionally, R-CNN reuses features by pre-train CNN on external datasets, then fine-tunes for specific detection tasks, offering better performance even with limited labelled dataset [72].

R-CNN cons: despite its success, R-CNNs have few drawbacks. First, R-CNN mainly plays as a classifier and does not predict object bounds (except for refining by bounding box regression). Its accuracy depends on the performance of the region proposal module [71]. Second, the training of R-CNN is slow and difficult because each stage must be trained separately in its multi-stage pipeline configuration [58, 64, 73, 74]. Third, although **Selective Search** can generate region proposals with relatively high recalls, the obtained region proposals are still redundant, and this procedure is time-consuming [58, 59, 75]. Lastly, R-CNN is also inefficient due to the fixed size input [58] and slow to test [64, 73].

SPPNet

The main bottleneck of R-CNN is that it requires extraction of CNN features from thousands of warped region proposals per image [64]. SPPNet [76] was introduced to improve R-CNN by combining traditional Spatial Pyramid Pooling (SPPNet) into CNN architecture [77, 78]. SPPNet enables CNN to generate a fixed-length representation regardless of the size of the image/ROI without rescaling it. SPPNet reuses the feature map to the project region proposal by only running the convolution layer once on the image. Then fixed-length representations of arbitrary regions can be generated for training the detectors, which avoids repeatedly computing the convolutional features [59]. In other words, SPPNet performs some information aggregation at a deeper stage of the network hierarchy (between convolutional layers and fully connected layers) to avoid the need for cropping or warping at the beginning [76]. The feasibility of the reusability of these feature maps is because they involve the strength of local responses and have relationships with their spatial positions [76].

SPPNet pros: compared to R-CNN, SPPNet achieves better efficiency by sharing the computation cost before the SPPNet layer among different proposals. It improves the detection performance with a correct estimation of different region proposals in their corresponding scale [58]. Moreover, SPPNet can generate a fixed-length representation regardless of input image size/scale. Pyramid pooling is also robust to object deformations [76]

SPPNet cons: however, SPPNet takes almost the same pipeline as R-CNN, but it still faces the drawbacks of multi-stage training [59]. In addition, SPPNet only fine-tunes its fully connected layer and cannot update the convolutional layers before the SPPNet layer, which limits the accuracy of very deep networks [58, 64]. Fast R-CNN [79] was proposed and solved these problems [59].

Fast R-CNN

Fast R-CNN [79] enables end-to-end training by streamlining the training with simultaneous learning of the softmax classifier and class-specific bounding box regression. The whole image is processed with a convolutional layer to produce feature maps, compared to multi-stage training in R-CNN. Fast R-CNN employs the idea of sharing convolution computation among region proposals. Subsequently, a RoI pooling layer is added between the last convolution layer and the first Fully Connected (FC) layer to extract fixed-length features for each region proposal.

The RoI pooling layer can be treated as a variant of the SPP layer, which has only one pyramid level; it uses feature-level warping to approximate warping at the image level. After the RoI pooling layer, the feature is fed into a sequence of FC layers. These branches into two sibling output layers: softmax probability for object category prediction and class-specific bounding box regression offsets for proposal refinement. All parameters in these procedures (except the generation of region proposals) are optimised via a multi-task loss in an end-to-end way [58].

Fast R-CNN pros: compared to R-CNN/SPP-Net, Fast R-CNN improves the efficiency and accuracy by developing the RoI-pooling layer to extract features from shared convolutional feature maps [64, 73, 75, 80]. In addition, the one-stage training pipeline also improves efficiency and accuracy by reducing the storage required for feature caching [58, 64].

Fast R-CNN cons: despite Fast R-CNN significantly speeding up the detection process, it still relies on additional methods, such as **Selective Search**, to generate external region proposals and poses as the bottleneck to improve efficiency [58, 59, 64, 75, 81, 82], for example, **Selective Search** needs 2 seconds to process one image [70]. Furthermore, the last layer output of a very deep CNN tends to be coarse, and neighbouring regions may overlap each other seriously; this results in Fast R-CNN struggles with small objects on

PASCAL VOC datasets and is most commonly performed on a single-scale feature map [75, 83].

Meanwhile, research [84, 85, 86] showed the feasibility of accurate detection in the convolutional layer rather than the FC layer for a CNN-based detector. Subsequently, Faster R-CNN [71, 87] was proposed by introducing a Region Proposal Network (RPN) instead of **Selective Search**.

Faster R-CNN

Faster R-CNN shares the same backbone network with Fast R-CNN and contains two modules: RPN for the task of region proposal and the Fast R-CNN detector for region classification. The entire system is a unified network for object detection with ‘attention’ mechanisms; the RPN module tells the Fast R-CNN module where to look [71]. A training scheme is proposed to unite RPNs with Fast R-CNN, alternating between fine-tuning for the region proposal task and fine-tuning for object detection while keeping the proposals fixed. This scheme converges quickly and produces a unified network with convolutional features shared between both tasks [71].

The RPN is a deep FC network that can take input images of arbitrary sizes. It simultaneously predicts object bounds and objectness scores at each position. It is trained end-to-end to output a set of rectangular object proposals, which Fast R-CNN uses for detection. RPN acts in a nearly cost-free way by sharing full-image convolutional features with the detection network. RPN operates on a specific convolutional layer with the preceding layers shared with the object detection network [58].

Faster R-CNN pros: the main contribution of Faster R-CNN is the introduction of RPN, which enables nearly cost-free region proposals. Like Fast R-CNN, Faster R-CNN also benefits from the one-stage end-to-end training pipeline [59], which leads to fast convergence. Furthermore, RPNs are designed to efficiently predict region proposals with a wide range of scales and aspect ratios [71].

Faster R-CNN cons: however, there is still computation redundancy at subsequent detection stage [59]; The alternate training algorithm is very time-consuming, and RPN produces object-like regions (including backgrounds) instead of object instances [58]; Lastly, Faster R-CNN struggles with extreme scales or shapes, such as small-size object detection and precise localisation, mainly due to the coarseness of its feature maps and limited information provided in particular candidate boxes [58, 75].

Mask R-CNN

Instance Segmentation can be regarded as two interrelated tasks of detection and segmentation, and much effort is proposed to solve these two problems in a unified stage [88, 89]. The multi-task scheme will create a spurious edge and exhibit systematic errors on overlapping instances [90]. The Mask R-CNN [91] is proposed to tackle pixel-wise object instance segmentation by extending Faster R-CNN.

Mask R-CNN adopts the same two-stage pipeline, with an identical first stage (RPN), but in the second stage, in parallel to predicting the class and box offset, Mask R-CNN adds a branch which outputs a binary mask for each RoI. The new branch is an FCN on top of a CNN feature map. To avoid the misalignments caused by the original RoI pooling layer, a RoI align layer was proposed to preserve the pixel-level spatial correspondence. This fully convolutional representation requires fewer parameters but is more accurate than that in [92].

Mask R-CNN pros: Mask R-CNN achieved top results for the COCO object instance segmentation and bounding box object detection. It is simple to train, generalises well, and adds only a small overhead to Faster R-CNN, running at five Frames Per Second (FPS) [64]. Another advantage of Mask R-CNN is the ease of generalisation of other related tasks. For example, Mask R-CNN allows the estimation of human poses in a similar environment [73]. It is a flexible and efficient framework for instance-level recognition, which can be easily generalised to other tasks with minimal modification [58].

As demonstrated by the authors, the model outperformed other state-of-the-art models on each COCO task for the dataset’s 2016 challenge.

R-FCN

Divided by the RoI pooling layer, Fast/Faster R-CNNs comprise two subnetworks: a shared fully convolutional subnetwork (independent of RoIs) and an unshared RoI-wise subnetwork. This decomposition originates from pioneering classification architectures, which consist of a convolutional subnetwork and several FC layers separated by a specific spatial pooling layer [58]. However, this naive design has considerably inferior detection accuracy, conjecturing that deeper convolutional layers are more sensitive to category semantics and less sensitive to translation. In contrast, object detection needs localisation representations that respect translation invariance [64].

R-FCN differs from Faster R-CNN only in the RoI sub-network. In Faster R-CNN, the computation after the RoI pooling layer cannot be shared, so R-FCN [93] is proposed using all convolutional layers to construct a shared RoI sub-network and RoI crops are taken from the last layer of convolutional features before prediction. A position-sensitive score map is proposed to address a dilemma between translation-invariance in image classification and translation-variance in object detection. Thus, naturally adopt fully convolutional image classifier backbones, such as the ResNets [94], for object detection.

R-FCN pros: with R-FCN, more powerful classification networks can be adopted to accomplish object detection in a fully convolutional architecture by sharing nearly all the layers. The state-of-the-art results are obtained on both PASCAL VOC and Microsoft COCO [37] data sets at a test speed of 170 ms per image [58]. R-FCN could achieve comparable accuracy to Fast R-CNN with higher efficiency [93].

FPN

Image pyramids have been widely applied in many object detection systems to extract features with scale invariance [95, 96, 97, 98]. However, training time and memory consumption increase rapidly. To this end, some techniques take only a single input scale to represent high-level semantics and improve the robustness of scale changes. Image pyramids are built at test time, which results in an inconsistency between train/test-time inferences. The in-network feature hierarchy in a deep ConvNet produces feature maps of different spatial resolutions while introducing large semantic gaps caused by different depths. To avoid using low-level features, pioneer works [99, 100] usually build the pyramid starting from middle layers or just sum transformed feature responses, missing the higher resolution maps of the feature hierarchy [58].

Different from these approaches, FPN [83] holds an architecture with a Bottom-Up pathway, a Top-Down pathway and several lateral connections to combine low-resolution and semantically robust features with high-resolution and semantically weak features. The Bottom-Up pathway, the basic forward backbone ConvNet, produces a feature hierarchy by downsampling the corresponding feature maps with a stride of 2. The layers of output maps of the same size are grouped into the same network stage, and the output of the last layer of each stage is chosen as the reference set of feature maps for the Top-Down pathway. Feature maps from higher network stages are upsampled at first and then enhanced with those of the same spatial size from the BU pathway via lateral connections to build the Top-Down pathway, [58].

FPN pros: FPNs benefit from image pyramid representation, as it not only allows arbitrary input scale but also further boost the detection performance [76, 83, 97, 101]; This is due to its principle advantage of featuring each level of an image pyramid is that it produces a multi-scale feature representation in which all levels are semantically strong, including the high-resolution levels [83, 97]. Moreover, RPN is a sliding window detector with a fixed window size, so scanning over pyramid levels can increase its robustness to

scale variance [83]. Lastly, FPN is independent of the backbone CNN architectures and can be applied to different stages of object detection (e.g., region proposal generation) and many other computer vision tasks (e.g., instance segmentation) [58, 83].

FPN cons: nevertheless, featuring each level of an image pyramid has apparent limitations. Inference time increases considerably, making this approach impractical for real applications. Moreover, training deep networks end-to-end on an image pyramid is infeasible in terms of memory. So, if exploited, image pyramids are used only at test time, which creates an inconsistency between train/test-time inference. For these reasons, Fast and Faster R-CNN opt not to use featured image pyramids under default settings [83].

2.4.3 Conclusion on two-stage detection algorithms

Summary of technical evolution

The evolution of two-stage detection methods has significantly advanced object detection. Beginning with R-CNN [67], these methods have utilised CNN to classify region proposals, fundamentally changing the approach of object detection algorithms. Successive developments have included:

- **Fast R-CNN** [79] simplified the training process by integrating classification and bounding box regression within a single network.
- **Faster R-CNN** [87] introduced a RPN, eliminating the need for separate proposal generation and enabling nearly real-time detection by sharing features across the network.
- **R-FCN** [93] and **Mask R-CNN** [91] refined feature extraction and region proposal classification, enhancing feature pooling techniques and reducing computational redundancy.

Advancements in RRN: the RPN has evolved to balance the need for informative features and reasonable resolution layers for proposal generation [75]. Adjustments to RPNs and detection heads in frameworks like Faster R-CNN and Mask R-CNN have improved the quality of the proposals and the detection results, enabling a more refined localisation and classification of objects [64].

Fast R-CNN significantly improves the speed of object detection compared to earlier methods; however, it still relies on **Selective Search** for region proposal generation, which is a major bottleneck due to its time-consuming bottom-up proposal process, taking approximately 2 seconds per image [58, 59, 64, 75, 81, 82]. Additionally, the coarse output of the final layer in deep CNN architectures and the overlapping of neighbouring regions hinder its ability to detect small objects effectively, as observed in datasets like PASCAL VOC [75, 83]. Research has highlighted the potential for accurate detection in the convolutional layers rather than fully connected layers, paving the way for the development of Faster R-CNN [84, 85, 86]. Faster R-CNN replaces **Selective Search** with a RPN, eliminating the external proposal generation process and significantly enhancing detection efficiency and performance [71, 87].

Challenges and future improvement

Despite their successes, two-stage detection algorithms face ongoing challenges, such as high computational demands and limitations in scale variance, and they are typically slower than one-stage detectors like YOLO and SSD, which do not involve preprocessing algorithms [64]. The integration of Light-Head R-CNN [102] aimed to reduce the computational load using a lightweight detection head. However, challenges with small backbones leading to imbalances and potential overfitting remain. While effective in identifying objects with high recall, the deep convolutional layers often perform poorly in localisation due to the coarseness of the feature maps [75]. Other strategies to mitigate these limitations include optimising the feature extractor or backbone network, which is the most time-consuming step [103].

Future improvements in two-stage detection methods are likely to focus on several key areas as the field continues to evolve towards more sophisticated and adaptable models:

- Reducing dependency on large annotated datasets through unsupervised and semi-supervised learning methods, addressing the scalability and resource-efficiency of training deep learning models [58].
- Developing more efficient network architectures that can process high-resolution inputs more rapidly, enhancing the speed and accuracy of object detection in diverse settings [73].
- Enhancing the robustness and versatility of detection models through cross-domain applications could improve generalisation across different types of data and environments [63].
- Further integration of R-CNN with newer or different types of neural networks, such as transformer models, could enhance object detection capabilities. Attention mechanisms could also improve the focus on relevant features of the input data, addressing real-time detection challenges where speed is crucial [64].

2.4.4 One-stage (unified) detection algorithms

OverFeat

OverFeat [104] were among the first few approaches to utilise a one-stage pipeline for object detection. It trains a convolutional network to perform localisation and adapt that localiser for detection. A fully connected layer is trained to predict the box coordinates for the localisation task that assumes a single object. The fully-connected layer is then turned into a convolutional layer for detecting multiple class-specific objects [71].

OverFeat pros: OverFeat has a significant speed advantage but is less accurate than R-CNN because training fully convolutional networks takes longer. The speed advantage

derives from sharing the convolution computation between overlapping windows in the fully convolutional network. OverFeat is similar to later frameworks such as YOLO and SSD, except that the classifier and the regressors in OverFeat are trained sequentially.

OverFeat cons: OverFeat performs sliding window detection efficiently but must still be more cohesive. OverFeat optimises for localisation, not detection performance. Like Deformable Parts Model (DPM), the localiser only sees local information when making a prediction. OverFeat cannot reason about global context and thus requires significant post-processing to produce coherent detections [74].

YOLO

YOLO [74] follows a different philosophy compared to the *proposal detection + verification* paradigm. YOLO unifies the separate object detection components into a single neural network, simultaneously predicting the bounding box and class probabilities for each bounding box. Unlike two-stage-based approaches that predict detection based on features from a local region, YOLO dropped the regional proposal generation stage; it trains all images with a loss function in an end-to-end manner to optimise detection performance directly.

YOLO pros: the main advantage of YOLO is its fast performance; it achieves real-time performance at 45 FPS while achieving comparable accuracy to other real-time systems. Fast YOLO- a smaller version of YOLO- processes at 155 FPS while still achieving double the mAP(Mean Average Precision) of other real-time detectors [74]; This is due to the single pipeline design, where the training can be processed in an end-to-end manner with global features; Furthermore, Since YOLO sees the entire image when making predictions, it implicitly encodes contextual information about object classes and is less likely to predict false positives in the background [64].

YOLO cons: however, compared to two-stage detection systems (such as Fast R-CNN), YOLO lags in accuracy because it makes more localisation errors (especially for small objects) [59, 64, 74]; This is due to few factors [74]: 1) The spatial constraint limits the number of nearby objects that our model can predict. 2) It struggles to generalise to objects in new or unusual aspect ratios or configurations. YOLO also uses relatively coarse features for predicting bounding boxes since the architecture has multiple downsampling layers from the input image. 3) The loss function treats errors similarly in small and large bounding boxes. A small error in a large box is generally benign, but a small one has a much more significant effect on Intersection over Union (IoU). The primary source of error is incorrect localisations.

SSD

In order to preserve real-time performance without sacrificing too much accuracy, SSD [99] is proposed by combining the ideas from RPN, YOLO and multi-scale convolutional features. Like YOLO, SSD predicts a fixed number of bounding boxes and scores, followed by an Non-Maximum Suppression (NMS) step to produce the final detection. SSD use ConvNet's pyramidal feature hierarchy by utilising a VGG16 backbone architecture, which would reuse the multi-scale feature maps from different layers computed in the forward pass and thus come free of cost [83]. Then, SSD adds several feature layers to the end of the network, which predict the offsets to default boxes with different scales and aspect ratios and their associated confidences. The network is trained with a weighted sum of localisation loss and confidence loss. Final detection results are obtained by conducting NMS on multi-scale refined bounding boxes [58].

SSD pros: the main contribution of SSD is its well-balanced detection speed and accuracy [58, 59]; this is due to the one-stage pipeline and the integration of multi-feature layers and multi-resolution detection techniques [59, 97] to handle objects with various sizes naturally. Another advantage is the visual perception in autonomous navigation sys-

tems; it favours overall scene understanding over the correctness of object classification; for instance, an advertisement with a person painted on a bus should be classified as a bus instead of a pedestrian [105].

SSD cons: however, SSD fails to capture deep semantics for shallow-layer feature maps since the bottom-up pathway in SSD can learn robust features only for deep layers but not for the shallow ones; it misses the opportunity to reuse the higher-resolution maps of the feature hierarchy [83]; which caused SSD struggling with detection small objects [58, 83, 97].

RetinaNet

The design of the RetinaNet detector shares many similarities with previous dense detectors, particularly the concept of ‘anchors’ introduced by RPN and the use of features pyramids as in SSD and FPN. RetinaNet is a single, unified network comprising a backbone network and two task-specific subnetworks. The backbone is responsible for computing a convolutional feature map over an entire input image and is an off-the-self convolutional network. The first subnet performs convolutional object classification on the backbone’s output; the second subnet performs convolutional bounding box regression. The two subnetworks feature a simple design explicitly proposed for one-stage, dense detection [106]. The author claimed that the extreme foreground-background class imbalance encountered during the training of dense detectors is the central cause. To this end, a new loss function named “focal loss” has been introduced in RetinaNet by reshaping the standard cross entropy loss so that the detector will focus more on complex, misclassified examples during training [59].

MultiBox

The MultiBox methods generate region proposals from a network whose last fully connected layer simultaneously predicts multiple class-agnostic boxes, generalising the *single*

box fashion of OverFeat. These class-agnostic boxes are used as proposals for R-CNN. In contrast to a fully convolutional scheme, the MultiBox proposal network is applied to a single image crop or multiple large image crops. MultiBox does not share features between the proposal and detection networks [71].

Deep MultiBox: Unlike R-CNN, a convolutional neural network was trained to predict regions of interest instead of relying on **Selective Search** [107]. MultiBox can also perform single object detection by replacing the confidence prediction with a single class prediction. However, MultiBox cannot perform general object detection and is still just a piece in a more extensive detection pipeline, requiring further image patch classification [74].

CornerNet

The use of anchor boxes in one-stage detectors has been critiqued for causing a significant imbalance between positive and negative examples, slowing down the training process, and introducing additional hyper-parameters [103]. A novel approach was introduced inspired by associative embedding techniques used in multi-person pose estimation. Bounding box object detection is formulated to detect paired top-left and bottom-right vital points. This method, implemented using two stacked Hourglass networks with a corner pooling strategy to enhance corner localisation, achieved a 42.1% AP on the MS COCO dataset, surpassing all previous one-stage detectors, albeit with a lower average inference speed of about 4 FPS on a Titan X Graphics Processing Unit (GPU). The main limitation of this approach is the difficulty in correctly pairing key points to form bounding boxes. To address this, a subsequent improvement introduced an additional critical point at the centre of a proposal to detect objects as a triplet of essential points, which raised the MS COCO AP to 47.0%. However, this modification resulted in even slower inference speeds than its predecessor [64].

DetectorNet

DetectorNet was among the first to explore CNN for object detection [108]. DetectorNet formulated object detection as a regression problem for object-bound box masks. They use AlexNet and replace the final softmax classifier layer with a regression layer. Given an image window, they use one network to predict foreground pixels over a coarse grid and four additional networks to predict the object's top, bottom, left and right halves. A grouping process then converts the predicted masks into detected bounding boxes. The network needs to be trained per object type and mask type and should not scale to multiple classes. DetectorNet must take many crops of the image and run multiple networks for each part on every crop, thus making it slow [64].

2.4.5 Conclusion on one-stage detection methods

Summary of technical evolution

One-stage detection frameworks have advanced object detection by simplifying the detection pipeline and improving processing speed. These models unify object localisation and classification into a single end-to-end trainable network, which leads to gains in efficiency and real-time detection capabilities. OverFeat [104] pioneered this approach by integrating localisation and detection into a unified framework. However, OverFeat is optimised for localisation, and the detection performance is undesirable. This innovation paved the way for frameworks like YOLO [74] and SSD [99], which further streamlined detection by incorporating end-to-end training and multi-scale feature utilisation, respectively. Innovations like focal loss in RetinaNet [106] and anchor-free detection in CornerNet [103] represent significant steps forward in addressing the inherent limitations of one-stage frameworks. These innovations aim to refine detection accuracy while maintaining the speed advantages. As seen in CornerNet [103] and CenterNet [109], the evolution towards anchor-free methods indicates a shift towards more flexible and potentially more accurate detection mechanisms.

Advantages and challenges

The primary advantage of one-stage detection algorithms lies in their efficiency. Frameworks like YOLO and SSD achieve real-time performance, making them ideal for applications requiring rapid decision-making, such as autonomous railway navigation. YOLO, for example, can process up to 45 frames per second (FPS) while maintaining competitive accuracy, and its global feature analysis reduces false positives in background regions. Similarly, SSD's multi-scale detection approach enhances its versatility for detecting objects of various sizes.

However, these methods have notable limitations. YOLO struggles with small object detection due to coarse feature representation and spatial constraints, which limit its ability to localise closely packed objects accurately [74]. SSD, while improving scale invariance, suffers from insufficient utilisation of high-resolution features, making it less effective for small objects [99]. Despite achieving state-of-the-art accuracy, CornerNet exhibits slow inference speeds, limiting its practicality for real-time systems [103]. These constraints highlight the trade-offs in one-stage detection architecture, where speed often comes at the expense of detection precision, especially for complex scenes involving occlusions or high object density.

Future improvement

The future of one-stage detection is likely to involve continued improvements in network architectures and training strategies. Emphasising feature integration across different scales and enhancing the network's ability to understand contextual and spatial relationships within an image could improve the speed and accuracy of one-stage detectors. Additionally, integrating advances from other areas of deep learning, such as attention mechanisms and transformers, could provide new pathways to overcome current limitations.

2.4.6 Discussion on learning-based object detection

Two-stage vs. one-stage

As discussed in Section 2.4.2 and 2.4.4, detection can be categorised into two-stage and one-stage based on the network pipeline. Two-stage approaches first generate a sparse set of candidate proposals that should contain all objects while filtering out the majority of negative locations, and the second stage classifies the proposals into foreground classes. The two-stage approaches include R-CNN, SPP-Net, Fast R-CNN, Faster R-CNN, R-FCN, Mask R-CNN and FPN. These two-stage methods offer several advantages: they utilise a two-stage structure complete with sampling heuristics that effectively handle class imbalance; they apply a two-step cascade approach for regressing the object box parameters; they employ features at two-stages to provide a more descriptive representation of the objects; they use two-step regression to predict the locations and sizes of the object boxes; and they incorporate RoI-wise operations to align features used for detection, enhancing the precision of the detection process [65, 66]. State-of-the-art two-stage detectors typically employ a robust detection mechanism that often results in better accuracy compared to their one-stage counterparts, albeit at a higher computational cost [66, 73, 102, 106].

In contrast, one-stage approaches use convolution layers to directly predict bounding boxes and class probabilities by regular and dense sampling over locations, scales and aspect ratios. The detection part of this category is composed of the additional layers to generate predictions; due to the single pipeline configuration, it can be optimised end-to-end directly on detection performance; thus, it is highly computationally efficient and has the potential for real-time applications compared to two-stage networks [110]. However, the detection accuracy of one-stage detectors is usually worse than two-stage methods. One of the main reasons is due to the class imbalance problem [65, 106]. Some recent works in the one-stage approach aim to solve the class imbalance problem to improve detection accuracy. Reverse Connection With Objectness Prior Networks (RON) [111]

significantly reduces the search space of objects via using the objectness prior constraint on convolutional feature maps. RetinaNet [106] reshapes the standard cross entropy loss to focus training on a sparse set of complex examples and down-weights the loss assigned to well-classified examples to address the class imbalance issue.

Real-time performance

Real-time performance is a critical challenge for object detection, particularly when immediate response and decision-making are required, such as in autonomous driving. Real-time performance often involves a trade-off between detection accuracy and the speed of processing. One-stage detection frameworks such as SSD detectors offer a boost in speed, allowing for near real-time performance; however, they may compromise on detection quality compared to two-stage models [63]. Advanced methods aim to optimise feature extraction and utilise cascade learning to streamline processing without severely impacting accuracy [82]. Furthermore, solutions for model compression shared feature maps in models like Faster R-CNN and effective architectural design, like lightweight neural networks, are essential in lowering computational needs [58]. These methods facilitate retaining high accuracy while reducing processing load and enabling faster reaction times. However, the problem still exists, particularly when scaling to high-resolution inputs or handling extensive object categories in dynamic environments. Future research should focus on improving the efficiency of these detection systems, potentially through more sophisticated machine learning algorithms, to better balance speed and accuracy. This will broaden the applicability of real-time object detection technologies in practical, real-world applications [73, 102].

Limited dataset

The performance of deep detectors relies heavily on extensive training data and large networks with millions or even billions of parameters. The availability of GPUs with high computational capabilities and large-scale detection datasets, such as ImageNet and Mi-

Microsoft COCO, often plays a crucial role in their effectiveness. These large datasets facilitate tackling more realistic and complex problems, characterised by significant intra-class variations and inter-class similarities. However, it is labour-intensive and time-consuming to prepare a large annotated dataset for training and poses the challenge of learning with a smaller training dataset [64].

Data augmentation is widely recognised as crucial for learning CNN and enhancing visual recognition performance [67, 79]. It involves perturbing images in ways that do not change the underlying category, such as cropping, flipping, rotating, scaling, translating, colour perturbations, and adding noise [64]. Data augmentation helps to reduce overfitting and improves generalisation by artificially increasing the number of training samples. It can be used for training or testing. Data augmentation may generate totally new training images, ensuring that these synthetic images generalise effectively to real-world circumstances remains challenging [112]. Researchers have recommended enriching datasets by pasting genuine segmented objects into natural photos to improve contextual comprehension, which is critical for placing objects accurately throughout scenes [113, 114].

Class imbalance

Class imbalance refers to the disproportionate representation of different classes within a dataset, where some classes have many more examples than others. This imbalance can significantly degrade the performance of detection algorithms, causing them to favour the majority class and perform poorly on the minority class.

In two-stage detectors like R-CNN and its variants, class imbalance is managed using a combination of a two-stage cascade and sampling heuristics. The proposal stage, using mechanisms like **Selective Search**, rapidly reduces the number of candidate object locations to a small number by filtering out most background samples. During the second classification stage, sampling heuristics such as a fixed foreground-to-background ratio or Online Hard Example Mining (OHEM) are employed to maintain a manageable balance between foreground and background [106].

In contrast, one-stage detectors like YOLO and SSD must process a much larger set of candidate object locations that densely cover spatial positions, scales, and aspect ratios. This scenario often results in handling up to 100k locations per image, with the majority being easy negatives. While similar sampling heuristics may also be applied, these are generally less efficient, as these easily classified negatives still dominate the training process. This inefficiency is a classic problem in object detection that is typically addressed via techniques such as bootstrapping or hard example mining [106].

Occlusion

Occlusion in object detection refers to when the object of interest is covered partially or completely. In real-world images, occlusions commonly result in loss of information from object instances and make accurate detection difficult. This section discusses various approaches to address occlusion. Deformable structures have been proposed to add flexibility to fixed geometric configurations in object detection models. Deformable RoI Pooling and deformable convolution, introduced by [115], allow for more adaptive feature extraction in occluded scenes. Author [112] suggests training an adversarial network to generate examples with occlusions, enhancing model robustness. Despite significant advancements, fully resolving occlusions remains a complex challenge, with Generative Adversarial Networks (GANs) proposed as a promising direction for future research [64].

Handling occlusions remains one of the most challenging aspects of object detection in complex environments. While the discussed strategies provide significant improvements, ongoing research continues to seek more effective solutions to ensure reliable object detection and tracking under various occluded scenarios.

Environmental challenges

Object detection algorithms face various challenges in complex real-world environments. These challenges include non-rigid object deformation, where different parts of an object may move independently, and camera motion, which changes the dynamics of back-

ground and foreground, complicating detection processes. Environmental factors such as changing illumination, shadows, and complex backgrounds interfere with the performance of detection algorithms. Rapid changes in lighting and overlapping shadows can drastically affect algorithms that depend on consistent visual appearances for object detection. These factors are compounded in situations with occlusions or sudden object movements, leading to potential detection failures or false positives [116, 117].

Compared to traditional object detection algorithms. Learning-based approaches offer inherent robustness to such variations. These methods leverage large volumes of data to learn distinguishing complex patterns and adapt to new, unforeseen conditions, thereby addressing some of the limitations faced by classical detection algorithms. However, they typically require significant computational resources and extensively annotated datasets to achieve high accuracy [64].

Object detection for railway maintenance environment

Object detection in railway maintenance environments faces several challenges, many of which are common to general object detection scenarios. These include class imbalance, where some objects appear far more frequently than others. Additionally, issues like occlusions, real-time processing, and environmental challenges such as different weather and complex backgrounds have been discussed in the broader context of object detection. One of the most critical challenges is dataset scarcity, as there are not enough annotated datasets for railway object detection, particularly those relevant to maintenance scenarios (For example, RailSem19 focuses on railway infrastructure objects, with less attention on maintenance-related objects). These challenges limit model training and the ability to generalise effectively to real-world applications.

Beyond these general challenges, object detection in railway maintenance environments faces unique challenges. Dynamic and complex backgrounds featuring elements like moving workers and other maintenance equipment can restrict detection performance and result in more false positives and missed detections. Motion blur and vibration caused

by the railway vehicle's motion on railway tracks also affect camera performance and complicate detection efforts. Additionally, railway maintenance tasks often occur at night, when low-light conditions could impact the camera for object detection. This challenge is further worsened by the UK's unpredictable weather, with frequent rain, fog, and sudden lighting changes. Lastly, the safety nature of railway operations demands high accuracy and reliability, as detection errors can lead to catastrophic consequences, which often requires the integration of multi-sensory systems, such as fusing multiple cameras or other sensors (e.g., LiDAR), which introduces additional challenges related to sensor calibration, synchronisation, and data fusion.

2.5 Object detection with transfer learning

2.5.1 Introduction

TL is a machine learning approach where knowledge learned in one source domain is utilised to enhance performance in a different but related target domain. It is beneficial when the target task has insufficient labelled training data, as it allows for exploiting large, labelled datasets from a source task to improve the learning process by transferring the learned features, patterns, and even model parameters. TL not only addresses the data scarcity problem by reducing the need for extensive training on the target task but also accelerates the model development process and improves its accuracy and generalisation capabilities. This approach is especially valuable when retraining models from scratch is impractical, as it facilitates domain adaptation and enhances model accuracy and efficiency [11, 118, 119, 120, 121].

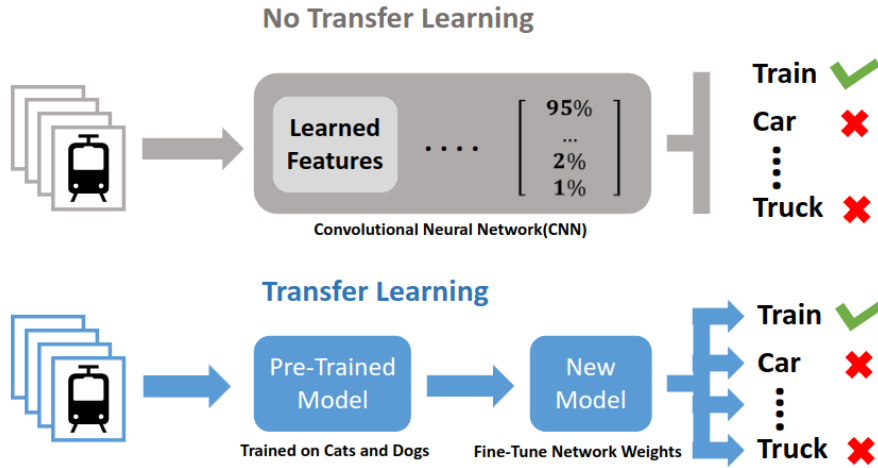


Figure 2.4: Different learning processes between (a) traditional machine learning and (b) TL

The significance of TL arises from its capabilities to address the challenges posed by traditional deep learning models, which can be challenging when collecting large amounts of labelled data for training in real-world scenarios. Compared to the rapid development in the automobile industry, the railway sector has received less attention, and there is a lack of sufficient dataset tailored for railway object detection, especially in maintenance scenarios. TL can bridge this gap. For instance, railway object detection can be improved by leveraging pre-existing models trained on sufficient data like COCO and fine-tuning with a smaller dataset, providing a solution to achieve robust object detection for autonomous railway maintenance vehicles without the need for a large, dedicated railway object dataset.

2.5.2 Transfer learning problem classification

The practical application of TL requires understanding and categorising the problems and solutions within specific contexts. There are multiple ways to classify TL; one popular approach is categorising TL based on the problem domain and solution domain [4].

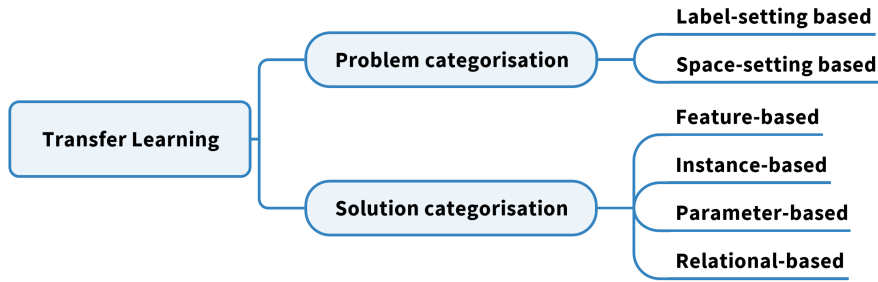


Figure 2.5: Categorisation of TL, adapted from [4].

The categorisation of TL problems is an essential aspect of the field and can help researchers better understand the challenges and opportunities of different TL scenarios. One popular approach for categorising TL problems is based on label availability (label-setting-based) and feature space similarity (space-setting-based) [4].

Label-setting-based categorisation

In label-setting-based categories, the TL problem is categorised based on the availability of labels in both the source and task domains. Inductive TL and Transductive TL differ primarily in label availability in the target domain. Inductive TL involves leveraging data from the source domain to improve performance on a related task in the target domain with label spaces. No matter whether the labels are available in the source domain or not. In contrast, Transductive TL focuses on the different tasks across domains, utilising labelled data from the source domain to the target domain without labelled data. Finally, unsupervised TL refers to where labelled data is unavailable or scarce in both source and target domains.

	Inductive TL	Transductive TL	Unsupervised TL
<i>Instance-transfer</i>	✓		
<i>Feature-representation-transfer</i>	✓	✓	
<i>Parameter-transfer</i>	✓		✓
<i>Relational-knowledge-transfer</i>	✓		

Table 2.1: Different approaches for different settings [11]

In Inductive TL, the source and target tasks differ. It is characterised by applying knowledge learned from the source domain to a different but related target task, often requiring labelled data in the target domain for effective transfer, regardless of whether the label is available in the source domain or not [122]. The focus is primarily on the target task, similar to multi-task learning, but with a concentration on the target task alone. When there is no labelled data in the source domain, inductive learning resembles self-taught learning. The key aim is to develop a target model with a small set of well-labelled data in the target domain [123]. In this type, the learning task in the target domain is different from the target task in the source domain ($T_s \neq T_t$).

Transductive TL is utilised when there is a lack of labelled data in the target domain but an abundance in the source. TL focuses on tasks where the source and target domains are different, but the tasks are the same by adapting the source domain model to fit the target domain despite their differing data distributions. The learning tasks are the same in both domains, while the source and target domains are different ($T_s = T_t, D_s \neq D_t$).

Unsupervised TL is employed in scenarios where labelled data is unavailable or scarce in both source and target domains. It leverages underlying patterns in the data to facilitate knowledge transfer across different tasks. Similar to inductive TL, it focuses on solving unsupervised learning tasks in the target domain, such as clustering, dimension reduction, and density estimation ($T_s \neq T_t$).

Space-setting-based categorisation

In space-setting-based categorisation, TL problems are distinguished based on the similarity of feature spaces between the source and target domains. Though data distributions may vary, the feature spaces are identical in Homogeneous TL. Heterogeneous TL deals with different feature spaces, necessitating methods to map or translate between these differing spaces to transfer knowledge effectively.

Homogeneous TL occurs when the feature space remains constant, but the data distribution varies between domains. This method focuses on bridging the gap in the data distributions between the domains as experienced in cross-domain transfer [118]. Homogeneous TL is defined as the scenario where the source domain (D_S) and the target domain (D_T) share the same feature space ($X_S = X_T$). Still, it may differ in their marginal probability distributions ($P(X_S) \neq P(X_T)$). This form of TL is applicable when there is an available dataset from a domain that is related to the target domain of interest, allowing the construction of a predictive model for the target domain as long as the input feature space remains the same. Homogeneous TL solutions are divided into categories such as instance-based, feature-based, parameter-based, and relational-based [124].

Heterogeneous TL deals with the case where the source and target domains are represented in different feature spaces ($X_S \neq X_T$). It encompasses methods to discover a common feature representation or to map one domain into another's feature space to enable knowledge transfer. This method requires feature and label space transformations to bridge the gap for knowledge transfer, as well as to handle the cross-domain data distribution differences [118]. It is particularly beneficial when there is a desire to utilise abundant data resources for machine learning tasks without the need to collect new data. Heterogeneous TL is valuable when there is an available dataset from a target domain of interest that has a different feature space from another target dataset [124].

2.5.3 Transfer learning approach classification

Section 2.5.2 discussed different Categories for TL problems. This section discusses different TL techniques. Broadly, TL methods can be categorised into four distinct groups: instance-based, feature-based, parameter-based, and relational-based. Instance-based methods focus on reusing the source instances that are most relevant to the target task, often by re-weighting them to reduce the distribution mismatch. Feature-based methods seek to find a common feature representation that minimises domain disparity, enhancing the

transferability of a model. Parameter-based methods involve the transfer of parameters or priors of a model from the source to the target task, which can significantly jump-start the learning process. Lastly, relational-based methods exploit the transfer of relational knowledge, capturing the dependencies or correlations among data in different domains. Each category offers unique mechanisms and strategies to overcome the challenges of TL, underscoring the versatility and adaptability of this approach in various application scenarios.

Feature-based TL

Feature-based TL aims to improve learning task performance by transferring feature representations from a source domain to a target domain [123]. This approach can be further classified into supervised and unsupervised algorithms based on the availability and type of source dataset; supervised methods utilise labelled data to learn effective feature representations for the target domain, while unsupervised techniques, which do not require labelled source data, offer solutions to more complex feature transfer challenges [123]. Feature-based approaches typically start by constructing a feature mapping between the source and target domain and leveraging the source domain's feature representation to train discriminative models for the target domain [123].

While feature-based TL has the advantage of utilising the shared feature between the domains, it also presents various challenges. The primary concern is the risk of negative transfer, where insufficient relatedness between the source and target domains can lead to performance degradation rather than improvement [125]. Additionally, pre-trained models, especially CNN trained on datasets like ImageNet, often carry inherent biases towards certain features such as textures, which may not be suitable for varied tasks or domains [125]. The process of aligning feature spaces across domains is fraught with complexities, requiring sophisticated mapping techniques that increase the TL process's complexity and computational cost [123]. The selection of appropriate "pivot features" for transfer poses another challenge, particularly with high-dimensional data and diverse domain charac-

teristics. The effectiveness of feature-based TL is also heavily dependent on the quality and quantity of data available from the source domain, where insufficient or poor-quality source data can significantly curtail the effectiveness of transferred features. Furthermore, there is a potential risk of model overfitting to the source domain's features, which may not represent the target domain's requirements, leading to poor generalisation.

Instance-based TL

Instance-based approaches focus on the selective reuse of source domain instances to improve learning in the target domain. This approach is beneficial when significant differences exist between the source and target domain data distributions. The primary strategies in instance-based TL are instance re-weighting and importance sampling, which prioritise parts of the source domain data suitable to the target domain task [11].

- Instance re-weighting modifies the influence of source domain data points based on their estimated relevance to the target domain. Techniques such as Kernel Mean Matching (KMM) and the Kullback-Leibler Importance Estimation Procedure (KLIEP) are employed to compute these relevancies, which are then used to adjust the weights of source instances in model training [11].
- Importance sampling is another critical technique in instance-based TL, where the source domain data's sampling distribution is adjusted to reflect the target domain's distribution more closely, thus addressing issues like covariate shift or sample selection bias [11].

Instance-based approaches are advantageous when the target domain has a limited dataset. They reuse instances from the source domain that are relevant to the target task. The approach is praised for its data efficiency, as it minimises the need for large-scale data collection by utilising existing datasets. Additionally, it enhances learning performance by increasing the target task with relevant information. This method also offers considerable flexibility, allowing adaptation to various domains and tasks [11].

However, it often requires a sophisticated algorithm to determine which instance from the source domain benefits the target domain. There is also a risk of negative transfer, where irrelevant source domain data can worsen performance rather than enhance it. Furthermore, balancing the selection and weighting of instances is challenging to avoid overfitting the source domain, which requires careful management to ensure effective learning without compromising the model's adaptability [11].

Parameter-based TL

Parameter-based approaches focus on optimising the target task by leveraging shared parameters between models trained on source and target domains. Compared to multi-task learning that optimises across several tasks simultaneously, parameter-based approaches focus primarily on enhancing the target task [123, 125].

This method assumes effectiveness when the source and target domains are closely aligned but diminish with more significant domain divergence, making it less suitable for domains with significant variances [123]. Practically, parameter-based TL often uses pre-trained models from large datasets like ImageNet, particularly for tasks such as object detection. These models provide a feature hierarchy that can be fine-tuned to suit the target task with potentially smaller datasets [126].

The primary advantages of the parameter-based approach include using pre-trained models and significantly reducing the resources and time required for model development from scratch. It is beneficial for tasks with limited labelled data in the target domain. Parameter-based methods such as fine-tuning on pre-trained models are well-established for object detection, taking advantage of the general feature extraction capabilities of CNN trained on extensive datasets. Such a strategy not only accelerates the deployment of new models but also enhances their robustness and ability to generalise across tasks [123, 126, 127]. Additionally, integrating regularisation techniques and selecting shared parameters can improve performance, specifically when the target dataset is small or when the tasks across domains are less similar. Regularisation-based techniques like Batch

Spectral Shrinkage and weight regularisation methods mitigate the risk of overfitting and help maintain performance even when domain similarity is low [128].

However, parameter-based approaches face challenges due to their dependency on the similarity between the source and target domains. Domain shifts can reduce effectiveness, sometimes resulting in negative transfer, where features or parameters optimal for the source task adversely affect the target task performance [123, 128]. Moreover, there is a risk of model over-specialisation; transferred parameters may not adapt well if the task similarity is low, potentially worsening performance issues. This requires careful tuning and validation to balance the benefits of TL against the risks of negative transfer. Finally, deciding which layers to transfer and the level of their adjustment involves complex decision-making. In CNN, while early layers often transfer well due to their generic nature, deeper layers typically require more substantial modification or may not be effectively transferable at all, making it critical to determine the optimal balance for freezing or fine-tuning specific layers [127, 128].

Relational-based TL

Relational-based approaches exploit the relational dynamics within and across domains, which does not require the source and target data to be independent and identically distributed, making it suitable for complex data environments where relational structures are inherent and informative. This approach utilises intra-domain and inter-domain relationships. It uses sophisticated models such as Markov Logic Networks (MLN) to understand and leverage these relationships. These networks can discern structural laws in the source domain that can be effectively applied to the target domain, thus facilitating a deeper and more contextual form of knowledge transfer.

- **Intra-domain relationships** reuses relationships from the source domain within the target domain, potentially by applying advanced MLN, which can capture and transfer structural patterns.

- **Inter-domain relationships** constructs relationships between datasets from different domains, using collaborative learning frameworks to integrate knowledge from various teacher networks. This results in a more versatile and generalised learning model.

Relational-based approaches offer a deeper comprehension of domain interactions, which is beneficial for domains where relational data is critical. Additionally, it encourages using multiple informational sources, exploiting structural similarities to improve learning outcomes. It is advantageous in social networks and bio-informatics scenarios, where the relational structure is critical to understanding and learning. The flexibility and robustness of relational-based methods make them promising for future research, especially given the adaptability required for real-world applications [123, 125, 129].

However, this approach requires a high computational resource due to the complexity of relational models and depends on domain knowledge for setting up relationships and logical rules. The success of the transfer is heavily influenced by the relevance and quality of the source domain's relational structures.

Discussion on TL approaches

TL approaches can be categorised into instance-based, feature-based, parameter-based, and relational-based approaches. Each approach presents advantages and challenges of adapting models from a source domain to a target domain [123, 11].

Instance-based approaches are good in scenarios with limited target domain data by selectively reusing relevant source domain instances [11]. This data efficiency is useful in railway object detection, where annotated datasets are scarce. However, they require sophisticated algorithms to identify relevant instances and have a high risk of negative transfer [11].

Feature-based approaches offer the advantage of learning transferable representations by aligning feature spaces between the source and target domains. This is useful for railway object detection, where domain-specific features like rail tracks and maintenance equipment differ significantly from those in general-purpose datasets. Nonetheless, the risk of negative transfer and the computational cost of aligning feature spaces necessitates careful design and implementation of these methods [123, 125].

Parameter-based approaches, such as fine-tuning pre-trained models, are beneficial in railway object detection due to their ability to leverage large-scale datasets like ImageNet for initial training. However, they face challenges in domain shifts, necessitating domain-specific fine-tuning to avoid negative transfer and over-specialisation [123, 126, 128].

Relational-based approaches provide solutions for understanding interactions and dependencies in complex railway scenarios, such as dynamic interactions between workers, vehicles, and infrastructure. While promising, the computational complexity and reliance on domain knowledge present challenges for real-time deployment in safety-critical railway environments [123, 129].

This study aims to develop an object detection system for autonomous railway maintenance vehicles and faces a multi-tiered dataset challenge:

- **Large general dataset:** a large-scale general-purpose dataset, such as COCO or ImageNet, provides a diverse set of object categories but lacks specific relevance to railway scenarios.
- **Smaller railway dataset:** a relatively smaller dataset (RailSem19) focused on railway scenarios offers domain-specific insights but primarily encompasses general railway objects, such as tracks, signals, and trains, without emphasis on maintenance-specific objects.
- **Custom railway maintenance dataset:** a small, custom dataset specifically created for railway maintenance tasks includes annotations for objects such as tools,

workers, and specialised machinery. This dataset is used to fine-tune the detection model.

Given the dataset constraints in railway object detection, parameter-based TL, such as leveraging pre-trained weights and fine-tuning, is advantageous for developing object detection systems in autonomous railway maintenance vehicles. They can transfer the knowledge learned from a large dataset like COCO to a target domain with a limited dataset. Additionally, fine-tuning enables the model to specialise in railway-specific tasks by adapting deeper layers to focus on railway maintenance-related objects, such as workers and other machinery, while retaining general features from earlier layers. This approach reduces training time and computational resources while improving model generalisation and provides an optimal solution for addressing the dataset scarcity challenges for railway object detection.

2.5.4 Negative transfer

Negative transfer occurs when knowledge transferred from a source domain negatively impacts the performance of the target task. This phenomenon typically arises when the domains involved are not sufficiently related, introducing misleading information that can degrade model performance [124, 128]. Negative transfer highlights the importance of assessing domain similarity and relevance for TL applications. It underscores the necessity of careful domain selection and alignment strategies to avoid the detrimental effects of inappropriate knowledge transfer [11, 130].

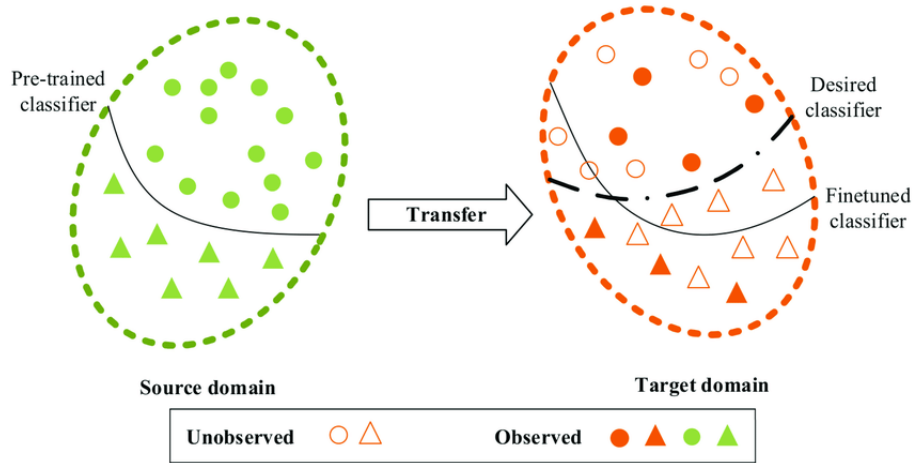


Figure 2.6: Illustration of negative transfer. The two oval dashed lines represent the distribution shape in the source and target domains. TL facilitates the training of a fine-tuned classifier in the target domain even with small samples, but distribution discrepancy may cause negative transfer, leading to poor generalisation in target tasks [5].

Causes of negative transfer

The underlying causes of negative transfer can be attributed to several factors. Specialisation of the source domain's features, inapplicability to the target task, and co-adapted neurons that do not generalise well to new domains contribute to this issue. The TL algorithm's inability to discern and align the relational structures between domains exacerbates the problem [118, 119, 120].

How to avoid negative transfer

- **Data-driven methods:** Increasing the amount of labelled target data can reduce the risk of the negative transfer by providing the model with sufficient relevant information [124].
- **Algorithmic approaches:** Employing algorithms capable of detecting and adjusting to the degree of similarity between source and target tasks can prevent negative transfer. For example, transfer-aware algorithms like the hierarchical version of the naive Bayes classifier adjust the similarity between model parameters based on the relevance to the new task [130].

- **Domain adaptation techniques:** Domain adaptation methods are explicitly designed to minimise the domain discrepancy and facilitate the transfer of only beneficial knowledge [126].
- **Selective transfer and fine-tuning:** Careful selection of what to transfer, including which layers and weights, and how to fine-tune them, is critical. For instance, freezing specific layers while retraining others can help retain beneficial knowledge while adapting to the target domain [128].
- **Regularisation strategies:** Choosing appropriate regularisation methods can prevent overfitting to the source domain's features. Techniques like L2-SP and DELTA have been proposed to preserve the generalisability of the model [128].

These strategies emphasise the nuanced application of TL, where the similarity between tasks, the amount and quality of data, and the specificity of algorithms converge to facilitate positive transfer and avoid the pitfalls of negative transfer.

2.5.5 Discussion on transfer learning

Good practices

Early investigations into deep TL have identified several best practices for enhancing performance across a range of tasks [128, 131]:

- **High-level feature reuse and low-level statistic assimilation:** the success of TL depends on how to reuse high-level features and the integration of low-level statistics. This foundational element enhances the learning process and ensures that models are not only reusing abstract concepts but also adapting to the finer statistics [131].
- **Fine-tuning pre-trained models:** fine-tuning allows models to adapt to specific tasks more effectively, especially when the target dataset is large or significantly

different from the source data, thus facilitating faster convergence and improved task-specific performance [128, 131].

- **Layer-specific transfer:** transferring knowledge from lower layers typically captures more universal features. In contrast, the upper layers contain more task-specific knowledge and are more sensitive to parameter adjustments [131]. The transfer of more layers is generally beneficial when source and target tasks are closely related. Conversely, fewer layers should be transferred if the tasks differ significantly to avoid negative transfers and enhance model adaptability [128].
- **Regularisation and fine-tuning parameters:** implementing appropriate regularisation techniques and adjusting fine-tuning parameters according to the size and nature of the target dataset is crucial for optimising TL outcomes. Techniques like Multi-tune and co-tuning can also improve accuracy and reduce training times, making them preferred methods for efficient TL [128]. Implementing appropriate regularisation techniques and adjusting fine-tuning parameters according to the size and nature of the target dataset is crucial for optimising TL outcomes. Techniques like Multi-tune and co-tuning can also improve accuracy and reduce training times, making them preferred methods for efficient TL [128].

Bad practices

While TL can significantly enhance learning efficiency and effectiveness, certain practices may lead to suboptimal results, particularly when the target dataset is large and differs substantially from the source:

- Pre-training weights may only sometimes lead to performance improvements and could result in negative transfer if the datasets are adequately aligned [128].
- Over-reliance on source domain features without sufficient adaptation can hinder model performance on the target task [128].

- There needs to be more fine-tuning or incorrect hyper-parameter settings for the learning rate, momentum, or duration before decay to prevent the model from adapting effectively to the new domain [128].
- specific regularisation techniques may only be suitable for some scenarios, mainly when the model requires the flexibility to learn from a large, diverse target dataset [128].

Future directions

- **Autonomy in learning** A significant direction for future research lies in enhancing the autonomy of TL systems. Most current TL algorithms require extensive human intervention to guide the transfer process. However, the development of autonomous TL systems has the potential to revolutionise the way knowledge is transferred across tasks. Inspiration can be taken from self-learning algorithms such as AlphaZero, which demonstrated remarkable performance in complex games without prior human understanding. Such systems could learn to adapt independently in various domains, improving over time with minimal external input [123].
- **Continuous and adversarial learning** Algorithms that support continuous learning and maintain robustness in adversarial environments represent another frontier for TL research. These algorithms would be capable of lifelong learning, adapting to new data and tasks throughout their deployment while resisting adversarial attacks that aim to exploit model weaknesses. Developing resilient TL models is crucial for applications where security and long-term autonomy are paramount [123].
- **Explainability in TL** The demand for explainability in TL models is growing, especially for applications in critical domains where decisions must be transparent and justifiable. Future research will likely focus on making TL models more interpretable, moving beyond the “black box” approach to allow for greater user trust

and understanding [123].

- **Domain adaptation and multi-source domain transfer** Domain adaptation and multi-source domain transfer are at the forefront of TL research. The challenge lies in reconciling the differences between source and target domain distributions to extract and transfer helpful information effectively. Future work may develop more efficient algorithms that can handle the computational complexity of multiple source domains and further refine instance-based methods that have shown promise in these settings [132].
- **Combating negative transfer and enhancing applicability** Mitigating the negative impacts of transfer when source and target domain knowledge are incompatible is an ongoing concern. Future TL models must identify and avoid negative transfer to ensure beneficial knowledge transfer. Additionally, enhancing the practical applicability of TL through methods such as joint learning and multi-view learning will be crucial in addressing the limitations of current Deep Transfer Learning (DTL) approaches in practical applications [129].
- **Benchmarking and evaluation** Finally, establishing comprehensive benchmarks and the systematic ranking of TL methods are essential to quantify progress and guide future developments. Researchers can better understand their relative strengths and weaknesses by comparing different TL approaches on standardised datasets and tasks, leading to more targeted and effective TL strategies [129].

2.6 Object tracking

2.6.1 Introduction

Object tracking can be defined as the process of estimating the location of the target (or targets) within an image sequence (time) [133]. Depending on the application domain, object-centric information like orientation, area or shape can also be plotted [60]. The

process is usually aided by different object features like motion, velocity, colour, texture, etc., within a specified target region (mostly a bounding box). Tracking is a necessary step for computer vision and has numerous applications in many domains, such as intelligent security surveillance, human-computer interaction, sports video analysis, autonomous navigation, etc. In this paper, object tracking is essential for collision avoidance as it allows the vehicle to monitor the movement of tracked objects continuously and enable proactive decision-making by predicting their future movement.

Tracking can be classified into Single Object Tracking (SOT) and Multiple Object Tracking (MOT) according to the number of target(s) being tracked. SOT aims to locate a given target in all frames of a video. To this end, tracking algorithms always extract specific features from the template of target appearance and a search frame and then match these features to locate the object. For retaining effective target templates, the appearance of the object in the first frame is taken as the initialisation and continuously updated during tracking. In contrast, the matching framework is manually designed and fixed during tracking. As a result, the extracted features must be representative to accurately distinguish the object from the background [134].

MOT can be viewed as a data association problem aiming to associate detection across the frames in a video sequence [135]. Differently from object detection algorithms, whose output is a collection of rectangular bounding boxes identified by their coordinates, height and width, MOT algorithms also associate a target ID to each box (known as detection) in order to distinguish among intra-class objects [63].

Section structure

The rest of this section is structured as follows:

- **Section 2.6.2** discusses common challenges faced for tracking.
- **Section 2.6.4** discusses the appearance model for tracking.
- **Section 2.6.5** discusses motion models for tracking.

- **Section 2.6.6** discusses the search strategy for tracking.

2.6.2 Tracking challenges

Although it has been studied extensively during the past decades, object tracking is still a challenging problem in the real world, such as prior knowledge of the object occlusion, pose variations, illumination changes, fast motion, background clutter, noise in images, complex object motion, as well as real-time processing requirements, etc. In the early years, almost all visual tracking methods assumed the interested object had a rigid body, so the trackers had difficulty coping with abrupt changes in appearance or size. This section discusses the main challenges to building a robust tracking system.

Environment & motion complexity

Environments with a large number of moving objects pose significant challenges in tracking. For example, the view of the interested target may be blocked by other objects, known as occlusion. Both partial and complete occlusion are significant research challenges; occlusion results in overlapping detected boundaries and wrong calculation of objects.

Minor movement in the backgrounds (or target motion) complicates the tracking process, and there is still a need for systems that can tolerate very rapid translations, rotations, and accelerations. These motions accumulate to significant prediction errors. Any prior knowledge of the target motion would greatly help in predicting the next position of the target. However, initial information about target movements in real-world scenarios must often be available [9, 80].

Lastly, the background is fixed in visual tracking with a static camera. This means foreground and background can be relatively separated easily. However, it is difficult to track with a moving camera. The camera's ego-motion could cause sensor measurement uncertainties, making it challenging to separate foreground and background. In addition, introducing a moving camera invalidates many simplifying techniques, such as

background subtraction and a constant ground plane assumption [136]. In tracking, the apparent motion of the target relative to the camera plays a vital role [7].

Imperfect sensor input

Working with imperfect input is a crucial requirement for a tracking system. Incomplete information can originate from sensor-related reasons, such as calibration issues, hardware malfunctions, uncertain detections, and asynchronous scans, or from scene perturbations, like occlusions, weather issues, and object shifting. The tracking system should be robust to the following complex scenarios.

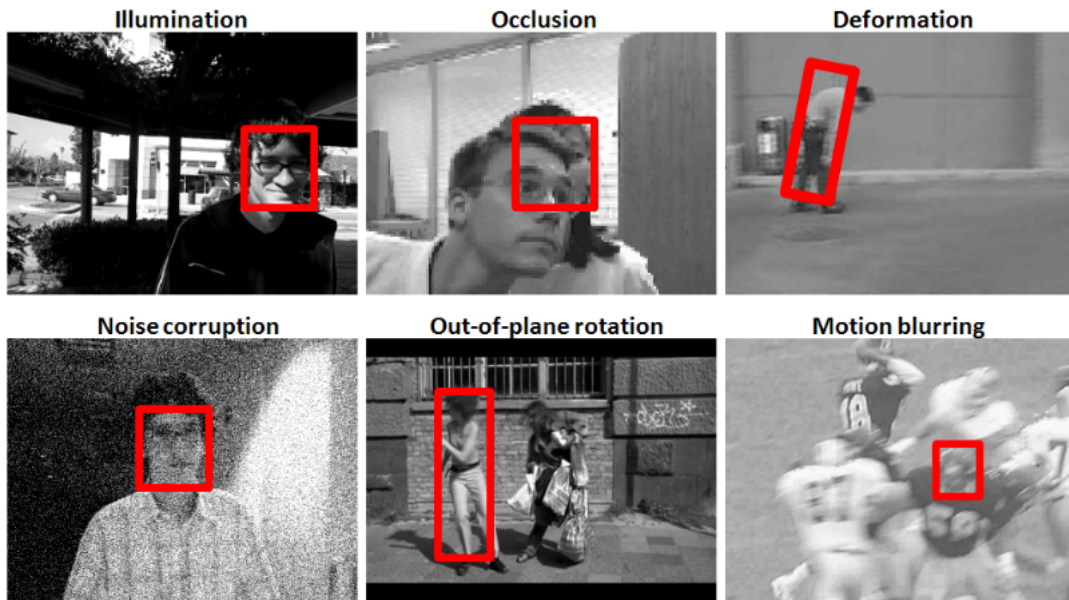


Figure 2.7: Illustration of imperfect sensor input and appearance change [6]

- **Shadows** coming from a background or moving object can be detected as foreground. Shadows must be detected and separated from the correct object.
- **Camera jitter** makes the captured object look blurred with prolonged boundaries. The proposed system shall provide practical ways to handle camera jitter due to high-velocity winds blowing during image capturing. The detection method shall be able to overcome this limitation [137].

- **Noisy input** One major challenge is associating noisy object detections in the current video frame with previously tracked objects. Combining different cues in computing similarity, such as appearance, motion, and location, is beneficial in handling ambiguities in association. Most previous works rely on heuristically selected parametric models for the similarity function and tune these parameters by cross-validation, which is not scalable to the number of features and does not necessarily guarantee the generalisation power of the model [138].
- **Illumination** Changes in illumination can occur because of the presence or disappearance of a light source in the background, e.g., a bulb, tube light, sun, etc. Rapid illumination changes can lead to false detections in consecutive frames or over multiple frames. The developed solution shall be able to work on different levels of illumination [137].

Real-time tracking

Another significant issue is computational cost. In order to be practical for autonomous navigation, tracking systems need to deal with live video streams with as little computation resources as possible. To achieve a smooth output video impression for human eyes, a frame rate of at least 15 FPS has to be established [139]. This task is difficult due to complex occlusions, fast motions, appearance changes, etc. Although real-time detection approaches have already been proposed in the vision community [140, 141, 142], they often build upon GPU processing, which implies significant energy consumption.

Adaptivity of appearance variation and drifting

As discussed earlier, object tracking relies heavily on how accurately the representation models describe the target under various challenging conditions. However, even if an accurate object is available, adaptivity to appearance variation is essential for robust tracking. This variation can arise from several complicated factors such as object rotations or geometrical transformations, texture changes, non-rigid shape deformation, and cam-

era viewpoint. Existing methods describe object appearance manually or trained using the first frame [143, 144]; these methods cannot deal with significant appearance changes and often cause tracking degradations and failures. Therefore, an adaptive appearance model is required to handle such variations simultaneously as the object changes its appearance. However, designing such a model involves many parameters that must be tuned to get good performance [145].

Furthermore, existing adaptive methods suffer from the drifting problem caused by the inaccurate estimation of the foreground and background. The foreground and background are usually divided by a bounding box or a region around the object. No matter how tight the region is, such a partition is too rough because some background regions are treated as a part of the foreground, especially when the location of the object is not precise or the object is occluded [146]. Another cause of drifting for online learners who perform self-updating is label jitter. The problem of label jitter arises if the bounding boxes of an object are not perfectly aligned with the target, although it is detected correctly. If label jitter occurs repeatedly over a tracking sequence, the tracker will likely start to lose the target object [147].

Multi-objects tracking

Compared to SOT algorithms, which rely on designing appearance models, motion models and other factors. MOT is more complex due to uncertainties in 1) prior unknown and time-varying number of targets and 2) identifying multiple targets [9].

Apart from the challenges above shared between SOT and MOT, further vital issues that complicate MOT are listed as follows:

- Frequent occlusions due to the crowd.
- Initialisation and termination of the tracking.
- Similar appearance in a crowded environment.
- Multiple objects interaction.

- Uncertain sensor measurements from any one of the targets.
- Targets that appear and disappear in the field of view.
- Complex state estimation for all present targets.
- Discrete combinatorial problem of data association.

Thus, it often causes target drift and ID switch errors, as such models usually struggle to distinguish intra-class objects with similar features by applying SOT models directly to solve MOT problems [63]. Furthermore, in most cases, SOT can be solved within a local region around the predicted position; the problem is much more complicated for MOT due to the data association problem and interactions between different targets [148].

Limited training dataset

In addition to the aforementioned technical issues in deep learning algorithms, there is a growing requirement for different huge datasets to perform an efficient training phase. First, deep models require a huge amount of training data, which has yet to be available in the case of multi-target tracking [149]. Second, as discussed in Section 2.6.2, appearance variation and drifting are a major concern for tracking. In contrast, these adaptive appearance models are data-dependent; there is insufficient data for online algorithms to learn [150]. Consequently, misaligned samples will likely be added and degrade the appearance models due to the drifting problem [150]. Third, both the data and the desired solution can be quite variable. One faces discrete and continuous variables, unknown cardinality for input and output, and variable lengths of video sequences [149]. In practice, the available online datasets are limited in terms of classes and instances that influence the training accuracy of deep learning algorithms [151].

Challenges in railway maintenance tasks

Railway maintenance environments present unique challenges for object tracking; they are often characterised by complex environmental conditions, such as frequent occlusion

caused by moving maintenance workers and equipment. The motion complexity is further increased by the vehicle's ego-motion, making it challenging to separate the foreground and background. Additionally, sensor measurements are often impacted by noise and uncertainties due to track vibration and calibration errors, particularly in imperfect weather or low-light conditions, such as tunnels or night-time operations. Moreover, the appearance of railway objects can vary due to changes in different FoV, the accumulation of dirt or wear, and the tracking system struggles to maintain accuracy over time. Railway maintenance often involves a crowded environment with multiple interested objects, such as maintenance workers and other equipment, moving simultaneously, which creates occlusions and leads to ID switches and target drift. Lastly, The scarcity of object-tracking datasets in the railway maintenance scenarios sector aggravates the above challenges; tracking models often struggle to operate accurately and reliably.

These interrelated challenges demand innovative solutions to enable reliable and scalable tracking systems for autonomous railway applications; a track-by-detect approach offers a practical solution; these methods rely on object detection models and track objects by associating detection results across frames and avoid the need for dedicated training for tracking. Additionally, Sensor-fusion-based approaches avoid reliance on large datasets by integrating multiple sensor inputs. For instance, LiDAR excels in providing precise spatial information, even in low-light environments, making it ideal for complex railway maintenance environments.

2.6.3 Tracking framework

The tracking task aims to localise objects within a given scene and generally can be divided into two sub-tasks: object observation and object localisation. The first one measures the similarity between objects in frames and represents the objects by observation models. The latter recovers the observation models and links them to identify various targets. The first issue involves the modelling of the target's appearance information (visual representation) and dynamics (motion model). The second issue consists of the inference

problem, which guides the search for an image patch that matches the target. A track system generally Fig. 2.8 shows the typical components in a tracking framework to resolve these issues [6, 145, 152, 153].

- **Appearance model**, characterise and distinguish objects at a particular location and update the object's appearance according to the newly acquired object location.
- **Motion model**, predicts the object's future location over time.
- **Search (match) strategy**, searches for which location the object most likely resides in the current frame.

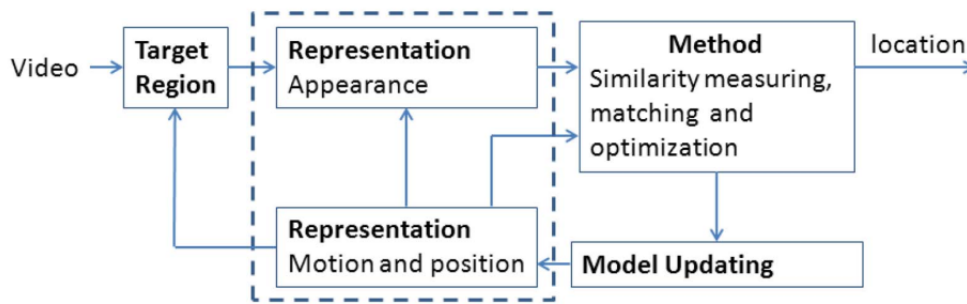


Figure 2.8: Model of trackers with five most important components [7]

However, not all tracking methods contain all the components. For example, model update is unnecessary for scenarios with no appearance change. On the other hand, additional components are necessary for specific cases—for example, object interaction model for MOT problems and occlusion handling for tracking within a complex environment. The tracking algorithm's performance relies heavily on the design and integration of these components. However, these components are not independent of each other [9]. They can be combined and integrated for satisfactory performance.

2.6.4 Appearance model

Every tracking method requires a target initialisation mechanism when the object appears in the video. The target initialisation method could be manual or automatic. Manual initialisation requires users to annotate object locations. In contrast, automatic initialisation

is achieved by object detectors [6]. A typical automatic approach for target initialisation is to use information in a single frame or the temporal information computed from a sequence of frames. Classical methods include point detectors, background subtraction, segmentation and supervised learning-based approaches [60]. In practice, selecting a target initialisation approach depends on the application [153]. This section discusses the common feature or representation used for object tracking.

Target representation

Target representation in the context of object tracking in computer vision refers to the method used to define and represent the object of interest (the ‘target’) within an image or a sequence of images. It is a critical aspect of the tracking process as it influences how the object is identified, tracked, and analysed over time. Figure 2.9 illustrates various target representations for tracking. The rest of the paragraph discusses the fundamental forms of target representation:

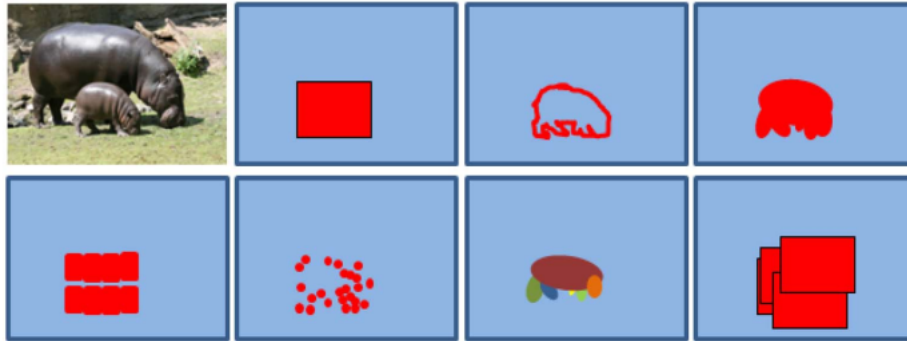


Figure 2.9: Target region representations used in tracking. From left to right, top to bottom: target bounding box, target contour, target blob, patch-based, sparse set of salient features, parts, and multiple bounding boxes [7].

Bounding box is the most widely used form of target representation and is a part of many successful trackers [63, 7]. It typically involves a rectangular (mostly axis-aligned) bounding box for initialisation and estimation of the target’s position in subsequent frames. Despite its simplicity, bounding boxes often include background pixels, leading to potential confusion with the target pixels [7]. Moreover, a simple rectangle may

not always accurately represent an object, leading to the inclusion of some background in the object's description [7, 154]. Bounding box regression has been explored to enhance tracking accuracy, particularly in improving the MOT score [63].

Patch-based initialisation considers the most informative parts of the target, allowing them to adapt to appearance changes independently. This method is beneficial for tracking objects with dynamic shapes or during occlusion, though it may not be advantageous for rigid or small targets [7]. Patch-based approaches may struggle when the object has an even appearance, as salient points may not lock onto their positions effectively. A combination of patches with multiple bounding boxes has been suggested as a robust solution [7].

Blob representation employs a rough segmentation or blob-representation of the target, enhancing discrimination from similar objects in the scene, mainly when the target occupies a small fraction of the surrounding box [7]. This approach is beneficial when the shape of the target is the primary focus, such as in gesture analysis.

Discussion, while there is a general trend toward distributed representations focusing on parts and patches, this approach has yet to prove itself in specific contexts fully. Including local background information in the target representation has demonstrated increased robustness against changes in illumination and appearance, as highlighted in recent surveys and studies [7]. The choice of target representation is pivotal in object tracking, significantly influencing the tracking performance under various conditions. Researchers continue to explore innovative methods to balance accuracy and robustness in dynamic and challenging tracking environments.

Generative methods in object tracking

Generative methods in object tracking focus on modelling the appearance of the target object, capturing its variability over time due to changes in viewpoint, illumination, and

occlusion. These methods, as described in [155], employ a model that accounts for the local information of an object, enhancing its ability to handle partial occlusions. The approach is extended in [156] by presenting a generative object detection algorithm that extracts objects of varying sizes and shapes from 3D range data, demonstrating the method's versatility across a wide range of object classes.

Generative trackers, as mentioned in [9, 80, 157], are particularly adept at reacquiring objects after occlusion or when they leave the field of view. They rely on appearance models updated online to adapt to variations in appearance and illumination, making them popular for scenarios where new objects are discovered, or existing ones disappear. However, challenges still need to be addressed, such as the drifting problem due to significant changes in appearance and the difficulty of verifying the correctness of the model in cluttered backgrounds.

Discriminative methods in object tracking

Discriminative methods treat object tracking as a binary classification problem, focusing on differentiating the target from the background. This approach is effectively implemented in [158], where a discriminative model using a graph-regularised discriminant analysis algorithm was developed. These methods, as stated in [145, 154, 159], involve online training to separate the object from the background and estimate its trajectory, employing techniques such as tracking-by-detection.

Despite their robust performance against occlusion and variation, discriminative methods face challenges. They often depend on simple features, which may need to be improved for tracking in complex environments. Additionally, the focus on enhancing classifiers rather than image features can lead to drift problems due to accumulated inaccuracies in object location detection.

Fusion of generative and discriminative methods in object tracking

The fusion of generative and discriminative methods aims to combine the strengths of both approaches in object tracking. As described in [6], Hybrid Generative-Discriminative Models (HGDAMs) integrate these methods in a single or multi-layer manner. This fusion strategy, however, only sometimes guarantees superior performance compared to individual models and may introduce additional constraints and parameters, leading to inflexibility in practical applications.

Nonetheless, the complementary nature of generative and discriminative models offers a promising direction for enhancing the robustness and adaptability of tracking systems, especially in handling complex scenarios like rapid appearance changes and occlusions.

2.6.5 Motion model

Object motion models, or dynamic models, are crucial in object tracking as they encode the dynamics of the object to facilitate target localisation in future frames. The motion pattern between consecutive frames or tracked targets significantly reduces the search space, thereby enhancing the robustness and effectiveness of tracking algorithms [134]. As discussed in Section 4.2.2, The object detector in this paper is trained to classify dynamic objects like humans, workers, and warthogs, all of which display varying motion dynamics. It is crucial to understand their movement patterns for reliable object tracking. For instance, workers and warthogs generally follow smoother, predictable motion paths, making linear motion models suitable for tracking their movements. In contrast, non-linear motion models are suitable for pedestrians (trespassers) due to their unpredictable and abrupt change in speed or direction. Fig. 2.10 illustrates the difference between these two motion models.

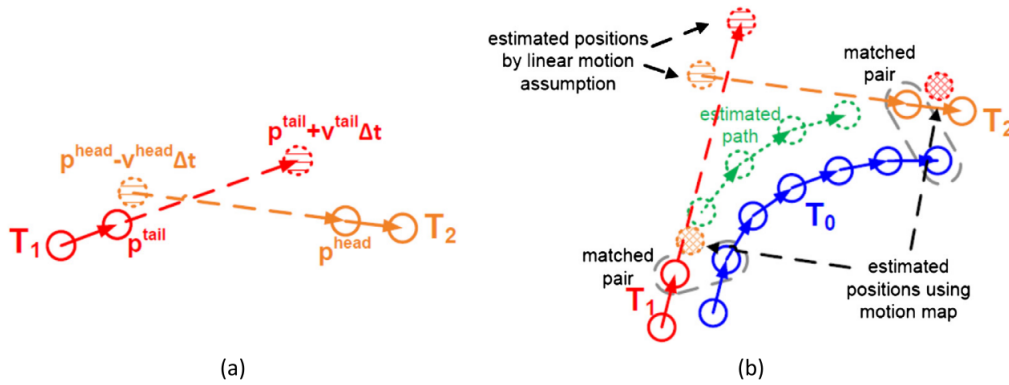


Figure 2.10: An image comparing the linear motion model (a) with the non-linear motion model (b) [8]

Linear motion model

Linear motion models exploit the smoothness of an object's velocity, position, or acceleration in video frames [9, 160]. These models enforce that the values of these properties change smoothly over successive frames. For example, velocity smoothness can be modelled by ensuring gradual changes in velocity, while position smoothness minimises the discrepancy between observed and estimated positions. Acceleration smoothness considers motion as a probability distribution of a state with a given observation tracklet at a specific time [9].

In practice, velocity can be estimated using first-order linear regression, assuming that the object's motion is linear over short periods. The Kalman Filter (KF), Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), and Particle Filter (PF) are various implementations of this concept, each suited to different assumptions about model and noise characteristics. The KF, in particular, is effective in practical applications, striking a balance between performance and computational complexity [153].

Non-linear motion model

While linear motion models are commonly used, they fall short in some scenarios. Non-linear motion models are more adaptable at capturing real-world motion complexity, providing more accurate motion affinity between tracklets. These models are advantageous

when the linear assumption is invalid, such as with abrupt changes in motion direction or speed [9, 160, 161].

2.6.6 Search strategy

In object tracking, the search strategy refers to locating and following the target object across video frames. Effective search strategies are crucial for accurate tracking, especially in challenging scenarios with occlusions, rapid movements, or changes in the object's appearance.

Online method

Online tracking methods, also known as sequential tracking, sequentially handle images and conduct learning during the tracking process [135, 138, 152, 162, 163, 164, 165]. These methods construct positive and negative training examples based on the tracking results, then train a similarity function for data association [138]. Figure 2.11 illustrates this concept with objects (represented as different circles) being tracked over time, showing the trajectory production based on up-to-time observations [9].

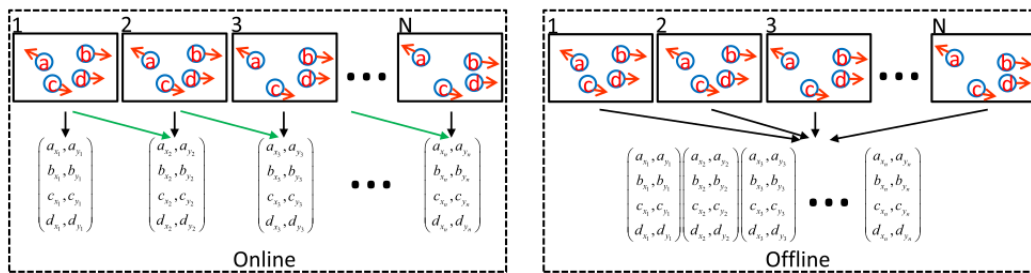


Figure 2.11: An illustration of online (left) and offline (right) tracking. (For interpretation of the references to colour in his figure, the reader is referred to the web version of this article.) [9].

Online method pros: online methods are particularly advantageous for real-time applications such as robotic navigation, where immediate response based on the current status and history of the target is essential. They are computationally efficient as they focus on

local features for data association [160].

Online method cons: however, online methods may suffer from limited training data and the possibility of learning from incorrect training examples, leading to tracking drift [138]. They are also prone to higher fragmentation rates, especially when the tracking target is occluded or detections are missing, as they are unable to anticipate future video frames to re-identify lost targets [63].

Offline method

Offline tracking analyses a batch or the entirety of frames collectively. As shown in Fig. 2.11, offline methods require pre-obtained observations from all frames and use them for joint analysis, treating tracking as a global data association problem [63, 138, 162]. These methods benefit from utilising past and future information, leading to better tracking quality [63, 138]. Recent results on datasets like MOT16 and MOT17 indicate that offline algorithms slightly outperform online ones, though the gap is narrowing [63].

However, offline methods struggle with long-term occlusions and demand substantial computational resources due to iterative associations for globally optimised results. They typically require pre-detected sequences and perform learning before tracking, limiting their application in dynamic, real-time scenarios [138, 162, 166]. A potential workaround is to process shorter video clips sequentially or hierarchically [9].

Global and local search: some tracking strategies employ a mix of global and local searches. Global search scans the entire image frame to locate the target, while local search narrows down to areas close to the target's last known position. An example of this approach is the dynamic attention-guided multi-trajectory tracking strategy [167], which merges local and global search results to enhance tracking in scenarios with heavy occlusions and out-of-view movements.

Probabilistic method

The probabilistic method is another way to classify searching methods based on the inference method, namely, probabilistic methods. Approaches based on probabilistic inference typically represent states of objects as a distribution with uncertainty [168, 169]. Tracking algorithms use different probability methods to estimate the probabilistic distribution of the target state based only on the existing observations, which are appropriate for online tracking tasks [9].

KF is an estimator that uses measurements observed over time to produce estimates. It has many applications in control, navigation, computer vision, etc. The main features of KF are prediction of the object's future location, noise reduction, and the ability to track single as well as multiple objects. KF is an efficient way to address multi-target tracking when the number of objects remains small. It is also well suited for real-time applications [161, 170]. The KF [171] is proven to be the optimal estimator in the case of a linear system and Gaussian-distributed object states. Moreover, it provides a good balance between computation load and tracking robustness [172].

However, when the number of objects increases, identity switches become more frequent and are difficult to correct due to the recursive nature of the method [161, 170]. Furthermore, experiment [173] tested this on high-resolution video streams, and its validity on lower-quality videos is not assured.

PF is extended form of KF [174], it is a technique for implementing a Bayesian inference filter using Monte Carlo simulation. The key is to construct the posterior probability density function of the state space by a set of random samples with associated weight using Monte Carlo integration. It has been widely applied to real-world tracking problems. Because it is computationally efficient and insensitive to local minima [175].

The advantage of PF is its ability to handle non-linear and non-Gaussian conditions, which makes it suitable for real-world tracking [61, 139, 161, 175]. Furthermore, the

computational complexity is simplified due to the reduction of sampling patches from Monte Carlo sampling [61, 139]. Another benefit is that the sampling effort can be kept constant, independent of the object's size to track, which is not the case with simply expanding the search region around the object with a fixed factor. Many state-of-the-art discriminative online learning-based tracking methods estimated the target's position directly from exhaustive search-based methods. It would be reasonable and inspiring to integrate these methods into a probabilistic inference framework [139].

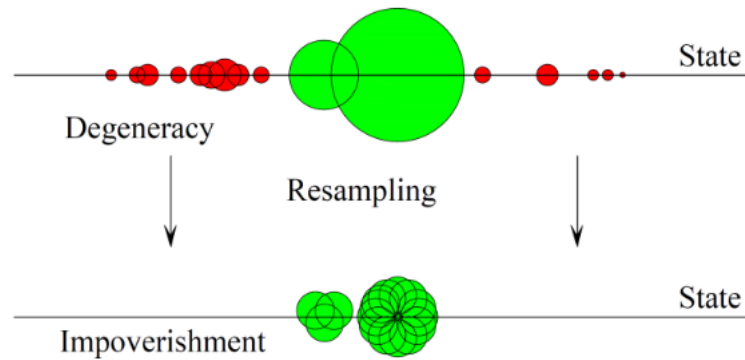


Figure 2.12: Sample degeneracy and impoverishment illustrated in 1D state space, where green circle indicates particles with significant weight and red circle indicates particles with small weight [10].

However, PF suffers from two fatal problems due to its suboptimal sampling mechanism, namely sample degeneracy and impoverishment, which is caused by the trade-off between the particle being distributed too widely and over-concentrated [10, 176]. Furthermore, PF-based tracking algorithms are more likely to perform reliably with more sampled particles. However, the computational cost grows exponentially with the number of particles [61, 161, 170, 174], which causes another trade-off between performance and efficiency.

Deterministic method

Compared to probabilistic inference methods, deterministic approaches focus on finding the maximum a posteriori (MAP) solution for MOT. This task involves casting data

association and target state inference as an optimisation problem. These methods are better suited for offline tracking since they require observations from all frames or at least a specific time window to be available in advance. Given the detection hypotheses from all frames, these methods aim to globally associate observations that belong to the same object into a single trajectory. The main challenge lies in determining the optimal association [9].

In practice, deterministic optimisation or energy minimisation is more popular than probabilistic approaches. Although the probabilistic approaches provide a more intuitive and complete solution to the problem, they are usually difficult to infer. On the contrary, energy minimisation could obtain a “good enough” solution in a reasonable time [9].

Mean shift is an iterative algorithm for locating the maxima of a density function given discrete data sampled from that function. It computes the similarity of object models and candidate image data. The algorithm constructs a confidence map (probability density function) in the new image based on the colour histogram of the object in the previous image. It reaches the target when the algorithm converges to the peak of the confidence map.

One significant advantage of mean-shift relies on it being modelled by applied kernel function, which means it is robust to edge occlusion, target rotation, deformation and background change [61]. Additionally, it is computationally efficient and suitable for real-time tracking [177]. However, the lack of essential update ability makes mean-shift likely to fail when there is the transformation of scale or form in tracking target [61]. Furthermore, it cannot differentiate the background and target when their colour distribution is similar because it only uses a colour histogram for the target description. Lastly, mean-shift can only track slow-moving objects because there is no overlapped area between two adjacent frames when an object moves quickly. A few methods based on mean shift are proposed to overcome these disadvantages, such as kernel-based tracking [178], ensemble tracking [179], and Continuously Adaptive Meanshift (CAMshift) [180].

Exclusion model

The exclusion model in object tracking considers spatiotemporal information and other constraints to address the challenge of object collision in MOT scenarios. This model is based on the physical principle that two distinct objects cannot occupy the same space simultaneously. In MOT, this principle translates into two primary constraints: detection-level exclusion and trajectory-level exclusion [9, 161].

Detection level exclusion asserts that two different detection responses in the same frame cannot correspond to the same target. Luo [9] categorised these into “soft” and “hard” models. The soft model minimises a cost term to penalise violations, such as assigning the same trajectory label to distant detection responses [9]. The hard model employs explicit constraints, like ‘cannot links’, to ensure that overlapping tracks are not assigned to the same cluster or trajectory [9]. Non-negative discretisations can be used to group detections into non-overlapping sets to adhere to the mutual exclusion constraint [9].

Trajectory level exclusion means that two trajectories cannot be infinitely close. This is often modelled by penalising scenarios where closely situated detection hypotheses are assigned different trajectory labels, effectively suppressing one of the labels. For example, a penalty term inversely proportional to the distance between two detection responses with different labels can be used [9]. If the detections are too close, this leads to a significantly large, or in extreme cases, infinite cost, preventing unrealistic overlaps of trajectories.

The exclusion model is particularly effective in crowded scenes where objects are nearby, and the risk of occlusion or overlap in tracking is high. By ensuring distinct and non-overlapping tracking paths for each object, the exclusion model significantly enhances the accuracy and integrity of object tracking systems, particularly in complex environments such as urban traffic or densely populated public spaces.

2.6.7 Tracking methods

Region-based tracking

Region-based tracking algorithms track objects according to variations of the image regions corresponding to the moving objects. For these algorithms, the background image is maintained dynamically [181, 182], and motion regions are usually detected by subtracting the background from the current image. Although they work well in scenes containing only a few objects (such as highways), region-based tracking algorithms cannot reliably handle occlusion between objects. Furthermore, as these algorithms only obtain the tracking results at the region level and are essentially procedures for motion detection, the outline or 3-D pose of objects cannot be acquired. (The 3-D pose of an object consists of the position and orientation of the object). Accordingly, these algorithms cannot satisfy the requirements for surveillance against a cluttered background or with multiple moving objects [183].

Contour-based tracking

Active contour-based algorithms track objects by representing their outline as a bounding contour and dynamically updating it in successive frames. These algorithms aim to extract subjects' shapes directly, providing more effective descriptions of objects than region-based algorithms [183]. The active contour representation usually uses the signed distance map to encode the boundary information of an object implicitly. By utilising level set evolution, this representation can accurately segment objects with complex shapes [6]. Similar to the Active Contour Models (ACM) approach in energy-based segmentation, contour-based approaches facilitate the use of energy functions according to the image's features and the smoothness of the contour profile [61].

Contour-based pros: contour-based model offers several advantages. Firstly, it can benefit from shape priors, which may be known through general geometric knowledge of the expected objects [172]. Secondly, contour-based tracking algorithms allow maximum

freedom for target appearance change and movement [7, 61], making them suitable for non-rigid object tracking [6]. Even under disturbances or partial occlusions, these algorithms may continuously track objects [183]. Finally, compared to region-based tracking, this approach reduces computational complexity [183].

Contour-based cons: however, tracking precision is limited at the contour level. Recovering the 3-D pose of an object from its contour on the image plane is challenging [183]. Moreover, this approach requires human assistance, such as specifying the number of foreground objects and their initial contours [61, 172], implying difficulty in dealing with occluded objects. Consequently, this approach is highly sensitive to initialisation [183], making automatic tracking startup challenging. The active contour model is a non-convex function, meaning it may converge to a local extreme or even diverge [61]. Overall, these models are found in specialised application areas, like pedestrian tracking, where precise shape detection is effective but less so for general applications [7].

CF-based tracking

Modern adaptive tracking approaches often use an online version of an object detection algorithm; Correlation Filter (CF) instead trains a regressor by exploiting the properties of circular correlation and performing operations in the Fourier domain [154], and minimise a least-squares loss for all circular shifts of the positive examples [184]. More specifically, a CF-based tracker models the object's appearance by filters trained on an example image; the target is initially selected based on a small tracking window centred on the object in the first frame. From this point on, tracking and filter training work together. The target is tracked by correlating the filter over a search window in the next frame; the location corresponding to the maximum value in the correlation output indicates the target's new position. An online update is then performed based on that new location [163]. CF-based trackers' performance has been notably improved with the adoption of multi-channel formulations [185], spatial constraints [186]. and deep features [187, 188].

CF pros: although CF might seem a weaker approximation of the true problem, it has been proven competitive with far more complicated tracking methods [175]. CF enables using densely-sampled examples and high-dimensional features in real-time using the Fast Fourier domain with low computational load [184, 189, 190]. It shows excellent robustness when tracking complex objects through rotations, occlusions, motion blur and illumination changes [163, 184].

CF cons: despite the advantages, existing CNN-based CF trackers have several limitations. First, these trackers learn CF for each layer independently without considering their relationship, making the Tracker more prone to drift away from target objects in cases of significant changes in appearance [174]. Second, most CF-Based Tracker can not correctly handle scale variation, appearance variation and deformation [163, 174, 175, 184], which makes the trackers unreliable in long-term tracking. Lastly, recent CF-based methods use deep features to improve the accuracy, but it broadly harms the speed during model update [81].

2.6.8 Discussion on object tracking

Appearance model

Section 2.6.4 discussed different aspects of the appearance model for object tracking, emphasising target initialisation and representation. Target initialisation methods, whether manual or automatic, are foundational for tracking and significantly influence the effectiveness of subsequent tracking stages [60, 6, 153].

Different target representations, such as bounding boxes, patches, and blob representations, offer unique advantages and challenges. Bounding boxes often include background pixels, potentially leading to confusion with the target [63, 7, 154]. Patch-based and blob representations can handle dynamic shapes and occlusions better but may struggle with rigid or small targets [7].

	Discriminative	Generative
Initialisation	Automatic, imperfect	Manual, perfect
Number of objects	Varying	Fixed
Applications	Specific type of objects (most cases)	Any type of objects
Pros	Handle a varying number of objects	Free of object detector
Cons	Performance depends on object detection	Manual initialisation

Table 2.2: Generative and discriminative methods comparison, adapted from [9]

Generative methods focus on modelling the target’s appearance, which helps in reacquiring objects after occlusions but can suffer from drift due to significant appearance changes [9, 80, 155, 156, 157]. Although generative methods appear to be identical to online methods (Section 2.6.6) due to the fact they process observation sequentially, exceptions exist [9], such as orderless tracking [191]. On the other hand, discriminative methods treat tracking as a binary classification problem, which has the advantage of distinguishing the target from the background but may face drift problems due to inaccuracies in object location detection [145, 154, 158, 159]. Table 2.2 shows a comparison between these two methods.

The fusion of generative and discriminative methods aims to leverage the strengths of both approaches, offering a promising direction for handling complex tracking scenarios [6, 192]. However, this fusion introduces additional constraints and parameters, which can lead to practical inflexibility.

Overall, the choice of target representation and the balance between generative and discriminative methods are crucial for achieving robust and adaptable tracking systems. Future research should focus on developing innovative strategies that enhance the accuracy and robustness of tracking in dynamic and challenging environments.

Motion model

In object tracking, a motion model is used to predict the future position and movement of a tracked object based on its past motion. Linear motion models are commonly employed due to their simplicity and efficiency in handling smooth and gradual changes in object

motion [160, 9]. However, they struggle with non-linear motion patterns. Non-linear motion models address these shortcomings by providing a more flexible framework to capture complex motion dynamics, although they often require more computational resources [161, 160, 9].

Both linear and non-linear models could not provide shape information of the object and fail to handle long inter-frame object occlusions or out-of-view movement since the motion estimation module tends to forget the appearance from the first frame [160, 134, 193].

Recently proposed motion estimation modules take both the search frame and other frames as inputs. However, excessive information from the initial frame will influence the overall tracking performance. More research is required to find the trade-off between the initial frame and other frames [134]. Furthermore, they are also challenged by complex motion scenarios. Hence, composite model-based tracking aims to strike a balance between motion and appearance modelling. However, such a balance is hard to achieve in the real world [194].

Search strategy

Tracking methods can be categorised into online (local) and offline (global) tracking according to how the data is processed (see Table 2.3). Online methods are more suitable for real-time applications due to their computational efficiency, sequentially handling images and updating tracking models dynamically. However, online tracking are more likely to drift due to limited training data and challenges in handling occlusions [138, 63, 9]. Offline methods, on the other hand, analyse batches of frames collectively, leveraging past and future information to enhance tracking quality. Despite their higher computational demands and limitations in dynamic real-time applications, offline methods generally achieve better tracking performance by addressing the global data association problem [63, 9, 162, 138].

Probabilistic methods, such as KF and PF, offer robust solutions by incorporating

	Online Methods	Offline Methods
Input	Up to time observations	All observations
Methodology	Gradually extend existing trajectories with current observations	Link observations into trajectories
Pros	Suitable for online tasks	Theoretically obtain global optimal solution
Cons	Shortage of observation	Delay in outputting final results

Table 2.3: Online and offline methods comparison, adapted from [9]

uncertainty and non-linearity in motion models. KF excel in scenarios with linear and Gaussian-distributed states, providing a balance between computational load and tracking robustness [172, 195]. PF extend these capabilities to non-linear and non-Gaussian conditions but face challenges such as sample degeneracy and increased computational cost with higher particle count [174, 161, 61, 196]. These methods explicitly account for object measurements and uncertainties to establish correspondences, but their output can vary over time. For instance, in detection-free tracking, the bounding box results differ when using PF due to the randomness in particle generation. Even in detection-based tracking, some studies refine detection bounding boxes using state-of-the-art methods, leading to different tracking results across runs [9].

On the other hand, deterministic methods use qualitative motion heuristics to constrain the correspondence problem, producing constant outputs when the methods are run multiple times. For instance, in tracking-by-detection, data association methods like the Hungarian algorithm yield deterministic tracking results. Deterministic tracking is typically associated with deterministic optimisation to derive the final output, ensuring consistency and predictability in the tracking results [9].

Finally, the exclusion model ensures non-overlapping tracking paths by applying spatial-temporal constraints, which is particularly beneficial in crowded scenes. This model enhances tracking accuracy by addressing object collisions at both detection and trajectory levels [9, 161].

Track-by-detection approaches

The performance of Track-by-detection approaches heavily relies on the quality of object detectors, which perform two tasks: detecting the presence and 2D location of objects within an image (detection) and categorising them into predefined classes (classification), thereby providing the detection hypotheses required to drive the tracking process.

Variations in detection quality can lead to significant differences in tracking performance, even when all other components of the tracking framework remain unchanged. However, object detection often suffers from false negatives and false positives in complex scenarios like dense pedestrian tracking or occlusion-heavy environments. While deep learning has improved detection accuracy, high-quality detections remain the most effective way to minimise false negatives. Combining SOT trackers with private detections offers a potential solution for recovering missed detections and improving tracking accuracy [63].

On the other hand, object classification can distinguish between the target object and the background, improving robustness in dynamic scenarios. Classifiers, such as multiple instance learning [197, 145], P-N learning [198], and online boosting [199, 200], enhance the system's adaptability to changes in appearance or illumination [177]. Advanced classifiers, like structured output SVM and Random Forests, have shown great potential for stabilising tracking performance and managing environmental variability [189, 201].

Ultimately, integrating detection and classification steps improves the stability and accuracy of trackers, particularly in challenging real-world scenarios. A precise balance between these components ensures that the system maintains high performance while adapting to varying conditions and object classes. Established benchmarks, such as Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI) and MOTChallenge, have made efforts to standardise the evaluation of detection and tracking, moving toward unified metrics to improve comparability [9].

Future directions

Multi-camera tracking: the appearance model based on a single camera can only provide limited information about the tracklets; tracking would benefit from multi-camera settings [202, 203]. There are two kinds of configurations of multiple cameras [9]. The first one is that multiple cameras record the same scene, i.e., multiple views. In this case, tracking in a large camera network needs to be established for long-term monitoring of the objects of interest [6]. However, one key question in this setting is how to fuse the information from multiple cameras. The second one is that each camera records a different scene, i.e., a non-overlapping multi-camera network. In that case, the data association across multiple cameras becomes a re-identification problem.

Tracking beyond human: most learning-based MOT algorithms have focused on pedestrian tracking [9, 63]. Since different types of targets pose different challenges, the extensions of the current MOT research to other kinds of targets (e.g., vehicles, animals, etc.) and scenarios (e.g., traffic scenes, aerial photographs, etc.) are also excellent research directions.

2D-3D fusion: most current approaches focus on multiple object tracking in 2D. However, 2D models cannot accurately estimate the poses of the tracked objects, leading to the sensitivity to occlusion and out-of-plane rotation. On the other hand, 3D appearance models could provide more precise position and size estimation, resulting in robust handling for occlusion and out-of-plane rotation [9, 6, 204]. However, 3D appearance models require a large parameter search space for 3D pose estimation, resulting in expensive computational costs. Furthermore, it involves camera calibration and overcoming other challenges for estimating camera poses and scene layout.

Meanwhile, 3D model design is another issue exclusive to 2D MOT. Therefore, combining the advantages of 2D and 3D appearance models is a challenging research topic. A possible solution to accelerate the pose estimation process of the 3D appearance models

is to use the tracking results of the 2D appearance models as the initialisation of the 3D appearance models. However, how to effectively transfer from 2D tracking to 3D tracking still needs to be solved.

Robustness v.s. accuracy: existing appearance models are incapable of simultaneously guaranteeing tracking robustness and tracking accuracy. To improve tracking accuracy, more visual features and geometric constraints are incorporated into the appearance models, resulting in precise object localisation in situations of particular appearance variations. However, these visual features and geometric constraints can also lower the generalisation capabilities of the appearance models in the aspect of undergoing other appearance variations. On the other hand, to improve the tracking robustness, the appearance models relax some constraints on a precise object localisation and thus allow for more ambiguity of the object localisation. Therefore, balancing tracking robustness and tracking accuracy is an interesting research topic [6].

Context-aware trackers: objects are usually embedded into specific contexts, and there is a broad agreement in the community on the valuable role that context plays in any video analysis and image understanding applications [139]. For Instance, objects are small and largely occluded in over-crowded scenarios. It is critical to understand the contextual information and scene structure to perform tracking tasks. Unreliable saliency detection leads to many feature mismatches across frames. Consequently, it is necessary to build an intelligent vision model that can robustly track these salient region features across frames like what human vision offers [6].

Application to autonomous railway maintenance tasks

Object tracking plays a critical role in context-aware collision avoidance systems. It provides essential information about the objects' movement, facilitates RIMV in predicting potential collisions, and makes proactive path-planning decisions.

The complex railway environment presents unique challenges, such as occlusions

caused by maintenance workers and equipment, ego-motion of the vehicle, and variations in object appearances due to dirt, wear, or changing weather conditions. The track-by-detection approach relies on high-confidence object detection for tracking and ensures reliable tracking in complex scenes [63]. Furthermore, the integration with detection models trained with TL makes it well-suited for applications where railway-specific datasets are limited [150, 134].

The reliance on both appearance and motion models aligns well with the railway context, where objects exhibit predictable motion patterns (e.g., workers following maintenance paths along the track) and require adaptive appearance modelling to handle environmental variability. Linear motion models are effective for tracking maintenance vehicles and equipment with smooth trajectories [9, 160]. In contrast, non-linear models are more suitable for pedestrians and dynamic objects because their movements are less predictable [161, 9]. Furthermore, railway maintenance tasks often operate in challenging environments, such as tunnels, night-time, or adverse weather conditions. The implementation of probabilistic models, such as particle filters, enhances robustness against noisy sensor inputs [61, 174].

Finally, the fusion of 2D and 3D tracking approaches is promising for object tracking in railway maintenance environments. While 2D trackers offer computational efficiency, 3D tracking improves RIMV's spatial understanding of surrounding obstacles with precise depth information [6, 9]. Additionally, fusion-based approaches often benefit from the strength of multiple sensors, such as combining stereo cameras and LiDAR. Cameras provide rich visual context, while LiDAR ensures precise depth information, even in low-light or adverse conditions [6, 7].

2.7 Depth estimation for tracked objects

Depth estimation is a fundamental problem in machine vision and image processing, involving mapping the distances of objects from the camera [205]. Effective distance esti-

mation allows for precise navigation, obstacle detection, and timely decision-making, all of which are crucial for maintaining the integrity of railway infrastructure.

Railway environments present unique challenges for distance estimation. The presence of dynamic obstacles, including other trains, maintenance vehicles, and trespassers, can significantly affect the accuracy of distance measurements. Environmental factors like weather conditions and lighting variations add another layer of complexity.

Addressing these challenges necessitates the adoption and refinement of sophisticated technologies and methodologies. This section of the literature review explores the existing research on depth estimation techniques, highlights the advancement, and identifies the gaps that need to be addressed.

2.7.1 Monocular vision-based methods

Monocular vision-based methods use monocular cameras to estimate distances and depths in a scene, leveraging principles of perspective geometry to infer three-dimensional information from two-dimensional images. These methods rely on two fundamental principles: perspective projection and motion parallax. Perspective projection means the apparent size of an object decreases with increasing distance from the camera [206]. Motion parallax implies that as the camera moves, closer objects appear to move faster across the field of view than distant ones [207].

These methods are simple, low-cost, and easy to integrate with existing camera systems. However, monocular vision methods face significant challenges. One major issue is the challenge of distinguishing between close small objects and large objects that are far away without additional cues [208]. Scale ambiguity also presents a problem, as the absolute scale of the scene cannot be determined from a single image; only relative distances can be inferred [209].

Additionally, occlusion and clutter within the scene can obscure objects and complicate the depth estimation [210]. Dynamic environments pose further challenges, as moving objects and changing lighting conditions can affect the accuracy of distance es-

timation [211]. Achieving accurate and reliable depth estimation with monocular vision remains a complex task requiring ongoing research and innovation.

2.7.2 Stereo vision-based methods

Stereo-based methods for depth estimation utilise two or more cameras positioned at different viewpoints to capture multiple views of the same scene. By analysing the disparities between these images, these methods infer depth information with higher accuracy compared to monocular techniques, mimicking human binocular vision.

Advantages:

- **Enhanced depth accuracy:** stereo vision provides reliable and precise depth measurements by directly comparing multiple viewpoints, reducing ambiguity and improving accuracy in distance estimation compared to monocular-based methods [212].
- **Real-time processing:** many stereo-based systems can process depth information in real-time, making them suitable for dynamic applications like autonomous driving and robotics [213, 212].
- **Scalability:** these systems can be scaled by adding more cameras to improve accuracy and coverage, which is beneficial for large-scale applications like virtual reality and 3D mapping [214].

Disadvantages:

- **High computational load:** processing multiple images to compute disparities requires significant computational power, limiting real-time applications with constrained resources [207, 212, 215].

- **Calibration complexity:** stereo systems require precise calibration to align the cameras correctly, and any misalignment can lead to errors in depth estimation. This process can be complex and time-consuming [208, 212, 215].
- **Occlusion issues:** stereo vision can struggle with occlusion, where parts of the scene visible to one camera are not visible to another, leading to incomplete or inaccurate depth information [211, 212, 215].
- **Effective range:** stereo-based methods are limited by the baseline distance between the cameras and the resolution of the disparity measurements; the accuracy decreases as the distance between the camera and the object increases due to the reduction in disparity between corresponding points in the images [215, 216].

2.7.3 LiDAR/radar-based methods

Advantages of LiDAR

LiDAR systems are ideal for 3D environment modelling and depth estimation due to their high resolution and accuracy. They are capable of detecting small objects and fine details in various environments [217, 218]. LiDAR systems also perform well in different lighting conditions, including total darkness, because they rely on laser pulses rather than ambient light [219]. This ability to operate in various lighting conditions enhances their reliability and applicability in diverse scenarios.

Disadvantages of LiDAR

Despite its high precision and robustness in various lighting conditions, LiDAR systems are typically expensive and complex, requiring precise calibration and maintenance, which can be a barrier to widespread adoption in cost-sensitive applications [217, 218]. Additionally, LiDAR performance can degrade in adverse weather conditions such as fog, rain, and snow, which scatter or absorb the laser pulses, reducing accuracy [219, 220].

Furthermore, LiDAR can be sparse in specific scenarios, requiring interpolation and additional processing to generate dense depth maps [221].

Advantages of radar

Radar systems are particularly effective at long-range distance measurement and can accurately detect the speed of moving objects, making them suitable for high-speed applications like autonomous driving [218, 222]. They are also more resilient to adverse weather conditions like fog, rain, or dust compared to LiDAR and cameras. This resilience makes radar reliable in various environments and enhances its applicability in challenging conditions [223, 220]. Radar's ability to penetrate through multiple materials further extends its utility in different applications [219].

Disadvantages of radar

However, radar typically offers lower resolution compared to LiDAR, which can result in less detailed environmental maps and limit its effectiveness in detecting small or closely spaced objects [224, 220]. Additionally, both LiDAR and radar systems require precise calibration to function accurately. Any misalignment can lead to significant errors in distance estimation, complicating the setup and maintenance process [218]. The computational load required to process radar data can also be high, which might be a limiting factor for real-time applications with limited resources [207].

Discussion

Autonomous railway maintenance vehicles are often deployed in environments with frequent occlusion caused by moving equipment, workers, and surrounding infrastructure; the high vibration levels and adverse weather lightning conditions further complicate accurate depth estimation. While Radar is effective for all-weather and long-range operations, it is less suitable for railway maintenance tasks that require close human-robot or robot-robot collaboration, as its lower resolution limits its effectiveness at short distances.

Additionally, Radar provides less detailed information about the objects, such as complex shapes and textures, making it less ideal for integration with object detection tasks that demand precise identification and classification of railway-specific components. However, LiDAR is too expensive for widespread adoption. Additionally, their performance can degrade in adverse weather conditions such as fog, rain, or snow. Despite these challenges, LiDAR remains a powerful solution for many of the complex demands of autonomous railway maintenance tasks.

2.7.4 Machine learning methods

Machine learning-based methods for depth estimation have advanced significantly with the advent of deep learning. These methods can be broadly categorised based on the type of data they use: images and pointcloud/LiDAR data.

Image-Based Machine Learning (ML) Methods typically use CNN and other deep learning architectures to estimate depth from single or multiple images. The networks are trained on large datasets of RGB images paired with ground truth depth maps, learning to predict depth by identifying patterns, textures, and features within the Images. Techniques such as supervised, unsupervised, or self-supervised learning are commonly employed [225].

Pointcloud/LiDAR-based Methods: These methods leverage the 3D spatial information provided by pointcloud and LiDAR sensors. Neural networks like PointNet or other pointcloud-specific architectures process and analyse the 3D data [226, 227]. The raw pointcloud or LiDAR data is fed into the network, which learns to predict depth by understanding the spatial relationships and geometric features within the 3D data [222].

Both image-based and LiDAR-based methods face several common challenges, such as high computational requirements and dataset dependency. Training deep learning models for both types of data requires significant computational resources, including powerful GPUs and large memory capacities. This can be a barrier for real-time applications and systems with limited computational power [228]. Furthermore, the performance of these

models relies on the quality and quantity of the training dataset. Inadequate or biased datasets can lead to poor generalisation and inaccurate depth predictions in real-world scenarios [225].

Image-based methods can handle large datasets and be adapted to various applications, such as autonomous driving, robotics, and augmented reality. They can also be used with different types of visual data, including single images, stereo images, and video sequences [229]. Compared to LiDAR-based ML methods, there are more extensive datasets available for training image-based systems, which enhances their performance and generalisation capabilities in object detection and tracking [225]. Furthermore, image-based methods can be seamlessly integrated with object detection and tracking systems, leveraging the same visual data. This integration simplifies the workflow and reduces the need for additional sensors, making it cost-effective and more accessible to implement [225].

On the other hand, LiDAR and pointcloud data provide highly accurate and detailed 3D spatial information, making them ideal for precise depth estimation in applications like autonomous driving and robotics [222]. LiDAR is more robust compared to cameras and, therefore, less affected by environmental conditions such as lighting and weather, providing reliable depth information in various scenarios [221]. However, large LiDAR datasets for training are limited. This can impact the development and generalisation of LiDAR-based depth estimation models compared to image-based systems [221].

In summary, while LiDAR-based methods are better suited for high-precision depth estimation, image-based methods offer advantages in object detection and tracking due to their integration capabilities and the richness of available training datasets. Combining these approaches can leverage their complementary strengths, enhancing both depth estimation and object detection/tracking performance.

2.7.5 Discussion on depth estimation

This section has studied the various methodologies for depth estimation, highlighting the strengths and limitations of each approach. Accurate depth estimation is crucial for practi-

cal navigation, obstacle detection, and decision-making in dynamic railway environments, which often feature complex and variable conditions.

Monocular and stereo methods

Monocular vision methods are simple and cost-effective but face challenges such as depth perception ambiguity and scale ambiguity [206, 207, 208]. Stereo vision improves depth accuracy by comparing multiple viewpoints, but it still faces challenges in a limited range.

Radar-based methods

Radar systems are effective for long-range distance measurement and operate reliably in adverse weather conditions, making them suitable for applications requiring robust performance in challenging environments [218, 222, 223]. However, radar typically offers lower resolution compared to LiDAR, which can limit its effectiveness in detecting small or closely spaced objects [220, 224].

LiDAR-based methods

LiDAR-based methods provide accurate depth information, making them ideal for railway object depth estimation. Their ability to perform well in different lighting conditions enhances their applicability in different working scenarios [217, 218, 219]. However, these systems are typically expensive and complex, requiring precise calibration and maintenance. Additionally, their performance can be degraded by adverse weather conditions such as fog, rain, and snow [220, 221].

Machine learning methods

Machine learning-based methods for depth estimation, including image-based and pointcloud/LiDAR-based techniques, have advanced significantly with deep learning. These methods leverage large datasets to train neural networks that can predict depth from images or pointcloud data. Image-based methods benefit from extensive training datasets and integration

capabilities, enhancing performance in object detection and tracking [225]. However, they are less accurate in in-depth estimation tasks. On the other hand, LiDAR-based ML methods excel in in-depth estimation, but they are less precise in object detection and have limited datasets.

Summary

Railway maintenance tasks present various challenges for depth estimation due to the dynamic and complex nature of the operating environment, such as multiple lighting conditions and adverse weather conditions, which can affect sensor performance.

The combination of LiDAR and image-based sensors offers a powerful solution: the image-based approach contributes to contextual understanding and object detection tasks, LiDAR provides a high precision depth estimation for the detected objects. Furthermore, LiDAR can also generate detailed 3D models of the surrounding environment, which can be supplementary information for object detection tasks. On the other hand, radar provides limited resolution and poor performance at short distances; it can be integrated as a supplementary system alongside LiDAR and image-based sensors, and railway maintenance vehicles can benefit from its resilience to all weather and long-range performance.

Chapter 3

Methodology

3.1 Introduction

Built on advancements in automotive collision avoidance, this work draws on several widely adopted techniques from road-based autonomous systems. Models such as YOLOv8 and ByteTrack were selected not only for their proven performance across various perception tasks, but also for their balance of speed, accuracy, and architectural simplicity, making them well-suited for real-time deployment. Similarly, extensive research in the automotive domain has demonstrated the trade-offs between different sensing modalities: cameras offer rich semantic information but struggle with depth estimation and low-light conditions, while LiDAR provides precise spatial measurements but has limited object recognition capability. These insights informed the decision to adopt a sensor fusion strategy that combines stereo vision and LiDAR to compensate for individual limitations and enhance system robustness in challenging railway maintenance environments. Furthermore, deep learning practices such as TL were employed to leverage pre-trained models and address the scarcity of annotated railway-specific datasets. Inspired by modular architectures used in other robotic systems, the proposed design follows a structured perception-to-action pipeline—including detection, tracking, depth estimation, and decision-making—to ensure robust, real-time operation. Environmental factors such as

variable lighting and sensor noise were also considered in the validation strategy.

These methodological adaptations reflect a thoughtful application of cross-domain knowledge to railway-specific needs and provide the foundation for the system development. As shown in Figure 3.1, the development is structured into five main sub-tasks: (1) designing the robotic framework, (2) implementing railway-specific object detection methods, (3) implementing a 3D object tracking system, (4) system integration and development of the collision avoidance system, and (5) validating the system through rigorous testing.

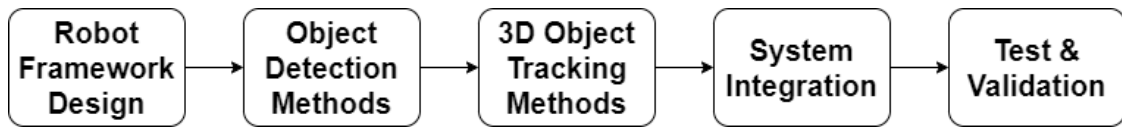


Figure 3.1: Overview of the methodological framework

The ROS2 robotic framework serves as a foundation for integrating various sensors and instruments to implement the autonomous capabilities of the robot [230]. ROS2 is an open-source robotic framework that provides tools and libraries to help design, build, and deploy robot applications. Compared to ROS1 [231], ROS2 offers several improvements, including enhanced performance, real-time capabilities, better security features, and improved support for multi-robot systems. The integration of ROS2 also facilitates modularity and scalability, allowing for future enhancements and expansions of the system. In this project, ROS2 is used to manage the communication between sensors and make decisions on the vehicle’s actuation by processing the sensor input for object detection and tracking algorithms.

Object detection allows the vehicle to identify and differentiate between various types of railway objects, such as trespassers and workers. This differentiation enables the vehicle to make an adaptive response according to the type of object detected beyond the simple “start-and-stop” solution, regardless of the object type. For example, the vehicle can continue to operate and collaborate with maintenance workers but must stop if any trespasser is detected.

3D object tracking enhances this capability by monitoring the movement and maintaining a history of the trajectories of these detected objects over time. This allows the robot to predict the future positions of objects and respond adaptively to their movements. For example, the robot can adjust its speed based on the objects' speed, therefore avoiding complete shutdown to improve efficiency.

The rest of the chapter is structured as follows:

- **Section 3.2** explains the design of the autonomous robotic framework using ROS2, highlighting hardware setup, software configuration, and system integration.
- **Section 3.3** explains the methods to differentiate between different railway objects with TL, including dataset preparation, model training, and evaluation metrics.
- **Section 3.4** explains the methods used to track objects in 3D space, focusing on the tracking algorithm, system integration and validation.

3.2 Robotic framework design and system integration



Figure 3.2: Original bance rail trolley with hand controller



Figure 3.3: Converted RIMV with autonomous capability

After identifying the two core subsystems for context-aware collision avoidance, object detection, and 3D object tracking, it is crucial to design a robotic framework to integrate these two systems. This framework serves as the backbone that integrates these

subsystems with the vehicle's perception, control and navigation systems to enable intelligent decision-making. The robotic framework should provide the essential infrastructure for integrating different components of the vehicle and ensure seamless communication between sensors, control systems, and the central processing unit, allowing the RIMV to make real-time decisions based on sensor data. Specifically, the objectives include:

- **ROS2 integration:** implement ROS2 as the backbone framework to ensure modularity, scalability, and robust communication between different components of the RIMV.
- **Modularity and scalability:** design a modular and scalable architecture to facilitate easy integration of additional sensors or functionalities as required. This scalability ensures that the system can adapt to different operational requirements and future upgrades.
- **Sensor integration:** integrate various sensors, including GPS, IMU, LiDAR, and cameras, to provide comprehensive environmental perception and localisation capabilities. Utilise these sensors for robotic perception, enabling the system to detect and interpret the surrounding environment.
- **Navigation and vehicle actuation:** implement advanced navigation and control algorithms that allow the vehicle to navigate complex railway environments while avoiding obstacles autonomously.
- **Real-time data processing:** develop computation-efficient data processing pipelines to ensure real-time processing of sensor data for immediate decision-making and control.
- **Validation and testing:** conduct testing and validation of the robotic framework in real-world environments to ensure its reliability and safety.

3.2.1 Overview of ROS2

ROS2 is an open-source robotic framework that provides a collection of tools and libraries to simplify the task of creating complex robot applications. ROS2 was chosen as the backbone of the robotic framework due to its flexibility, modularity, and extensive support within the robotics community. One of the critical strengths of ROS2 is its ability to integrate with a wide range of hardware interfaces, sensors, actuators, and control algorithms. ROS2 has a large and active community, providing third-party open-source libraries to streamline the development process. Additionally, ROS2's modular architecture enables developers to create reusable and interchangeable modules, facilitating rapid prototyping and iterative development. The communication infrastructure of ROS2 is based on a publish-subscribe messaging system, which provides reliable data exchange between different subsystems, facilitating real-time robotic applications. ROS2 introduces multiple enhancements over its predecessor, ROS1, particularly regarding real-time capabilities, flexibility, and security:

- **Real-time capabilities:** ROS2 employs the Data Distribution Service (DDS) as its communication middleware, which supports real-time communication, meeting the demanding requirements of real-time distributed embedded systems [232]. Studies have shown that ROS2 can achieve significant improvements in communication latency and jitter, providing more reliable and predictable performance for time-critical applications [233].
- **Improved security:** ROS2 includes built-in security features through DDS, which provide support for authentication, encryption, and access control. This ensures a more secure robotic ecosystem compared to ROS1 [234].
- **Modular and scalable architecture:** ROS2 supports a more modular architecture, which allows for better scalability and flexibility. This is beneficial for complex robotic systems that require distributed processing and easy integration of new components and devices [235].

- **Cross-platform support:** unlike ROS1, which is primarily limited to Unix-based systems, ROS2 offers better cross-platform support, including Windows and macOS. This enhances the development and deployment flexibility for various users and environments [236].

In summary, ROS2 offers significant advancements over ROS1. Its real-time capabilities, enhanced security, improved middleware, cross-platform support, and modularity make ROS2 an ideal choice for the backbone of the robotic framework for autonomous railway vehicles. Detailed instructions¹ for installing ROS2.

3.2.2 System architecture

Figure 3.4 illustrates the system architecture and simulation framework interacting between its real-world and simulated environments through the ROS2. In the real-world environment, the sensory system detects external influences on the RIMV. These sensor inputs are fed into the ROS2-based computer, which processes the data and makes decisions regarding navigation and actuation commands. These commands are sent to the ESP32 microcontroller using the `micro_ros_arduino` package² tailored for embedded systems [237], to execute actuation commands such as motion and lighting.

The right part of Figure 3.4 shows the simulated counterpart. The Gazebo simulator provides a physics engine and graphics, as well as programmatic and graphical interfaces. The simulated components match their real-world counterparts on the left block, with relevant dynamics and external influences such as sensor noise. This simulation environment reduces the time and cost associated with real-world testing and streamlines the development, iteratively refining the system in simulation before deployment.

¹ Available at: <https://docs.ros.org/en/foxy/Installation/Ubuntu-Install-Debians.html>

² Available at: https://github.com/micro-ROS/micro_ros_arduino/tree/foxy

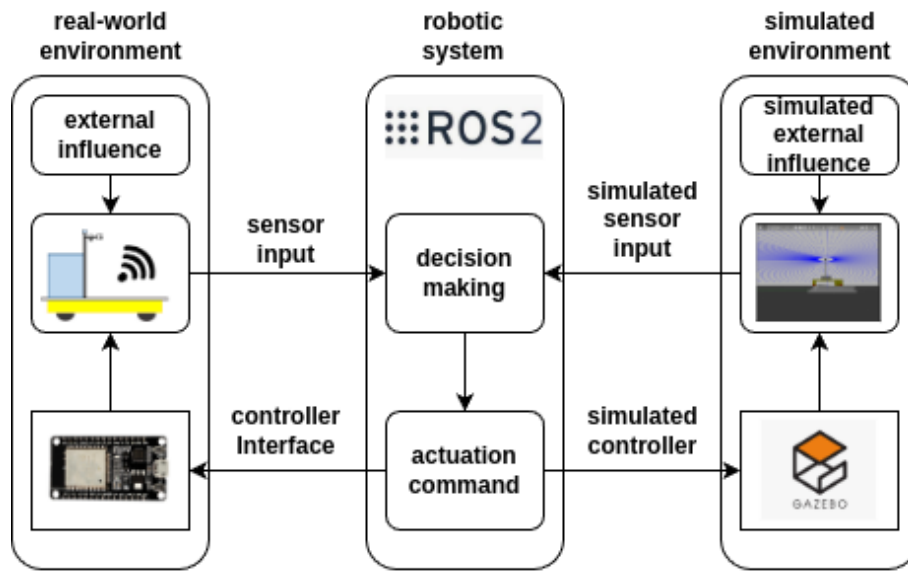


Figure 3.4: System architecture and simulation framework

3.2.3 Hardware integration and vehicle actuation

Figure 3.5 illustrates the system architecture and dataflow and highlights the integration of various hardware components and their interactions through the ROS2 framework. The left part of the diagram shows all the sensory systems responsible for the fundamental functionalities of RIMV, such as vehicle localisation and environmental perception. These sensors communicate to the ROS2 machine in the middle part of the diagram. It processes the input from the sensory system from the left part of the diagram and makes decisions according to the vehicle's and surrounding obstacles' position. The ROS2 machine also allows human operators to control the vehicle remotely or activate emergency stop(E-Stop). Finally, the right part of the diagram shows the actuation devices, such as the motor, alarm, and LED. The motor is responsible for the vehicle's movement, and the alarm is activated before any movement to alert nearby users. The LED light indicates the speed and direction the vehicle travels. It also reserves a USB serial and analogue connection for other inspection and maintenance tasks, such as the ultrasonic kit.

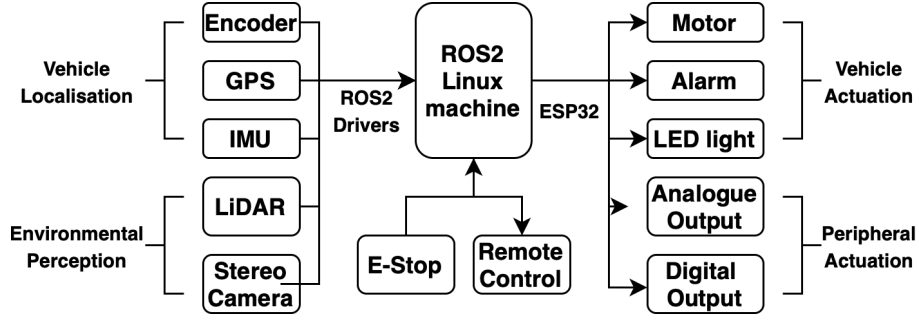


Figure 3.5: System architecture and dataflow for RIMV

3.2.4 Sensor list

Table 3.1 shows the sensors used for the robotic framework. They provide perception input for the vehicle, and they can be categorised based on their functionalities: localisation, object detection, and depth estimation.

Item	Provider	Model (Specs)
IMU	Parker LORD	3DM-GX5-AHR
GPS	Novatel	Smart 7
LiDAR	Velodyne	Ultra Puck
Wheel encoder	Generic	(600 pulses/rev)
Stereo camera	Intel	RealSense D435i
Environmental camera	Axis	M5525-E

Table 3.1: Sensors and equipment used in RIMV

For localisation, the RIMV primarily relies on the GPS to navigate along railway tracks. When GPS signals are denied or unavailable, the IMU and wheel encoder become crucial. The IMU provides measurements of the vehicle’s orientation, acceleration, and angular velocity, making them widely used in modern localisation systems [238, 239]. The wheel encoder detects and records the rotation of a wheel. It provides incremental movement data to estimate the position of a vehicle. For object detection, The environmental camera and depth camera provide video streams that can be used by machine learning algorithms for real-time monitoring and inspection, allowing for the detection and identification of various objects in the vehicle’s path. For depth estimation (3D object tracking), the LiDAR sensor generates detailed 3D maps of the environment, facilitat-

ing obstacle detection and avoidance. Additionally, the depth camera integrated into the stereo camera provides further depth information.

3.2.5 Evaluation of sensor selection

The sensor selection was evaluated based on their ability to meet the demands of two primary tasks: object detection and depth estimation. For object detection, vision-based systems—particularly RGB cameras—are highly advantageous due to their ability to capture rich semantic information suitable for classification tasks. While LiDAR has been explored for object detection, its limited semantic resolution—owing to the absence of texture and colour—makes it less effective for discriminating between object categories. LiDAR provides valuable geometric cues such as shape, size, and position, but the lack of annotated railway-specific pointcloud datasets restricts the development and generalisability of learning-based LiDAR detection models. Thermal and infrared cameras, though capable in low-light conditions, offer limited classification capability and are costly, with performance degradation in warmer environments. Radar also lacks the semantic richness and spatial resolution necessary for object detection in complex railway scenes.

In contrast, depth estimation focuses on recovering accurate spatial information. LiDAR is widely regarded as the gold standard, offering high-resolution 3D pointcloud across a broad range of distances and lighting conditions. Stereo vision is another effective option, providing mid-range depth perception through disparity triangulation between image pairs. When fused with RGB data, stereo cameras enable both semantic and geometric interpretation. However, stereo depth accuracy diminishes with distance due to reduced disparity. Monocular depth estimation was also considered for its low cost and software-only implementation, but it is inherently less reliable, particularly at long distances and under varied lighting. These methods require large, annotated datasets and suffer from poor generalisation in new environments. Radar offers coarse range estimation for large metallic objects but lacks the resolution for fine-grained 3D tracking or obstacle localisation. Infrared sensors are generally ineffective for reliable depth estimation due to

their limited range and sparse spatial representation.

Sensor robustness was also evaluated. Cameras are sensitive to lighting changes, glare, and adverse weather such as rain or fog. While LiDAR is more resilient to lighting variation, it may experience degradation in snow, dust, or heavy rain due to scattering and signal noise from airborne particles. Given these trade-offs, a stereo camera and LiDAR combination was selected to balance semantic and geometric perception. LiDAR offers robust distance measurements under diverse conditions, while stereo vision enables mid-range depth estimation and real-time object classification. This fusion ensures accurate and adaptable perception in railway environments. Although radar and ultrasonic sensors were excluded in this iteration due to resolution and range limitations, they remain viable options for enhancing redundancy and robustness in future designs, particularly under extreme or weather-compromised conditions.

3.2.6 Collision avoidance architecture overview

Figure 3.6 illustrates the high-level logic for the proposed collision avoidance system, where the input module integrates stereo vision and LiDAR to enable robust perception and context-awareness — identifying both the obstacle type and its spatial position. This contextual information is passed to the decision-making module, which interprets the threat level based on object type (e.g., pedestrian vs. maintenance worker) and spatial location. The actuation module then executes these commands, which may involve changing speed, activating an emergency brake. This modular perception-to-actuation design ensures the system can operate reliably under real-time constraints and dynamic railway environments.



Figure 3.6: Context-aware decision logic for collision avoidance.

3.2.7 Sensor reliability and safety mechanisms

To ensure safety and robustness under sensor degradation, the proposed collision avoidance system employs a two-layered safety mechanism. The primary layer relies on stereo vision for object classification and LiDAR for spatial localisation, forming the foundation of the context-aware decision-making process. In parallel, a secondary fail-safe mechanism operates independently of the perception pipeline. It defines a fixed spatial buffer zone along the vehicle's forward travel path, which is continuously monitored by LiDAR to detect any potential intrusions. If any object—regardless of type or appearance—enters this buffer zone, the system immediately triggers an emergency stop. This design provides a critical safeguard in situations where the stereo camera may fail to detect obstacles due to poor lighting, vibration-induced blurring, or algorithmic limitations. Additional redundancies include four onboard emergency stop buttons, a wireless E-stop remote, and a monitoring system that halts the vehicle if the remote becomes disconnected or moves out of range.

At present, the system does not implement formal conflict-resolution logic for discrepancies between LiDAR and camera data, such as inconsistent depth estimates. In practice, for single-object scenarios, LiDAR measurements are prioritised, as they offer more reliable distance information. Moreover, because the LiDAR module activates only after a valid detection from the stereo camera, isolated false positives from the vision system are inherently suppressed. While this approach is effective in the current application, it does not fully address more complex, multi-object scenarios. Future improvements may involve consistency checks or confidence-based fusion to better handle cross-modal disagreements. Further enhancements such as radar integration for adverse weather conditions (e.g., fog, rain, or snow) or filtering logic to ignore low-risk objects like vegetation could also strengthen system reliability.

Despite the advancements, the system remains a prototype developed for research purposes and is not certified under any formal industry safety standard. However, its design aligns with key principles from railway relevant safety standard and broader autonomous

systems domains. In the railway sector, the EN 50126 standard outlines a structured lifecycle approach for achieving RAMS (Reliability, Availability, Maintainability, and Safety) [240], while EN 50129 specifies requirements for validating safety-related electronic systems [241]. RIMV’s design reflects EN 50126-aligned principles through structured system decomposition, staged validation, and traceable modifications. Similarly, it implements EN 50129-aligned practices such as functional separation of perception and emergency stop modules, redundant fail-safe actuation, and runtime monitoring of sensor health and wireless connectivity.”

From an automation perspective, the railway sector currently lacks a universally adopted autonomy standard. Therefore, the classification framework defined by the Society of Automotive Engineers (SAE) in standard J3016 is used as a point of reference. SAE J3016 outlines six levels of driving automation (Levels 0–5), describing increasing degrees of autonomy, though it serves as a descriptive taxonomy rather than a formal safety certification standard [242]. Within this framework, the RIMV roughly meets SAE Level 2, which is defined as partial driving automation. In this context, the RIMV performs driving tasks autonomously under normal operating conditions. However, a human operator remains responsible and can intervene at any time through multiple emergency stop buttons. RIMV incorporates foundational safety mechanisms that align with Level 2 autonomy and support future alignment with both railway and cross-domain safety expectations.

3.2.8 Modularity and scalability

To ensure long-term adaptability and facilitate the integration of future functionalities, the proposed robotic framework was designed with both modularity and scalability as core principles of its system architecture.

ROS2 architecture: the system builds on ROS2’s node-based architecture, where each subsystem (such as detection, tracking, and localisation) is implemented as an independent ROS2 node. Data exchange between nodes is achieved through ROS interfaces (top-

ics, services, or actions). For example, a sensor driver node publishes data to a specific topic, and a processing node receives the data by subscribing to that topic. This message-passing architecture ensures that nodes remain loosely coupled, interacting through standardised communication interfaces rather than direct function calls. As a result, components can be tested, upgraded, or replaced independently, facilitating flexibility and ease of integration for future extensions.

Algorithm and code organisation: the software is structured as a sequence of modular processing stages, with each stage responsible for a distinct task. This structure avoids monolithic codebases and enables independent development, debugging, and replacement of specific functionalities. New algorithms can be introduced by substituting individual modules without rewriting the full pipeline. Runtime configuration is supported through YAML files and modular launch structures, simplifying tuning and testing processes.

Simulation support: a Gazebo-based simulation environment emulates sensor dynamics and robot–environment interactions, supporting early-stage development, testing, and validation. This setup enables scalable experimentation, such as evaluating new sensing strategies before hardware deployment, thereby reducing risk and accelerating iteration cycles.

Sensor and system extensibility: the system is designed without hardcoded assumptions about sensor types or object classes. New sensors or additional cameras can be integrated by introducing new ROS2 nodes. On the hardware side, the vehicle includes reserved digital and analogue ports to accommodate future expansion without requiring structural modification.

Distributed deployment: ROS2’s DDS-based communication enables real-time, distributed operation across multiple components or machines. This supports scalability to larger platforms or multi-robot systems while preserving modularity and performance un-

der real-time constraints.

3.3 Railway object detection

3.3.1 Detection framework selections

As discussed in Section 2.4, object detection is a critical component of a collision avoidance system; it differentiates between different types of railway objects and enables the vehicles to respond differently to the object types. Machine learning-based techniques have advanced significantly over the past decades and can be categorised into one-stage and two-stage detectors, each with its distinct advantages and disadvantages.

One-stage detectors such as YOLO and SSD directly predict bounding boxes and class probabilities in a single forward pass of the network. They treat object detection as a regression problem over spatial locations and categories, without a separate proposal stage. This streamlined pipeline significantly improves inference speed and computational efficiency, making one-stage detectors ideal for real-time applications on embedded or mobile platforms. However, their simplified architecture often leads to lower detection accuracy, particularly in detecting small or densely packed objects, as they rely on coarse feature maps and lack the refinement process available in multi-stage frameworks [110, 106]. Two-stage detectors, such as those in the R-CNN family, follow a different strategy. The first stage generates region proposals using methods like selective search or an RPN, which hypothesise object locations. The second stage then classifies these proposed regions and refines their bounding box coordinates. This two-step process allows for more accurate object localisation and class prediction, particularly beneficial in complex or cluttered scenes. However, this additional computation increases latency and resource requirements, making two-stage detectors less suitable for real-time applications [66, 65].

Given the need for both speed and accuracy in real-time railway object detection, YOLOv8 was chosen for this project. It introduces several improvements in network ar-

chitecture, including advanced anchor-free detection heads and a more efficient C2f-based backbone, which contribute to higher detection accuracy and faster inference compared to its predecessors, making YOLOv8 suitable for safety-critical real-time applications. As an open-source framework, YOLOv8 benefits from wide community support and documentation. It includes a mature ecosystem for training, evaluation, deployment, and integrates well with widely used machine learning platforms such as PyTorch and TensorRT. These features simplify the development pipeline and facilitate seamless integration into robotic systems like ROS2.

From a data perspective, YOLOv8 is well-suited to small, specialised datasets. Although two-stage detectors like Faster R-CNN can theoretically offer higher detection performance, they typically require a large training dataset. In contrast, YOLOv8 is designed for fast adaptation to small datasets and requires no architectural changes, enabling straightforward fine-tuning with high-quality pre-trained weights from the COCO dataset. The support for TL makes YOLOv8 suitable for domain-specific applications such as railway maintenance environment, where annotated data is scarce.

Lastly, railway maintenance vehicles require long stopping distances and close-proximity human-machine interactions, where perception systems must operate with low latency. YOLOv8's streamlined architecture ensures fast inference, enabling fast decision-making and responsive actuation. Its lightweight design makes it suitable for deployment on mobile robotic platforms with limited processing power and computational capacity. Additionally, its compatibility with ROS2 support easy integration into the RIMV's robotic architecture.

3.3.2 Problem definition

Limited dataset for railway object detection

Learning-based object detection algorithm typically requires extensive datasets for effective training [243]. However, one primary challenge in railway maintenance object

detection is the need for specialised datasets. While datasets like COCO provide extensive annotated images for general object detection [37], they lack specific contextual information for railway scenarios. This scarcity creates a bottleneck in training a model to effectively detect railway objects despite the recent advance in object detection algorithms.

TL is a machine learning technique where a model developed for a particular task is reused as the starting point for a model on a relevant task (Section 2.5). This approach can leverage knowledge gained from one domain to enhance the performance of a new model in a relevant domain with a limited dataset. In the context of railway object detection, TL can mitigate the challenge of dataset scarcity by using a pre-trained model on a large, diverse dataset, such as COCO or RailSem19, and fine-tuning it with a smaller, domain-specific railway dataset. This approach reduces the need to create an extensive railway maintenance-specific dataset and streamlines the development cycle, ultimately advancing the capabilities of autonomous railway maintenance vehicles.

Mathematical definition of transfer learning

TL involves leveraging knowledge acquired from one or more source problem domains and tasks to enhance learning in a related but distinct target domain and task. The following explains the mathematical foundation for TL.

Domains and tasks in TL: a domain $\mathcal{D}$ is defined by a feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$, where $X = \{x_1, x_2, \dots, x_n\}$ are samples from $\mathcal{X}$. A task $\mathcal{T}$, associated with domain $\mathcal{D}$, involves a label space $\mathcal{Y}$ and an objective predictive function $f(\cdot)$, inferred from training data comprising feature-label pairs (x_i, y_i) , with each x_i as a feature vector in $\mathcal{X}$ and y_i as the corresponding label in $\mathcal{Y}$.

TL framework: the objective of TL is to improve the learning of a target predictive function $f_T(\cdot)$ in a target domain $\mathcal{D}_T$ by leveraging knowledge from a source domain $\mathcal{D}_S$ with its related task $\mathcal{T}_S$. Mathematically, this involves using the knowledge where

$f_S(\cdot)$ from $\mathcal{T}_S$ assists in learning $f_T(\cdot)$ in $\mathcal{D}_T$ under the condition that either $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$. This transfer is predicated on the relationship between the source and target domains and tasks, which can significantly affect the efficacy of the transferred knowledge [11, 118].

Importance of relatedness in TL: the success of TL is contingent upon the relatedness between the source and target domains ($\mathcal{D}_S$ and $\mathcal{D}_T$) and their respective tasks ($\mathcal{T}_S$ and $\mathcal{T}_T$). It is through understanding and quantifying this relatedness that TL algorithms can effectively apply knowledge from the source to improve performance in the target domain [11, 118].

What to transfer in transfer learning?

Determining what is more beneficial for TL - using a larger, general dataset or a relatively smaller, railway-specific dataset - remains a fundamental question. Larger datasets like COCO offer a broad range of objects and sufficient images, providing extensive feature representations for object detection models. However, these datasets lack the specific contextual details for railway environments. On the other hand, RailSem19 [38] is a smaller, railway-specific dataset; it contains images and objects directly relevant to the railway domain. These datasets capture the unique characteristics and provide specific contextual details for the railway environments. The challenge lies in determining whether the extensive but general features from larger datasets outweigh the benefits of specialised but limited railway-specific data.

Another critical aspect is understanding how object class numbers can impact TL performance. When transferring learning from a source dataset, the number of object classes and their variety play a significant role. A dataset with a higher number of object classes can provide richer feature representations, potentially enhancing the model's ability to generalise across different objects. Contrastingly, more object classes might dilute the model's effectiveness, leading to worse performance. Therefore, it is essential to find

the trade-off in object class to cover diverse railway objects while maintaining sufficient examples of each class to train the model effectively.

Experiment objectives

The object detection experiment aims to investigate the following aspects of TL:

- Evaluate the TL performance from different source targets: COCO and RailSem19.
- Evaluate how the object class number in RailSem19 impacts TL performance.
- Evaluate different TL and fine-tuning strategies.

3.3.3 Experiment overview

The following list provides an overview of the methodology for the railway object detection experiment. For a detailed experiment design, refer to Section 4.3. As shown in Figure. 3.7, The following datasets were used in the experiment:

- **Custom railway (target) dataset** comprises 103 images with six classes, specifically created to capture the unique characteristics and objects found in railway maintenance scenarios, including pedestrians, workers, signals, and other relevant railway objects.
- **COCO (source) dataset [37]**, a large, general dataset containing around 330k images with 80 classes, was used with pre-trained weights from YOLOv8 for TL experiments, providing extensive feature representations despite lacking specific railway contextual details.
- **RailSem19 (source) dataset [38]**, consisting of 8,500 images with 21 classes, focuses on urban railway environments and includes a variety of railway-specific objects.

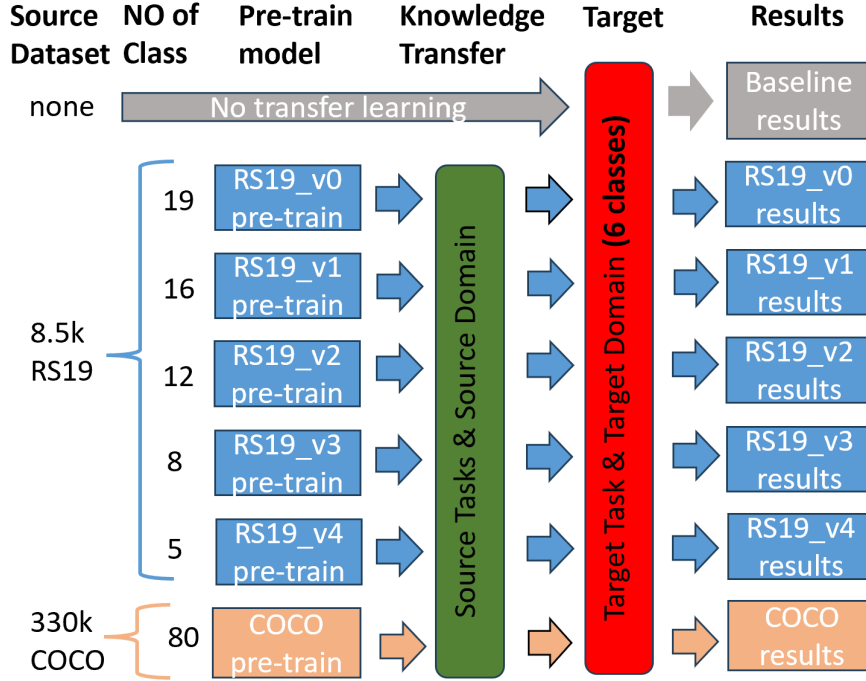


Figure 3.7: Experimental design for object detection with TL

Figure 3.7 shows an overview of the object detection experiment. The original RailSem19 dataset annotations utilise polygons and polylines to describe object boundaries and linear features. These annotations need to be converted to a bounding box representation compatible with YOLO architecture. Polyline annotations, representing linear features like rails and guardrails, were removed as they do not align with the detection objectives of this study.

The RailSem19 dataset was then modified by reducing the number of classes to evaluate how the class diversity in the source dataset can impact TL. **RS19_v0** contains 19 classes after removing the rail and guardrail. **RS19_v1** contains 16 classes by combining similar types of switches (left, right, unknown) and signals (front and back). **RS19_v2** focuses more on railway scenarios by removing non-relevant components such as trucks and cars, resulting in 12 classes. **RS19_v3** aligns more closely with the custom dataset, including only objects that appear in (or help to locate) the custom dataset, reducing the count to 8 classes. Finally, **RS19_v4** contains only five classes that appear in the custom dataset. These modifications allow for a systematic evaluation of how the number

of object classes in the source dataset affects the TL performance on the custom railway dataset. By creating these subsets, the study aims to identify the optimal balance between dataset relevance and class diversity for effective object detection in railway scenarios.

The best-performing subset from the RailSem19 modifications will be chosen and then be compared with the model pre-trained on the COCO dataset to evaluate what is more beneficial for TL as a source dataset: a larger, general dataset like COCO or a smaller, domain-specific dataset like RailSem19. This comparison aims to provide insights into the effectiveness of dataset relevance versus quantity in enhancing railway object detection. Additionally, both models will be compared with the baseline experiment (without TL) to assess the overall improvement and effectiveness of TL. Table 3.2 shows the comparison for the source dataset used.

Feature	COCO Dataset	RailSem19 Dataset
Dataset size	Large (over 330K images)	Medium (8.5K)
Classes NO.	80	19
Annotation type	Bounding boxes	Polygons
Application	General object detection	Railway scene understanding
Image source	Varied	Railway environments

Table 3.2: Source dataset comparison

3.4 3D object tracking

In this project, 3D object tracking refers to the tasks of object detection and tracking with distance estimation. These tasks are challenging due to the trade-offs between the task requirement of detection tracking and distance measurement, as well as the different sensors' characteristics and capabilities.

Vision-based sensors, such as cameras, capture high-resolution images that contain rich visual information, such as texture, colour, and shape, enabling the systems to distinguish between different objects and making them suitable for object detection and track-

ing. However, vision-based systems are less accurate in measuring the distance compared to LiDAR, especially in varying lighting conditions and complex environments.

On the other hand, LiDAR excels in distance measurement and environment perception; it generates a detailed 3D pointcloud map that accurately describes the geometric information about the surrounding environment and allows the system to determine the location and movement of objects much more accurately compared to vision-based sensors. However, LiDARs are less effective at object detection compared to vision-based systems.

To address these limitations, this project adopts a sensor fusion-based approach that combines the strengths of both sensors, enabling a robust and accurate 3D object tracking system suitable for complex railway environments.

3.4.1 Methodological approach overview

The following list outlines the methodology employed in developing a multi-modal 3D object-tracking algorithm for a context-aware collision avoidance system. This approach combines stereo camera and LiDAR sensors to achieve accurate object detection and distance measurement. The ability to accurately detect and track objects in a 3D environment enables the collision avoidance system to make informed decisions to avoid collisions and facilitate more adaptive path planning based on the position and movement of objects. The methodology includes the following key components:

- **Integration of YOLOv8 for object detection and tracking:** Implementation of YOLOv8, which incorporates both BoT-Tracking and ByteTrack algorithms, for real-time object detection and tracking within ROS2. This component detects object types, identities, and their 2D locations within the image frame.
- **Data processing and transformation:** Convert 2D bounding boxes from the camera's coordinate into 3D bounding boxes in the LiDAR coordinate system based on the transformation matrix. This process constructs an adjustable frustum based on

the object type and the initial depth estimation from the stereo camera.

- **Pointcloud clustering and distance measurement:** Determination of suitable point-cloud clustering algorithms for object clustering within the frustum and more accurate distance measurement using the LiDAR pointcloud data.
- **System integration and collision avoidance development:** Integration of the object detection and 3D tracking modules into a real-time collision avoidance system. This includes decision-making algorithms that process detected obstacles' object types and movements and generate appropriate actuation commands.
- **Validation:** Indoor validation to test the system's robustness under various lighting conditions. Outdoor validation to test the system's performance in real-world railway environments.

By combining advanced object detection algorithms with robust sensor data fusion, the proposed methodology aims to enhance the reliability and precision of collision avoidance systems for autonomous railway vehicles, contributing to safer and more efficient autonomous railway operations.

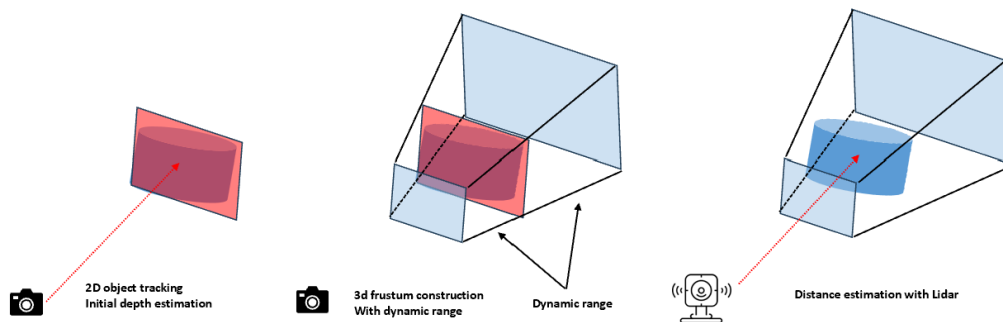


Figure 3.8: 3D tracking methodology illustration

As illustrated in Figure 3.8, the algorithm begins with the 2D object tracking using a stereo-vision based on the YOLOv8 framework. The algorithm identifies objects within the stereo camera's field of view and provides their object type, object identification and 2D bounding box coordinates. These 2D coordinates are then used with stereo camera

data to calculate the disparity between images captured from the stereo camera's different angles. This step provides a preliminary estimate of the object's distance from the camera. Following this, a 3D frustum is constructed based on the initial depth estimation and the size of the 2D bounding box. The frustum extends from the camera through the bounding box into 3D space, with its range dynamically adjusted according to the object class. For instance, the frustum may extend by ± 1 meters for a human and ± 2 meters for larger objects. Finally, the algorithm uses LiDAR to refine the distance measurements within the frustum, allowing for precise distance estimation. The LiDAR focuses on the pointcloud data only within the frustum, filtering out irrelevant information and ensuring accurate tracking and distance measurement compared to solely relying on the vision-based system. This combination of visual detection, frustum construction, and LiDAR refinement creates a robust system for 3D object tracking in complex railway environments.

3.4.2 Validation plan

The validation of the 3D object tracking and integrated collision avoidance system for RIMV is structured into two phases: **robustness validation** (indoor) and **collision avoidance system validation** (outdoor). This approach ensures a comprehensive evaluation of the system's performance, accuracy, and reliability under varying conditions.

- **Robustness validation (indoor)** assesses the system's resilience under various environmental challenges within a controlled indoor setting. This phase examines the system's performance under different lighting conditions, including various brightness across different times of the day, artificial light from RIMV or lights from other passing vehicles during low-light environments. Ensuring the system's tracking and distance measurement remain reliable in varied operational settings.
- **Collision avoidance system validation (outdoor)** evaluates the fully integrated collision avoidance system in both indoor and real-world railway settings at the Railway Innovation Test Area (RITA) at Cranfield University. This phase tests the

system's ability to accurately classify and respond to dynamic obstacles, track and predict their movements in real-time and execute appropriate actuation decisions based on the detected objects' types and movement patterns. This comprehensive assessment confirms the system's readiness for deployment in live railway maintenance operations, ensuring safe and efficient autonomous functioning.

Each validation phase is designed to target specific aspects of the system, providing a thorough examination of its capabilities and identifying areas for further improvement. Detailed methodologies, experimental setups, and evaluation metrics for each validation phase can be found in Section 5.3.

Chapter 4

Object Detection with Transfer Learning

4.1 Introduction

4.1.1 Experiment overview

TL is a machine learning technique that reuses a pre-trained model as the starting point for a new task. This approach is beneficial when the new task has a limited (target) dataset, allowing the model to leverage the knowledge gained from a larger, related (source) dataset. By reusing and fine-tuning the pre-trained models, TL can significantly enhance performance and reduce the time and resources required to train a model from scratch.

This experiment uses three datasets: two source datasets for pre-training and one target dataset for fine-tuning and evaluation. The source datasets include the COCO [37] and RailSem19 [38] datasets. The COCO dataset is a large, general-purpose dataset with 330k images and 80 object classes, widely used for various computer vision tasks. The RailSem19 dataset focuses on the urban railway environment and includes 19 object classes and 8.5k images. Finally, the custom dataset created for this study, which consists of 6 object classes and 103 images relevant to railway maintenance scenarios, is used as the target dataset.

The experiment follows a systematic approach to evaluate the TL performance by pre-training models from different source datasets (COCO and RailSem19), then leverage knowledge from these models to re-train with the custom railway dataset. This experiment is designed to answer critical research questions: whether pre-training on a large, general dataset like COCO or a smaller, domain-specific dataset like RailSem19 provides better performance in terms of mAP50 and F1 scores for railway-specific object detection tasks.

Additionally, the experiment also investigates how the number of object classes in the RailSem19 dataset affects the performance of TL. The custom railway dataset has six classes, while the original RailSem19 dataset contains 21 classes. For this study, the RailSem19 dataset is modified into different variations by reducing the number of classes and, finally, matching the object class within the custom dataset. This allows for a systematic analysis of how class diversity impacts performance in TL.

Furthermore, one-stage pre-training involves training models solely on the COCO or RailSem19 dataset. In contrast, two-stage pre-training involves an initial training phase on more than one dataset. The one-stage approach directly adapts the model to the railway domain, potentially capturing more relevant features for railway object detection. In contrast, the two-stage approach first benefits from the broad and diverse features learned from the COCO dataset, which are then refined and adapted to the railway domain through further pre-training on the RailSem19 dataset. This comparison aims to determine which strategy provides better performance for railway object detection.

Lastly, Layer freezing is a technique used during fine-tuning where specific layers of a pre-trained model are kept unchanged or “frozen”. Freezing early convolutional layers can help preserve low-level features such as edges and textures, which are generally beneficial across different tasks and domains. However, freezing too many layers, including middle and late layers that capture more complex and task-specific features, can restrict the model’s ability to adapt to the new dataset. This study investigates the impact of different freezing strategies to understand how they affect the performance of TL for railway object detection.

By systematically comparing TL with different pre-trained models and fine-tuning techniques, This experiment seeks to provide insights into the most effective strategies for enhancing object detection in railway maintenance with a limited dataset, ultimately contributing to the advancement of autonomous railway vehicle navigation and safety systems.

4.1.2 Objective and hypotheses

The objective of this chapter is to evaluate the impact of various TL strategies on railway object detection, it specifically aims at:

- Investigate the effectiveness of TL when pre-training on a larger, general-purpose dataset (COCO) versus a smaller, domain-specific dataset (RailSem19).
- Compare the performance of one-stage pre-training against two-stage pre-training approaches.
- Examine how the number of object classes in the source dataset (RailSem19) influences the effectiveness of TL.
- Assess the impact of different layer freezing techniques on the TL process.

Based on these objectives, the study formulates the following hypotheses:

- TL from a larger, general dataset like COCO will improve the overall detection performance.
- TL from a more relevant but smaller railway-specific dataset like RailSem19 will yield better performance for railway-specific objects.
- Reducing the number of object classes in the source dataset to the most relevant ones will improve performance on the custom railway dataset.
- Two-stage TL (first on COCO, then on RailSem19) will outperform one-stage TL on either dataset alone.

- Freezing early convolutional layers will improve performance while freezing too many layers will decrease performance.

4.1.3 Structure of the chapter

The rest of the chapter is structured as follows:

- **Section 4.2 Dataset preparation:** documents the construction process of the custom railway dataset, the modification of the RailSem19 dataset into subsets, and the rationale for these modifications.
- **Section 4.3 Experiment design:** outlines the design and setup of the experiment, including the baseline experiment, pre-training on RailSem19 datasets, TL and fine-tuning on the custom railway dataset.
- **Section 4.4 Results and discussion:** provides the evaluation metrics and performance comparison. It also includes an analysis of the best-performing model and dataset and a discussion of findings and implications.
- **Section 4.5 Conclusion:** summarises the key findings of the experiment, discusses the contributions to the field, outlines the practical implications for the railway industry, and provides recommendations for future work.

4.2 Dataset preparation

4.2.1 Overview of dataset used in this experiment

This experiment uses three datasets: COCO [37], RailSem19 [38], and a custom railway dataset. The following list and Table 4.1 show the different characteristics of each dataset. It is important to note that the COCO dataset is not directly used in the experiment. Instead, this study uses the pre-trained weights provided by the YOLOv8 model, which has been initially trained on the COCO dataset.

COCO [37]: is a large-scale benchmark designed for object detection and segmentation tasks. It was introduced by Microsoft in 2014 to facilitate the development of computer vision models capable of recognising objects in diverse and complex scenes. The dataset contains over 330k images, with more than 200k labelled, featuring 80 different object classes ranging from animals to household and outdoor items. Each image includes detailed annotations, such as object bounding boxes, instance segmentation masks and object classes. This dataset provides a wide-ranging and large number of annotations, making COCO an excellent source dataset for pre-training models to learn a wide range of features to be transferred to other detection tasks. However, it does not include railway-specific objects, limiting its applicability to tasks in the railway domain.

RailSem19 [38]: is developed specifically to address the need for annotated data in the railway domain, with a focus on railway objects captured from the train’s ego-perspective. It provides 8.5k images with annotations for 19 railway object classes¹. Although much smaller than COCO, RailSem19 is a specialised dataset for railway infrastructure detection, with annotations for objects like signals, switches, and platforms. However, it does not include railway maintenance objects such as workers, tools, or maintenance vehicles, limiting its applicability for object detection in railway maintenance tasks.

For this study, RailSem19 is further adapted to explore the impact of object class diversity in TL. The dataset was systematically modified into five versions, reducing the number of object classes incrementally: RS19_v0 (19 classes), RS19_v1 (16 classes), RS19_v2 (12 classes), RS19_v3 (8 classes), and RS19_v4 (5 classes). These adaptations enable a detailed analysis of how class diversity in a source dataset influences TL performance.

Custom dataset: is specifically created for this study to evaluate how to achieve high object detection performance with the limited dataset. It consists of 103 images with an-

¹The original dataset contained 21 object classes; this study removed object classes represented by polylines, such as rail tracks, to focus on objects annotated with bounding boxes and polygons; Section 4.2.3.

notations for six distinct object classes relevant to railway maintenance scenarios. These classes include objects such as pedestrians and workers captured at the Northampton & Lamport Railway. This dataset is used to fine-tune with pre-trained models from other source datasets and evaluate their performance in a focused domain. Section 4.2.2 includes the details of the development of the custom dataset.

Dataset	Images No.	Classes No.	Type	Purpose
COCO	330k+	80	Source	Pre-training
RailSem19 (RS19_v0)	8,500	19	Source	Pre-training
RailSem19 (RS19_v1)	8,500	16	Source	Pre-training
RailSem19 (RS19_v2)	8,500	12	Source	Pre-training
RailSem19 (RS19_v3)	8,500	8	Source	Pre-training
RailSem19 (RS19_v4)	8,500	5	Source	Pre-training
Custom dataset	103	6	Target	Fine-tuning

Table 4.1: Overview of datasets used in the experiment

4.2.2 Custom dataset construction

Role of custom dataset in this experiment

The custom railway dataset was explicitly created for this study to meet the requirements of object detection within railway maintenance scenarios and serves as a target dataset for evaluating the impact of different source datasets on TL. During the fine-tuning phase, models initially pre-trained on the COCO and RailSem19 datasets are fine-tuned with the custom railway dataset. This process leverages the learned general and domain-specific features of the railway maintenance environment. This dataset facilitates assessing how well the models transfer learned knowledge to a new, specialised context with different source datasets.

Image collection

The video collection for the railway dataset was conducted in parallel with the experiment on evaluating mobile manipulators for railway track maintenance with the warthog rail-

way robot, as detailed in [244]. During this validation experiment, a GoPro camera and a mobile phone were attached to the warthog robot to capture the first-person view from different angles. This setup provided multiple perspectives, enhancing the dataset’s comprehensiveness and capturing a wider variety of scenes. The video was captured at the Northampton & Lamport Railway, a heritage railway serving tourist purposes but maintaining the same standards and features as a passenger railway. This setting provided a realistic depiction of railway maintenance scenarios. Data capture sessions varied in times of day and lighting conditions to enhance the dataset’s ability to generalise across various settings and ensure robustness by reflecting actual operational environments.

To prevent overfitting, images were selected by adjusting the intervals accordingly: longer intervals when the warthog robot was moving slowly and shorter intervals during faster travel. This strategy helped avoid capturing similar scenes, preserve the dataset’s diversity, and enhance its relevance. All images were resized to 1920x1080 pixels to ensure uniformity and consistency across the dataset.

Image annotation (labelling)

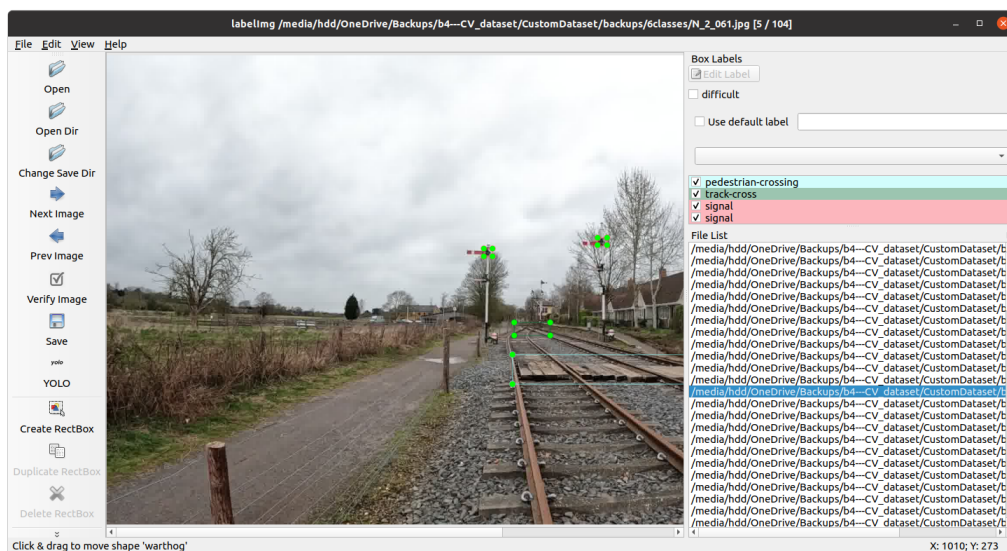


Figure 4.1: Dataset annotation with LabelImg

Image annotation labels the image with metadata to describe various objects. This process involves drawing bounding boxes around objects and assigning class labels to these boxes. Image annotation is essential for object detection algorithms; it provides information for machine learning models to learn and predict the location and types of objects within an image. LabelImg was used to annotate the images; it provides a user-friendly interface and supports the COCO annotation format for the YOLO detection framework. Figure 4.1 illustrates the annotation process using the LabelImg application.

Object classes in custom dataset

Table 4.2 provides an overview of the object types included in the dataset. These classes are chosen to reflect common objects that appear in railway maintenance scenarios, ensuring both safety and operational efficiency. For example, detecting a pedestrian (treated as a trespasser if within a restricted boundary) would prompt the vehicle to stop, while identifying a worker distinguished by high-visibility clothing would signal the vehicle to slow down and proceed cautiously. Pedestrian crossings are included because they require extra caution to ensure safety in areas where interactions with humans (workers or trespassers) are expected. Recognising signal and switch (track-cross) points is beneficial for navigation, operational decisions and potential areas for inspection tasks. The inclusion of the warthog robot reflects collaboration with other autonomous railway robots, providing for future development and integration of such technologies in railway maintenance. Figure 4.2 shows the example of the annotated dataset with object type and bounding box.

Object Class	Description
Pedestrian	Pedestrian present in the railway environment (treated as trespasser).
Worker	Identified by high-visibility clothing, mandatory for railway maintenance.
Warthog	Another type of autonomous railway robot.
Signal	Railway signals control the movement of robots and trains.
Switch	Points where a track splits, possible area for inspection.
Pedestrian-cross	Areas where the robot might expect pedestrians are important for safety measures.

Table 4.2: Object classes in the custom dataset



Figure 4.2: Example of custom dataset with annotation

Dataset statistics

The custom railway dataset contains 103 images in total. The list below shows the statistical information for the custom dataset:

- **Annotations per image**, on average, each image contains approximately 3.04 annotations in the training set and 4.05 annotations in the validation set, indicating a diverse range of objects within the images.

- **Resolution of images**, all images in the dataset are resized to a resolution of 1920x1080 pixels to ensure uniformity.
- **Train/validation split**, the dataset is divided into 82 images for training and 21 images for validation. This split helps in evaluating the model's performance on unseen data.
- **Training set:**
 - Mean bounding box area: 130,660 pixels
 - Median bounding box area: 28,619 pixels
 - Standard deviation of bounding box area: 209,245 pixels
- **Validation set:**
 - Mean bounding box area: 101,391 pixels
 - Median bounding box area: 30,810 pixels
 - Standard deviation of bounding box area: 191,501 pixels

Dataset	Class	Instances
Training	Pedestrian	51
	Worker	38
	Warthog	32
	Signal	45
	Switch	50
	Ped-cross	33
Validation	Pedestrian	14
	Worker	13
	Warthog	18
	Signal	11
	Switch	17
	Ped-cross	12

Table 4.3: Distribution of object classes in the training and validation datasets

Table 4.3 shows the object class distribution in the custom datasets. Although classes such as warthog and ped-cross are less frequent compared to other object classes, their real-world occurrence in railway maintenance environments is lower than reflected in the dataset. Given the small size of the custom dataset, maintaining real-world class frequencies would have resulted in too few examples and limited the model’s ability to learn their representations. These classes were therefore included less frequently than the dominant categories, but more frequently than they appear in real-world deployment. This decision was made to balance between real-world distribution and the need to provide enough examples of rare but safety-critical objects for effective learning.

The class imbalance in the dataset could introduce potential bias during model training, favouring more frequent classes and leading to under-detection or misclassification of rare object classes, which raises concerns for safety-critical applications. To ensure fair evaluation, the validation dataset was designed to mirror the class distribution of the training dataset. Although this does not mitigate the imbalance itself, it avoids introducing additional evaluation bias and enables reliable comparison. No explicit mitigation strategies, such as class weighting, oversampling, or targeted augmentation, were applied during training. Moreover, given the limited size of the dataset, resampling or augmentation of rare classes could have introduced overfitting and reduced generalisability. Addressing

class imbalance directly was considered out of scope, as the focus of this study is to evaluate TL performance across different pre-training strategies under naturally imbalanced and operationally realistic conditions.

To address the potential safety concerns associated with missed detections, RIMV's architecture incorporates a two-layer safety mechanism. While the object detection module serves as the first layer, a second independent layer uses LiDAR to detect the presence of any object within a designated safety region (See Section 3.2.7). If an object is detected, the system triggers an emergency stop regardless of its class, providing a robust fail-safe against potential collisions.

4.2.3 RailSem19 dataset modification

This subsection includes the implementation steps for the modification of the RailSem19 dataset and the reasons for the modifications:

- Convert the annotations to be compatible with the YOLO framework
- Removing polyline representations
- Reducing the object class numbers to better align with the custom dataset

Firstly, the RailSem19 dataset uses polygon annotations to outline object boundaries, which provide a precise outline of objects and is ideal for detailed segmentation tasks. However, to integrate this dataset with the YOLO object detection framework, it is necessary to convert these polygon annotations into bounding box format. Bounding boxes are defined by their top-left and bottom-right corners, which reduces the complexity of object detection and makes them computationally efficient compared to polygon annotation. Figure 4.3 illustrates the difference between polygon and bounding box annotation.

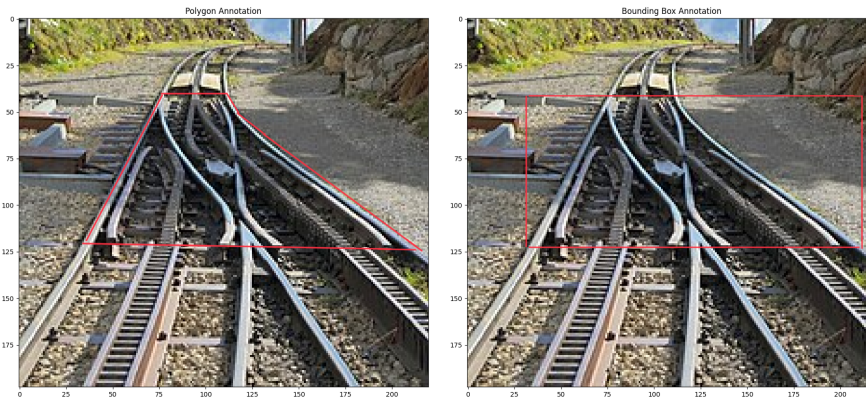


Figure 4.3: Comparison between polygon and bounding box annotation

Secondly, the original RailSem19 dataset includes polyline annotations to represent linear features such as rails and guardrails. While polylines are useful for detailed mapping and segmentation tasks, they are not suitable for the objectives of this study, which focuses on detecting discrete objects rather than linear features. Furthermore, as illustrated in Figure 4.4, It often results in very large bounding boxes and covers significant

portions of the image when converting polylines to bounding boxes. This can lead to several issues:

- Increased false positives: large bounding boxes can encompass multiple objects, leading to incorrect detections and increased false positive rates.
- Decreased model precision: the model may struggle to accurately distinguish between different objects within the large bounding box, reducing overall precision.
- Inefficiency in computation: processing larger bounding boxes requires more computational resources, making the model less efficient, particularly in real-time applications.
- Loss of specificity: large bounding boxes fail to provide detailed localisation of specific objects, which is crucial for tasks that require high precision in object positioning and identification.

Therefore, object classes with polyline annotations (rail and guard rail) were removed to streamline the dataset and concentrate on object detection tasks.



Figure 4.4: Comparison of polyline and bounding box annotation (where the teal line is the polyline representation, and the red line is the bounding box representation, better seen in colour).

Finally, the original RailSem19 dataset includes 21 object classes (19 classes after removing rail and guard-rail), many of which are outside of railway maintenance and

not included in the custom railway dataset. The Object class number in RailSem19 was systematically reduced to align more closely with railway maintenance scenarios and the custom railway dataset.

Polygon to bounding box conversion

This section describes the methodology for converting RailSem19 datasets from polygon representation of the bounding box representation.

The conversion from a polygon to a bounding box is determined by identifying the minimum and maximum x and y coordinates of the polygon points:

$$x_{\min} = \min(\{x_1, x_2, \dots, x_n\})$$

$$y_{\min} = \min(\{y_1, y_2, \dots, y_n\})$$

$$x_{\max} = \max(\{x_1, x_2, \dots, x_n\})$$

$$y_{\max} = \max(\{y_1, y_2, \dots, y_n\})$$

After identifying the minimum and maximum in both x-y axes, the following equation converts the bounding box format required by YOLO, which uses the centre coordinates along with the width and height of the bounding box:

$$dw = \frac{1}{\text{width}}$$

$$dh = \frac{1}{\text{height}}$$

$$x = \left(\frac{x_{\min} + x_{\max}}{2} \right) \times dw$$

$$y = \left(\frac{y_{\min} + y_{\max}}{2} \right) \times dh$$

$$w = (x_{\max} - x_{\min}) \times dw$$

$$h = (y_{\max} - y_{\min}) \times dh$$

Object class modification

Table 4.4 compares the object classes across different versions of the RailSem19 dataset, illustrating the modifications made to create subsets for analysis. RS19_v0 is the modified RailSem19 dataset with the original 21 object classes reduced to 19 by removing classes with polyline representation (e.g., *rail* and *guard-rail*). From RS19_v0 to RS19_v1, the classes *track-signal-front* and *track-signal-back* were combined into a single class, *track-signal*, and *switch-left*, *switch-right*, and *switch-unknown* were combined into *switch* to reduce redundancy. RS19_v2 further refined the dataset by removing non-relevant railway components such as *cars* and *trucks*. RS19_v3 focused on retaining only those object classes that appear in or help identify objects in the custom dataset, such as *switch-indicator* and *switch-static*, which provide context for the presence of switches. Finally, RS19_v4 retained only those classes that directly overlap with the custom dataset. This reduction ensures the dataset’s relevance and manageability, optimising TL for railway-specific object detection tasks.

To maintain consistency and reliability in training and evaluation, all versions of the RailSem19 dataset were split into 80% for training and 20% for validation. This consistent split ensures that each version of the dataset can be effectively used to assess the impact of class diversity on the performance of TL models.

Object Class	RS19_v0	RS19_v1	RS19_v2	RS19_v3	RS19_v4
Switch-left	✓				
Switch-right	✓				
Switch-unknown	✓				
Switch (combined)		✓	✓	✓	✓
Switch-indicator	✓	✓	✓	✓	
Track-sign-front	✓	✓	✓	✓	
Track-signal-front	✓				
Track-signal-back	✓				
Track-signal (combined)		✓	✓	✓	✓
Switch-static	✓	✓	✓	✓	
Pole	✓	✓	✓		
Platform	✓	✓			
Crossing	✓	✓	✓	✓	✓
Train-car	✓	✓			
Rail-occluder	✓	✓			
Fence	✓	✓			
Buffer-stop	✓	✓	✓		
Person	✓	✓	✓	✓	✓
Car	✓	✓			
Person-group	✓	✓	✓	✓	✓
Truck	✓	✓			
total class number	19	16	12	8	5

Table 4.4: Comparison of object classes in different RailSem19 versions

4.3 Experiment setup

4.3.1 Experiment design

This experiment aims to evaluate how different source datasets impact TL for railway object detection. The goal is to investigate whether it is more beneficial to transfer the model trained on a larger, general-purpose dataset (COCO) or a smaller, domain-specific dataset (RailSem19). Additionally, the effect of the number of object classes in RailSem19 on the overall and class-specific performance is also investigated. Furthermore, the experiment explores the impact of one-stage pre-training compared to two-stage pre-training strategies. **two-stage pre-training** refers to the models that have undergone a two pre-training process: the pre-training with the RailSem19 dataset leverages the default weights trained

on COCO. **one-stage pre-training** refers to models that underwent a single pre-training process, trained from scratch using only the RailSem19. Lastly, both models were later fine-tuned with the custom dataset, allowing for evaluation of the effectiveness of two-stage versus one-stage pre-training on the final performance.

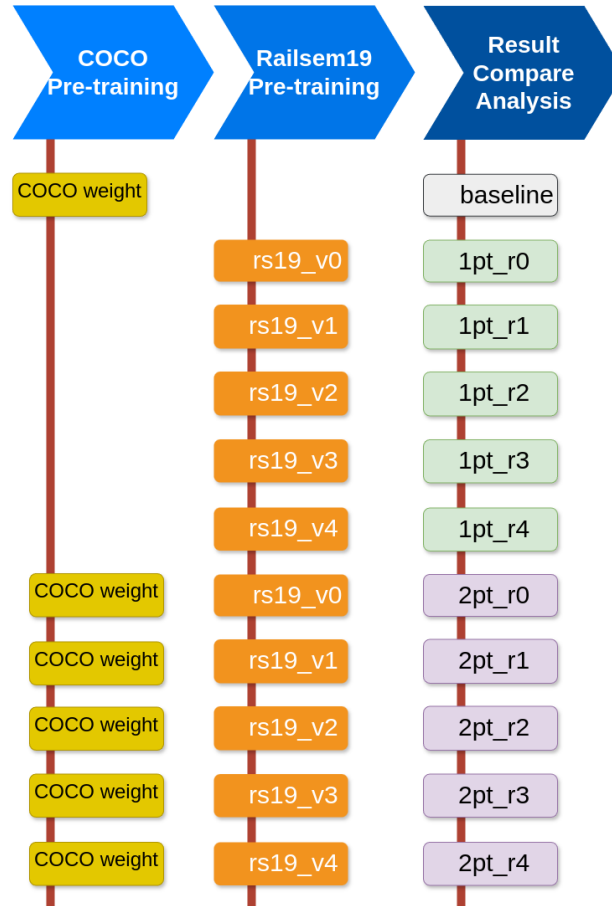


Figure 4.5: Overview of TL experiment design (better to view in colour), “1pt_r0” means one-stage pre-training with RailSem19 v0, “2pt_r0” means two-stage pre-training with COCO and RailSem19 v0.

Figure 4.5 illustrates the four critical steps in this experiment:

- Baseline experiment: one-stage pre-training with the default COCO weight provided by YOLOv8.
- One-stage pre-training with various RailSem19 versions.
- Two-stage pre-training with the default COCO weight and various RailSem19 versions.

- Layer freeze impact study: repeating all experiments with different freezing techniques.

After the pre-training stage, each weight is transferred to train with the custom dataset. This approach allows the identification of the optimal combination of object class numbers and the pre-training method that maximises TL effectiveness for railway object detection tasks. The fine-tuning process uses the same parameters as the pre-training phase: a YOLOv8s model trained for up to 200 epochs with a batch size of 16 and an input image size of 640×640 pixels. Early stopping was applied across all configurations during training, with a criterion of 50 consecutive epochs without improvement in validation loss, to prevent overfitting and ensure convergence. A detailed analysis of the training and validation loss curves is provided in Figure 4.6 in Section 4.4.2. The primary aim of this experiment is not to maximise the model’s performance but to evaluate and compare the effectiveness of different source datasets in TL. By using consistent training parameters and convergence monitoring, the study focuses on understanding the relative impact of each dataset modification rather than achieving the highest possible detection accuracy.

Baseline experiment: one-stage pre-training with the default COCO

The baseline experiment involves using a one-stage pre-training approach with the default weights provided by the YOLOv8, which is trained on the COCO dataset. COCO’s extensive and diverse dataset helps the model learn transferable general features, enhancing its ability to adapt to various tasks [245]. As a well-known benchmark in computer vision, COCO provides reliable performance metrics, making it an excellent reference point [246]. Furthermore, the variety of objects in COCO enhances the model’s generalisation capability, allowing it to perform well on new and different tasks [247].

The pre-trained weights from COCO are fine-tuned with the custom railway dataset and serve as a reference to compare one-stage pre-training with various RailSem19 versions and two-stage pre-training with both COCO and various RailSem19 versions.

One-stage pre-training with RailSem19

In this step, the model is pre-trained using different versions of the RailSem19 dataset (RS19_v0 to RS19_v4 as discussed in Section 4.2). Each version varies in the number of object classes, allowing for an analysis of how class diversity impacts model performance. The one-stage pre-training on these versions aims to identify the optimal RailSem19 dataset configuration for railway object detection for each class and compare it with the baseline results.

Two-stage transfer learning with COCO and RailSem19

This phase involves two-stage pre-training, where the model is initially pre-trained on the COCO dataset and then further pre-trained on the various versions of the RailSem19 dataset. This approach leverages the extensive feature representation learned from COCO and adapts it to the more specific features in RailSem19, potentially enhancing the model's performance for railway object detection tasks.

Impact study on layer freezing

In the final step, the experiment investigates the effects of freezing different layers of the YOLOv8 model during the fine-tuning process. The goal is to determine how freezing specific layers can impact the TL performance for railway object detection. Eight different freezing strategies were evaluated, including freezing early convolutional layers, intermediate layers, and a combination of layers. These strategies are chosen based on their potential to preserve low-level features while allowing high-level features to adapt to the custom dataset.

The freezing strategies tested in this experiment are as follows:

- Non-freeze: all layers are trainable during fine-tuning.
- F0123: freeze layers 0, 1, 2, and 3 (early layers).
- F0135: freeze layers 0, 1, 3, and 5 (early convolutional layers).

- F01357: freeze layers 0, 1, 3, 5 and 7 (early convolutional layers).
- F2468: freeze layers 2, 4, 6, and 8 (C2f layers).
- F0to5: freeze layers from 0 to 5 (early freezing).
- F0to8: freeze layers from 0 to 8 (moderate freezing).
- F0to15: freeze layers from 0 to 15 (extensive freezing).
- F0to21: freeze all layers from 0 to 21 (more extensive freezing).

4.3.2 Hardware and software specifications

Table 4.5 shows the experiment's hardware and software configurations are detailed in this subsection to ensure clarity for reproducibility. The hardware setup for this experiment includes an NVIDIA RTX A5000 GPU, an Intel i7-10700 Central Processing Unit (CPU), and 96GB of Random-Access Memory (RAM). This combination provides the necessary computational power and memory capacity to efficiently handle the training of deep learning models on large datasets.

Type	Component	Specification
Hardware	GPU	NVIDIA RTX A5000
Hardware	CPU	Intel i7-10700
Hardware	Memory	96GB RAM
Software	Operating system	Ubuntu 20.04
Software	Frameworks	PyTorch 1.9, YOLOv8, Jupyter notebook
Software	Additional tools	LabelImg, OpenCV, TensorBoard

Table 4.5: Hardware and software specifications for the experiment

The experiments were conducted on the Linux-based operating system Ubuntu 20.04 with the following software components:

- **Frameworks:**

- **PyTorch**² is an open-source machine learning framework developed by Facebook's AI Research lab. It is widely used in research and industry for its flexible and dynamic environment, supporting dynamic computational graphs and GPU acceleration.
 - **YOLOv8**³ is a state-of-the-art object detection framework known for its speed and accuracy. It provides weights pre-trained with the COCO dataset for TL. The one-stage architecture is efficient and suitable for real-time object detection in railway scenarios.
 - **Jupyter notebook**⁴ is an open-source web application that allows the creation and sharing of documents containing live code, equations, visualisations, and narrative text. It is widely used for machine learning.
- **Additional tools:**
- **LabelImg**⁵ is an open-source graphical image annotation tool used for labelling images in the dataset. The annotations are saved in COCO format and compatible with the YOLO framework. Section 4.2.2 includes the details of constructing a custom dataset with LabelImg.
 - **OpenCV**⁶ is a computer vision library used for image processing tasks such as resizing, augmentation, and preprocessing before training. Its versatile functions facilitate the manipulation of images to prepare them for model input.
 - **TensorBoard**⁷ is a visualisation tool used for monitoring and visualising the training process. It provides real-time metrics such as loss and accuracy over epochs, assisting in the evaluation and tuning of model performance.

²available at <https://pytorch.org/>

³available at <https://yolov8.com/>

⁴available at <https://jupyter.org/>

⁵available at: <https://github.com/HumanSignal/labelImg>

⁶available at <https://opencv.org/>

⁷available at: <https://www.tensorflow.org/tensorboard>

4.3.3 Evaluation metrics

F1 Score is the harmonic mean of precision and recall, providing a balanced measure that accounts for both false positives and false negatives. It is particularly useful when the dataset has an uneven class distribution. The F1 score is calculated using the following equation:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Justification for using F1 score and mAP50: The F1 score is chosen due to its ability to provide a single metric that balances both precision and recall, which is crucial in scenarios where both false positives and false negatives need to be minimised. Mean Average Precision at 50% (mAP50) is selected over Mean Average Precision at 95% (mAP95) because it is a widely used metric that provides a clear indication of the model's overall performance at a commonly used IoU threshold (50%). mAP50 is generally less stringent than mAP95, making it more suitable for evaluating models in practical applications where a slightly lower overlap between predicted and ground truth bounding boxes is acceptable. mAP95 is more commonly used in tasks requiring high-precision localisation, such as autonomous driving and robotics. This project aims to identify objects rather than require precise localisation; thus, mAP50 is a more appropriate metric. This choice helps in understanding the model's performance in real-world scenarios where perfect overlaps are less common.

F1 score is the harmonic mean of precision and recall, providing a balanced measure that accounts for both false positives and false negatives. It is particularly useful when the dataset has an uneven class distribution. The F1 score is calculated using the following equation:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The F1 score is chosen due to its ability to provide a single metric that balances both

precision and recall, which is crucial in scenarios where both false positives and false negatives need to be minimised. mAP50 is selected over mAP95 because it is a widely used metric that provides a clear indication of the model's overall performance at a commonly used IoU threshold (50%). mAP50 is generally less stringent than mAP95, making it more suitable for evaluating models in practical applications where a slightly lower overlap between predicted and ground truth bounding boxes is acceptable. This choice helps in understanding the model's performance in real-world scenarios where perfect overlaps are less common.

4.4 Results and discussion

4.4.1 Introduction

As discussed in Section 4.2, the RailSem19 dataset consists of 8.5 images with 21 railway object classes. Although the dataset size is smaller compared to the 330k images in COCO data, the RailSem19 dataset focuses on urban railway environments. It includes a variety of railway-specific objects such as signals, tracks, etc. This offers an opportunity to pre-train models on different source datasets, fine-tune the learned knowledge with the custom dataset, and compare their performance.

Additionally, this section evaluates the performance of the one-stage and two-stage pre-training approaches. In one-stage pre-training, the model only utilises knowledge from one source dataset (COCO or RailSem19), while two-stage pre-training leverages both datasets sequentially. Initially, the model is pre-trained on the COCO dataset to learn general features, followed by further pre-training on the RailSem19 dataset to adapt to the specific railway domain.

Furthermore, the impact of freezing different layers during fine-tuning is examined. Layer freezing layers is a technique where specific layers of a pre-trained model are kept unchanged during fine training. This study tests different freezing strategies, from freezing early convolutional layers to more extensive freezing of middle and later layers, to un-

derstand how these techniques affect the model’s performance in detecting both railway-specific and non-railway objects.

Lastly, the RailSem19 dataset includes broader railway infrastructure classes which are not directly relevant to the six maintenance-specific classes in the custom dataset. The 21 original classes are gradually reduced to align with the custom dataset, enabling evaluation of how the alignment between source and target datasets influences the effectiveness of TL.

The evaluation includes both overall class performance and class-specific results. The class-specific results focus on key metrics of F1 and mAP50 scores for each object class. Additionally, it compares the performance of railway-specific objects (track switch, signal, pedestrian crossing) and non-railway objects (pedestrian, worker, warthog), providing insights into the role of different source datasets in influencing detection performance. The all-class results are calculated as the average of the same metrics across all object classes. By analysing these different aspects, this section aims to provide comprehensive insights into the most effective TL strategies for enhancing railway object detection performance using a limited custom dataset.

4.4.2 Baseline experiment results

As shown in Table 4.6, The baseline experiment provides an overview of the model’s performance using one-stage pre-training with the default COCO weights supplied by the YOLOv8 framework. The model achieves an all-class F1 score of 0.660 and a mAP50 of 0.663. However, the mAP95 is lower at 0.409, reflecting the stricter criteria for higher IoU thresholds. For the remainder of this section, the baseline results are denoted as **1s-coco**.

Class	P	R	F1	mAP50	mAP95
All-class	0.726	0.605	0.660	0.663	0.409
Pedestrian	0.914	0.929	0.921	0.965	0.609
Worker	0.989	0.846	0.912	0.862	0.485
Warthog	0.924	1.000	0.960	0.995	0.838
Signal	0.490	0.264	0.343	0.334	0.121
Switch	0.413	0.176	0.247	0.286	0.083
Ped-cross	0.622	0.417	0.499	0.537	0.320

Table 4.6: Baseline results for one-stage pre-training with COCO

The class-specific performance of the model demonstrates high detection accuracy for non-railway object classes such as pedestrians, workers, and warthogs, achieving strong F1 scores and mAP50 metrics (e.g., 0.921 F1 and 0.965 mAP50 for pedestrian), indicating reliable detection performance. However, for railway-related object classes such as signals, switches, and pedestrian crossings, the model struggles, achieving lower F1 and mAP50 scores (e.g., 0.247 F1 and 0.286 mAP50 for switch). These results highlight the challenge of TL for railway-specific objects with COCO and serve as a benchmark for comparing the effectiveness of different pre-training strategies in subsequent experiments.

As described in Section 4.3, the baseline training was set to 200 epochs with early stopping applied if no improvement in validation loss was observed for 50 consecutive epochs. In line with the default behaviour of the YOLOv8 framework, early stopping monitors validation box and stop training when the patience condition is met. Figure 4.6 shows the box loss plot for both training and validation. In object detection, box loss measures accuracy by comparing predicted bounding boxes with the ground truth. As shown in the graph, the training box loss shows a steady decline over epochs, reflecting the model's improving fit on the training data. In contrast, the validation box loss shows no substantial improvement after epoch 97. Consequently, the early stopping criteria was met, and training terminated at epoch 147, thereby preventing unnecessary additional epochs and mitigating the risk of overfitting.

Similar trends across other configurations in this study support the choice of 200 epochs with 50-epoch early stopping, balancing training opportunity and prevention of

overfitting. One contributing factor is the relatively small size of the training dataset (103 images), which allows the model to converge faster than larger-scale benchmarks. In addition, the use of TL accelerates convergence. The fixed early stopping criteria provides a uniform and fair basis for comparing different pre-training strategies and configurations.

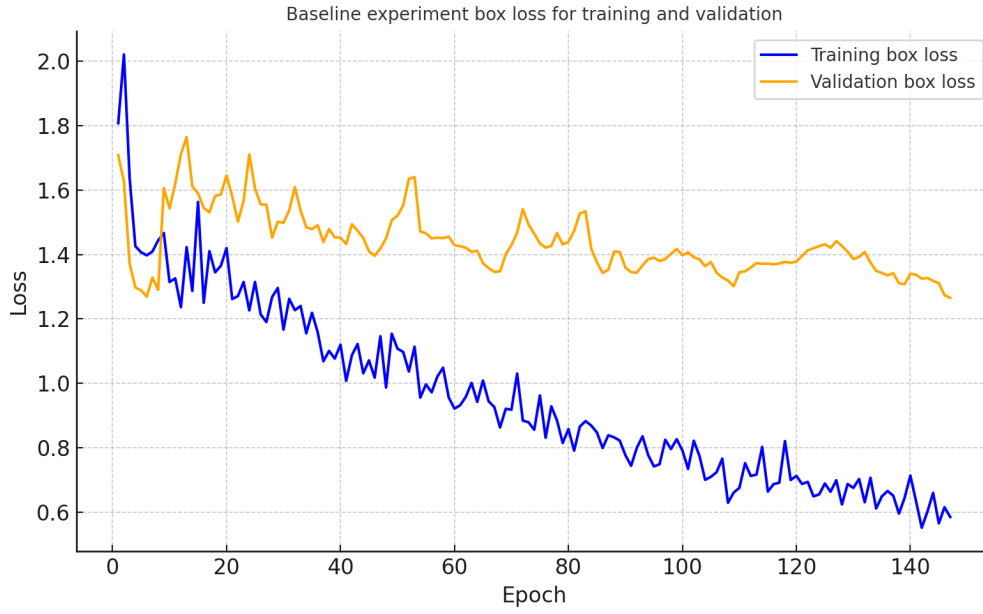


Figure 4.6: Baseline Experiment Box Loss for Training and Validation

4.4.3 Analysis between one-stage & two-stage pre-training

This subsection examines the performance differences between one-stage and two-stage pre-training approaches. The one-stage approach pre-trains models exclusively with the RailSem19 dataset, while the two-stage approach pre-trains models first with the COCO dataset, followed by further pre-training on RailSem19. In both cases, the pre-trained models are fine-tuned with the custom dataset to adapt to railway maintenance scenarios. Additionally, this subsection evaluates the impact of different freezing strategies during fine-tuning.

Non-freezing results

All-class results: as is shown in Figure 4.7, two-stage pre-training outperforms one-stage pre-training in both F1 and mAP50. For instance, the best-performing two-stage pre-training version (2sv1) achieved an F1 score of 0.672 and a mAP50 of 0.657, compared to an F1 score of 0.623 and a mAP50 of 0.607 in the best one-stage pre-training version (1sv2). This suggests that incorporating a second stage of pre-training helps the model refine its feature representations and improve detection accuracy across all object classes. On average, the two-stage approach improves the performance by approximately 9.6% and 8.8% in F1 and mAP50, respectively, across all RailSem19 variations.

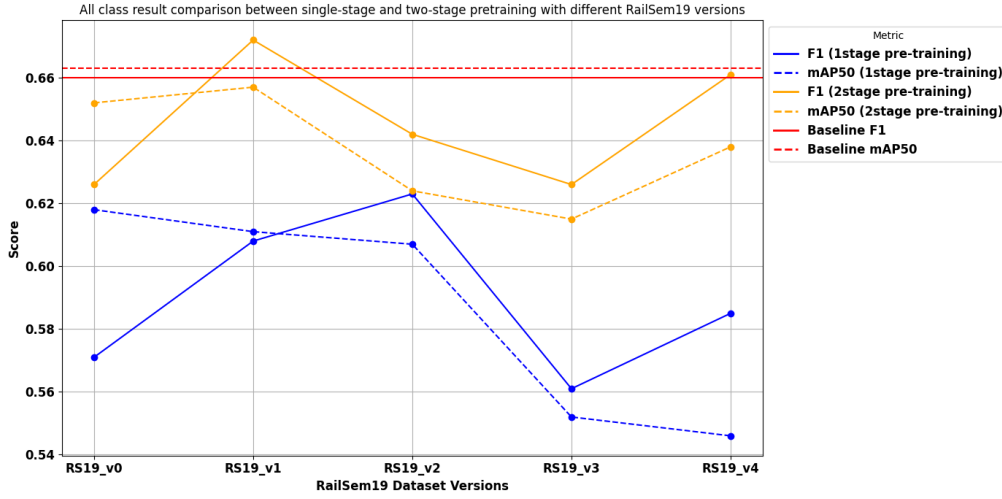


Figure 4.7: All-class results comparison between one-stage and two-stage pre-training with different RailSem19 versions

Class-specific results: as shown in Tables 4.7 and 4.8, two-stage pre-training generally outperforms one-stage pre-training across most object classes. However, there are a few exceptions. For instance, for the ‘signal’ class in RS19_v0, the one-stage pre-training has a higher mAP50 score (0.437) compared to the two-stage pre-training (0.337). Similarly, for the ‘ped-cross’ class in RS19_v1, the one-stage pre-training shows a higher mAP50 score (0.435) compared to the two-stage pre-training (0.377). These exceptions could be attributed to specific characteristics of the training data in these versions, where the additional pre-training stage might not provide significant new information for these particular

classes.

Class	RS19_v0 (1stage/2stage)	RS19_v1 (1stage/2stage)	RS19_v2 (1stage/2stage)	RS19_v3 (1stage/2stage)	RS19_v4 (1stage/2stage)
Pedestrian	0.662 / 0.871	0.851 / 0.892	0.880 / 0.906	0.675 / 0.957	0.716 / 0.775
Worker	0.799 / 0.898	0.695 / 0.829	0.812 / 0.807	0.725 / 0.817	0.468 / 0.746
Warthog	0.856 / 0.907	0.929 / 0.928	0.921 / 0.947	0.951 / 0.966	0.965 / 0.930
Signal	0.386 / 0.370	0.331 / 0.494	0.364 / 0.522	0.378 / 0.349	0.444 / 0.430
Switch	0.348 / 0.334	0.301 / 0.311	0.236 / 0.190	0.090 / 0.245	0.355 / 0.423
Ped-cross	0.306 / 0.350	0.495 / 0.445	0.440 / 0.441	0.452 / 0.328	0.432 / 0.524

Table 4.7: Class-specific F1 comparison for one-stage and two-stage pre-training across RS19 versions.

Class	RS19_v0 (1stage/2stage)	RS19_v1 (1stage/2stage)	RS19_v2 (1stage/2stage)	RS19_v3 (1stage/2stage)	RS19_v4 (1stage/2stage)
Pedestrian	0.816 / 0.983	0.934 / 0.990	0.878 / 0.979	0.760 / 0.990	0.894 / 0.960
Worker	0.818 / 0.910	0.775 / 0.819	0.765 / 0.810	0.814 / 0.859	0.481 / 0.795
Warthog	0.992 / 0.995	0.992 / 0.995	0.988 / 0.990	0.995 / 0.995	0.995 / 0.995
Signal	0.437 / 0.337	0.297 / 0.446	0.342 / 0.442	0.323 / 0.385	0.338 / 0.342
Switch	0.305 / 0.300	0.235 / 0.313	0.247 / 0.217	0.095 / 0.136	0.215 / 0.377
Ped-cross	0.342 / 0.387	0.435 / 0.377	0.425 / 0.307	0.326 / 0.327	0.352 / 0.358

Table 4.8: Class-specific mAP50 comparison between one-stage and two-stage pre-training across RailSem19 versions

Freezing 0135 results

All-class results: as shown in Figure 4.8, two-stage pre-training with the f0135 freezing scenario generally outperforms one-stage pre-training in both F1 and mAP50 scores (except F1 score in RS_v4). The best-performing two-stage pre-training version (RS_v0) achieved an F1 and mAP50 score of 0.702, 0.693 respectively, compared to an F1 score of 0.611 and a mAP50 of 0.602 in the best one-stage pre-training version (RS_v1). On average, the two-stage approach improves the performance by approximately 8.0% and 11.2% in F1 and mAP50, respectively, across all RailSem19 variations.

Class-specific results: as shown in Tables 4.9 and 4.10, two-stage pre-training with f0135 outperforms one-stage pre-training across most object classes. However, there are a

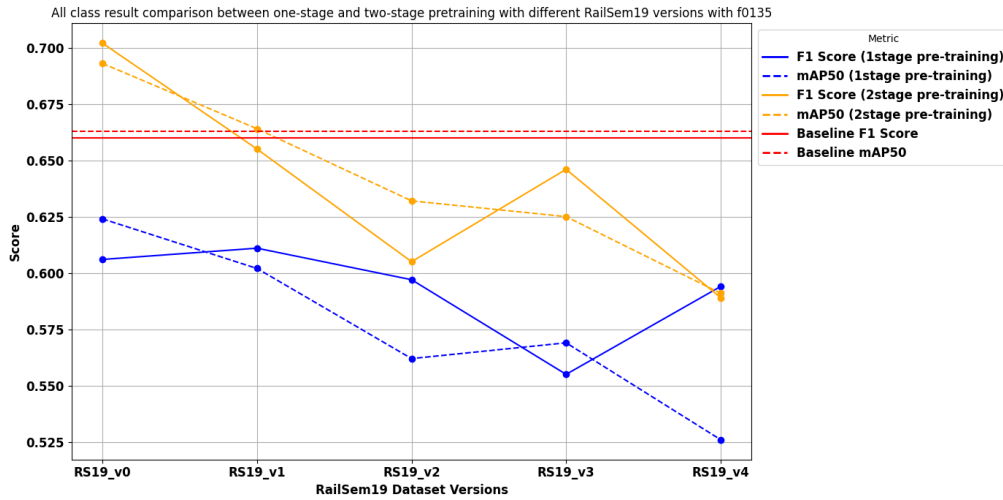


Figure 4.8: All-class results comparison between one-stage and two-stage pre-training with different RailSem19 versions with f0135

few exceptions. For instance, for the ‘signal’ class in RS19_v2, the one-stage pre-training has a higher F1 score (0.423) compared to the two-stage pre-training (0.404).

Class	RS19_v0 (1stage/2stage)	RS19_v1 (1stage/2stage)	RS19_v2 (1stage/2stage)	RS19_v3 (1stage/2stage)	RS19_v4 (1stage/2stage)
Pedestrian	0.689 / 0.889	0.918 / 0.897	0.773 / 0.855	0.766 / 0.934	0.772 / 0.607
Worker	0.804 / 0.832	0.646 / 0.851	0.739 / 0.877	0.763 / 0.762	0.421 / 0.615
Warthog	0.896 / 0.945	0.949 / 0.948	0.846 / 0.912	0.933 / 0.951	0.974 / 0.889
Signal	0.452 / 0.474	0.270 / 0.484	0.423 / 0.404	0.292 / 0.441	0.364 / 0.377
Switch	0.321 / 0.431	0.309 / 0.249	0.256 / 0.254	0.145 / 0.229	0.346 / 0.344
Ped-cross	0.403 / 0.519	0.495 / 0.408	0.438 / 0.290	0.374 / 0.499	0.384 / 0.563

Table 4.9: Class-specific F1 comparison for one-stage between two-stage pre-training across RailSem19 versions with f0135

Class	RS19_v0 (1stage/2stage)	RS19_v1 (1stage/2stage)	RS19_v2 (1stage/2stage)	RS19_v3 (1stage/2stage)	RS19_v4 (1stage/2stage)
Pedestrian	0.847 / 0.973	0.957 / 0.983	0.839 / 0.969	0.832 / 0.995	0.858 / 0.789
Worker	0.788 / 0.879	0.714 / 0.827	0.733 / 0.841	0.811 / 0.752	0.396 / 0.781
Warthog	0.995 / 0.995	0.992 / 0.995	0.978 / 0.988	0.995 / 0.995	0.995 / 0.992
Signal	0.448 / 0.423	0.323 / 0.501	0.335 / 0.442	0.305 / 0.363	0.351 / 0.241
Switch	0.280 / 0.390	0.192 / 0.260	0.175 / 0.237	0.156 / 0.219	0.271 / 0.287
Ped-cross	0.384 / 0.501	0.433 / 0.417	0.309 / 0.314	0.313 / 0.426	0.283 / 0.456

Table 4.10: Class-specific mAP50 comparison for one-stage between two-stage pre-training across RailSem19 versions with f0135

4.4.4 Analysis on impact of layer freezing techniques

Why freezing?

The YOLOv8 architecture includes 21 layers, each playing a crucial role in the process of object detection. For example, convolutional layers for feature extraction and Cross-Stage-Partial (CSP) layers enhance gradient flow and feature propagation. The early layers in YOLOv8 detect basic image features such as edges, textures, and simple shapes. These foundational features are useful across a variety of tasks. The late layers capture abstract, task-specific features. These layers refine the combined features from earlier layers to make accurate predictions and perform the final object detection tasks.

Freezing specific layers during fine-tuning is a technique used to preserve the knowledge acquired in those layers from the pre-training phase while allowing other layers to adapt to the new task. Understanding the function of each layer helps in selecting appropriate layers for freezing to optimise the model’s performance for a new task. Table 4.11 shows different freezing scenarios tested in this experiment. By freezing different combinations of layers, this study aims to determine the optimal freezing strategy that balances retaining useful pre-trained features and adapting to new data.

Freeze Scenario	Description
non-freeze	all layers are trainable during fine-tuning
f0123	freeze layers 0, 1, 2, and 3 (early layers).
f0135	freeze layers 0, 1, 3, and 5 (early convolutional layers)
f01357	freeze layers 0, 1, 3, 5, and 7 (early convolutional layers)
f2468	freeze layers 2, 4, 6, and 8 (C2f layers)
f0to5	freeze layers from 0 to 5 (early freezing)
f0to8	freeze layers from 0 to 8 (moderate freezing)
f0to15	freeze layers from 0 to 15 (extensive freezing)
f0to21	freeze all layers from 0 to 21 (more extensive freezing)

Table 4.11: Different freezing scenarios tested

Freezing early layers (f0123, f0135, f01357) can potentially improve model performance by retaining low-level features such as edges and textures, which are beneficial across different tasks. These strategies help stabilise the learning process, especially

when dealing with small target datasets, by preserving essential features from the pre-trained model. Moderate freezing (f0to5, f0to8) achieves a balance between retaining features and allowing adaptability, which can also enhance performance.

On the other hand, extensive freezing strategies (f0to15, f0to21) may decrease performance due to the reduced flexibility of the model. The model's ability to adapt to the new target dataset is significantly limited by freezing too many layers, which can be detrimental if the target dataset differs considerably from the source dataset.

Best all-class performance

Figure 4.9 and 4.10 shows the F1 and mAP50 scores across different freezing combinations; the f0135, f01357, and f0to5 freezing combinations yield the best results compared to other scenarios:

- In the f0135 scenario, 2sv0 achieves the best F1 score of 0.702 and mAP50 score of 0.693, surpassing the non-freeze F1 score of 0.626 and mAP50 score of 0.652.
- In the f01357 scenario, 2sv2 achieves an F1 score of 0.688 and mAP50 score of 0.642, significantly higher than the non-freeze F1 score of 0.642 and mAP50 score of 0.624.
- In the f0to5 scenario, one-stage pre-training with COCO achieves an F1 score of 0.694 and mAP50 score of 0.694, showing better performance than the non-freeze F1 score of 0.660 and mAP50 score of 0.663.

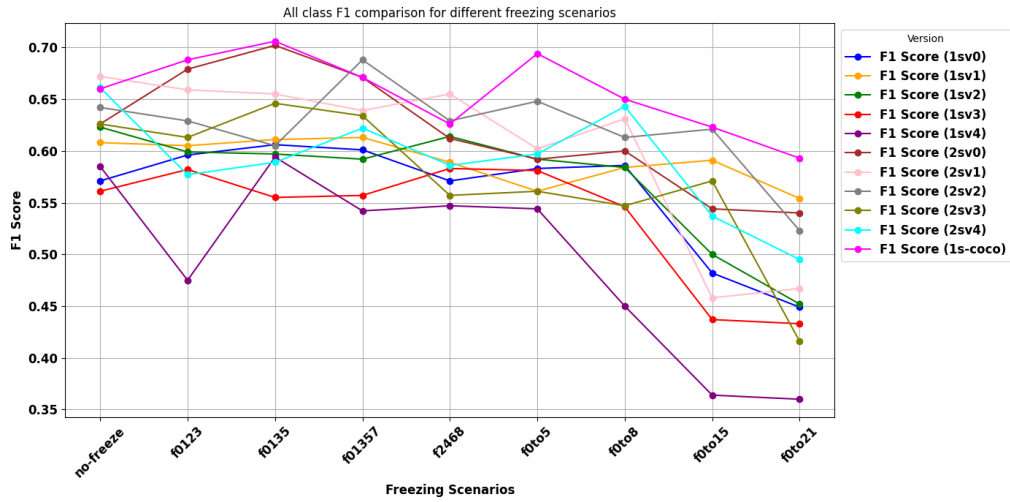


Figure 4.9: All-class F1 comparison for different freezing scenarios

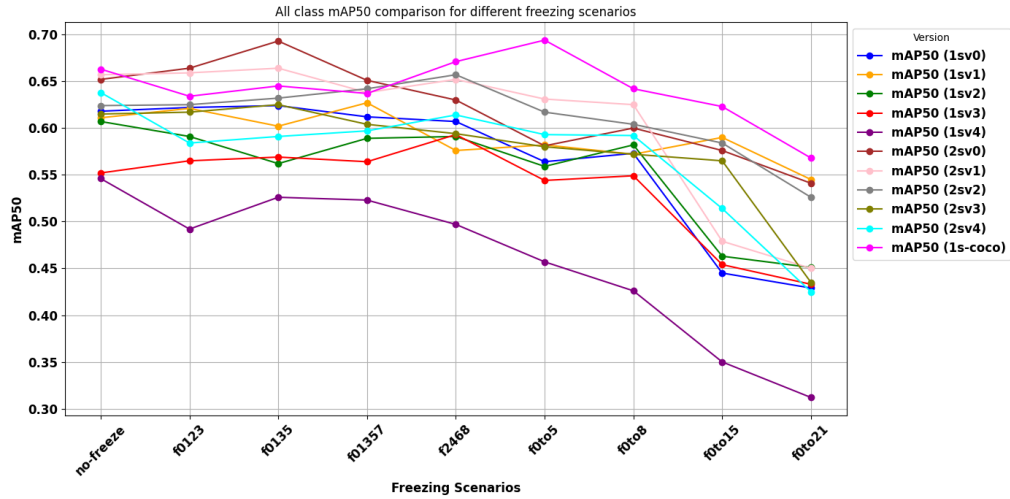


Figure 4.10: All-class mAP50 comparison for different freezing scenarios

The improved performance of the aforementioned scenarios can be attributed to the freezing of the early convolutional layers. In YOLOv8, layers 0, 1, 3, 5, and 7 are convolutional layers responsible for detecting basic image features. Freezing convolutional layers retains the general feature extraction capabilities learned from the pre-trained dataset. Layers 2, 4, 6, and 8 are C2f layers, which enhance feature propagation and gradient flow; freezing C2f layers help in retaining complex, high-level feature representations. Compared to later layers, which capture complex, dataset-specific features. Early layers capture low-level features such as edges, textures, and basic shapes. Freezing the early

convolutional layers can help the model to focus on the later layers for high-level tasks.

Poorer all-class performance

As shown in Figure 4.9 and 4.10, the f0to21 and f0to15 scenarios consistently perform worse for both F1 and mAP50 scores across all freezing combinations in all dataset variations. As discussed before, it can be beneficial to freeze the early layers to preserve the learned features and transfer them to different datasets. However, freezing too many layers, including middle and late layers (as in the f0to21 and f0to15 scenarios), prevents the model from adapting to the specific characteristics of the new dataset. The model cannot fine-tune the higher-level features necessary for accurate object detection in the given context. Consequently, it leads to poor performance.

Discussion on the best performing freezing scenarios f0135

All-class results: Figure 4.11 shows the F1 and mAP50 scores for the f0135 freezing combination across different pre-training scenarios for 2sv0. Although the baseline results (**1s-coco**) achieve the highest F1 score of 0.706, the two-stage pre-training with RailSem19 v0 (2sv0) is more balanced with high scores in both F1 (0.702) and mAP50 (0.693), indicating that two-stage pre-training by combining COCO and RS_v0 provides substantial benefits.

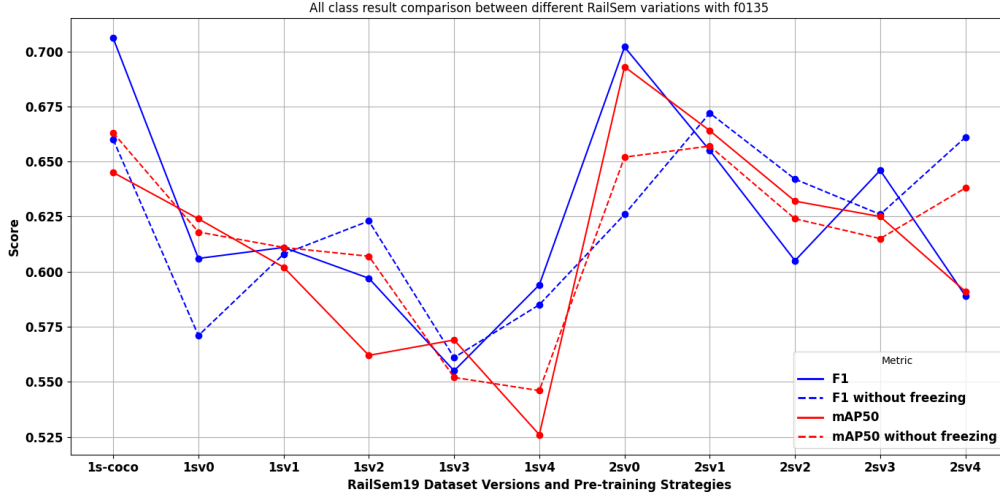


Figure 4.11: All-class results comparison between different RailSem19 variations with f0135

Freezing layers 0, 1, 3, and 5 (f0135) follow a similar trend to the non-freeze scenario across different datasets. The difference is that 2sv0 achieves the peak performance instead of 2sv1, and the trend between 2sv3 and 2sv4 is different.

In the f0135 scenario, 2sv0 achieves the peak performance instead of 2sv1. This can be attributed to the fact that RailSem19 v0 has the most object classes, providing a richer and more diverse set of features during the pre-training phase. The frozen layers retain these broad features, which helps in achieving better performance when fine-tuned on the COCO dataset.

The trend between 2sv3 and 2sv4 differs significantly. While the non-freeze scenario shows an increase, the f0135 scenario shows a sharp decline at 2sv4. RailSem19 v4 has the least number of object classes, which means fewer diverse features are captured during the pre-training. When early convolutional layers (0, 1, 3, and 5) are frozen, the model's ability to adapt to this less diverse dataset is restricted, leading to poorer performance in 2sv4 compared to the non-freeze scenario where the model can adjust all layers to compensate for the reduced class diversity.

Class-specific results: Figures 4.12 and 4.13 illustrate the mAP50 and F1 scores for each class in the f0135 freezing scenario across different pre-training strategies. Com-

pared to the non-freeze scenario (represented by dashed lines), the f0135 freezing scenario shows a similar trend with few exceptions:

- **1s-coco:** for the one-stage pre-training with COCO, the f0135 scenario shows an increase in F1 score compared to the non-freeze scenario. The pedestrian crossing and signal classes primarily contribute to this improvement. These classes benefit from the retained early convolutional features, which are essential for detecting smaller and less distinct objects like signals and pedestrian crossings.
- **2sv0:** for the two-stage pre-training with COCO and RailSem19 v0, the f0135 scenario achieves higher F1 and mAP50 scores compared to the non-freeze scenario. This improvement is mainly contributed by the railway-specific object classes: pedestrian crossing, signal, and switches. Freezing the early convolutional layers (0, 1, 3, and 5) preserves the basic feature representations and leads to better feature extraction and generalisation, improving the model's performance in detecting these railway objects.
- **2sv4:** conversely, for the two-stage pre-training with COCO and RailSem19 v4 (2sv4), the f0135 scenario exhibits a decrease in both F1 and mAP50 scores compared to the non-freeze scenario. This decline is primarily due to the non-railway object classes: pedestrian and worker. The reduction in object class diversity in RailSem19 v4 limits the model's ability to generalise, and freezing the early convolutional layers restricts the model's capacity to adapt to these non-railway objects. Consequently, the model's performance in these classes drops, leading to an overall decline in performance.

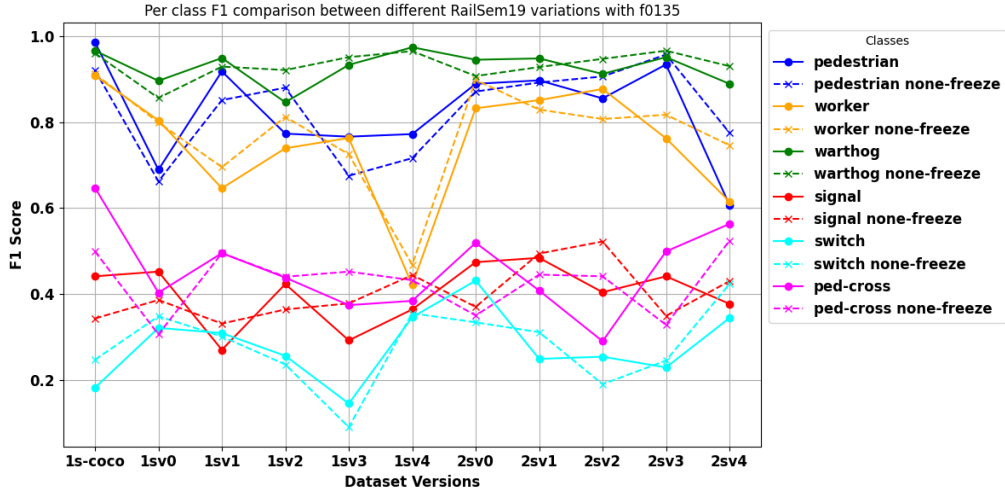


Figure 4.12: Class-specific F1 comparison between different RailSem19 variations with f0135

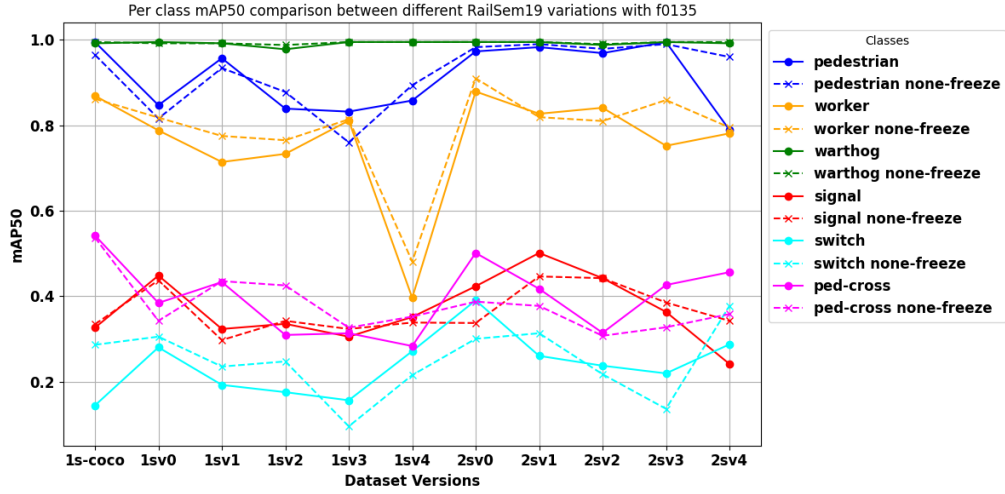


Figure 4.13: Class-specific mAP50 comparison between different RailSem19 variations with f0135

4.4.5 Analysis on the impact of different source datasets

Figures 4.12 and 4.13 illustrate the all-class results for different RailSem19 versions across various freezing scenarios. Although two-stage pre-training achieves better overall results by leveraging both datasets, this subsection focuses on comparing the performance of TL from a large, general-purpose dataset (COCO) and a smaller, domain-specific dataset (RailSem19) under one-stage pre-training.

COCO and RailSem19: non-freezing

Table 4.12 presents the F1 and mAP50 scores for all classes and individual object classes without freezing. The comparison focuses on the best performance of one-stage pre-training with COCO versus RailSem19 version 2 (1sv2) without freezing.

Class	1s-coco (F1/mAP50)	1sv2 (F1/mAP50)
All-class	0.660 / 0.663	0.623 / 0.607
Pedestrian	0.921 / 0.965	0.880 / 0.878
Worker	0.912 / 0.862	0.812 / 0.765
Warthog	0.960 / 0.995	0.921 / 0.988
Signal	0.343 / 0.334	0.364 / 0.342
Switch	0.247 / 0.286	0.236 / 0.247
Ped-cross	0.499 / 0.537	0.440 / 0.425

Table 4.12: Results comparison between 1s-coco and rs_1sv2 without freezing

All-class results: the one-stage pre-training with COCO achieved slightly better performance with an F1 score of 0.660 and a mAP50 of 0.663, which is 6% and 9.2% improvement compared to the F1 score of 0.623 and a mAP50 of 0.607 with one-stage pre-training with RailSem19 version 2 (1sv2).

Non-railway object classes: for non-railway object classes, COCO achieved higher F1 scores and mAP50 values for pedestrian (F1: 0.921, mAP50: 0.965), worker (F1: 0.912, mAP50: 0.862), and warthog (F1: 0.960, mAP50: 0.995) compared to 1sv2's scores of pedestrian (F1: 0.880, mAP50: 0.878), worker (F1: 0.812, mAP50: 0.765), and warthog (F1: 0.921, mAP50: 0.988). On average, COCO improved F1 and mAP by 7.1% and 7.8% for non-railway object classes. This indicates that COCO's larger and more diverse dataset provides better generalisation for non-railway objects.

Railway object classes: in terms of railway object classes, RailSem19 performed better for the signal class, achieving an F1 score of 0.364 and mAP50 of 0.342, compared to COCO's F1 score of 0.343 and mAP50 of 0.334. For the switch and pedestrian crossing

classes, although COCO showed better performance with F1 scores of 0.247 and 0.499 and mAP50 values of 0.286 and 0.537, respectively, the performance gap between COCO and 1sv2 was smaller compared to non-railway object classes. Specifically, 1sv2 achieved F1 scores of 0.236 and 0.440 and mAP50 values of 0.247 and 0.425 for switch and pedestrian crossing, respectively. On average, COCO improved F1 and mAP by 4.1% and 13.3% for railway object classes.

COCO and RailSem19: best freezing scenarios

Table 4.13 presents the F1 and mAP50 scores for all classes and individual object classes under the best freezing scenarios. The comparison focuses on the best performance of one-stage pre-training with COCO (f0to5) versus RailSem19 version 1 (1sv1) (f01357) with their best freezing scenarios.

Class	1s-coco (F1/mAP50)	1sv2 (F1/mAP50)
All-class	0.694 / 0.694	0.613 / 0.627
Pedestrian	0.925 / 0.976	0.796 / 0.927
Worker	0.813 / 0.836	0.679 / 0.822
Warthog	0.945 / 0.995	0.947 / 0.995
Signal	0.540 / 0.605	0.371 / 0.390
Switch	0.250 / 0.192	0.389 / 0.276
Ped-cross	0.602 / 0.562	0.411 / 0.355

Table 4.13: Results comparison between 1s-coco (f0to5) and 1sv1 (f01357) with the best freezing scenario

All-class results: the one-stage pre-training with COCO using the f0to5 freezing scenario achieved an F1 score of 0.694 and an mAP50 of 0.694. In contrast, one-stage pre-training with RailSem19 version 1 (1sv1) using the f01357 freezing scenario resulted in an F1 score of 0.613 and a mAP50 of 0.627. On average, COCO outperformed RailSem19 by 13.2% and 10.7% for F1 and mAP50.

Non-railway object classes: for non-railway objects, COCO with f0to5 achieved higher F1 scores and mAP50 values for pedestrian (F1: 0.925, mAP50: 0.976), worker (F1: 0.813, mAP50: 0.836), and warthog (F1: 0.945, mAP50: 0.995) compared to 1sv1 with f01357, which achieved scores of pedestrian (F1: 0.796, mAP50: 0.927), worker (F1: 0.679, mAP50: 0.822), and warthog (F1: 0.947, mAP50: 0.995). On average, COCO improved F1 by 12.0% and decreased -2.1% for non-railway object classes.

Railway object classes: for railway objects, RailSem19 version 1 (1sv1) with the f01357 freezing scenario outperformed COCO with f0to5 in detecting switches, achieving an F1 score of 0.389 and a mAP50 of 0.276 compared to COCO's F1 score of 0.250 and mAP50 of 0.192. However, for signals and pedestrian crossings, COCO with f0to5 performed significantly better. COCO achieved F1 and mAP50 scores of 0.540 and 0.605 for signals, 0.602 and 0.562 for pedestrian crossings, whereas 1sv1 with f01357 scored lower with 0.371 and 0.390 for signals, and 0.411 and 0.355 for pedestrian crossings. On average, COCO improved F1 and mAP by 18.8% and 32.9% for railway object classes.

In summary, TL with one-stage pre-training, COCO generally outperforms RailSem19, both with and without freezing scenarios. COCO's larger and more diverse dataset provides better all-class performance and generalisation for non-railway object classes. However, RailSem19 shows a slight advantage in detecting railway-specific objects, particularly switches and signals, when the appropriate layer-freezing strategy is applied. Despite this, COCO's performance remains superior for most classes, highlighting its robustness as a pre-training dataset for diverse object detection tasks.

4.4.6 Analysis on the impact of RailSem19 variations

Introduction

This subsection analyses the performance of models pre-trained on different versions of the RailSem19 dataset, focusing on the impact of dataset variations and layer freezing

(f0135 strategy) on model stability and accuracy. The aim is to evaluate how changes in the number and types of object classes in RailSem19 influence all-class and class-specific performance, with attention to the preservation of low-level features and noise reduction. Additionally, the study compares the effectiveness of two-stage TL with COCO and various RailSem19 datasets.

The analysis assesses F1 and mAP50 scores across all RailSem19 versions compared with the baseline results for both all-class and class-specific performance. Additionally, the study of the impact of RailSem19 variations focuses on how different dataset configurations influence both all-class and class-specific model performance.

The original dataset, RailSem19, contains 21 object classes, capturing a wide variety of railway-specific objects. To better align with the custom dataset, several modifications were made; table 4.14 summarises the different versions of the RailSem19 dataset used in pre-training.

Dataset	Classes No.	Purpose
RS19_v0	19	removed classes with polyline annotation
RS19_v1	16	combined various types of switch and signal
RS19_v2	12	removed non-railway object classes
RS19_v3	8	focused on relevant classes to custom dataset
RS19_v4	5	only classes appearing in the custom dataset

Table 4.14: Comparison of different RailSem19 datasets

One-stage pre-training

All-class performance analysis: as shown in Figure 4.14, the performance differences among the various versions of RailSem19 are noticeable. RS19_v2 achieves the highest F1 score at 0.623. Despite the mAP50 being slightly lower than RS19_v0, this indicates that reducing the dataset to the most relevant railway objects improves the model's precision and all-class performance. In contrast, RS19_v3 and RS19_v4, which focus on the most relevant classes to the custom dataset, show a decline in performance metrics compared to RS19_v1 and RS19_v2.

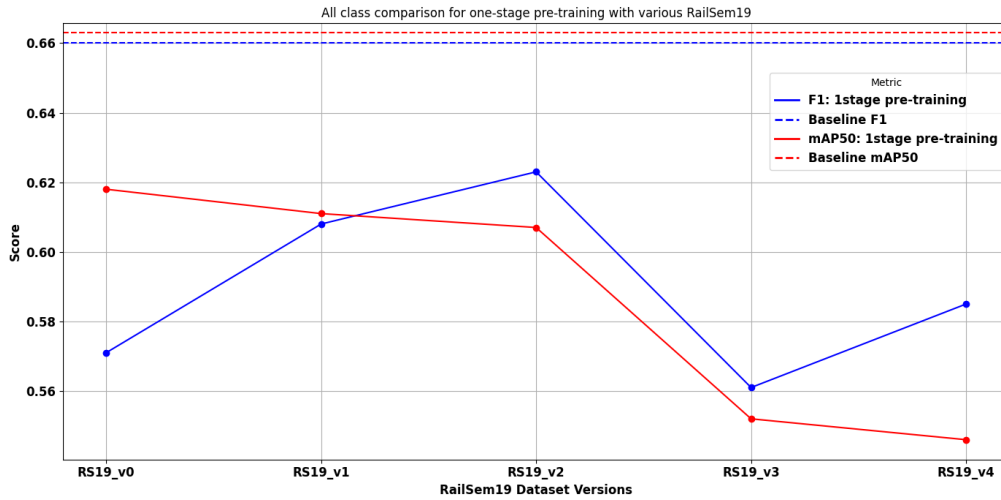


Figure 4.14: All-class comparison for one-stage pre-training with different RailSem19 versions

Class-specific comparison with baseline results: Figure 4.15 and 4.16 show the class-specific performance for each RailSem19 variation with the baseline results. COCO outperforms RailSem19 in most object classes. However, there are exceptions where RailSem19 versions perform better. For instance, RS19_v0 and RS19_v4 show slightly better performance in detecting switches and signals. The improved performance can be attributed to the specialised nature of the RailSem19 dataset, which includes more relevant and contextual examples for railway objects. Switches and signals are included in the RailSem19 dataset but are not in the COCO dataset.

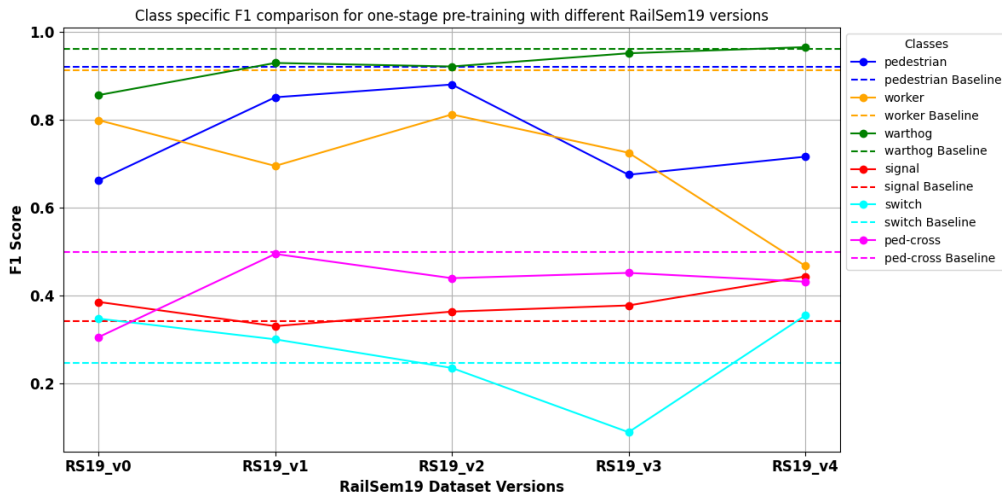


Figure 4.15: Class-specific F1 comparison for one-stage pre-training with different RailSem19 versions

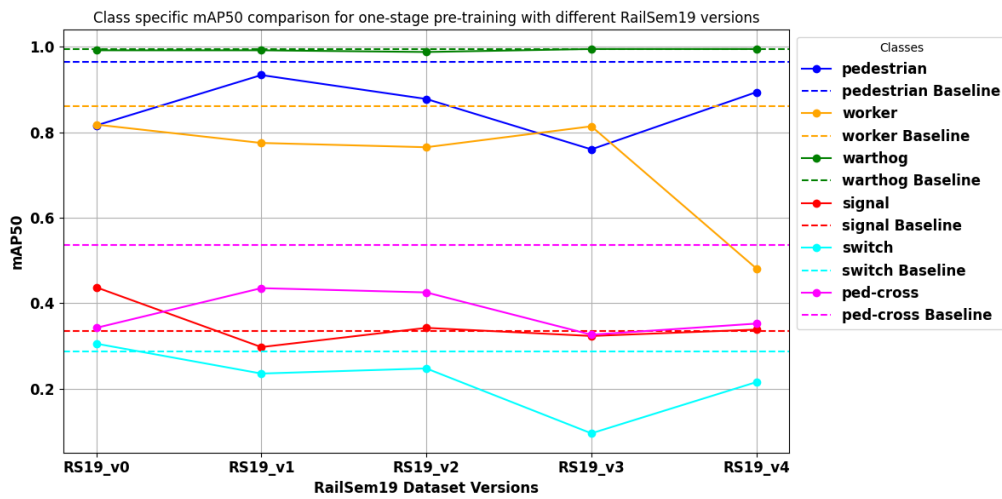


Figure 4.16: Class-specific mAP50 comparison for one-stage pre-training with different RailSem19 versions

Class-specific RailSem19 version impact study: Table 4.15 summarises the results for each class across the RailSem19 variations with one-stage pre-training, highlighting the best-performing RailSem19 version for each object class. The results aim to identify trends in model performance for each object class as the dataset is gradually reduced to align with the custom dataset.

Class	RS19_v0 (F1/mAP50)	RS19_v1 (F1/mAP50)	RS19_v2 (F1/mAP50)	RS19_v3 (F1/mAP50)	RS19_v4 (F1/mAP50)
Pedestrian	0.662/0.816	0.851/0.934	0.880/0.878	0.675/0.760	0.716/0.894
Worker	0.799/0.818	0.695/0.775	0.812/0.765	0.725/0.814	0.468/0.481
Warthog	0.856/0.992	0.929/0.992	0.921/0.988	0.951/0.995	0.965/0.995
Signal	0.386/0.437	0.331/0.297	0.364/0.342	0.378/0.323	0.444/0.338
Switch	0.348/0.305	0.301/0.235	0.236/0.247	0.090/0.095	0.355/0.215
Ped-cross	0.306/0.342	0.495/0.435	0.440/0.425	0.452/0.326	0.432/0.352

Table 4.15: Class-specific results comparison across different RailSem19 versions (one-stage pre-training), with best-performing versions highlighted.

- For the pedestrian class, RS19_v2 achieves the highest F1 score of 0.880 and a mAP50 of 0.878, indicating the best detection accuracy. RS19_v1 follows closely with an F1 score of 0.851 and a mAP50 of 0.934. Despite the initial increase in performance from RS19_v0 to RS19_v1 and RS19_v2, the performance in RS19_v3 and RS19_v4 decreased in both F1 score and mAP50 metrics.
- For the worker class, RS19_v2 achieves the highest F1 score of 0.812, and RS19_v0 has the highest mAP50 of 0.818. The performance fluctuates (remains stable) between RS19_v0 and RS19_v3, with minor variations in F1 score and mAP50. A notable decline is observed in RS19_v4, where both the F1 score and mAP50 significantly drop.
- For the warthog class, RS19_v4 achieves the highest F1 score of 0.965 and an mAP50 of 0.995; all versions from RS19_v0 to RS19_v4 show a consistently good performance in both F1 score and mAP50.
- For signal class, RS19_v4 achieves the highest F1 score of 0.444, while RS19_v0 has the highest mAP50 of 0.437, indicating the best detection accuracy for the signal class. The F1 score for the signal class shows a slight increase from RS19_v0 to RS19_v4, while the mAP50 exhibits a slight decrease.
- For switch class, RS19_v4 achieves the highest F1 score of 0.355, while RS19_v0 has the highest mAP50 of 0.305. Both the F1 and mAP50 score shows a general

decrease from RS19_v0 to RS19_v3 followed by a significant increase in RS19_v4

- For pedestrian crossing class, RS19_v1 achieves the highest F1 score of 0.495 and the highest mAP50 of 0.435. The F1 score and mAP50 for the pedestrian crossing class show a fluctuating trend with an increase in RS19_v1 and a slight decrease afterwards.

Two-stage pre-training

All-class performance analysis: as illustrated in Figure 4.17, the impact of different RailSem19 versions follows a similar trend to the one-stage pre-training results, showing an initial increase, then a decrease, and finally a further increase at RS19_v4. RS19_v1 shows the best performance in both F1 and mAP50 scores, indicating that a balanced dataset with relevant object classes enhances the model's ability. On the other hand, the decrease in RS19_v2 and RS19_v3 suggests that further reduction in class number might reduce the model's performance. Interestingly, there is a noticeable performance increase at RS19_v4, which could be attributed to the removal of noise introduced by less relevant classes.

It can also be observed that RS19_v1 is the best-performing RailSem19 variation than others. Although the mAP50 is slightly lower than the baseline result of 0.663, The F1 score of 0.672 surpasses the baseline F1 score of 0.660. This indicates that two-stage pre-training with COCO and RS19_v1 can improve the model's overall performance, especially in terms of F1 score.

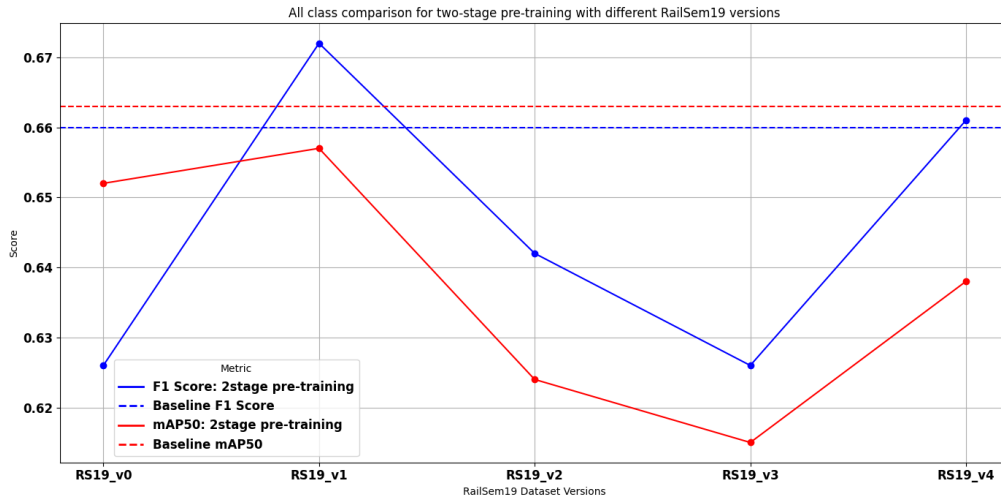


Figure 4.17: All-class comparison for two-stage pre-training with different RailSem19 versions

Class-specific comparison with baseline results: Figures 4.18 and 4.19 illustrate the class-specific performance for each RailSem19 variation compared to the baseline under the two-stage pre-training scenario. While COCO achieves better results in most non-railway object classes, several RailSem19 versions surpass the baseline for railway-specific objects. For example, RS19_v2 and RS19_v4 show improved performance in detecting signals and switches, respectively. This improvement can be attributed to the domain-specific feature of the RailSem19 dataset, providing an advantage over COCO, which lacks these railway-specific objects.

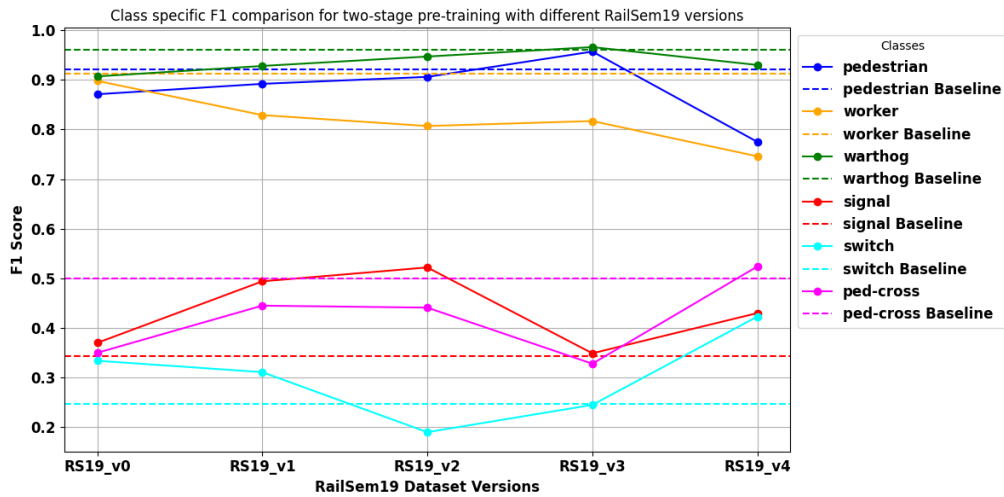


Figure 4.18: Class-specific F1 comparison for two-stage pre-training with different RailSem19 versions

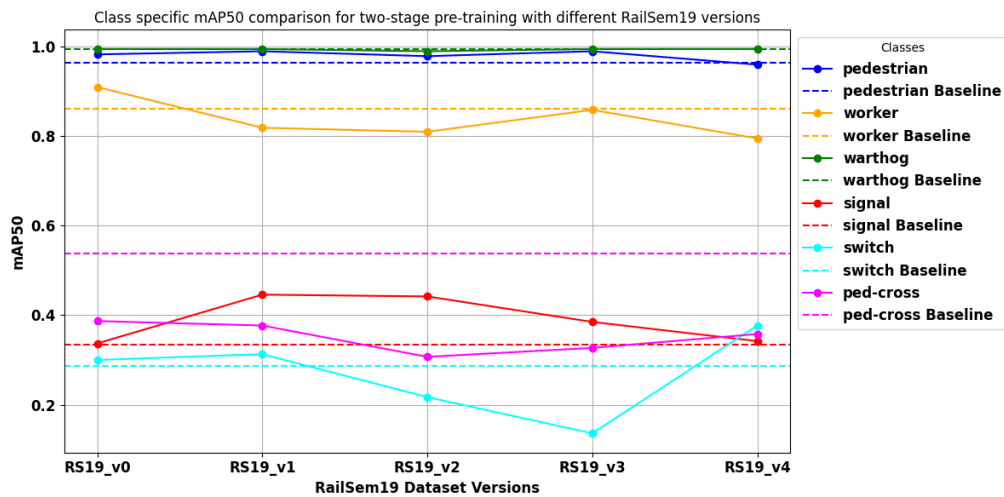


Figure 4.19: Class-specific mAP50 comparison for two-stage pre-training with different RailSem19 versions

Class-specific RailSem19 version impact study: Table 4.16 summarises the results for each class across the RailSem19 variations with two-stage pre-training, highlighting the best-performing RailSem19 version for each object class. The results aim to identify trends in model performance for each object class as the dataset is gradually reduced to align with the custom dataset.

Class	RS19_v0 (F1/mAP50)	RS19_v1 (F1/mAP50)	RS19_v2 (F1/mAP50)	RS19_v3 (F1/mAP50)	RS19_v4 (F1/mAP50)
Pedestrian	0.871/0.983	0.892/0.990	0.906/0.979	0.957/0.990	0.775/0.960
Worker	0.898/0.910	0.829/0.819	0.807/0.810	0.817/0.859	0.746/0.795
Warthog	0.907/0.995	0.928/0.995	0.947/0.990	0.966/0.995	0.930/0.995
Signal	0.370/0.337	0.494/0.446	0.522/0.442	0.349/0.385	0.430/0.342
Switch	0.334/0.300	0.311/0.313	0.190/0.217	0.245/0.136	0.423/0.377
Ped-cross	0.350/0.387	0.445/0.377	0.441/0.307	0.328/0.327	0.524/0.358

Table 4.16: Class-specific results comparison across different RailSem19 versions (two-stage pre-training), with best-performing versions highlighted.

- For the pedestrian class, RS19_v3 achieves the highest F1 score of 0.957 and a mAP50 of 0.990, indicating the best overall detection accuracy for the pedestrian class. The F1 score generally increases from RS19_v0 to RS19_v3. However, there is a notable decline in the F1 score in RS19_v4. mAP50 remains a good result over 0.96 for all scenarios.
- For worker class, RS19_v0 achieves the highest F1 score of 0.898 and a mAP50 of 0.910, The F1 score shows a general decrease from RS19_v0 to RS19_v4, with RS19_v3 having a slight increase compared to RS19_v2. The mAP50 also follows a similar trend, showing a decrease overall with a slight increase in RS19_v3.
- For warthog: RS19_v3 achieves the highest F1 score of 0.966 and a mAP50 of 0.995. The F1 score generally increases from RS19_v0 to RS19_v3, showing consistent improvements in detection performance before slightly decreasing in RS19_v4. The mAP50 remains consistently high across all versions.
- For signal, RS19_v2 achieves the highest F1 score of 0.522 and 0.442. The F1 score for the signal class shows an initial increase from RS19_v0 to RS19_v2, followed by a decrease in RS19_v3 and a slight recovery in RS19_v4. The mAP50 exhibits a similar pattern, with a peak at RS19_v1, followed by a decline.
- For switch, RS19_v4 achieves the highest F1 score of 0.423 and the highest mAP50 of 0.377, The performance trend shows a general decrease from RS19_v0 to RS19_v3,

followed by a significant increase in RS19_v4 for both F1 score and mAP50.

- For pedestrian crossing, RS19_v4 achieves the highest F1 score of 0.524. The highest mAP50 is achieved by RS19_v0 at 0.387. The performance trend shows fluctuations in both F1 score and mAP50, with a significant increase in F1 score in RS19_v4, while mAP50 does not follow a clear trend.

One-stage pre-training with f0135

All-class performance analysis: as shown in Figure 4.20, all RailSem19 versions under the f0135 freezing scenario perform worse than the baseline results, which achieves an F1 score of 0.660 and a mAP50 of 0.663. Within each RailSem19 variation, the impact across different RailSem19 versions with the f0135 freezing scenario exhibits a different trend to the non-freezing setup; RS_v0 achieves the best all-class performance, with an F1 score of 0.606 and a mAP50 of 0.624. The results suggest that freezing layers 0, 1, 3, and 5 help retain essential features from RS_v0, which benefits more from this approach compared to the other versions. This difference could be due to RS_v0's broader inclusion of object classes, which might provide a richer feature set that benefits more from the preserved low-level features during freezing.

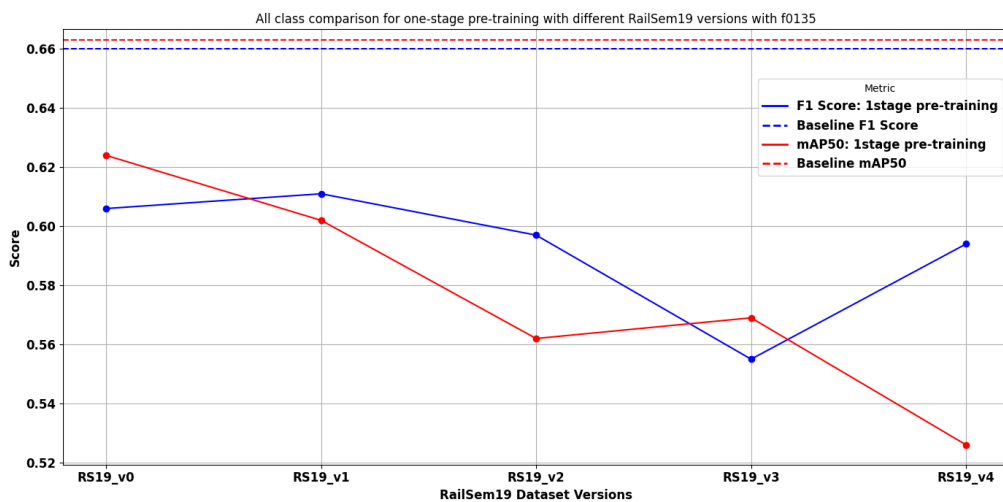


Figure 4.20: All-class comparison for one-stage pre-training with different RailSem19 versions with f0135

Class-specific RailSem19 version impact study: Table 4.17 and Figure 4.21, 4.22 summarise the results for each class across the RailSem19 variations with one-stage pre-training under the f0135 freezing scenario. The results aim to identify trends in model performance for each object class as the dataset is gradually reduced to align with the custom dataset.

Class	RS_v0 (F1/mAP50)	RS_v1 (F1/mAP50)	RS_v2 (F1/mAP50)	RS_v3 (F1/mAP50)	RS_v4 (F1/mAP50)
Pedestrian	0.689/0.847	0.918/0.957	0.773/0.839	0.766/0.832	0.772/0.858
Worker	0.804/0.788	0.646/0.714	0.739/0.733	0.763/0.811	0.421/0.396
Warthog	0.896/0.995	0.949/0.992	0.846/0.978	0.933/0.995	0.974/0.995
Signal	0.452/0.448	0.270/0.323	0.423/0.335	0.292/0.305	0.364/0.351
Switch	0.321/0.280	0.309/0.192	0.256/0.175	0.145/0.156	0.346/0.271
Ped-cross	0.403/0.384	0.495/0.433	0.438/0.309	0.374/0.313	0.384/0.283

Table 4.17: Class-specific results comparison across different RailSem19 versions (one-stage pre-training with f0135), with best-performing versions highlighted.

- **Non-railway object classes:** for non-railway object classes, the performance of different RailSem19 versions using f0135 shows consistent results with slight variations. The F1 scores for the pedestrian and warthog classes remain relatively stable across versions v0 to v4. However, there is a notable drop in the F1 and mAP50 scores for the worker class at v4. This significant drop in v4 could be due to the dataset's reduced diversity, which may limit the model's ability to generalise and accurately detect workers. The reduced number of classes in v4 might have led to overfitting on the remaining classes, thus not providing enough variety for effective training of the worker class.
- **Railway object classes:** the performance for railway object classes using the f0135 freezing scenario reveals several trends. Signal and pedestrian crossing detection remains relatively stable across different RailSem19 versions, with minor variations. In contrast, switch detection shows a gradual decline in performance until RS_v3, followed by an improvement in RS_v4. This pattern suggests that removing object classes that assist in identifying switches, such as switch indicators, in

v4 likely reduced noise and improved the model's accuracy for switch detection. Overall, the trends are consistent with the non-freezing scenario, underscoring the importance of dataset composition in influencing model performance.

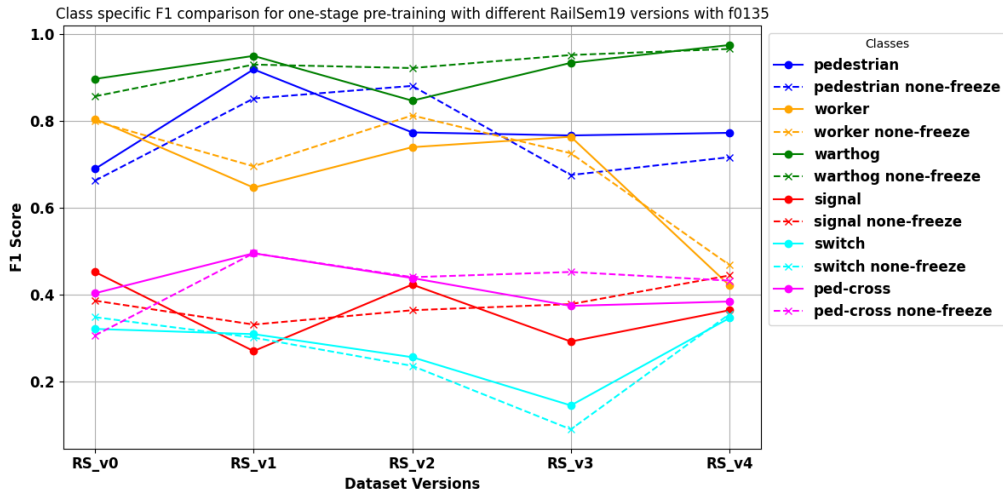


Figure 4.21: Class-specific F1 comparison for one-stage pre-training with different RailSem19 versions with f0135

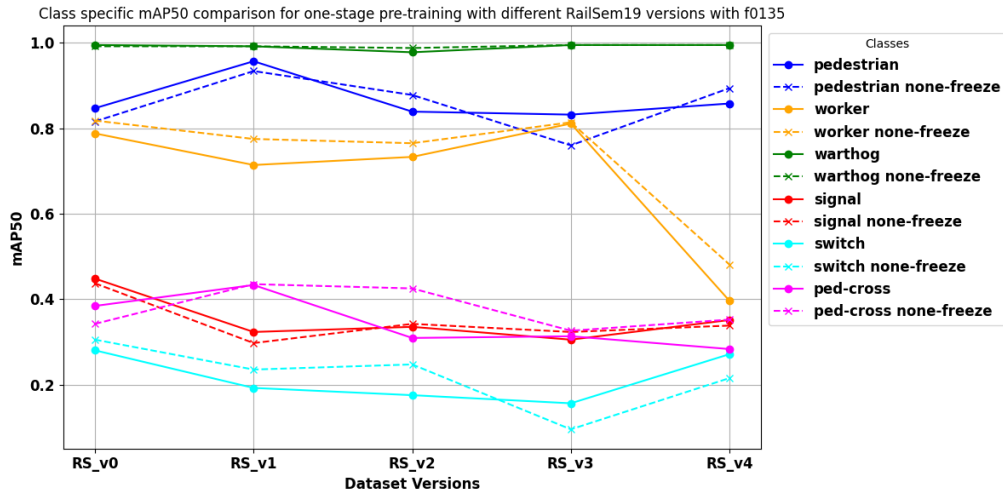


Figure 4.22: Class-specific mAP50 comparison for one-stage pre-training with different RailSem19 versions with f0135

Two-stage pre-training with f0135

All-class performance analysis: as is shown in Figure 4.23, the all-class performance for two-stage pre-training with the f0135 freezing scenario shows a general decline as the

number of object classes in the RailSem19 dataset is reduced. Initially, performance is higher with more object classes but decreases as classes are removed. However, there is a temporary increase in performance at RS_v3 before it drops again at v4.

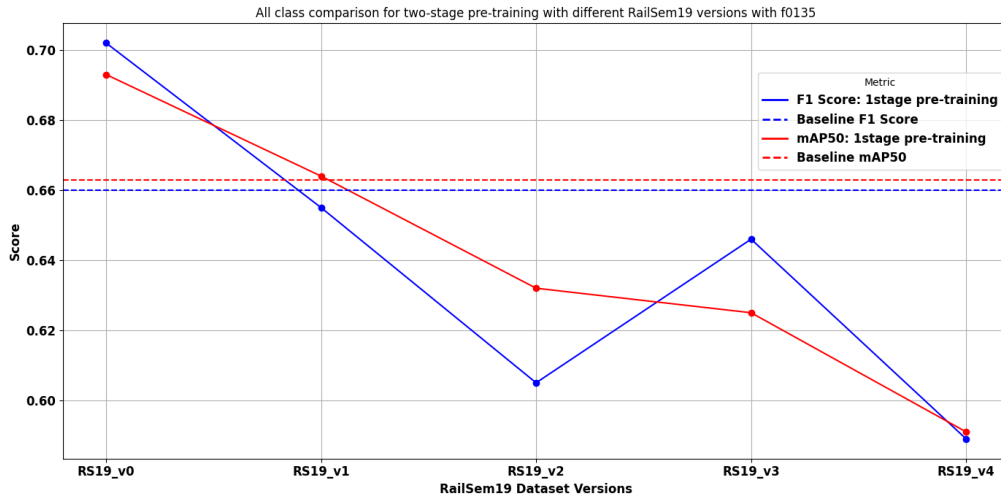


Figure 4.23: All-class comparison for two-stage pre-training with different RailSem19 versions with f0135

Class-specific RailSem19 version impact study: Table 4.18 and Figure 4.24, 4.25 summarise the results for each class across the RailSem19 variations with two-stage pre-training under the f0135 freezing scenario. The results aim to identify trends in model performance for each object class as the dataset is gradually reduced to align with the custom dataset.

Class	RS_v0 (F1/mAP50)	RS_v1 (F1/mAP50)	RS_v2 (F1/mAP50)	RS_v3 (F1/mAP50)	RS_v4 (F1/mAP50)
Pedestrian	0.889/0.973	0.897/0.983	0.855/0.969	0.934/0.995	0.607/0.789
Worker	0.832/0.879	0.851/0.827	0.877/0.841	0.762/0.752	0.615/0.781
Warthog	0.945/0.995	0.948/0.995	0.912/0.988	0.951/0.995	0.889/0.992
Signal	0.474/0.423	0.484/0.501	0.404/0.442	0.441/0.363	0.377/0.241
Switch	0.431/0.390	0.249/0.260	0.254/0.237	0.229/0.219	0.344/0.287
Ped-cross	0.519/0.501	0.408/0.417	0.290/0.314	0.499/0.426	0.563/0.456

Table 4.18: Class-specific results comparison across different RailSem19 versions (two-stage pre-training with f0135), with best-performing versions highlighted.

- **Non-railway object classes:** for non-railway object classes, the performance gen-

erally remains stable from versions v0 to v3 with the f0135 freezing scenario. The detection accuracy for the pedestrian and warthog classes is relatively consistent, with the pedestrian class showing a peak in RS_v3. The worker class also maintains stable performance up to RS_v3, with the highest performance in RS_v2. However, there is a noticeable drop in performance for the pedestrian, worker, and warthog classes in RS_v4.

- **Railway object classes:** for railway object classes, the results demonstrate stability across versions v0 to v3, likely due to preserved low-level features from the f0135 freezing strategy. The detection accuracy for the signal class shows minor fluctuations but is relatively consistent. In contrast, the switch class experiences a gradual decline in performance from v0 to v3. The pedestrian crossing class exhibits the most variability, maintaining stable performance with peaks in versions v0 and v3. However, in RS_v4, the signal class performance declines, while the switch and pedestrian crossing classes show an increase in performance. This improvement in v4 can be attributed to the removal of less relevant classes (Section 4.2.3), such as switch indicators, which may have reduced noise and contributed to the improved detection of these specific railway objects.

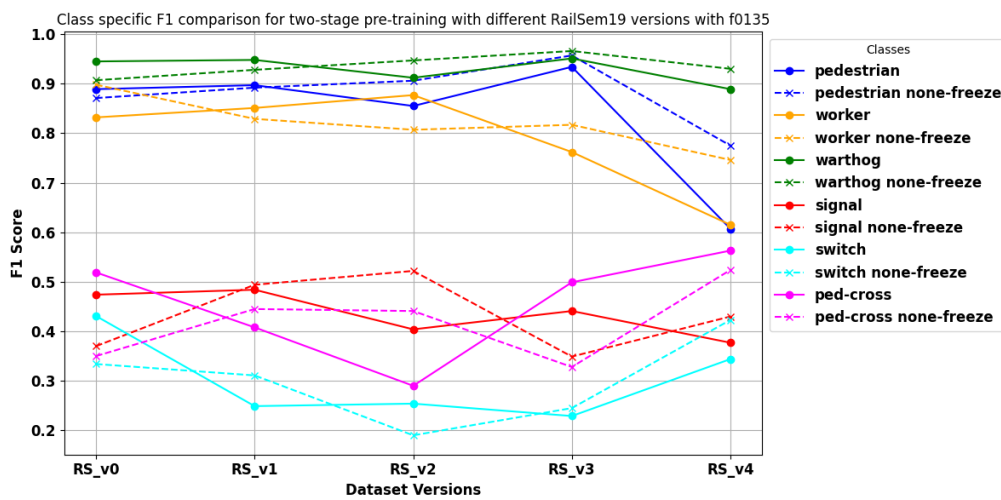


Figure 4.24: Class-specific F1 comparison for two-stage pre-training with different RailSem19 versions with f0135

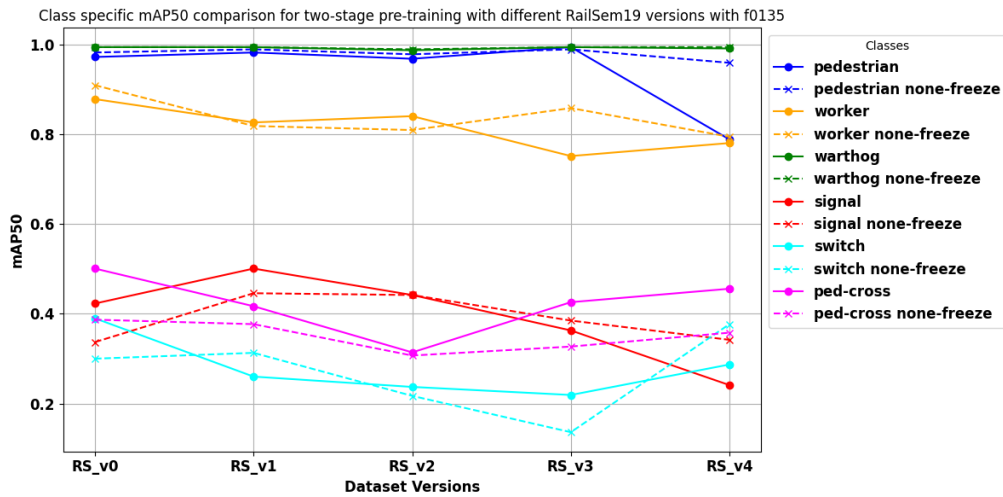


Figure 4.25: Class-specific mAP50 comparison for two-stage pre-training with different RailSem19 versions with f0135

4.4.7 Key findings and discussions

One-stage vs. two-stage pre-training

Two-stage pre-training outperforms one-stage pre-training for both F1 scores and mAP50. By calculating the average across all RailSem19 versions, the F1 score improves by approximately **9.6%**, while the mAP50 improves by around **8.8%**. These improvements are further enhanced with fine-tuning. Specifically, with the best-performing freezing strategy (f0135), two-stage pre-training achieved an average F1 score improvement of **8.0%** and an average improvement of **11.2%** in mAP50. The additional pre-training stage on COCO before fine-tuning on RailSem19 helps the model refine its feature representations, leading to improved detection accuracy across most object classes.

Discussion: the benefit of two-stage pre-training over one-stage pre-training can be attributed to the progressive refinement of transferable features. [123] explains that multi-stage TL enables a more effective alignment of source and target domain features, allowing the model to develop more robust and transferable feature representations. This gradual refinement process reduces domain shifts and mitigates negative transfer, as features are progressively tuned to be more task-specific. Similarly, [128] highlights that

incorporating an intermediate stage of pre-training on a related but more general dataset enhances final task performance compared to direct transfer from a single source. This intermediate stage allows for the extraction of more generalised features, which can later be specialised for the target task. Furthermore, [248] suggests that multi-stage unsupervised feature learning leads to better feature discrimination, as early stages promote coarse-grained feature learning, while later stages refine these features to support fine-grained object detection. This is particularly beneficial for multi-object environments, such as railway scenes, where distinguishing between visually similar objects (e.g., switches and rails) is crucial. Additionally, [249] proposes that learning to transfer knowledge across multiple stages allows for the systematic optimisation of transferable features, which is more effective than a single pre-training stage. This approach not only improves feature generalisation but also reduces the reliance on large, labelled target datasets.

The observed performance gains in F1 score and mAP50 validate the theoretical benefits of multi-stage TL. The findings demonstrate that the two-stage approach promotes better feature alignment, enhanced discrimination of visually similar objects, and improved generalisation. These outcomes are consistent with the insights of [123, 128, 248, 249], reinforcing the efficacy of progressive feature refinement to address domain shifts and improve object detection in railway maintenance scenarios.

Transfer from large general or smaller relevant datasets

TL using a large, general dataset (COCO) generally outperforms the smaller domain-relevant dataset (RailSem19) in the one-stage pre-training approach with the best fine-tuning strategies applied. The results show an improvement in the F1 score by **13.1%** and an improvement in mAP50 by **10.7%**. Non-railway object classes show an F1 improvement of **11.9%**, with a slight decrease in mAP50 by **2.1%**. Railway object classes demonstrate significant improvements with approximately **18.86%** and by **32.9%** in F1 and mAP50. However, RailSem19 demonstrates slightly better performance for the warthog class, with an improvement of **0.2%** in F1 and **5.0%** in mAP50. For the switch class,

RailSem19 significantly outperforms COCO, with a substantial improvement of **35.7%** in F1 and **9.4%** in mAP50.

The results indicate that TL from a larger general dataset enables the model to learn more generalised and transferable features. This advantage is particularly evident for non-railway objects. RailSem19 slightly outperforms COCO in the warthog class, likely due to incidental overlaps in domain-specific features, though the improvement is marginal since warthogs are not explicitly included in either dataset. However, RailSem19 significantly excels in the switch class, achieving a **35.73%** improvement in F1, as switches are included and well-represented in the dataset. Conversely, although pedestrian crossings and signals are railway-related, they are not strictly railway infrastructure and are less represented in the RailSem19 dataset, limiting its advantage. COCO's generalisation capabilities allow it to perform better in these classes, reflecting its broader applicability across diverse object types.

Discussion: the observed results reveal that TL from a large, general dataset like COCO outperforms the RailSem19, a smaller domain-specific dataset, for most object classes. This performance boost can be attributed to several key factors supported by existing literature. Large, general datasets like COCO expose the model to a wider variety of object classes and visual concepts, enabling the development of low-level and mid-level features that are more transferable to new tasks [250, 4]. The extensive class diversity in COCO allows for the extraction of more general-purpose features that are useful across multiple target tasks [124, 11, 251]. Additionally, large datasets facilitate the development of more robust initial weights that are less susceptible to divergence during fine-tuning [128]. This is particularly beneficial for non-railway object detection (such as pedestrians and workers), where COCO's broader visual representation improves the model's ability to identify these classes.

On the other hand, domain-specific datasets like RailSem19 provide certain advantages for railway-related classes. For example, the "switch" class benefits significantly

from RailSem19, which is well-represented in the dataset but inadequately captured in COCO. This is consistent with findings that domain-specific datasets offer focused and task-relevant features that are not always learned from general datasets [251, 129]. Furthermore, TL with domain-related source datasets can reduce the computational complexity for more efficient learning of representations for those specific objects [127].

In summary, the results highlight the advantages of TL with large, general datasets and smaller, domain-specific datasets. While general datasets like COCO offer broad generalisation capabilities that support non-railway object detection, domain-specific datasets like RailSem19 are beneficial for railway-specific objects. These findings align with the broader insights from [124, 128, 251, 129], which collectively emphasise the importance of leveraging both generalised and domain-specific datasets to achieve robust TL performance in highly specialised tasks like railway object detection.

Impact of RailSem19 versions

In the non-freezing scenario, both one-stage and two-stage pre-training saw an initial improvement with RS19_v1 (e.g., **7.3%** and **0.7%** improvement for F1 and mAP50 under a two-stage approach). However, these improvements are negligible to overall performance. Interestingly, although RS_v4 performed worse than RS_v0, it showed a notable recovery compared to RS_v3 in the two-stage pre-training approach. RS_v4 improved F1 score by **5.6%** and a mAP50 by **3.7%** over RS_v3. By investigating class-specific performance, both the initial improvement and recovery are contributed by railway object classes (as shown in Figures 4.18 and 4.19), where railway object detection benefits from reduced noise and a more focused dataset with fewer object classes.

In the freezing scenario (f0135), both one-stage and two-stage pre-training show a consistent decline in performance as the dataset is reduced from RS19_v0 to RS19_v4. For the two-stage approach, the F1 score decreases by approximately **16.1%**, while the mAP50 decreases by around **14.7%**. By examining class-specific performance, the overall decline is primarily driven by non-railway objects. For example, as shown in Figure

4.24, the pedestrian class decreased by approximately **31.7%** in F1, while the pedestrian crossing class increased by around **8.5%**. Other railway-specific objects, such as signals and switches, demonstrated stable or slightly varying performance.

Discussion: the observed results demonstrate the trade-off in object detection when reducing the number of object classes in RailSem19. For railway-specific objects, the reduction in class diversity enables the model to focus on domain-relevant features, leading to improved detection performance. [252, 128, 253] highlight that datasets with fewer domain-specific classes reduce learning complexity, minimise class competition, and decrease semantic confusion, allowing the model to prioritise essential features and achieve more stable convergence. The removal of non-relevant classes in RS19_v4 reduces dataset noise, allowing the model to develop more discriminative feature representations for target objects, thereby improving the detection of railway-specific objects.

On the other hand, for non-railway objects, reducing the number of classes constrains the model's exposure to a diverse range of object representations, leading to a decline in generalisation performance. [251, 128, 254] explain that reducing class diversity limits generalisation capacity, weakens feature robustness, and causes the model to overfit to the smaller set of target-specific features. As a result, non-railway object detection is negatively impacted in RS19_v4, as the model struggles to generalise to unseen classes that are no longer present in the training data.

These findings align with broader TL principles. [255, 256, 257] suggest that reducing the number of object classes allows for a more focused latent feature space, enhancing task-specific transferability. However, this comes at the cost of reduced generalisation to broader object categories. In summary, reducing the number of object classes in RailSem19 improves railway object detection by reducing noise and enables the model to focus on domain-relevant features. However, the performance of non-railway object detection declines due to the loss of dataset diversity. Ultimately, the choice of dataset should align with application-specific priorities, whether optimising for railway-specific

objects or balancing performance across a broader range of objects.

Layer freezing techniques

The study also examines the impact of freezing different layers during fine-tuning. The f0135 freezing scenario, which involves freezing early convolutional layers, generally leads to better performance than other freezing strategies. For example, in two-stage pre-training with RS_v0, the F1 and mAP50 improved **12.1%** and **6.3%** respectively. This approach helps preserve low-level features that are crucial for object detection. However, extensive freezing, such as f0to15, significantly hampers performance. For instance, with the one-stage pre-training approach with the COCO dataset, the F1 score decreased by approximately **5.6%** and the mAP50 by **6.0%**, highlighting the limitations of excessive freezing, which restricts the model's ability to adapt to the new dataset.

Discussions: the observed results are consistent with the hierarchical feature learning properties of CNN. [119] explains that early layers of CNNs learn general features, such as edges and textures, which are transferable across a wide range of tasks. Freezing these layers, as done in the f0135 freezing strategy, helps preserve general features while allowing deeper layers to adapt to the target dataset. [120] reinforces this idea by showing that freezing the initial layers allows for the transfer of common, reusable features while allowing deeper layers to adapt to new, task-specific features.

However, freezing too many layers, as in the f0to15 strategy, restricts the model's ability to adapt to domain-specific features. This occurs because deeper layers capture source-specific features, and freezing them prevents adaptation to new domains [258, 259]. Excessive freezing also limits the model's capacity to learn domain-relevant representations, particularly when source and target datasets differ in feature distribution [259, 260]. [127] highlights that deeper layers are more susceptible to co-adaptation, which hinders transferability if frozen. Additionally, [261] argues that effective TL requires deeper layers to remain adaptable, as they play a crucial role in capturing domain-specific representations.

4.4.8 Future work

The findings suggest that it is more beneficial to TL from larger datasets (COCO) than relatively smaller but domain-relevant datasets (RailSem19). Future research should focus on quantifying the relationship between source dataset size and performance improvement by analysing how the increment in source dataset size impacts object detection accuracy. It is important to establish a TL benchmark to facilitate the comparison of different pre-training strategies, dataset sizes, and TL techniques. Investigating the impact of different object detection architectures on TL performance is another promising direction, as more efficient or domain-specific architectures (e.g., lightweight models for edge deployment) may better handle the unique challenges of railway environments. Finally, future work should expand the custom dataset by including more varied weather conditions (such as low light or rain) and a broader range of object classes relevant to the railway maintenance environment (such as other equipment and tools) for real-world applications.

4.5 Chapter conclusion

This chapter aims to investigate the impact of various TL strategies on railway object detection. By leveraging pre-training on large general-purpose datasets (e.g., COCO) and domain-specific datasets (e.g., RailSem19), the study explored how dataset composition, pre-training strategies, and layer-freezing techniques influence detection performance for railway maintenance environments. The chapter also evaluated one-stage and two-stage pre-training approaches, comparing the effectiveness of TL for fine-tuning a custom dataset tailored for railway maintenance environments.

4.5.1 Key contributions

This chapter contributes to the understanding of TL for railway object detection as follows:

- **Dataset contribution:** created a custom dataset with annotations for six object classes that commonly appear in railway maintenance environments, essential for detection in autonomous maintenance operations.
- **Technical contribution:** demonstrated how to optimise TL performance for railway maintenance object detection by leveraging a two-stage pre-training approach that combines general (COCO) and domain-specific (RailSem19) datasets.
- **Theoretical contribution:** provided evidence on the trade-offs between large, diverse datasets and smaller, domain-specific datasets, highlighting the benefits of dataset alignment and diversity for object detection.
- **Methodological contribution:** introduced a systematic approach to dataset configuration by reducing object classes in RailSem19 and assessing its impact on railway-specific object detection with TL. The study also evaluated different layer-freezing strategies, identifying the optimal freezing strategy to improve detection performance with the custom dataset.

4.5.2 Summary of key findings

The key findings from this chapter are as follows:

- **Two-stage pre-training outperforms one-stage pre-training:** two-stage pre-training resulted in an average increase of **9.6%** in F1 score and **8.8%** in mAP50 across all RailSem19 versions, demonstrating the advantage of sequential knowledge transfer.
- **Large general-purpose datasets are more effective than smaller domain-specific datasets:** TL using COCO outperformed RailSem19 by an average of **13.1%** in F1 and **10.7%** in mAP50. However, RailSem19 showed superior performance in detecting railway-specific objects like switches.
- **Dataset class reduction in RailSem19 impacts detection performance:** reducing the number of object classes in RailSem19 improves the detection performance

of railway-specific objects while reducing the detection accuracy for non-railway objects.

- **Layer-freezing enhances feature preservation:** freezing early convolutional layers (f0135 strategy) improves performance, leading to an F1 increase of **12.1%** and a mAP50 increase of **6.3%** under the two-stage pre-training approach.

4.5.3 Research impact

The research presented in this chapter has a significant impact and broader implications for the development of autonomous railway maintenance systems. By advancing object detection capabilities through the aforementioned TL strategies, this study contributes to the creation of an object detection system for autonomous railway maintenance vehicles. The methodological insights offer transferable knowledge applicable to other domains that are also facing dataset scarcity challenges. Additionally, the ability to detect maintenance-specific objects improves the safety, operational efficiency, and reliability of autonomous railway maintenance systems. This research not only addresses key industry challenges but also lays the groundwork for future innovations in railway automation.

Chapter 5

3D Object Tracking and Collision

Avoidance

5.1 Introduction

The advancement of autonomous railway navigation requires a context-aware collision avoidance system that is capable of continuously tracking detected objects. The RIMV can anticipate their future trajectories by determining both the type and movement of the objects, therefore enabling proactive and intelligent motion planning and decision-making. For instance, dynamic objects such as trespassers require different avoidance strategies compared to maintenance workers. Furthermore, the autonomous system can optimise its movement to avoid collision by analysing the detected objects' movement. This predictive capability not only enhances safety but also improves efficiency by minimising unnecessary stops and delays.

As discussed in Chapter 3, cameras are suitable for object detection tasks by capturing rich visual data to differentiate different railway objects. However, their ability to measure distances accurately is limited, especially under challenging lighting conditions like low light. On the other hand, LiDAR can provide accurate distance measurements despite the ambient lighting. However, LiDAR needs more comprehensive visual information for

object detection compared to cameras. By integrating cameras and LiDAR, the system benefits from the strengths of both sensors: cameras enable comprehensive object detection and detection, while LiDAR provides accurate distance measurements. This section provides the methodological approach, experimental setup, validation experiments, and analysis of results, providing a comprehensive examination of the multi-sensor fusion strategy for 3D object tracking.

The objectives of this chapter are:

- Develop a robust camera-LiDAR calibration framework ensuring accurate transformation between sensor coordinate systems.
- Implement a multi-modal sensor fusion algorithm for effective 3D object tracking and distance measurement.
- Develop a context-aware collision avoidance system to make responsive actuation based on the detected railway object type and movement.
- Validate the integrated collision avoidance system under various environmental conditions and operational scenarios.

The rest of this chapter is structured as follows:

- **Section 5.2** describes the methodology approach and implementation steps, which include camera-LiDAR calibration, data processing, transformation techniques, point-cloud clustering and distance measurement.
- **Section 5.3** describes the design of the validation experiments, covering experimental setup, system robustness assessments, and final outdoor validation.
- **Section 5.4** presents the results obtained from the experiments and discusses their implications.
- **Section 5.5** Discussion on the chapter, summarising key findings and suggestions.

5.2 Methodology

5.2.1 Methodology approach

Tracking framework

YOLOv8 can be integrated with tracking capabilities by using algorithms like ByteTrack [262] and BoT-Tracking [263]. These algorithms enhance YOLOv8’s object detection by maintaining consistent identities of detected objects across multiple frames. Figure 5.1 illustrates an example of 2D tracking. Each detected object is assigned an ID and object type (e.g., id:1 for the warthog and id:2 for a human) along with a detection confidence score (e.g., 0.92 for the warthog). Bounding boxes highlight the spatial location of the objects, and positional labels (e.g., Up-Left, Down-Right) are used to indicate object movement within the camera’s 2D coordinate system.



Figure 5.1: Example of 2D tracking result, the “warthog” vehicle refers to another type of autonomous railway robot used for maintenance tasks.

Building on YOLOv8’s 2D tracking, the proposed 3D object tracking system implements a multi-sensor fusion approach by combining camera and LiDAR. The system detects objects in 2D image space with camera data and projects these detections into 3D space using LiDAR data for distance measurement. The key components of the system

include:

- Camera-to-LiDAR Transformation: mapping Bounding Box (BBX) detected in camera 2D coordinate system into the LiDAR's 3D coordinate system, providing spatial alignment between the two sensors.
- Frustum Construction: extending the 2D BBX into a 3D frustum that serves as a spatial filter, allowing LiDAR to focus on the area for efficiency.
- Distance measurement: applying pointcloud clustering algorithms within the frustum to group relevant LiDAR points and measure distance.

5.2.2 Camera-LiDAR calibration

Introduction

Figure 5.2 shows the comparison between the camera and LiDAR's coordinate system, in order to transform the detected object from camera to LiDAR's coordinate system, it is necessary to determine the transformation parameters (rotation and translation) between the camera and LiDAR.

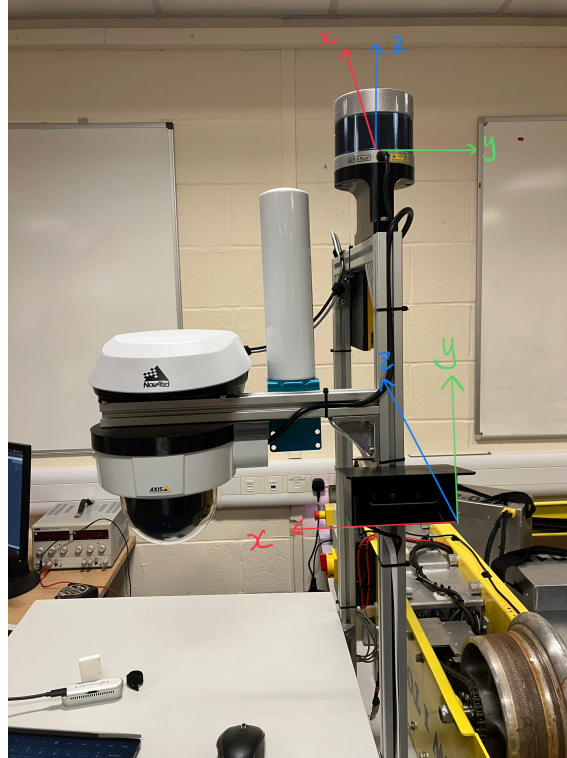


Figure 5.2: Camera and LiDAR coordinate comparison

Figure 5.3 shows the chessboard selected as the calibration target. The high-contrast pattern can be easily detected by both cameras and LiDAR [264]. The grid pattern of a chessboard provides sufficient points of reference that can be identified to facilitate the alignment process [265]. Additionally, the planar and uniform surface of a chessboard simplifies the geometric modelling required for calibration, ensuring that the intrinsic and extrinsic parameters of the sensors can be precisely determined [266]. The regular geometric arrangement of the squares aids in reducing calibration errors and allows for accurate feature extraction and matching between two sensors [267].



Figure 5.3: Chessboard used for camera LiDAR calibration

Mathematical foundation of camera-LiDAR calibration

The calibration between the camera and LiDAR is the process of identifying the parameters that define the rotation and translation between the two sensors' coordinate systems. This process involves extracting and aligning features from both sensors, computing transformation parameters through optimisation techniques, and evaluating the calibration quality using metrics like the condition number and Variability of Quality (VOQ) [268].

Extrinsic calibration: the mathematical foundation in camera-LiDAR calibration is to determine the transformation matrix between the camera and LiDAR coordinate systems. This transformation matrix $\mathbf{T}$ is defined by a rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$$

where $\mathbf{R}$ is a 3×3 rotation matrix, and $\mathbf{t}$ is a 3×1 translation vector. Given a point $\mathbf{p}_C$ in the camera coordinate system, its corresponding point $\mathbf{p}_L$ in the LiDAR coordinate system can be expressed as:

$$\mathbf{p}_L = \mathbf{R}\mathbf{p}_C + \mathbf{t}$$

Feature extraction: to determine the transformation matrix, it is necessary to extract and match features from both camera and LiDAR. In the camera images, features such as chessboard corners are detected. These 2D coordinates in the image plane can be converted to 3D coordinates using the intrinsic parameters of the camera. In the LiDAR point-cloud, the corresponding features in the camera image can be identified by fitting planes and finding intersections using algorithms like Random Sample Consensus (RANSAC).

Normal matrix and rotation estimation: after extracting the normal vectors of the chessboard planes in both the camera and LiDAR coordinate systems. These normal vectors are used to estimate the rotation matrix that aligns the camera coordinate system with the LiDAR coordinate system.

Let $\mathbf{N}_C$ and $\mathbf{N}_L$ be the normal matrices of the camera and LiDAR, respectively:

$$\mathbf{N}_C = \begin{bmatrix} \mathbf{n}_{C1} & \mathbf{n}_{C2} & \mathbf{n}_{C3} \end{bmatrix}, \quad \mathbf{N}_L = \begin{bmatrix} \mathbf{n}_{L1} & \mathbf{n}_{L2} & \mathbf{n}_{L3} \end{bmatrix}$$

The rotation matrix $\mathbf{R}$ can be estimated by aligning these normal vectors:

$$\mathbf{R}\mathbf{N}_C = \mathbf{N}_L$$

This equation can be solved using optimisation techniques such as the Levenberg-Marquardt algorithm, which minimises the difference between the observed and predicted positions of the features in the camera and LiDAR coordinate systems.

Optimisation techniques: several optimisation techniques are used to refine the transformation parameters and maximise calibration accuracy. These techniques focus on minimising errors and ensuring stability in the estimation process. The optimisation includes several key components:

Condition number and matrix stability The condition number $\kappa(\mathbf{N})$ of the normal matrix is an important metric for assessing the stability of the rotation estimation. It is defined as:

$$\kappa(\mathbf{N}) = \|\mathbf{N}^{-1}\mathbf{N}\|$$

A lower condition number indicates a more stable and reliable estimation of the rotation matrix. Linear dependency in the normal matrix can cause instability, making the condition number a critical factor in evaluating the quality of the calibration. To ensure a robust estimation, the normal matrix should have a low condition number, indicating that the matrix inversion is stable.

Translation estimation The translation vector $\mathbf{t}$ is estimated by aligning the centre of the chessboard in both coordinate systems. Errors in feature extraction, especially in LiDAR measurements, can affect the accuracy of this estimation. The sum of dimension errors e_{dim} for the chessboard is used to quantify these inaccuracies:

$$e_{\text{dim}} = \sum_{i=0}^3 |l_{L,i} - l_{M,i}|$$

where $l_{L,i}$ are the LiDAR measurements of the chessboard edges, and $l_{M,i}$ are the known physical dimensions. Minimising these errors helps in accurately determining the translation vector, ensuring that the points in the camera's 2D coordinate system are correctly mapped to the points in the LiDAR's 3D coordinate system.

VOQ metric VOQ metric is introduced to evaluate the overall quality of the calibration. It combines the condition number of the normal matrix and the chessboard dimension errors:

$$\text{VOQ} = \kappa(\mathbf{N}) + e_{\text{dim}}$$

A lower VOQ score indicates a more robust and generalisable calibration. This metric helps in selecting the best poses for calibration and assessing the accuracy of the estimated parameters. By minimising the VOQ score, the calibration process ensures that the transformation parameters are both stable and accurate, leading to reliable 3D object

detection and tracking.

Calibration process

The calibration process utilised the `cam_lidar_calibration` package¹ to align the camera and LiDAR coordinate systems. The calibration target was a chessboard printed on A3 paper, with 55 mm squares and 6x4 vertices (7x5 squares). This chessboard was attached to a rectangular A3-sized cardboard, ensuring that both centres were precisely aligned. A total of 27 sample captures were taken at various angles and distances up to 5 meters, as shown in Figure 5.4. These captures ensure the calibration parameters generalise well across different perspectives and ranges. After completing the calibration, the following intrinsic camera matrix was obtained:

$$\begin{bmatrix} 908.035 & 0 & 646.650 \\ 0 & 906.078 & 353.831 \\ 0 & 0 & 1 \end{bmatrix}$$

This matrix represents the intrinsic parameters of the camera, where the focal lengths are [908.035, 906.078], and [646.650, 353.831] is the principal point where the optical axis intersects the image plane. These parameters are essential for converting points in the camera's 2D coordinate system to points in the LiDAR's 3D coordinate system.

¹available at: https://github.com/acfr/cam_lidar_calibration

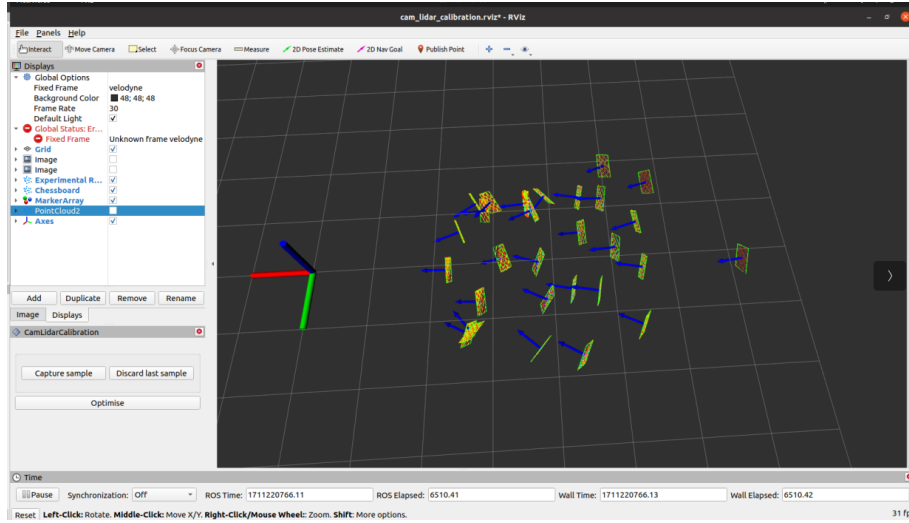


Figure 5.4: Visualisation of calibration process with RVIZ2

The transformation parameters from the camera to the LiDAR coordinate system were determined as follows:

$$\text{Translation vector: } \mathbf{t} = \begin{bmatrix} -0.00122 \\ 0.00784 \\ -0.49320 \end{bmatrix}$$

$$\text{Rotation matrix: } \mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

The rotation matrix $\mathbf{R}$ is computed using the Euler angles (roll, pitch, yaw): (-1.65923, 0.02178, 1.58054) radians. The conversion from Euler angles to the rotation matrix involves the following steps:

1. ****Roll (rotation around the x-axis):****

$$\mathbf{R}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix}$$

2. **Pitch (rotation around the y-axis):**

$$\mathbf{R}_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$

3. **Yaw (rotation around the z-axis):**

$$\mathbf{R}_z(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The overall rotation matrix $\mathbf{R}$ is then obtained by combining these individual rotations:

$$\mathbf{R} = \mathbf{R}_z(\psi)\mathbf{R}_y(\theta)\mathbf{R}_x(\phi)$$

5.2.3 Camera to LiDAR transformation

The previous section discussed the calibration process to identify the transformation matrix between the camera and LiDAR. However, the transformation matrix alone is not sufficient for accurate distance measurement. An initial depth estimation from the stereo camera is also required to determine the distance of the object. This section discusses the mathematical foundation of depth estimation with a stereo camera. Additionally, stereo cameras often face noisy estimation, especially around the edges of objects. This section also discusses the mitigation techniques.

Mathematical foundation for depth estimation with stereo camera

This method involves capturing two images of the same scene from slightly different angles using a pair of cameras mounted at a known distance apart. The key steps are as follows:

- **Image capture:** The stereo camera captures two images simultaneously.
- **Image rectification:** The images are geometrically aligned to ensure that corresponding points lie on the same horizontal line.
- **Disparity calculation:** By comparing the horizontal displacement (disparity) between corresponding points in the two images, the depth information can be computed. Points that are closer to the camera will have a larger disparity, while points that are farther away will have a smaller disparity.
- **Depth map generation:** The disparity values are converted into depth values using the formula:

$$\text{Depth} = \frac{f \cdot B}{\text{Disparity}}$$

Where f is the focal length of the cameras, B is the baseline distance between the cameras, and Disparity is the difference in the position of a point in the two images.

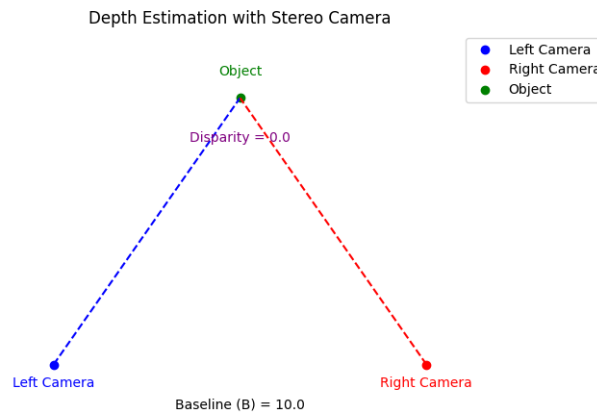


Figure 5.5: Depth estimation with stereo camera

Figure 5.5 shows the two cameras (left and right), the captured images, and how the disparity between corresponding points is used to calculate depth. The closer the object is to the cameras, the larger the disparity, and vice versa.

Depth estimation process

The `depthimage_to_laserscan` package² is used to measure the initial depth by converting depth images into two-dimensional laser scans, allowing stereo cameras to function as laser range finders. This package measures distances for all pixels along the x-axis within the image frame as a laser scan array. This section discusses the process of using the tracked objects' 2D BBX and image ratios to estimate the distance to the tracked objects.

YOLOv8 detect objects and represent their location as BBX, each BBX is defined by its coordinates $(x_{\min}, y_{\min}, x_{\max}, y_{\max})$. the package estimates the distance to each horizontal point and returns the measurements as an array of distances, denoted as $D[i]$, corresponding to different angles. Therefore it is possible to associate the BBX by mapping the horizontal of the bounding box $(x_{\min}, x_{\max})$ to indices in the laser scan array $D[i]$, and the index corresponding to the bounding box edges is calculated as follows:

$$\begin{aligned} \text{min_index} &= \left\lfloor \left(\frac{\text{image_width} - x_{\max}}{\text{image_width}} \right) \times \text{total_ranges} \right\rfloor \\ \text{max_index} &= \left\lfloor \left(\frac{\text{image_width} - x_{\min}}{\text{image_width}} \right) \times \text{total_ranges} \right\rfloor \end{aligned}$$

Where `image_width` is the width of the image (1280 pixels), and `total_ranges` is the size of the distance array $D[i]$.

Mitigation for noisy estimation

It is observed that the depth estimation near the edges of the object tends to be less reliable. This instability is often due to edge effects, where the precision of distance measurements decreases at the boundaries of the bounding box. Factors contributing to this include:

- Stereo matching errors: the disparity calculation near the edges can be less accurate due to occlusions or lack of texture, leading to inconsistent depth readings.

²available at: https://github.com/ros-perception/depthimage_to_laserscan

- Resolution limitations: the effective resolution of the depth camera may vary across the field of view, causing less precise measurements at the edges.
- Sensor noise: noise in the sensor data can result in unreliable readings, particularly near the boundaries of objects.

To enhance the accuracy of the distance estimation, a percentile filter is applied to remove outliers and focus on the central 60% of the data:

$$\text{lower bound} = \text{percentile}(\text{ranges}, 5)$$

$$\text{upper bound} = \text{percentile}(\text{ranges}, 60)$$

This filtering step mitigates the impact of noisy measurements at the edges without affecting the central, more reliable data. The choice of the 5th and 60th percentiles is based on empirical observations, ensuring that most extreme outliers are excluded while retaining a significant portion of the data for accurate estimation. Finally, the average distance from the filtered range measurements is computed:

$$\text{average_distance} = \frac{\sum \text{filtered_ranges}}{\text{number of filtered ranges}}$$

This average distance represents the estimated depth of the object within the bounding box. By implementing these filtering techniques, the system can produce a more accurate and robust depth estimation.

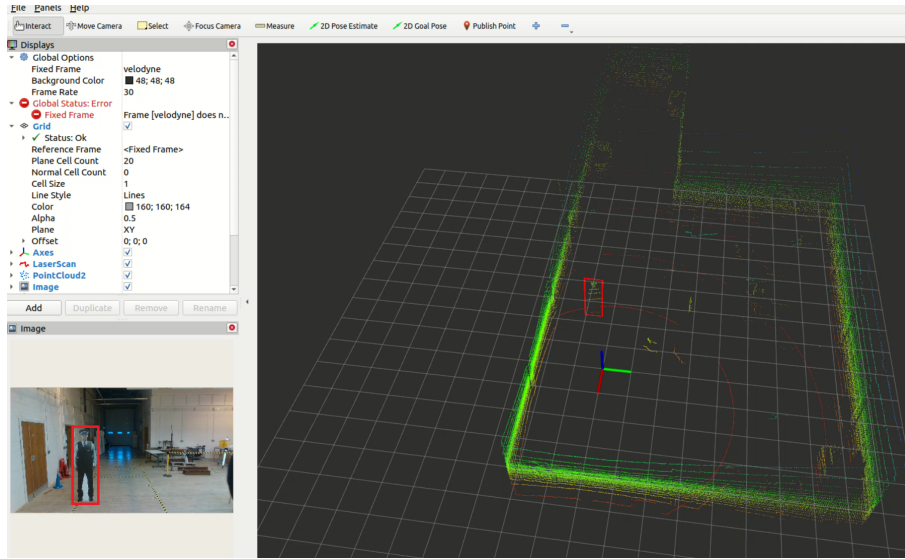


Figure 5.6: Visualisation of the camera to LiDAR transformation

After obtaining the initial depth estimation, it is possible to transform the BBX from the camera's 2D coordinate to BBX in LiDAR's 3D coordinate. Figure 5.6 shows the visualisation. On the left side, the camera image shows the detected pedestrian with a red 2D bounding box that marks the location of the object in the camera's coordinate system. On the right side, the corresponding pointcloud on the right side, the corresponding 3D pointcloud is marked within another 3D bounding box in LiDAR's 3D coordinate system. The green and yellow points in the 3D space represent the LiDAR pointcloud, capturing the spatial information of the environment, including the walls and other objects in the room.

5.2.4 3D frustums construction for object filtering

Figure 5.7 illustrates the frustum construction process within the proposed 3D tracking system. The stereo camera is responsible for both object detection and generating initial depth estimates, producing 2D bounding boxes, object class labels, and associated depth values for each detected object. Using the results from the earlier camera–LiDAR calibration step, these depth values are transformed from the stereo camera coordinate frame into the LiDAR coordinate frame to enable spatial alignment across modalities.

A 3D frustum is then constructed by extending the 2D bounding box into 3D space based on the object's estimated class and depth. The range of the frustum is dynamically adjusted to account for class-specific characteristics—for example, ± 1 metre for smaller objects such as pedestrians and ± 2 metres for larger ones like warthogs. Finally, a clustering algorithm is applied to extract LiDAR pointcloud data within the frustum volume, which is then analysed to estimate the object's distance relative to the vehicle.

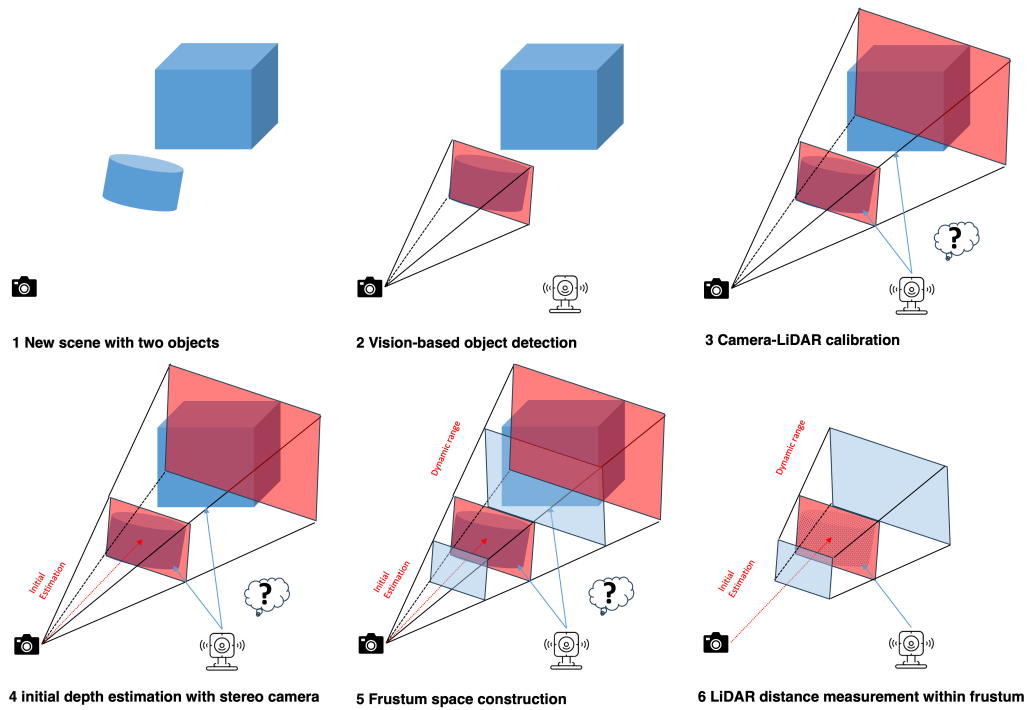


Figure 5.7: Multimodal 3D tracking pipeline using stereo and LiDAR fusion

Figure 5.8 shows an example of frustum constructed from the 3D BBX, which extends 2D BBX into a 3D frustum. The initial distance to the object, estimated using the stereo camera, was 3 metres, while the actual distance was measured at 3.2 metres. This distance is further refined by the LiDAR sensor, improving the overall accuracy of the distance measurement and ensuring robust object tracking in the 3D space. These visual confirmations, alongside the quantitative validation using ground truth distance measurements, confirm the robustness and precision of the calibration.

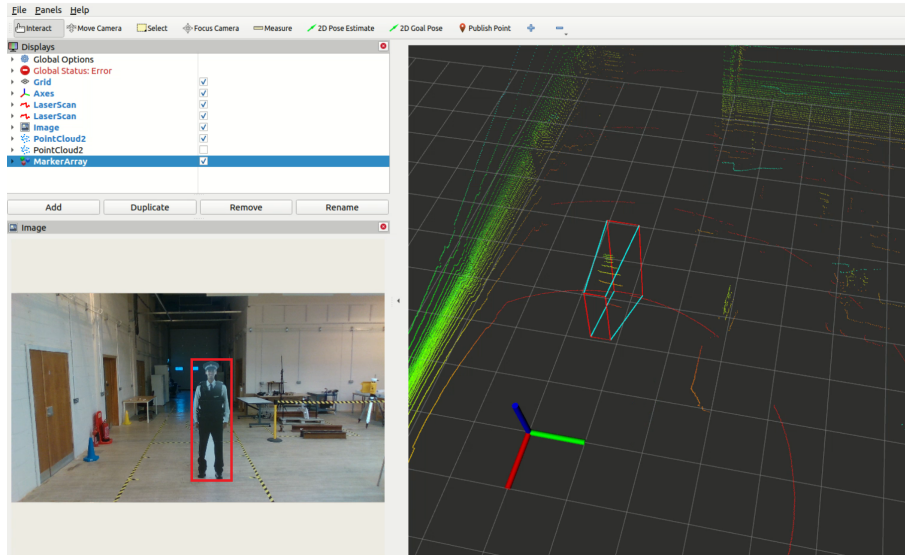


Figure 5.8: Visualisation of 3D frustum

5.2.5 Distance measurement

Pointcloud clustering algorithm

This step groups LiDAR reading that represents the detected objects within the 3D frustum space from the previous step. A convex hull is constructed using Delaunay Triangulation around the frustum's corners to refine the spatial boundaries and ensure that only LiDAR points within the frustum are retained. Subsequently, clustering algorithms such as Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and K-means are used to group the points into distinct clusters. DBSCAN is effective for identifying clusters of arbitrary shapes and sizes and eliminating noise points that do not belong to any object cluster. This is especially useful in complex railway environments with cluttered backgrounds. K-means offers a more structured and efficient clustering method by partitioning points into pre-defined clusters, although it requires specifying the number of clusters beforehand. By filtering points within the frustum, the system reduces redundant data processing and enhances real-time performance.

Distance measurement

After clustering the pointcloud clustering within the 3D frustum space, the distance measurement calculates the minimum Euclidean distance from the sensor to the nearest point in the clustered point. This approach is driven by the safety requirements that the closest point of an obstacle is more relevant for collision avoidance than the cluster's centre. To further improve the measurement accuracy, a percentile filter is applied to the clustered points. The top and bottom 20% of the distance measurements are discarded, and the remaining points are used to calculate the average distance. This approach helps reduce the impact of noise commonly found in LiDAR data, particularly in cases where reflections or outliers are present.

5.3 Experiment design

This section discusses the experimental designed to evaluate the performance and reliability of the 3D object tracking algorithm and the integrated collision avoidance system developed for the RIMV. The validation is divided into **robustness validation (indoor)** and **collision avoidance validation(outdoor)**.

Robustness validation assesses the system's resilience under various lighting environments within an indoor setting. The objective is to ensure that both the object tracking and distance measurement capabilities remain consistent and reliable in a real-world railway maintenance scenario.

Collision avoidance validation tests the fully integrated collision avoidance system in real-world conditions at the RITA (Railway Innovation and Testing Area) at Cranfield University (Figure 5.9). This outdoor validation evaluates the system's ability to classify different dynamic obstacles, track and anticipate their movements in real-time, and execute appropriate actuation decisions based on the detected objects' types and movement patterns. This comprehensive assessment ensures that the RIMV can operate safely and efficiently in live railway environments.

Together, these two validation phases provide a comprehensive examination of the RIMV's 3D object tracking and collision avoidance capabilities, ensuring its readiness for deployment in operational railway maintenance tasks.



Figure 5.9: RITA contains a ballastless track with a standard track gauge of 1.435 metres and a total length of 18 metres, used for testing and validating autonomous railway systems.

5.3.1 Experiment setup

In Figure 5.10, the lighting configuration includes overhead, on-vehicle and external dynamic lights. The overhead lights are turned on incrementally to simulate the brightness at different times of the day. The on-vehicle lights offer direct illumination for night-time or tunnel environments. Lastly, the external dynamic lights are provided by a portable light to simulate the effect of passing vehicles or other equipment. Figure 5.11 shows the placement of two objects, a pedestrian and a warthog robot, positioned 5 and 6 metres away from the RIMV, respectively. These objects, representing typical railway maintenance obstacles, are used to test the system's 3D object tracking performance under various lighting conditions.



Figure 5.10: Light setup



Figure 5.11: Object setup

Hardware and measurement tool

Section 3.2.2 discusses the details of the hardware system used in the robotic framework, which includes a stereo camera for 2D bounding boxes and depth estimation and a LiDAR sensor for 3D pointcloud. CPU and GPU processing units to run the YOLOv8 framework. In addition to the sensors, the lighting conditions are measured using the **URCERI digital light meter**. This meter features a range of 0.1 to 200k Lux, with an accuracy of $\pm 4\%$. The light meter will be used to measure the lighting intensity near both the sensor (camera and LiDAR) and the objects (pedestrian and warthog).

5.3.2 System robustness validation

Railway maintenance vehicles operate across various lighting environments, such as tunnels, stations and light sources from other vehicles and equipment. Inconsistent lighting could impact the performance of object tracking and distance measurement, posing potential risks to the vehicle's navigation and safety. This experiment focuses on evaluating the system's object tracking and distance estimation capability under different lighting conditions.

Light sources analysis

This section discusses three different lighting scenarios that typically occur in railway maintenance scenarios and could affect the camera and LiDAR: environmental lighting, on-vehicle lighting, and dynamic lighting. For example, in low-light environments, the camera may struggle to capture enough detail for reliable object detection, reducing the overall tracking performance. On the other hand, bright lighting conditions may cause overexposure, making it difficult for the camera to distinguish objects. While LiDAR is generally less affected by lighting conditions due to its active sensing, it may still experience noise in extremely bright environments, such as direct sunlight or strong reflections.

Environmental lighting refers to the ambient light present in the environment, which can vary significantly during different times of the day. Additionally, artificial lighting, such as track-side lights or station illumination, can also add complexity.

On-vehicle lighting refers to the lighting systems mounted directly on the RIMV, such as headlights. They illuminate the surrounding area in a low-light environment, such as tunnels or night-time.

Dynamic lighting refers to light from passing vehicles, portable work lights, or light from other equipment and machinery. Unlike static environmental lights, dynamic lighting sources can introduce sudden changes in illumination, adding complexity to detection and tracking.

Lighting impact locations analysis

Cameras operate by capturing ambient light reflected from objects, relying on illumination to determine visual features. Consequently, the light near the object affects the camera's performance. On the other hand, When light directly illuminates the camera, it can lead to glare and overexposure to strong light. Therefore, it is necessary to investigate the brightness both near the object and near the camera.

In contrast, LiDAR operate by emitting its light and could be affected by strong light sources near the sensor, which may cause sensor saturation. While lighting near the object is generally a secondary concern for LiDAR. Table 5.1 summarises the primary concerns related to camera and LiDAR.

Measurement Location	Camera Impact	LiDAR Impact
Near the object	Determines how well the object is illuminated, affecting visibility, contrast, and detection accuracy.	Measure high reflectivity or under a strong local light source.
Near the sensor	Identifies potential issues with glare, lens flare, or sensor saturation.	Evaluate potential light interference affecting the LiDAR.

Table 5.1: Camera and LiDAR impact at different measurement locations

Test input variables

Table 5.2 summarises the controlled input variables and the method of variation for each lighting source.

Variable	Light Intensity	Light Directions
Overhead light	12	1
On-vehicle light	9	2
Dynamic light	13	6

Table 5.2: Controlled variables in robustness experiments

The overhead light is controlled by adjusting the number of fixed-position overhead lights from 1 to 12 to simulate varying environmental lighting conditions, from low-light tunnels to bright daytime.

On-vehicle light simulated nine different light intensities to simulate the high-beam headlights of the RIMV in a low-light environment. As shown in Figure 5.12, the light direction can either be directly pointed to the object or angled away from it. When the on-

vehicle light aims directly at the object, it simulates the ideal lighting condition where the target is fully illuminated. On the other hand, the partial or indirect illumination simulates more realistic scenarios when the RIMV is moving.

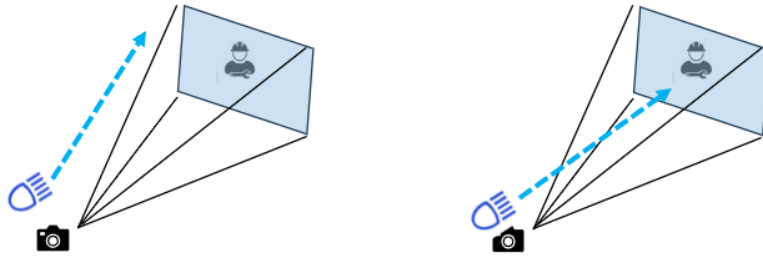


Figure 5.12: On-vehicle light setup

Dynamic light simulated 13 different light intensities to simulate the external light in a low-light environment. Figure 5.13 and 5.15 demonstrate different scenarios of how external dynamic light can affect the camera; the light source can either point directly at the camera or away to avoid direct illumination. Additionally, three scenarios were designed to evaluate different situations:

- The light source is out of the camera's FoV.
- The light source is within the camera's FoV and near the object.
- The light source is within the camera's FoV but occluding the object.

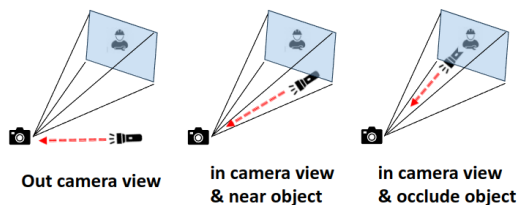


Figure 5.13: Light point to camera

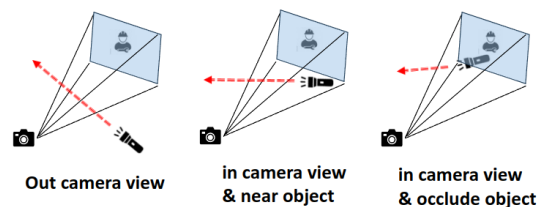


Figure 5.14: Light not point to camera

Figure 5.15: External light setup scenarios

These scenarios are essential for railway maintenance, where dynamic lighting from worksite floodlights, passing vehicles, and portable lights can impact the RIMV's sensors. The system must handle situations where lights either avoid or enter the camera's FoV, potentially causing glare, overexposure, or object occlusion.

Evaluation metrics

This research extends the 2D object tracking algorithm by integrating LiDAR, which allows distance measurement of tracked objects. The implemented tracking algorithms follow a Track-by-detection approach, where the overall tracking quality is inherently influenced by detection accuracy. Instead of focusing on tracking performance metrics, such as tracking continuity or ID switches, this experiment aims to evaluate the essential aspects that affect a collision avoidance system: object detection, distance measurement accuracy, and real-time performance. Table 5.3 shows the measurement methods and expected results for each metric.

- **Detection confidence rate** evaluates how confident in identifying objects, ranging from 0 to 1; this is assigned to each tracked object from the YOLOv8 framework. A higher result indicates better detection performance.
- **Distance accuracy** evaluates how accurate the system's distance measurements are. The distance estimated by the pointcloud clustering algorithm with LiDAR input is compared with the ground truth measured by a laser distance meter (**distance error**). Additionally, the evaluation includes the accuracy for objects at different ranges (**accuracy at various ranges**)

Metric	How to Measure	Expected Results
Confidence rate	Measure with the YOLOv8 framework	Higher confidence rate with better lightning
Distance accuracy	Comparison estimation against ground truth measured with a digital distance meter.	Consistent, accurate measurement despite different lighting conditions.

Table 5.3: Metric details, measurement methods, and expected outputs

5.3.3 Collision avoidance system validation

The experiment to validate the context-aware collision avoidance system was conducted at the RITA, an 18-metre railway track at Cranfield University, to replicate real-world railway conditions. Compared to simulation or indoor testing, outdoor validation introduces additional uncertainties such as environmental noise, lightning, vibration, etc., which are essential to ensure the system’s robustness and deployment.

As illustrated in Figure 5.16, the red square marks the restricted area around the vehicle with a high risk of collision. Any object detected within the area triggers the emergency stop and alarm to warn the surrounding operator. If the object is detected outside this area, a different actuation command will be activated based on the object type. Table 5.4 summarises the expected actuation commands for each scenario:

- **No detection:** the vehicle navigates at normal speed
- **Trespasser detection:** stop the vehicle immediately regardless of its position and activate an audible alarm
- **Worker detection:** the vehicle navigates at a lower speed, activates the light indicator
- **Worker near track:** stop the vehicle immediately and activate an audible alarm

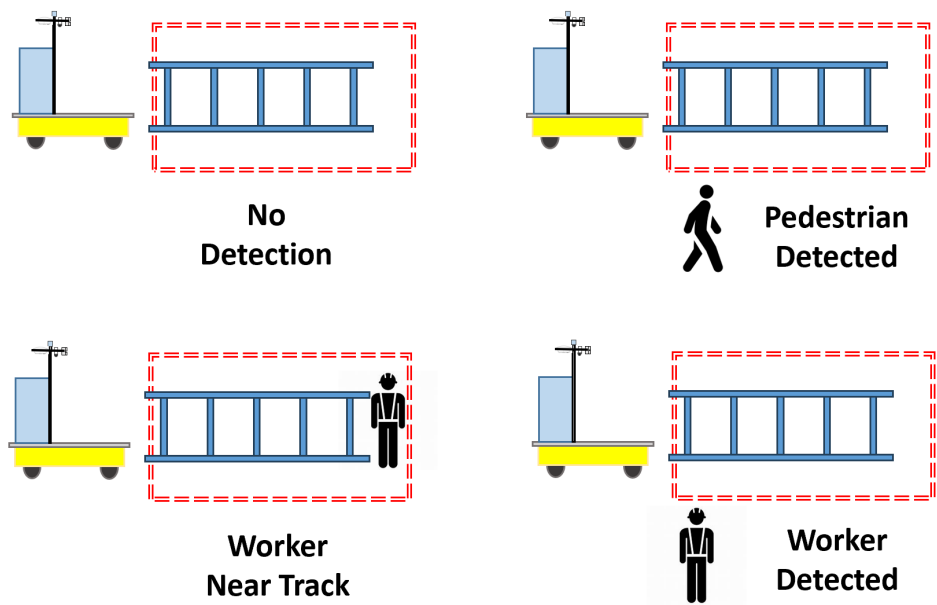


Figure 5.16: Collision avoidance test scenarios

Scenarios	Expected Actuation Command
No detection	Navigate at speed 2.
Trespassers	Emergency stop, activate light and alarm.
Worker	Navigate at speed 1, activate light.
Worker near track	Emergency stop, activate light and alarm.

Table 5.4: Expected actuation for each test scenario

5.4 Results and discussion

5.4.1 Introduction

This section presents the findings from the validation experiments designed to evaluate the system’s performance. The results are structured around two key validation experiments: **indoor robustness validation**, and the **outdoor collision avoidance** tests. The metrics used to assess the system include detection confidence, distance measurement accuracy, real-time performance, and the effectiveness of collision avoidance. The results focus on

evaluating the system's ability to detect, track, and respond to obstacles under various conditions.

5.4.2 Discussion on robustness validation results

This section discusses the camera and LiDAR's performance under eight lighting scenarios as shown in Table 5.5, which combine different sources of light, including overhead lights (simulating natural ambient daylight), on-vehicle lights, and dynamic external lights (simulating passing vehicle lights) under both daylight and night-time settings.

No.	On-Vehicle Light	External Light	Ambient Light
1	Yes	Yes	Daylight
2	No	No	Daylight
3	Yes	No	Daylight
4	No	Yes	Daylight
5	Yes	Yes	Night-time
6	No	No	Night-time
7	Yes	No	Night-time
8	No	Yes	Night-time

Table 5.5: Lighting conditions scenarios tested

Lighting impact on LiDAR

The LiDAR distance measurement tests were tested independently without object detection to focus on evaluating LiDAR's performance under various lighting conditions. Across these scenarios, as summarised in Table 5.5, the brightness for LiDAR ranges from 0 lux in total darkness to a maximum of 1752 lux. For the brightness of the warthog, lux readings ranged from 0 lux to approximately 498.8 lux. The distance measurement remains consistent at 5.33, with a 98.15% accuracy compared to 5.43 ground truth. The results indicate that the LiDAR sensor maintains reliable performance regardless of the light near the sensor or the object.

Environmental light impact on camera

Figure 5.17 shows the impact of environmental lighting on the detection; the left part measures the brightness near the camera, and the right part measures the brightness near the warthog. Both plots show that the brightness has a direct impact on the detection performance. The detection confidence rises from 55% to 90% as lighting improves, stabilising after 200 lux. The results suggest that adequate lighting is critical for reliable object detection, but beyond that, additional light provides no further benefits.

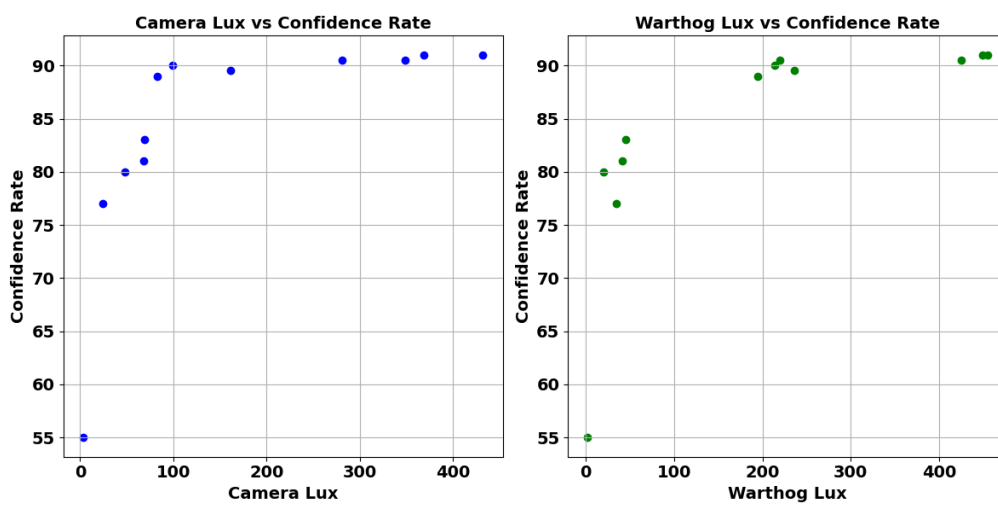


Figure 5.17: Environmental light impact on camera

On-vehicle light impact on camera

The setup is similar to the environmental lighting scenario; the on-vehicle light increases gradually to illuminate the target object while keeping the camera brightness at 0. This setup isolates the effect of lighting around the camera.

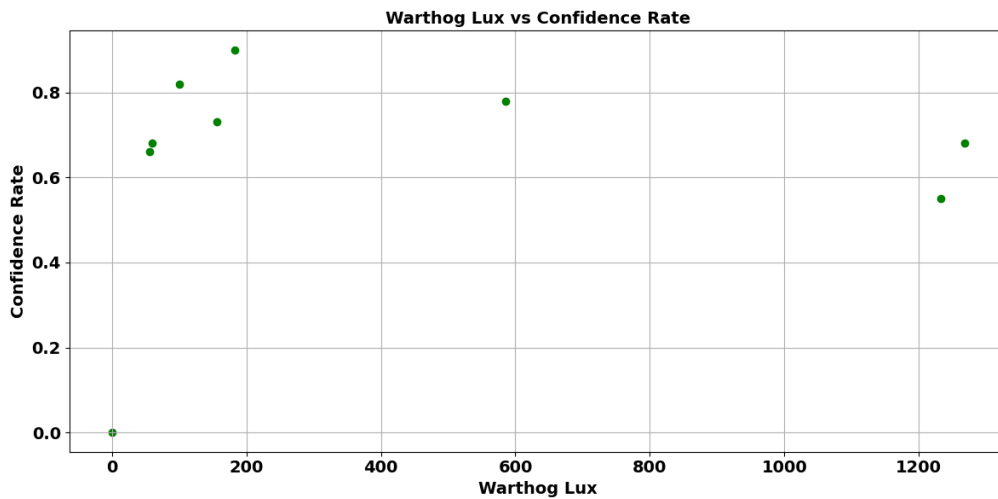


Figure 5.18: On-vehicle light impact on camera

As illustrated in 5.18, the results show a similar trend in environmental lighting scenarios, where the confidence rate improves as the brightness near the warthog increases, reaching the best performance of 90% around 200 Lux. However, the performance declines when the brightness near the object increases and only reaches 55% confidence with 1268 lux, which is due to the overexposure caused by strong beam as shown in 5.19. Another interesting observation is that shadows can reduce detection accuracy despite brightness, which suggests that the direction of on-vehicle lighting plays a crucial role.



Figure 5.19: The confidence rate decreases with higher brightness at 1286 Lux

Dynamic light impact on camera

This experiment investigates the impact of dynamic lighting on the camera's detection performance under specific environmental lighting conditions. In real-world applications, dynamic lights (such as vehicle headlights or portable work lights) are generally not used during high-brightness daytime conditions. Therefore, this study excludes scenarios where environmental light exceeds 1000 Lux [269]. Similarly, as discussed in the previous section, the detection capability is compromised when the environment light is under 200 Lux. The dynamic lightning is gradually increased to evaluate the impact on the camera's detection.

When the light source is out of the camera's FoV, two scenarios were explored: where the dynamic light beam is directed away from the camera and directed towards the camera.

- When the light beam is directed away from the camera, the detection confidence rate remains constant at around 91%, which is due to the light beam not being directed to

the camera or the target object; the brightness does not experience changes despite the significant change in the dynamic light source. The camera lux remains between 342.3 and 418.3 lux, and the lux near the warthog varies only slightly from 388.9 to 427.8 lux.

- When the light beam is directed towards the camera, the brightness near the target object remains relatively constant, ranging from 407 to 439 Lux. Figure 5.20 shows the detection confidence remains steady at around 90% despite the brightness near the camera increases; however, when the brightness near the camera reaches 13.9k Lux, the detection rate drops significantly to 50% due to the lens flare caused by strong beam.

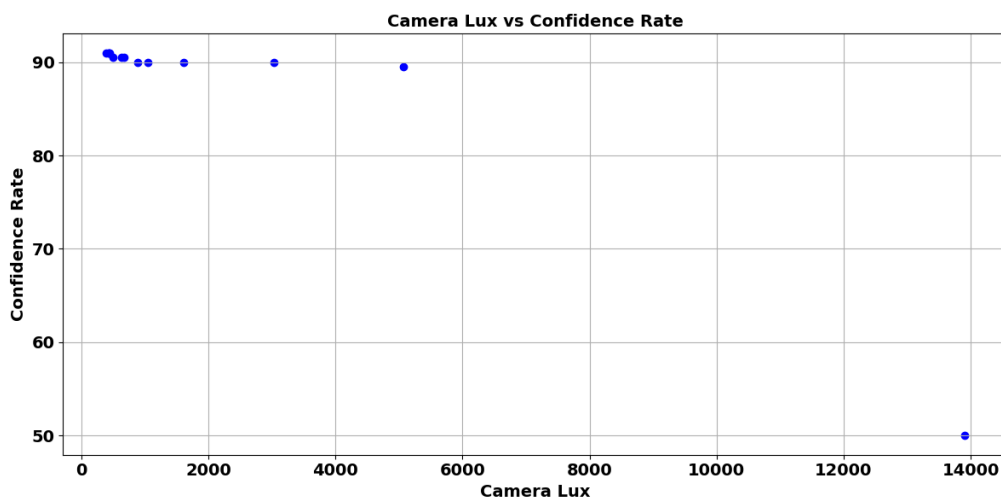


Figure 5.20: When dynamic light out camera's FoV, and point towards the camera

When the light source within of the camera's FoV, same scenarios were explored: where the dynamic light beam is directed away the camera and directed towards from the camera.

- When the light beam is directed away from the camera, the dynamic light does not illuminate the camera or the target object, and the detection confidence rate remains constant at around 84-86%, which suggests the dynamic light does not significantly change the brightness near the object nor the camera. It has a small impact on the

detection rate. However, the detection rate is lower than the baseline result of 91% without dynamic light. This reduction is likely due to subtle glare effects or indirect illumination changes impacting the camera's view.

- When the light beam is directed towards the camera, Figure 5.21 shows the results follow a similar trend compared to when the light was outside the camera's FoV: the confidence rate remains constant around 90-91% when the brightness near the camera is low. However, when the brightness near the camera reaches 2000 Lux, detection performance degrades, which is earlier than 14k Lux when the camera is out of the camera's FoV. This trend suggests that glare effects are more noticeable (Figure 5.22) when the dynamic light source is within the camera's FoV. The impact on the camera is more severe compared to when the camera is out of the camera's FoV.

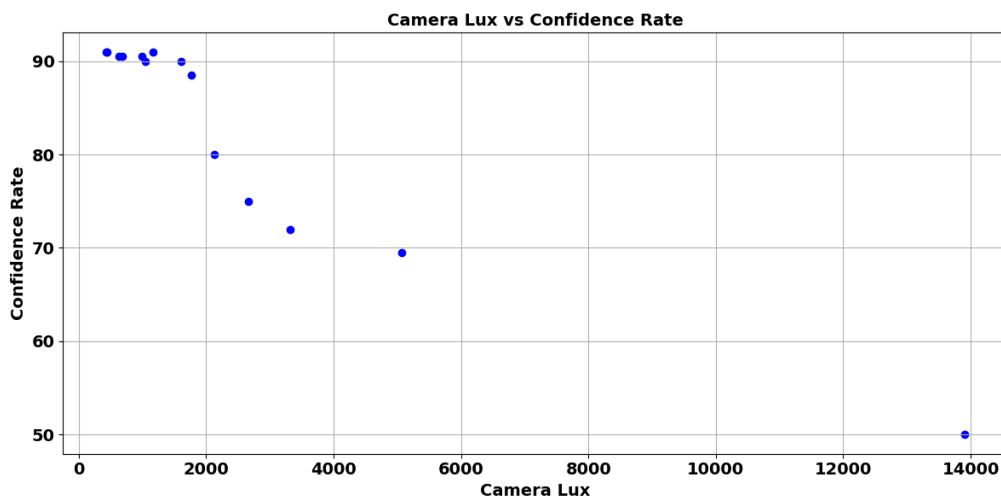


Figure 5.21: When dynamic light within camera's FoV, and point towards the camera

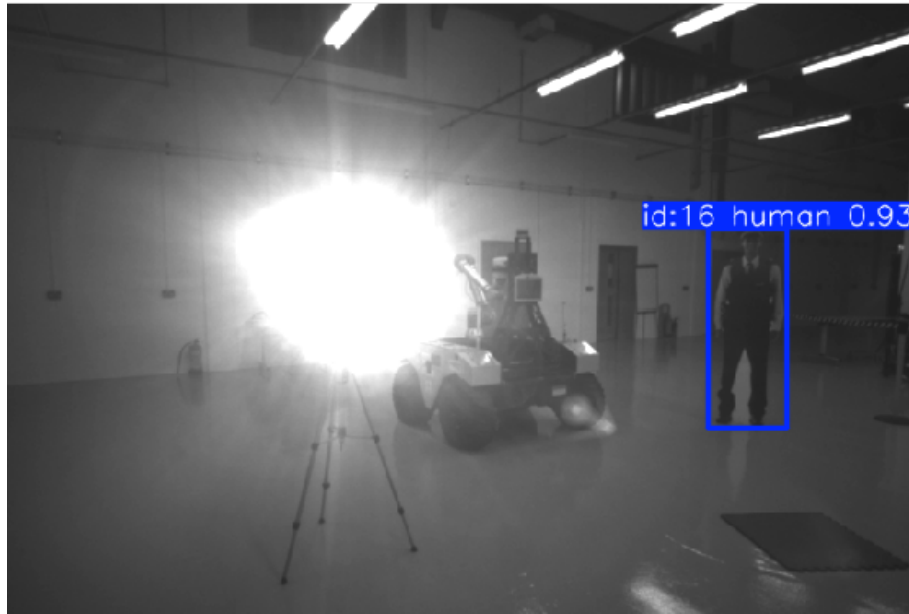


Figure 5.22: Example of how glare impacts camera's detection

5.4.3 Discussion on collision avoidance validation results

The collision avoidance system was validated on a real-world test track to evaluate its ability to distinguish between trespassers and workers, track their 3D positions in real-time, and execute context-aware responses based on the type and position of detected objects. As shown in Figure 5.23, The worker and trespasser are differentiated by the hi-vis jackets. The environmental illumination during the test was approximately 1500 lux, which exceeds the 200 lux threshold for ideal illumination conditions identified in Section 5.4.2.

The validation test was repeated 10 times, with each iteration evaluating the system's performance under standard conditions. As the 3D tracking system extends from a 2D track-by-detection approach, its performance heavily relies on object detection. To ensure a focused assessment of the system's core functionality, complex scenarios that may adversely affect detection, such as occlusion and crowded scenes, were intentionally excluded.



Figure 5.23: Collision avoidance validation experiment setup

Validation results

The vehicle achieved a success rate of 80%, with 8 out of 10 tests successfully identifying the object type and responding with appropriate actuation commands. For workers detected outside the restricted zone, the vehicle continues operation by slowing down. However, when a worker was detected within the restricted zone or a trespasser was identified—regardless of their position—the system immediately activated an emergency stop and triggered an audible alarm to ensure safety, demonstrating its ability to adapt to dynamic scenarios in a railway environment while prioritising safety and efficiency. Figure 5.24 illustrates the vehicle’s field of view alongside the operational interface of the collision avoidance system. The right panel showcases the vehicle’s visual and sensor feedback displayed in RViz, while the left console provides a real-time display of detection status and actuation commands.

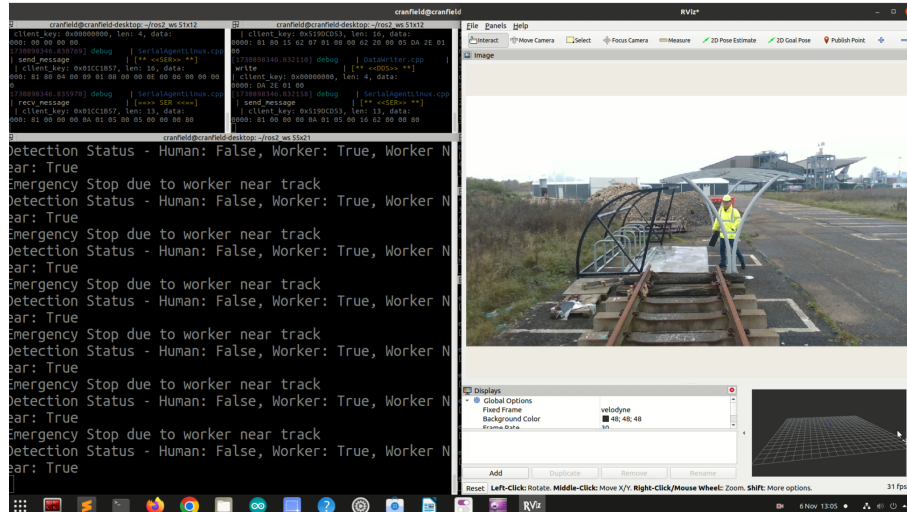


Figure 5.24: Vehicle's perspective and control console during testing

Analysis on system limitation

In the two failed cases, the system mis-detected the worker as a pedestrian, leading to unnecessary emergency stops, which highlights the need for improvement in object detection performance. Furthermore, within the successful test, two tests experienced a noticeable delay of 3-4 seconds for the scenario when a worker is near the track, compared to the average latency of approximately 1 second. As the vehicle moves, the system faces multiple tracking challenges due to motion, relative motion of obstacles, and vibrations from the track.

When the vehicle is in motion, the camera's relative speed to the environment prevents the system from capturing clear, stabilised images, leading to motion blur [187, 270, 163, 7, 193, 159]. Visual tracking systems that rely on inter-frame correspondences (e.g., feature tracking or optical flow) are especially vulnerable, as image blur disrupts the alignment of visual landmarks used for continuous tracking [163, 7].

Relative motion between the vehicle and dynamic obstacles poses additional complexity. When objects in the scene have relative motion with respect to the vehicle, the system must resolve the correspondence of objects across multiple frames, increasing the computational load required to maintain continuous tracking [63, 271, 61]. For tracking-by-detection systems, the computational demands are especially high, as new objects must be

detected in every frame before being associated with prior tracks [9, 7]. This relative motion introduces frame-to-frame displacements that complicate matching, often requiring predictive models or data association algorithms to resolve the correct object correspondence [159, 200].

Lastly, vibrations from the track worsen motion blur and affect tracking stability. Continuous movement along railway tracks introduces sustained vibrational noise, causing image distortions that reduce the system's ability to extract clear object features [159]. Furthermore, vibrations affect frame stability, making it difficult to maintain alignment across consecutive frames [187, 270].

The combination of these factors (vehicle motion, relative object motion, and vibrations) introduces interrelated challenges. This complexity leads to system latency, where delays in processing can cascade into broader system delays, impacting real-time responsiveness [200, 63]. Systems designed for real-time operation face difficulty balancing computational speed with tracking accuracy [9, 159].

5.4.4 Future works

Although RIMV demonstrates reliable performance, several areas could be improved. First, due to the constraint of the stereo camera's hardware, the stereo camera can only estimate object distance up to 10 meters, which restricts the system's ability to detect objects at greater range. Future improvements include upgrading the hardware with an extended range or incorporating alternative software-based depth-sensing approaches.

Second, the current distance measurement approach focuses on estimating the closest object within the 3D frustum. This approach is effective for isolated objects but faces challenges in crowded environments where multiple objects may fall within the same frustum. This could be improved with advanced clustering techniques that could allow for differentiation between multiple objects.

Third, the vehicle faced occasional mis-detection. Training the model with a larger and more diverse dataset could enhance its ability to generalise to different object types

and environments. Alternatively, exploring other advanced TL techniques or leveraging one-shot learning methods could enable the system to learn effectively from limited railway-specific data.

Finally, the tracking system faced occasional latency caused by the relative motion and vibrations from the track. The proposed system relies on a camera for object detection and LiDAR for distance estimation. Future work could focus on leveraging LiDAR's input for object detection, providing an alternative detection approach when the camera's performance is restricted, particularly in low-light or night-time conditions. Additionally, integrating LiDAR-based detection would address challenges associated with motion blur affecting the camera, ensuring consistent and robust distance estimation for 3D object tracking.

5.5 Chapter conclusion

This chapter presented a multimodal 3D object tracking and collision avoidance system for autonomous railway vehicles. By fusing 2D image data with 3D LiDAR pointcloud, the system extends 2D object tracking into 3D and enables precise tracking for detected objects.

The key contributions of this chapter include the calibration between the camera and LiDAR, the implementation of a frustum-based multimodal 3D tracking framework, and the design and validation of a context-aware collision avoidance system. These innovations collectively enhance real-time responsiveness and the adaptability of the RIMV to the dynamic and unstructured railway maintenance environment.

The robustness of the 3D tracking system was demonstrated through comprehensive testing under various lighting conditions. While LiDAR consistently maintained 98% distance measurement accuracy across all tested lighting conditions. The camera's detection performance was found to be sensitive to changes in brightness near the target object, in the presence of glare from dynamic light sources, or due to overexposure when solely

relying on the vehicle's headlight in low-light environments.

The collision avoidance system was validated on a real-world railway test track, achieving an 80% success rate. However, system latency increased to 3-4 seconds in high-motion scenarios due to track-induced vibrations and relative motion between the RIMV and dynamic objects, complicating the tracking of fast-moving objects in real-time.

Two limitations were identified. First, the mis-detection of workers as trespassers occurred in 20% cases, which triggered unnecessary emergency stops and alarms. Second, relative motion and track-induced vibrations led to processing delays, with latency increasing to 3-4 seconds under certain scenarios. To improve detection performance, future work will focus on model generalisation for worker-trespasser differentiation through the use of TL on diverse datasets. To address the long latency issue, future work will explore the possibility of using the pointcloud data from LiDAR to track. Furthermore, LiDAR-based object detection can reduce reliance on the camera, particularly under low-light or high-glare conditions, therefore improving the robustness of the collision avoidance system.

Chapter 6

Conclusion

6.1 Research overview

Autonomous technology has seen rapid development over recent years, contributing to improvement in safety and productivity across various industries. However, the railway sector has yet to receive comparable attention. For example, self-driving cars can achieve a high level of autonomy by reacting to complex situations [34, 272], autonomous railway vehicles still rely on a simple “start-stop” approach without considering the dynamics of surrounding obstacles[17, 29]. Acknowledging this gap, this project proposed a novel approach to design and implement a context-aware collision avoidance system to enhance both safety and efficiency in railway maintenance activities. The goal is to allow the RIMV (see Figure 1.2 in Section 1.3.2) to classify and anticipate the movement of potential obstacles, therefore enabling proactive decision-making based on the type and movement of detected objects.

The collision avoidance system for the RIMV integrates two primary components: object detection and 3D object tracking. Object detection focuses on classifying six object classes that frequently appear in railway maintenance environments, such as workers, pedestrians, signals, etc. Meanwhile, 3D object tracking allows RIMV to anticipate the obstacle’s future position by analysing its historical movement. A sensor-fusion approach

is implemented to enhance the performance of both tasks. The camera is ideal for object detection tasks due to the rich visual information it captures. On the other hand, LiDAR is more accurate and robust in distance measurement. The sensor-fusion approach benefits from both visual recognition and spatial accuracy by leveraging the advantages of each sensor.

This system provides real-time situational awareness, allowing the vehicle to respond proactively to detected obstacles. For instance, recognising trespassers triggers RIMV’s emergency stop to ensure safety. When a worker is detected, the vehicle can adjust its speed based on the worker’s movement. This increases efficiency by avoiding unnecessary stops and leads to better task scheduling or predictive maintenance, contributing to overall railway operation optimisation.

6.1.1 Research gaps, objectives and novelties

The development of context-aware collision avoidance for autonomous railway vehicles faces several critical research gaps (**Objective 1**, identified in **Chapter 2**). This study addresses these gaps through novel methodologies and system-level innovations, each aligned with specific targeted objectives. Table 6.1 summarises how each identified gap corresponds to the research objectives and the associated novelty introduced by this research.

Gap	Objective	Novelty
Gap A	2	Created dataset for object detection in railway main-tenance
Gap B	3	Investigated various TL strategies
Gap C	4	Multi-sensor fusion for 3D tracking and system inte-gration
Gap D	5,6	Development and validation of Context-aware colli-sion avoidance system

Table 6.1: Mapping research gaps to objectives and novelties.

Gap A: limited dataset for object detection in railway maintenance environment

To address this gap, a custom dataset was constructed by collecting images from the Northampton & Lamport heritage railway (**Objective 2**, addressed in **Section 4.2.2**), focusing on six object classes that frequently appear in railway maintenance scenarios and not covered by existing datasets. The study demonstrates its practicability for fine-tuning object detection models, bridging the gap of lacking domain-specific datasets for railway maintenance tasks.

Gap B: absence of transfer learning optimised for railway object detection

To address this gap, this research designed an experiment to investigate various factors that impact object detection with TL (**Objective 3**, addressed in **Chapter 4**). The experiment evaluated the effectiveness of one-stage and two-stage pre-training approaches. Secondly, It compared the performance of TL from a large general and smaller domain-relevant dataset. Thirdly, the role object class alignment between the source and target dataset was explored. Lastly, the experiment evaluated the fine-tuning strategy by freezing different layers in the YOLOv8 object detection architecture.

Gap C: absence of multi-sensor fusion and unified robotic framework for real-time 3D object tracking

A novel multimodal sensor fusion framework was developed (**Objective 4**, addressed in **Chapter 5**), integrating stereo cameras for object detection and LiDAR for distance estimation, enabling the transition from 2D to 3D object tracking. This system operates within a ROS2-based robotic framework, providing real-time processing, modularity, and scalability.

Gap D: need for a context-aware collision avoidance system with system validation

The research developed a context-aware collision avoidance system for autonomous railway vehicles; it classifies objects (e.g., workers vs trespassers) and dynamically adjusts

the vehicle's response based on object type and position (**Objective 5,6**, addressed in **Chapter 5**). System validation was conducted in both indoor and outdoor environments. Indoor validation simulated diverse lighting conditions, including low light and glare, to replicate real-world railway maintenance scenarios. Outdoor validation was performed at a real-world controlled test track to ensure high TRL

6.2 Summary of contributions

6.2.1 Railway object detection with transfer learning

One key challenge to detecting railway objects is the scarcity of large, annotated datasets to train machine learning models. Object detection in the railway domain, particularly for maintenance scenarios, requires the identification of railway infrastructure components and other machinery. Most existing large object detection datasets focus on general scenes and do not represent the unique characteristics of the railway maintenance environment. It is crucial to address this gap and utilise the strength of machine learning to detect objects in a railway.

Datasets and experiment design

The TL experiments used three datasets: COCO, RailSem19, and a custom railway dataset. The COCO is a large dataset with 80 classes and 330k images; it focuses on general object detection rather than railway objects. In contrast, RailSem19 focuses exclusively on 19 different railway object classes but is relatively smaller, with 8.5k images. Lastly, the custom railway dataset developed for this experiment contains six object classes that commonly appear in railway maintenance scenarios. This is used as the target dataset in the fine-tuning step.

The experiments pre-train models both on COCO and RailSem19, then fine-tune the models to reuse the learned features; it assesses the impact of transferable features learned from a large general dataset versus smaller, railway-focused datasets. Addition-

ally, although the RailSem19 dataset covers 19 railway object classes, the custom dataset is smaller, with only six object classes relevant to railway maintenance scenarios, and contains objects not represented by the RailSem19. The experiments pre-train models by gradually reducing object classes in the RailSem19 dataset and comparing the performance difference by fine-tuning with the custom dataset. This experiment seeks to understand how object class number in the source dataset influences performance for TL. Lastly, The experiments investigated the impact of different fine-tuning strategies by freezing various layers in the YOLOv8 architecture to determine which part of the model is most adaptable to the new, specialised dataset and provide the optimised fine-tuning strategy for TL.

Key findings

Larger general dataset or smaller relevant dataset: when comparing the performance of TL with a larger general dataset (COCO) and a relatively smaller relevant dataset (RailSem19), Overall, COCO outperforms RailSem19 by approximately **13.1%** in F1 score and **10.7%** in mAP50. For non-railway objects, COCO shows approximately **11.9%** improvement than RailSem19 in F1 and **2.1%** decrease in mAP50. Railway object classes demonstrate significant improvements with approximately **18.86%** and **32.9%** in F1 and mAP50.

One-stage v.s. two-stage pre-training approach: it is more effective to employ a two-stage pre-training approach to achieve optimal performance. In this approach, the model is first pre-trained on COCO and followed by additional pre-training on RailSem19 to adapt to railway-specific characteristics. This approach led to an improvement of **8.0%** in F1 and **11.2%** in mAP50 compared to only pre-training with RailSem19. This results indicate that the multi-stage pre-training enables the model to benefit from both datasets, with the later stage having a more substantial impact. By pre-training first on COCO and then on RailSem19, the model gains broad feature knowledge and then fine-tunes

to the unique characteristics of railway-specific objects, resulting in notable performance improvements.

Impact of number of object classes in source dataset: reducing the number of object classes in RailSem19 improves railway object detection by reducing noise and enables the mode to focus on domain-relevant features. However, the performance of non-railway object detection declines due to the loss of dataset diversity. For example, in the best-performing setup (fine-tuned two-stage pre-training), the pedestrian class (non-railway) decreased by approximately **31.7%** in F1, while the pedestrian crossing class (railway) increased by around **8.5%**.

Impact of fine-tuning techniques: the results show that freezing early convolutional layers in YOLOv8 improved performance by up to **6-12%** in F1 and mAP50 scores, especially when pre-training with a larger source dataset like COCO. Convolutional layers capture features from images: early layers detect low-level features such as shapes and textures, while later layers focus on high-level task-specific details. Freezing early layers retains these foundational features learned from a large dataset. However, freezing too many layers restricts adaptability, leading to performance declines. Moderately freezing early layers (e.g., f0135) is most effective.

Final remarks

The results demonstrate the benefits of TL for railway object detection. By pre-training with a large dataset (COCO) and refining with domain-specific dataset RailSem19, the model leverages the learned knowledge by further fine-tuning with a custom dataset specifically tailored to railway maintenance settings. This model improved railway object detection with a small annotated dataset, enabled the RIMV to identify railway objects such as signals, workers, etc., and prevented accidents for autonomous railway vehicles by ensuring a context-aware collision avoidance system.

Broadly, this experiment contributes to the field of machine learning by highlighting

the role of TL for object detection. The experiments investigated the role of the source dataset for TL from different perspectives and addressed key questions:

- **Quantity vs. quality in source datasets:** it explores whether a large general dataset or a smaller, domain-specific dataset serves better for TL.
- **Impact of object class diversity:** the experiment investigates how the number of object classes in the source dataset affects model adaptability, both general and domain-specific objects.
- **Effective TL techniques:** the experiment assessed the effectiveness of strategies like multi-stage pre-training and selective layer freezing.

In conclusion, this experiment provides a practical framework for applying TL to object detection in specialised domains with limited annotated data. This approach ensures robust low-level feature retention from large datasets while adapting high-level features to the specialised railway domain. Such methodological improvements are not only beneficial for this scenario but also extend to other domains (e.g., medical imaging or agricultural robotics) with limited annotated datasets.

6.2.2 Multi-modal 3D object tracking

A robust collision avoidance system requires the ability to detect various objects and estimate their position. The camera is ideal for object detection due to the rich visual information it captures. However, it struggles with distance estimation, especially under challenging lighting conditions. On the other hand, LiDAR provides accurate distance measurement regardless of lighting but is less precise in object detection. This project proposed a sensor-fusion approach to combine both sensors, enabling a real-time 3D object tracking ability and ensuring robustness to varying operational environments.

The system builds on the YOLOv8 framework with ByteTrack, extending 2D object tracking into 3D. First, a calibration process aligns the stereo camera and LiDAR's

coordinate systems, establishing spatial coherence between the two sensors. This step integrates the depth information from the stereo camera with the 2D bounding boxes in the camera's coordinate system, resulting in a 3D bounding box in the LiDAR's coordinate system. Subsequently, a 3D frustum is constructed around each detected object, with the size dynamically adjusted based on the object type. This frustum serves as a spatial filter and allows LiDAR to focus on measuring the distance to objects within this area. This multi-modal approach significantly reduces computational requirements and improves measurement accuracy compared to the stereo camera alone.

Impact of lighting

Railway maintenance tasks often occur across different environments with various lighting conditions, from bright daytime to minimal lighting at night-time or within a tunnel. Such variability presents a challenge for both the camera and LiDAR. To systematically evaluate the impact of illumination on the 3D tracking system, three lighting scenarios were tested to simulate real-world lighting conditions: environmental lighting, on-vehicle lighting, and dynamic lighting.

- **Environmental lighting** simulates naturally varying daylight conditions by adjusting overhead lighting levels within an indoor setting. By gradually increasing light intensity, the experiment identifies the minimum required illuminance for reliable system performance, establishing a benchmark for open-track railway maintenance.
- **On-vehicle lighting** represents the headlights of the RIMV, simulating operations in low-light environments such as tunnels or night-time settings. This scenario evaluates whether on-vehicle lighting can provide comparable detection and tracking performance to natural daylight. The findings offer insights into how to optimise lighting configurations on the RIMV to ensure maximum system performance in low-visibility environments.
- **Dynamic lighting** replicates sudden, unpredictable illumination changes, such as

the effects of passing vehicle headlights or portable work lights that are frequently encountered in active railway maintenance. This step evaluates the system's resilience to "intrusive light". The findings offer insights into how to optimise external lightning configuration around the RIMV to ensure maximum system performance in low-visibility environments.

Key findings

The experiment evaluated the system's performance in two aspects: the LiDAR's distance estimation accuracy and the camera's detection confidence rate. Results showed that the LiDAR sensor maintained consistent distance estimation accuracy regardless of lighting conditions, achieving 98.15% accuracy relative to ground truth measurements, which confirmed LiDAR is robust and well-suited for railway maintenance environment.

However, the camera's performance varies with lighting. Under environmental lighting scenarios, the camera failed to detect any object. The detection performance improved as illumination increased, reached the peak of 91% confidence with 200 Lux near the object, and stabilised regardless of further light increase. The results followed a similar trend under the on-vehicle lighting scenario and reached a 90% confidence rate with 200 Lux near the object. However, the performance declines when the brightness near the object increases and only reaches 55% confidence with 1268 lux, which is due to the overexposure caused by strong beam from the on-vehicle light. Lastly, the impact of dynamic lighting was assessed in two cases. When the light source was outside the camera's FoV, detection confidence remained stable at around 90%, and the performance dropped to 50% confidence under extreme scenarios when the camera experienced lens flare while 13.9k Lux near the camera. This lens flare effect became more severe when the dynamic light source was within the camera's FoV, where glare effects started to appear at around 2000 Lux near the camera—much earlier than in scenarios with the light source outside the FoV.

Final remarks

The 3D object tracking system extends 2D tracking into 3D space by combining the strength from the stereo camera and LiDAR, improving the trade-off between object detection and distance estimation requirements. The robustness validation experiment demonstrated the robustness of LiDAR under various lighting conditions. Furthermore, it studied the impact of illumination on the camera, identified the ideal operational range and provided clear guidelines for the minimum level required for RIMV to operate in a low-light environment. The results also highlighted the challenges of overexposure and lens flare effect and identified the area for future improvement.

In addition to illumination-based limitations, several operational constraints were identified during validation. The RealSense stereo camera has a limited effective range for depth estimation: accuracy deteriorates beyond approximately 10 metres due to disparity limitations. It also becomes unreliable at very close distances (below 0.5 metres), where insufficient disparity prevents accurate triangulation. These limitations affect the reliability of the initial depth estimates used for frustum construction. When objects are located beyond the maximum effective range, the resulting frustum may fail to encapsulate the true position of the object in 3D space, preventing the LiDAR from capturing relevant pointcloud data for distance measurement. However, the near-range limitation does not affect system functionality, as the vehicle is not expected to operate with obstacles within such proximity. In safety-critical cases where an object does enter this zone, the LiDAR-based proximity module acts independently to trigger an emergency stop. To overcome the stereo camera's range limitations, future work could explore the use of higher-specification stereo or depth sensors, or adopt software-based depth estimation alternatives, such as deep learning models trained on monocular or stereo image inputs.

During outdoor testing, vibration from track irregularities impacted system performance, leading to increased latency and degraded tracking accuracy. Although not quantitatively measured, these findings suggest that mechanical vibration adversely affects both

perception delay and detection stability. Future work could explore the use of dampers, camera stabilisers, or gimbal-mounted rigs to mitigate vibration and improve system robustness under real-world operating conditions.

Lastly, the system faced challenges in crowded scenes where the frustums of multiple objects overlapped, leading to less accurate distance measurements and object association errors. This highlights limitations in handling complex, cluttered environments. Future work could explore more advanced clustering algorithms capable of distinguishing multiple objects within a single frustum or consider replacing the vision-based detection approach with pointcloud-based methods for direct 3D object localisation.

6.2.3 Collision avoidance system and real-world validation

Finally, the collision avoidance system relies on inputs from the object detection and 3D tracking modules to make context-aware actuation decisions based on detected objects and their position. Instead of testing the system's functionality isolated within a controlled indoor setting, a real-world robotic framework was developed and tested in a real-world railway environment. The robotic framework is built on ROS2, an open-source robotic middleware that supports real-time communication, modular design, and cross-platform compatibility. Apart from the camera and LiDAR used for object detection and tracking, the system relies on GPS, encoder and IMU for localisation. The output includes motor control for vehicle movement, alarm and LED indicator for alert and warning, and both digital and analogue channels for other peripheral devices used for railway inspection and maintenance tasks. Lastly, the outdoor validation was conducted at RITA, an 18-metre railway track at Cranfield University (see Figure 5.9 in Section 5.3), to ensure a high technology readiness level under realistic railway conditions.

During the test, RIMV achieved an 80% success rate by distinguishing between workers and trespassers and responding dynamically to their positions. When a trespasser is detected, the vehicle takes proactive measures to activate an emergency stop and alarm to warn the nearby operators. When a worker is detected, the vehicle operates at a slower

speed and activates the LED light to ensure safety and efficiency without unnecessary stops. However, if the worker is detected too close to the track, the vehicle also activates an emergency stop to prioritise safety. The result confirms the collision avoidance capability to execute context-aware responses to the type and position of detected objects.

6.3 Research impact

This research addresses a critical gap in autonomous railway vehicles. It improved the existing collision avoidance system from the basic “start-stop” to a “context-aware” approach by distinguishing different railway objects and estimating their position. By integrating it into a ROS2-based robotic framework and testing in a real-world setting, this work advanced the TRL from conceptual designs to near-operational prototypes.

Additionally, the findings may influence the evolution of industry standards, such as defining optimal lighting conditions and how to mitigate the effect of external light. Apart from the railway industry, the underlying principles— sensor fusion, TL, robotic framework, and real-time system integration—can also be extended to other industries facing complex and unpredictable environments, such as construction, mining, or precision agriculture.

Lastly, RIMV is designed to support various railway inspection and maintenance tasks by integrating additional specialised equipment. The enhanced context awareness and proactive obstacle avoidance can reduce operational risks by minimising human exposure to hazards and improve railway maintenance efficiency by operating in conditions where human presence is limited, such as during night-time or in inaccessible areas. More broadly, this promotes the adoption of rail transport and contributes to more sustainable travel.

6.4 Challenges and limitations

Despite the system's advancements, several challenges remain that require further investigation and optimisation to enhance its effectiveness and reliability in complex railway environments:

- **Object detection:** during the outdoor validation, the system occasionally treats workers as trespassers, which leads to unnecessary stops. Under indoor testing, the system also struggles under extreme conditions, such as operating in dark environments. The system could potentially encounter issues such as overexposure and lens flare. These challenges highlight the need for improvement in object detection.
- **Stereo camera depth estimation:** the 3D tracking relies on a stereo camera for initial depth estimation and further refinement with LiDAR. Due to the hardware constraint, the implemented stereo camera is limited to a 10-metre range, limiting the system's ability to detect and track objects at greater distances.
- **Crowded scene:** in the 3D tracking system, the LiDAR measures distance within a 3D frustum. This approach faces challenges in densely populated scenes, where overlapping clusters can lead to ambiguous measurements and reduce tracking precision.
- **Latency in tracking:** due to relative motion between the vehicle and detected objects and vibrations from the railway tracks, the camera experiences motion blur and additional computational demands, leading to a longer-than-average delay.

To address the limitations in object detection, future efforts should focus on expanding the training dataset to include broader railway-specific scenarios, such as varied lighting conditions (e.g., dawn, tunnels), adverse weather (e.g., fog, rain), and densely crowded scenes. In addition to real-world datasets, leveraging artificial images, such as simulated scenes or AI-generate datasets, can significantly enhance performance and accelerate development without extensive manual annotation. Finally, exploring advanced learning

techniques, such as one-shot learning and other TL approaches, will further improve the models in handling complex scenarios.

The challenges related to object tracking, such as limited depth estimation, crowded scene handling and long latency, are inherently interconnected. It is possible to address these challenges individually. The limited stereo depth estimation can be improved by upgrading the hardware for a longer operation range or employing software-based techniques, which rely on monocular cameras and advanced neural networks to estimate object types and locations. Crowded scene handling and longer latency could be improved with advanced LiDAR clustering algorithms to distinguish and track different objects based on their size, shape and spatial features, providing an alternative solution when the camera is not available.

A unified solution could also be pursued by training a single system to combine visual data and LiDAR pointcloud for simultaneous object detection and tracking. The implemented 3D object tracking system currently relies on each sensor handling separate tasks: the camera is used exclusively for object detection, while LiDAR provides depth measurements for the detected object. The unified approach trains a model using a dataset containing synchronised image data and LiDAR pointcloud. This dataset allows the system to be trained end-to-end with both object detection and tracking tasks combined. A unified approach improves robustness under challenging conditions like poor visibility or crowded scenes. By eliminating separate processes, it also reduces system delays between object detection and tracking, enabling faster and more reliable performance.

6.5 Edge cases and ethical considerations

In addition to the technical constraints mentioned above, the RIMV also faces limitations when addressing edge cases and ethical concerns. The system currently relies on a two-layered safety mechanism: the primary layer uses an object detection algorithm to classify six predefined railway object classes and adjusts vehicle actuation based on object type

and movement, while the secondary fail-safe layer uses LiDAR to continuously monitor the vehicle's direction of travel, triggering an emergency stop if any object is detected within this restricted zone, regardless of type.

However, due to the limited object detection module, the vehicle cannot recognise certain object classes—such as children near the track or wild animals—which, although unrelated to railway maintenance, may still pose significant safety risks if left undetected. Additionally, the system lacks confidence-based reasoning and may fail to respond appropriately to partially occluded objects, particularly in cluttered or low-visibility environments. Although the secondary safety layer provides generic obstacle detection, it lacks the capacity for context-sensitive responses based on object type. This absence of contextual understanding limits the system's ability to initiate proactive measures for high-risk situations, such as a person bending over the track or a child chasing a toy. These constraints highlight the need for higher-level reasoning that goes beyond geometric presence, enabling the system to assess situational risk and respond more intelligently.

Beyond these limitations in perception and actuation, the system also faces fundamental ethical challenges. Even with future enhancements for detecting a broader range of object classes, the system would still lack the capacity to resolve dilemmas involving conflicting safety priorities or unclear human intent. For instance, should the vehicle always stop—compromising mission continuity—whenever it detects an ambiguous shape, such as overgrown vegetation or a stray animal? Or in situations where the system detects a risk, but the human supervisor is distracted or disagrees, who ultimately bears responsibility for the consequences? Such scenarios highlight the need not only for improved perception capabilities but also for an ethical reasoning framework capable of navigating trade-offs between safety, accountability, and operational efficiency.

Addressing these challenges requires not only improved perception and actuation but also a system architecture capable of adapting to new demands. The modular and scalable design of the robotic framework offers a strong foundation for such advancement. For instance, the existing node-based architecture allows for the integration of new de-

tection modules or behaviour analysis modules to support human–machine collaboration. Furthermore, as autonomous systems increasingly operate in life-critical environments, it is essential to align their behaviour with established ethical guidelines—such as the IEEE Ethically Aligned Design framework [273] and the European Commission’s Guidelines for Trustworthy AI [274]—which prioritise the protection of human life over property, operational efficiency, or task completion. Embedding these principles into both technical design and operational protocols will be essential to ensure the safe and responsible deployment of autonomous railway systems.

6.6 Future works

Advancing to higher TRL

To achieve a higher TRL, future work should focus on testing and validation from controlled test tracks to live operational tracks, where the system encounters unpredictable scenarios such as operational train schedules, complex worker interactions, and dynamic weather conditions. Additionally, it is essential to integrate RIMV with the ETCS to ensure interoperability with railway signalling and traffic management systems. Achieving compliance with industry standards will be crucial for system certification and industry acceptance. Addressing these factors will enable the system to progress from a proof-of-concept prototype to a fully operational, deployable solution.

Human-robot and robot-robot collaboration

Future work should investigate collaborative frameworks to enable effective human-robot and robot-robot interactions. For human-robot collaboration, the research could focus on worker behaviour recognition algorithms and wearable devices to anticipate worker actions. For robot-robot collaboration, future work could focus on V2V and multi-agent coordination systems to facilitate the concurrent operation of other autonomous railway vehicles on the same track. These collaborative strategies will improve operational effi-

ciency and system scalability for autonomous railway maintenance tasks.

Enhanced object detection for railway maintenance

While the current object detection system focuses on collision avoidance tasks, future efforts should focus on expanding detection capabilities to identify small objects related to railway maintenance, such as bolts, fish plates and tools. Enhanced detection systems could also focus on identifying track wear, surface cracks, and degradation. Future studies should consider creating a domain-specific railway maintenance dataset to support object detection for these objects. Synthetic data generation through techniques like Generative Adversarial Networks (GANs) could be used to augment datasets, improving model robustness under varied environmental and operational conditions.

AI-driven adaptation and online learning

Future research should explore the potential of adaptive AI systems that can self-improve during live deployment. Current approaches require pre-trained models, but future systems could adopt online learning and reinforcement learning (RL) methods to enable on-site model adaptation. This would allow RIMV to continuously improve its detection, tracking, and actuation logic based on feedback from new railway environments.

References

- [1] Rail Safety and Standards Board (RSSB), *GERT8000-HB11: Duties of the Person in Charge of the Possession (PICOP)*, September 2021. [Online]. Available: <https://customer-portal.rssb.co.uk>
- [2] M. Arastounia, “Automated recognition of railroad infrastructure in rural areas from lidar data,” *Remote Sensing*, vol. 7, no. 11, pp. 14 916–14 938, 2015.
- [3] D. Stein, *Mobile laser scanning based determination of railway network topology and branching direction on turnouts*. KIT Scientific Publishing, 2018, vol. 38.
- [4] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, “A comprehensive survey on transfer learning,” *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2021.
- [5] L. Zhang, J. Zhang, Y. Peng, and J. Lin, “Intra-domain transfer learning for fault diagnosis with small samples,” *Applied Sciences*, vol. 12, no. 14, p. 7032, 2022.
- [6] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. V. D. Hengel, “A survey of appearance models in visual object tracking,” *ACM transactions on Intelligent Systems and Technology (TIST)*, vol. 4, no. 4, pp. 1–48, 2013.
- [7] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, “Visual tracking: An experimental survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1442–1468, 2014.

- [8] B. Yang and R. Nevatia, “Multi-target tracking by online learning of non-linear motion patterns and robust appearance models,” in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 1918–1925.
- [9] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim, “Multiple object tracking: A literature review,” *Artificial Intelligence*, vol. 293, p. 103448, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0004370220301958>
- [10] T. Li, S. Sun, T. P. Sattar, and J. M. Corchado, “Fight sample degeneracy and impoverishment in particle filters: A review of intelligent approaches,” *Expert Systems with applications*, vol. 41, no. 8, pp. 3944–3954, 2014.
- [11] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.
- [12] Office of Rail and Road, “Passenger rail usage 2022–23,” 2023, accessed: 2024-11-20. [Online]. Available: <https://dataportal.orr.gov.uk/statistics/usage/passenger-rail-usage/>
- [13] Eurostat, “Railway passenger and freight transport statistics 2023,” 2023, accessed: 2024-11-20. [Online]. Available: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Railway_passenger_transport_statistics_-_quarterly_and_annual_data
- [14] M. Rahman, M. Rahimi, A. Starr, I. D. Cardenas, A. Hall, and R. Anderson, “Challenges for a railway inspection and repair system from railway infrastructure,” in *2022 10th International Conference on Control, Mechatronics and Automation (ICCMA)*. IEEE, 2022, pp. 210–215.
- [15] M. Rahimi, H. Liu, I. D. Cardenas, A. Starr, A. Hall, and R. Anderson, “A review on technologies for localisation and navigation in autonomous railway maintenance systems,” *Sensors*, vol. 22, no. 11, p. 4185, 2022.

- [16] H. Liu, M. Rahman, M. Rahimi, A. Starr, I. Durazo-Cardenas, C. Ruiz-Carcel, A. Ompusunggu, A. Hall, and R. Anderson, "An autonomous rail-road amphibious robotic system for railway maintenance using sensor fusion and mobile manipulator," *Computers and Electrical Engineering*, vol. 110, p. 108874, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0045790623002987>
- [17] M. Rahman, H. Liu, I. D. Cardenas, A. Starr, A. Hall, and R. Anderson, "Towards an autonomous rirs: design, structure investigation and framework," in *2021 7th International Conference on Mechatronics and Robotics Engineering (ICMRE)*. IEEE, 2021, pp. 164–168.
- [18] R. Lagay and G. M. Adell, "The autonomous train: a game changer for the railways industry," in *2018 16th international conference on intelligent transportation systems telecommunications (ITST)*. IEEE, 2018, pp. 1–5.
- [19] B. Fremont, A. Menhaj, P. Deloof, and M. Heddebaut, "A cooperative collision avoidance and communication system for railway transports," in *ITSC2000. 2000 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 00TH8493)*. IEEE, 2000, pp. 216–221.
- [20] P. Singh, M. A. Dulebenets, J. Pasha, E. D. S. Gonzalez, Y.-Y. Lau, and R. Kampmann, "Deployment of autonomous trains in rail transportation: Current trends and existing challenges," *IEEE access*, vol. 9, pp. 91 427–91 461, 2021.
- [21] M.-P. Pacaux-Lemoine, Q. Gadmer, and P. Richard, "Train remote driving: A human-machine cooperation point of view," in *2020 IEEE International Conference on Human-Machine Systems (ICHMS)*. IEEE, 2020, pp. 1–4.
- [22] M. Herrb, "H2bis robot platform," <https://homepages.laas.fr/matthieu/robots/h2bis.shtml>, 2024, accessed: 2024-11-20.

- [23] “Railpod: Our platform,” <https://www.rail-pod.com/our-platform/>, 2024, accessed: 2024-11-20.
- [24] SINTEF, “Bob - boots off ballast - robots for safe and efficient railway maintenance,” 2023, accessed: 2024-11-20. [Online]. Available: <https://www.sintef.no/en/projects/2023/-bob-boots-off-ballast-robots-for-safe-and-efficient-railway-maintenance/>
- [25] W. contributors, “Track geometry car,” https://en.wikipedia.org/wiki/Track_geometry_car, 2024, accessed: 2024-11-20.
- [26] E. Magel, M. Roney, J. Kalousek, and P. Sroba, “The blending of theory and practice in modern rail grinding,” *Fatigue & Fracture of Engineering Materials & Structures*, vol. 26, no. 10, pp. 921–929, 2003.
- [27] R. Smith and N. Johnson, “Evaluation of the sperry ultrasonic wheel inspection unit.” Battelle Pacific Northwest Labs., Richland, WA (United States), Tech. Rep., 1970.
- [28] “Assets4rail: Measuring and monitoring railway assets,” <https://cordis.europa.eu/project/id/730849>, 2024, accessed: 2024-11-20.
- [29] F. He, I. Durazo-Cardenas, H. Liu, M. Rahman, M. Rahimi, A. Starr, and M. Poulter, “A practical demonstration of autonomous ultrasonic testing for rail flaws inspection,” 2022.
- [30] European Commission, “Etc levels and modes,” 2024, accessed: 2024-11-21. [Online]. Available: https://transport.ec.europa.eu/transport-modes/rail/ertms/what-ertms-and-how-does-it-work/etc-levels-and-modes_en
- [31] W. Yihui, L. Ren-shi, C. Fang, and N. Bin, “Research on the key issues of railway collision avoidance system,” *Journal of the China Railway Society*, vol. 34, pp. 46–50, 2012.

- [32] L. Jiang, C. Bai-gen, and W. Jian, “Early warning system for potential train collisions: Enhancing safety levels of high-speed trains,” *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, vol. 230, no. 1, pp. 295–312, 2016.
- [33] B. Mishra, “Tmcas: An mqtt based collision avoidance system for railway networks,” in *2018 18th International Conference on Computational Science and Applications (ICCSA)*. IEEE, 2018, pp. 1–6.
- [34] S. Nordhoff, J. D. Lee, S. C. Calvert, S. Berge, M. Hagenzieker, and R. Happee, “(mis-) use of standard autopilot and full self-driving (fsd) beta: Results from interviews with users of tesla’s fsd beta,” *Frontiers in psychology*, vol. 14, p. 1101520, 2023.
- [35] G. Lan and Q. Hao, “End-to-end planning of autonomous driving in industry and academia: 2022-2023,” 2023. [Online]. Available: <https://arxiv.org/abs/2401.08658>
- [36] UK Government, “National ai strategy,” 2021, accessed: 2024-11-25. [Online]. Available: <https://www.gov.uk/government/publications/national-ai-strategy>
- [37] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *European conference on computer vision*. Springer, 2014, pp. 740–755.
- [38] O. Zendel, M. Murschitz, M. Zeilinger, D. Steininger, S. Abbasi, and C. Belez-nai, “Railsem19: A dataset for semantic rail scene understanding,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.
- [39] J. Li, I. Durazo-Cardenas, C. Ruiz-Carcel, F. He, R. Anderson, A. Hall, D. Burbridge, and A. Starr, “Smart railways: the design and construction of an au-

- tonomous inspection and maintenance vehicle,” *Procedia CIRP*, vol. 128, pp. 61–65, 2024.
- [40] J. Kuchar and A. C. Drumm, “The traffic alert and collision avoidance system,” *Lincoln laboratory journal*, vol. 16, no. 2, p. 277, 2007.
- [41] M. Strohmeier, V. Lenders, and I. Martinovic, “On the security of the automatic dependent surveillance-broadcast protocol,” *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 1066–1087, 2014.
- [42] S. Ramasamy, R. Sabatini, A. Gardi, and J. Liu, “Lidar obstacle warning and avoidance system for unmanned aerial vehicle sense-and-avoid,” *Aerospace Science and Technology*, vol. 55, pp. 344–358, 2016.
- [43] B. Hu, R. W. Liu, K. Wang, Y. Li, M. Liang, H. Li, and J. Liu, “Statistical analysis of massive ais trajectories using gaussian mixture models,” in *2017 2nd International Conference on Multimedia and Image Processing (ICMIP)*. IEEE, 2017, pp. 113–117.
- [44] S. Campbell, M. Abu-Tair, and W. Naeem, “An automatic colregs-compliant obstacle avoidance system for an unmanned surface vehicle,” *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, vol. 228, no. 2, pp. 108–121, 2014.
- [45] N. Souweidane and B. Smith, “State of adas, automation, and connectivity,” *Center for Automotive Research March*, pp. 1–40, 2023.
- [46] E. Pastucha, “Catenary system detection, localization and classification using mobile scanning data,” *Remote Sensing*, vol. 8, no. 10, p. 801, 2016.
- [47] B. Yang and L. Fang, “Automated extraction of 3-d railway tracks from mobile laser scanning point clouds,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 12, pp. 4750–4761, 2014.

- [48] S. Kaewunruen and Q. Lian, “Digital twin aided sustainability-based lifecycle management for railway turnout systems,” *Journal of Cleaner Production*, vol. 228, pp. 1537–1551, 2019.
- [49] D. Maier and A. Kleiner, “Improved gps sensor model for mobile robots in urban terrain,” in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 4385–4390.
- [50] A. Ess, K. Schindler, B. Leibe, and L. Van Gool, “Object detection and tracking for autonomous navigation in dynamic environments,” *The International Journal of Robotics Research*, vol. 29, no. 14, pp. 1707–1725, 2010.
- [51] A. Kamann, P. Held, F. Perras, P. Zaumseil, T. Brandmeier, and U. T. Schwarz, “Automotive radar multipath propagation in uncertain environments,” in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 859–864.
- [52] E. G. Zimelman and R. F. Keefe, “Real-time positioning in logging: Effects of forest stand characteristics, topography, and line-of-sight obstructions on gnss-rf transponder accuracy and radio signal propagation,” *PLoS One*, vol. 13, no. 1, p. e0191017, 2018.
- [53] R. Manduchi, A. Castano, A. Talukder, and L. Matthies, “Obstacle detection and terrain classification for autonomous off-road navigation,” *Autonomous robots*, vol. 18, no. 1, pp. 81–102, 2005.
- [54] J.-S. Yoon, M. Sagong, J. Lee, and K.-s. Lee, “Feature extraction of a concrete tunnel liner from 3d laser scanning data,” *Ndt & E international*, vol. 42, no. 2, pp. 97–105, 2009.
- [55] I. Puente, H. González-Jorge, J. Martínez-Sánchez, and P. Arias, “Automatic detection of road tunnel luminaires using a mobile lidar system,” *Measurement*, vol. 47, pp. 569–575, 2014.

- [56] K. Grauman and B. Leibe, “Visual object recognition,” *Synthesis lectures on artificial intelligence and machine learning*, vol. 5, no. 2, pp. 1–181, 2011.
- [57] X. Zhang, Y.-H. Yang, Z. Han, H. Wang, and C. Gao, “Object class detection: A survey,” *ACM Computing Surveys (CSUR)*, vol. 46, no. 1, pp. 1–53, 2013.
- [58] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, “Object detection with deep learning: A review,” *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.
- [59] Z. Zou, Z. Shi, Y. Guo, and J. Ye, “Object detection in 20 years: A survey,” *arXiv preprint arXiv:1905.05055*, 2019.
- [60] A. Yilmaz, O. Javed, and M. Shah, “Object tracking: A survey,” *Acm computing surveys (CSUR)*, vol. 38, no. 4, pp. 13–es, 2006.
- [61] Z. Pan, S. Liu, and W. Fu, “A review of visual moving target tracking,” *Multimedia Tools and Applications*, vol. 76, no. 16, pp. 16 989–17 018, 2017.
- [62] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015. [Online]. Available: <https://doi.org/10.1038/nature14539>
- [63] G. Ciaparrone, F. L. Sánchez, S. Tabik, L. Troiano, R. Tagliaferri, and F. Herrera, “Deep learning in video multi-object tracking: A survey,” *Neurocomputing*, vol. 381, pp. 61–88, 2020.
- [64] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, “Deep learning for generic object detection: A survey,” *International journal of computer vision*, vol. 128, no. 2, pp. 261–318, 2020.
- [65] S. Zhang, L. Wen, Z. Lei, and S. Z. Li, “Refinedet++: Single-shot refinement neural network for object detection,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 2, pp. 674–687, 2020.

- [66] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Single-shot refinement neural network for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4203–4212.
- [67] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.
- [68] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.
- [69] K. E. Van de Sande, J. R. Uijlings, T. Gevers, and A. W. Smeulders, "Segmentation as selective search for object recognition," in *2011 international conference on computer vision*. IEEE, 2011, pp. 1879–1886.
- [70] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.
- [71] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.
- [72] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, pp. 142–158, 2015.
- [73] A. M. Hafiz and G. M. Bhat, "A survey on instance segmentation: state of the art," *International Journal of Multimedia Information Retrieval*, vol. 9, no. 3, p. 171–189, Jul 2020. [Online]. Available: <http://dx.doi.org/10.1007/s13735-020-00195-x>

- [74] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.
- [75] T. Kong, A. Yao, Y. Chen, and F. Sun, “Hypernet: Towards accurate region proposal generation and joint object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 845–853.
- [76] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.
- [77] S. Lazebnik, C. Schmid, and J. Ponce, “Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories,” in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*, vol. 2. IEEE, 2006, pp. 2169–2178.
- [78] K. Grauman and T. Darrell, “The pyramid match kernel: Discriminative classification with sets of image features,” in *Tenth IEEE International Conference on Computer Vision (ICCV’05) Volume 1*, vol. 2. IEEE, 2005, pp. 1458–1465.
- [79] R. Girshick, “Fast r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.
- [80] P. Li, D. Wang, L. Wang, and H. Lu, “Deep visual tracking: Review and experimental comparison,” *Pattern Recognition*, vol. 76, pp. 323 – 338, 2018. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0031320317304612>
- [81] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, “High performance visual tracking with siamese region proposal network,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8971–8980.

- [82] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, “A unified multi-scale deep convolutional neural network for fast object detection,” in *European conference on computer vision*. Springer, 2016, pp. 354–370.
- [83] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.
- [84] K. Lenc and A. Vedaldi, “R-cnn minus r,” *arXiv preprint arXiv:1506.06981*, 2015.
- [85] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Object detectors emerge in deep scene cnns,” *arXiv preprint arXiv:1412.6856*, 2014.
- [86] —, “Learning deep features for discriminative localization,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2921–2929.
- [87] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.
- [88] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, “Simultaneous detection and segmentation,” in *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part VII 13*. Springer, 2014, pp. 297–312.
- [89] K. Chen, J. Pang, J. Wang, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Shi, W. Ouyang *et al.*, “Hybrid task cascade for instance segmentation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4974–4983.

- [90] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei, “Fully convolutional instance-aware semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2359–2367.
- [91] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.
- [92] J. Dai, K. He, and J. Sun, “Instance-aware semantic segmentation via multi-task network cascades,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3150–3158.
- [93] J. Dai, Y. Li, K. He, and J. Sun, “R-fcn: Object detection via region-based fully convolutional networks,” in *Advances in neural information processing systems*, 2016, pp. 379–387.
- [94] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [95] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille, “Attention to scale: Scale-aware semantic image segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [96] P. Dollár, R. Appel, S. Belongie, and P. Perona, “Fast feature pyramids for object detection,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 8, pp. 1532–1545, 2014.
- [97] T. Kong, F. Sun, C. Tan, H. Liu, and W. Huang, “Deep feature pyramid reconfiguration for object detection,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 169–185.
- [98] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, “Joint 3d proposal generation and object detection from view aggregation,” in *2018 IEEE/RSJ Inter-*

- national Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–8.
- [99] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in *European conference on computer vision*. Springer, 2016, pp. 21–37.
- [100] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick, “Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2874–2883.
- [101] S. Gidaris and N. Komodakis, “Object detection via a multi-region and semantic segmentation-aware cnn model,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1134–1142.
- [102] Z. Qin, Z. Li, Z. Zhang, Y. Bao, G. Yu, Y. Peng, and J. Sun, “Thudernet: Towards real-time generic object detection on mobile devices,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6718–6727.
- [103] H. Law and J. Deng, “Cornernet: Detecting objects as paired keypoints,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 734–750.
- [104] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” *arXiv preprint arXiv:1312.6229*, 2013.
- [105] L. Chen, Z. Yang, J. Ma, and Z. Luo, “Driving scene perception network: Real-time joint detection, depth estimation and semantic segmentation,” in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 1283–1291.

- [106] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.
- [107] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, “Scalable object detection using deep neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 2147–2154.
- [108] C. Szegedy, A. Toshev, and D. Erhan, “Deep neural networks for object detection,” *Advances in neural information processing systems*, vol. 26, 2013.
- [109] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, “Centernet: Keypoint triplets for object detection,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6569–6578.
- [110] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun, “Multinet: Real-time joint semantic reasoning for autonomous driving,” in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1013–1020.
- [111] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen, “Ron: Reverse connection with objectness prior networks for object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5936–5944.
- [112] X. Wang, A. Shrivastava, and A. Gupta, “A-fast-rcnn: Hard positive generation via adversary for object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2606–2615.
- [113] D. Dwibedi, I. Misra, and M. Hebert, “Cut, paste and learn: Surprisingly easy synthesis for instance detection,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1301–1310.

- [114] N. Dvornik, J. Mairal, and C. Schmid, “Modeling visual context is key to augmenting object detection datasets,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 364–380.
- [115] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, “Deformable convolutional networks,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 764–773.
- [116] M. Yazdi and T. Bouwmans, “New trends on moving object detection in video images captured by a moving camera: A survey,” *Computer Science Review*, vol. 28, pp. 157–177, 2018.
- [117] N. Al-Najdawi, H. E. Bez, J. Singhai, and E. A. Edirisinghe, “A survey of cast shadow detection algorithms,” *Pattern Recognition Letters*, vol. 33, no. 6, pp. 752–764, 2012.
- [118] O. Day and T. M. Khoshgoftaar, “A survey on heterogeneous transfer learning,” *Journal of Big Data*, vol. 4, pp. 1–42, 2017.
- [119] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?” *Advances in neural information processing systems*, vol. 27, 2014.
- [120] S. Akçay, M. E. Kundegorski, M. Devereux, and T. P. Breckon, “Transfer learning using convolutional neural networks for object classification within x-ray baggage security imagery,” in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 1057–1061.
- [121] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, “Transfer learning using computational intelligence: A survey,” *Knowledge-Based Systems*, vol. 80, pp. 14–23, 2015, 25th anniversary of Knowledge-Based Systems. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0950705115000179>

- [122] W. Pan, “A survey of transfer learning for collaborative recommendation with auxiliary data,” *Neurocomputing*, vol. 177, pp. 447–453, 2016.
- [123] S. Niu, Y. Liu, J. Wang, and H. Song, “A decade survey of transfer learning (2010–2020),” *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 2, pp. 151–166, 2020.
- [124] K. Weiss, T. M. Khoshgoftaar, and D. Wang, “A survey of transfer learning,” *Journal of Big data*, vol. 3, pp. 1–40, 2016.
- [125] R. Ribani and M. Marengoni, “A survey of transfer learning for convolutional neural networks,” in *2019 32nd SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, 2019, pp. 47–57.
- [126] R. Liu, Y. Shi, C. Ji, and M. Jia, “A survey of sentiment analysis based on transfer learning,” *IEEE Access*, vol. 7, pp. 85 401–85 412, 2019.
- [127] I. Athanasiadis, P. Mousoulitis, and L. Petrou, “A framework of transfer learning in object detection for embedded systems,” *arXiv preprint arXiv:1811.04863*, 2018.
- [128] J. Plested and T. Gedeon, “Deep transfer learning for image classification: a survey,” 2022.
- [129] F. Yu, X. Xiu, and Y. Li, “A survey on deep transfer learning and beyond,” *Mathematics*, vol. 10, no. 19, p. 3619, Oct 2022. [Online]. Available: <http://dx.doi.org/10.3390/math10193619>
- [130] M. T. Rosenstein, Z. Marx, L. P. Kaelbling, and T. G. Dietterich, “To transfer or not to transfer,” in *NIPS 2005 workshop on transfer learning*, vol. 898, no. 3, 2005.
- [131] B. Neyshabur, H. Sedghi, and C. Zhang, “What is being transferred in transfer learning?” *Advances in neural information processing systems*, vol. 33, pp. 512–523, 2020.

- [132] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, “A survey on deep transfer learning,” in *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III* 27. Springer, 2018, pp. 270–279.
- [133] N. Wang and D. Y. Yeung, “Learning a deep compact image representation for visual tracking,” *Advances in neural information processing systems*, 2013.
- [134] X.-Q. Zhang, R.-H. Jiang, C.-X. Fan, T.-Y. Tong, T. Wang, and P.-C. Huang, “Advances in deep learning methods for visual tracking: Literature review and fundamentals,” *International Journal of Automation and Computing*, Mar. 2021. [Online]. Available: <https://doi.org/10.1007/s11633-020-1274-8>
- [135] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” in *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2016, pp. 3464–3468.
- [136] B. Leibe, K. Schindler, N. Cornelis, and L. Van Gool, “Coupled object detection and tracking from static cameras and moving vehicles,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 10, pp. 1683–1698, 2008.
- [137] M. Kaushal, B. S. Khehra, and A. Sharma, “Soft computing based object detection and tracking approaches: State-of-the-art survey,” *Applied Soft Computing*, vol. 70, pp. 423–464, 2018.
- [138] Y. Xiang, A. Alahi, and S. Savarese, “Learning to track: Online multi-object tracking by decision making,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4705–4713.
- [139] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, “Recent advances and trends in visual tracking: A review,” *Neurocomputing*, vol. 74, no. 18, pp. 3823–3831, 2011.

- [140] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool, “Pedestrian detection at 100 frames per second,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2903–2910.
- [141] V. Prisacariu, I. Reid *et al.*, “fasthog-a real-time gpu implementation of hog,” *Department of Engineering Science*, vol. 2310, no. 9, 2009.
- [142] P. Sudowe and B. Leibe, “Efficient use of geometric constraints for sliding-window object detection in video,” in *International Conference on Computer Vision Systems*. Springer, 2011, pp. 11–20.
- [143] V. Lepetit and P. Fua, “Keypoint recognition using randomized trees,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 9, pp. 1465–1479, 2006.
- [144] A. Adam, E. Rivlin, and I. Shimshoni, “Robust fragments-based tracking using the integral histogram,” in *2006 IEEE Computer society conference on computer vision and pattern recognition (CVPR’06)*, vol. 1. IEEE, 2006, pp. 798–805.
- [145] B. Babenko, M.-H. Yang, and S. Belongie, “Robust object tracking with online multiple instance learning,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 8, pp. 1619–1632, 2010.
- [146] L. Matthews, T. Ishikawa, and S. Baker, “The template update problem,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 6, pp. 810–815, 2004.
- [147] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof, “Prost: Parallel robust online simple tracking,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 723–730.
- [148] A. Andriyenko and K. Schindler, “Multi-target tracking by continuous energy minimization,” in *CVPR*, vol. 2, no. 6, 2011, p. 7.

- [149] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler, "Online multi-target tracking using recurrent neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.
- [150] K. Zhang, L. Zhang, and M.-H. Yang, "Fast compressive tracking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 10, pp. 2002–2015, 2014.
- [151] Q. Abbas, M. E. Ibrahim, and M. A. Jaffar, "A comprehensive review of recent advances on deep vision systems," *Artificial Intelligence Review*, vol. 52, no. 1, pp. 39–76, 2019.
- [152] H. Li, Y. Li, and F. Porikli, "Deeptrack: Learning discriminative feature representations by convolutional neural networks for visual tracking," in *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014.
- [153] Z. Cai, Z. Gu, Z. L. Yu, H. Liu, and K. Zhang, "A real-time visual object tracking system based on kalman filter and mb-lbp feature matching," *Multimedia Tools and Applications*, vol. 75, no. 4, pp. 2393–2409, 2016.
- [154] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. Torr, "Fast online object tracking and segmentation: A unifying approach," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1328–1338.
- [155] X. Cheng, N. Li, T. Zhou, L. Zhou, and Z. Wu, "Object tracking via collaborative multi-task learning and appearance model updating," *Applied Soft Computing*, vol. 31, pp. 81–90, 2015.
- [156] R. Kaestner, J. Maye, Y. Pilat, and R. Siegwart, "Generative object detection and tracking in 3d range data," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 3075–3081.

- [157] B. Yang and R. Nevatia, “Online learned discriminative part-based appearance models for multi-human tracking,” in *Computer Vision – ECCV 2012*, A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 484–498.
- [158] T. Zhou, Y. Lu, and H. Di, “Locality-constrained collaborative model for robust visual tracking,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 2, pp. 313–325, 2015.
- [159] X. Dong, J. Shen, D. Yu, W. Wang, J. Liu, and H. Huang, “Occlusion-aware real-time object tracking,” *IEEE Transactions on Multimedia*, vol. 19, no. 4, pp. 763–771, 2017.
- [160] S. Sun, N. Akhtar, H. Song, A. Mian, and M. Shah, “Deep affinity network for multiple object tracking,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 1, pp. 104–119, 2019.
- [161] L. Fan, Z. Wang, B. Cail, C. Tao, Z. Zhang, Y. Wang, S. Li, F. Huang, S. Fu, and F. Zhang, “A survey on multiple object tracking algorithm,” in *2016 IEEE International Conference on Information and Automation (ICIA)*. IEEE, 2016, pp. 1855–1862.
- [162] S.-H. Bae and K.-J. Yoon, “Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1218–1225.
- [163] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, “Visual object tracking using adaptive correlation filters,” in *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 2544–2550.

- [164] S. Hong, T. You, S. Kwak, and B. Han, “Online tracking by learning discriminative saliency map with convolutional neural network,” in *International conference on machine learning*, 2015, pp. 597–606.
- [165] H. Possegger, T. Mauthner, and H. Bischof, “In defense of color-based model-free tracking,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2113–2120.
- [166] N. Wojke, A. Bewley, and D. Paulus, “Simple online and realtime tracking with a deep association metric,” in *2017 IEEE international conference on image processing (ICIP)*. IEEE, 2017, pp. 3645–3649.
- [167] X. Wang, Z. Chen, J. Tang, B. Luo, Y. Wang, Y. Tian, and F. Wu, “Dynamic attention guided multi-trajectory analysis for single object tracking,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 12, pp. 4895–4908, 2021.
- [168] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, “Multiple hypothesis tracking revisited,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4696–4704.
- [169] S. H. Rezatofighi, A. Milan, Z. Zhang, Q. Shi, A. Dick, and I. Reid, “Joint probabilistic data association revisited,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 3047–3055.
- [170] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua, “Multiple object tracking using k-shortest paths optimization,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 9, pp. 1806–1819, 2011.
- [171] D. Reid, “An algorithm for tracking multiple targets,” *IEEE transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.

- [172] D. K. Prasad, D. Rajan, L. Rachmawati, E. Rajabally, and C. Quek, "Video processing from electro-optical sensors for object detection and tracking in a maritime environment: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 8, pp. 1993–2016, 2017.
- [173] D. Bloisi and L. Iocchi, "Argos—a video surveillance system for boat traffic monitoring in venice," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, no. 07, pp. 1477–1502, 2009.
- [174] Y. Zhang, Y. Huang, and L. Wang, "Multi-task deep learning for fast online multiple object tracking," in *2017 4th IAPR Asian Conference on Pattern Recognition (ACPR)*. IEEE, 2017, pp. 138–143.
- [175] T. Liu, G. Wang, and Q. Yang, "Real-time part-based visual tracking via adaptive correlation filters," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4902–4912.
- [176] J. S. Liu, R. Chen, and T. Logvinenko, "A theoretical framework for sequential importance sampling with resampling," in *Sequential Monte Carlo methods in practice*. Springer, 2001, pp. 225–246.
- [177] F. Porikli, "Achieving real-time object detection and tracking under extreme conditions," *Journal of Real-Time Image Processing*, vol. 1, no. 1, pp. 33–40, 2006.
- [178] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003.
- [179] S. Avidan, "Ensemble tracking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 2, pp. 261–271, 2007.

- [180] S. Huang and J. Hong, “Moving object tracking system based on camshift and kalman filter,” in *2011 International Conference on Consumer Electronics, Communications and Networks (CECNet)*. IEEE, 2011, pp. 1423–1426.
- [181] K.-P. Karmann, “Moving object recognition using an adaptive background memory,” *Proc. Time Varying Image Processing*, 1990.
- [182] M. Kilger, “A shadow handler in a video-based real-time traffic monitoring system,” in *Proceedings IEEE Workshop on applications of computer vision*. IEEE Computer Society, 1992, pp. 11–12.
- [183] W. Hu, T. Tan, L. Wang, and S. Maybank, “A survey on visual surveillance of object motion and behaviors,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 34, no. 3, pp. 334–352, 2004.
- [184] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. Torr, “Staple: Complementary learners for real-time tracking,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1401–1409.
- [185] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, “High-speed tracking with kernelized correlation filters,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 3, pp. 583–596, 2014.
- [186] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, “Learning spatially regularized correlation filters for visual tracking,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4310–4318.
- [187] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg, “Eco: Efficient convolution operators for tracking,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 6638–6646.
- [188] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. Torr, “End-to-end representation learning for correlation filter based tracking,” in *Proceedings of the*

- IEEE conference on computer vision and pattern recognition*, 2017, pp. 2805–2813.
- [189] M. Wang, Y. Liu, and Z. Huang, “Large margin object tracking with circulant feature maps,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4021–4029.
- [190] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Young Choi, “Action-decision networks for visual tracking with deep reinforcement learning,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2711–2720.
- [191] S. Hong, S. Kwak, and B. Han, “Orderless tracking through model-averaged posterior estimation,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2296–2303.
- [192] S. Sridhar, F. Mueller, M. Zollhöfer, D. Casas, A. Oulasvirta, and C. Theobalt, “Real-time joint tracking of a hand manipulating an object from rgb-d input,” in *European Conference on Computer Vision*. Springer, 2016, pp. 294–310.
- [193] G. S. Walia and R. Kapoor, “Recent advances on multicue object tracking: a survey,” *Artificial Intelligence Review*, vol. 46, no. 1, pp. 1–39, 2016.
- [194] G. Ning, Z. Zhang, C. Huang, X. Ren, H. Wang, C. Cai, and Z. He, “Spatially supervised recurrent convolutional neural networks for visual object tracking,” in *2017 IEEE international symposium on circuits and systems (ISCAS)*. IEEE, 2017, pp. 1–4.
- [195] F. Farahi and H. S. Yazdi, “Probabilistic kalman filter for moving object tracking,” *Signal Processing: Image Communication*, vol. 82, p. 115751, 2020.
- [196] Z. Zhou, D. Wu, and Z. Zhu, “Object tracking based on kalman particle filter with lssvr,” *Optik*, vol. 127, no. 2, pp. 613–619, 2016.

- [197] K. Zhang and H. Song, “Real-time visual tracking via online weighted multiple instance learning,” *Pattern recognition*, vol. 46, no. 1, pp. 397–411, 2013.
- [198] Z. Kalal, K. Mikolajczyk, and J. Matas, “Tracking-learning-detection,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 7, pp. 1409–1422, 2011.
- [199] J. Shen, X. Yang, Y. Jia, and X. Li, “Intrinsic images using optimization,” in *CVPR 2011*. IEEE, 2011, pp. 3481–3487.
- [200] H. Grabner, C. Leistner, and H. Bischof, “Semi-supervised on-line boosting for robust tracking,” in *European conference on computer vision*. Springer, 2008, pp. 234–247.
- [201] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof, “On-line random forests,” in *2009 IEEE 12th international conference on computer vision workshops, iccv workshops*. IEEE, 2009, pp. 1393–1400.
- [202] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi, “Performance measures and a data set for multi-target, multi-camera tracking,” in *European conference on computer vision*. Springer, 2016, pp. 17–35.
- [203] E. Ristani and C. Tomasi, “Features for multi-target multi-camera tracking and re-identification,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6036–6046.
- [204] Y. Park, V. Lepetit, and W. Woo, “Multiple 3d object tracking for augmented reality,” in *2008 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*. IEEE, 2008, pp. 117–120.
- [205] T. E. Cognard, C. Dainty, and N. Devaney, “Estimating axial resolution with diffraction theory,” *Applied Optics*, vol. 57, no. 22, pp. E138–E141, 2018.

- [206] T. Lemaire, C. Berger, I.-K. Jung, and S. Lacroix, “Vision-based slam: Stereo and monocular approaches,” *International Journal of Computer Vision*, vol. 74, no. 3, pp. 343–364, 2007.
- [207] V. Mohanty, S. Agrawal, S. Datta, A. Ghosh, V. D. Sharma, and D. Chakravarty, “Deepvo: A deep learning approach for monocular visual odometry,” *arXiv preprint arXiv:1611.06069*, 2016.
- [208] Q. Jia, L. Chang, B. Qiang, S. Zhang, W. Xie, X. Yang, Y. Sun, and M. Yang, “Real-time 3d reconstruction method based on monocular vision,” *Sensors*, vol. 21, no. 17, p. 5909, 2021.
- [209] V. Guizilini, I. Vasiljevic, D. Chen, R. Ambruş, and A. Gaidon, “Towards zero-shot scale-aware monocular depth estimation,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 9233–9243.
- [210] S. Xue, X. Li, and Q. Liu, “Research on three-dimensional measurement method based on monocular vision,” in *Proceedings of the 2019 4th International Conference on Robotics, Control and Automation*, 2019, pp. 238–241.
- [211] Z. Zhang, C. Xu, J. Yang, J. Gao, and Z. Cui, “Progressive hard-mining network for monocular depth estimation,” *IEEE Transactions on Image Processing*, vol. 27, no. 8, pp. 3691–3702, 2018.
- [212] A. Saxena, J. Schulte, A. Y. Ng *et al.*, “Depth estimation using monocular and stereo cues,” in *IJCAI*, vol. 7, 2007, pp. 2197–2203.
- [213] M. Persson, T. Piccini, M. Felsberg, and R. Mester, “Robust stereo visual odometry from monocular techniques,” in *2015 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2015, pp. 686–691.
- [214] L. M. Paz, P. Piniés, J. D. Tardós, and J. Neira, “Large-scale 6-dof slam with stereo-in-hand,” *IEEE transactions on robotics*, vol. 24, no. 5, pp. 946–957, 2008.

- [215] S. Praveen, “Efficient depth estimation using sparse stereo-vision with other perception techniques,” *Coding Theory*, vol. 111, no. 10.5772, 2020.
- [216] Y.-C. Lim, C.-H. Lee, S. Kwon, and W.-Y. Jung, “Distance estimation algorithm for both long and short ranges based on stereo vision system,” in *2008 IEEE Intelligent Vehicles Symposium*. IEEE, 2008, pp. 841–846.
- [217] R. Xu, W. Dong, A. Sharma, and M. Kaess, “Learned depth estimation of 3d imaging radar for indoor mapping,” in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 13 260–13 267.
- [218] M. Jamelska, “Radar vs. lidar sensors in the automotive industry,” 2017, accessed: 2024-05-20. [Online]. Available: <https://mse238blog.stanford.edu/2017/08/mj2017/radar-vs-lidar-sensors-in-automotive-industry/>
- [219] C. V. Poulton, A. Yaacobi, D. B. Cole, M. J. Byrd, M. Raval, D. Vermeulen, and M. R. Watts, “Coherent solid-state lidar with silicon photonic optical phased arrays,” *Optics letters*, vol. 42, no. 20, pp. 4091–4094, 2017.
- [220] R. Yanase, D. Hirano, M. Aldibaja, K. Yoneda, and N. Suganuma, “Lidar-and radar-based robust vehicle localization with confidence estimation of matching results,” *Sensors*, vol. 22, no. 9, p. 3545, 2022.
- [221] M. Cui, Y. Zhu, Y. Liu, Y. Liu, G. Chen, and K. Huang, “Dense depth-map estimation based on fusion of event camera and sparse lidar,” *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–11, 2022.
- [222] J.-T. Lin, D. Dai, and L. Van Gool, “Depth estimation from monocular images and sparse radar data,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 10 233–10 240.

- [223] M. Xiong, X. Xu, D. Yang, and E. Steinbach, “Robust depth estimation in foggy environments combining rgb images and mmwave radar,” in *2022 IEEE International Symposium on Multimedia (ISM)*. IEEE, 2022, pp. 34–41.
- [224] S. A. Siddiqui, A. Vierling, and K. Berns, “Multi-modal depth estimation using convolutional neural networks,” in *2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2020, pp. 354–359.
- [225] A. Masoumian, H. A. Rashwan, J. Cristiano, M. S. Asif, and D. Puig, “Monocular depth estimation using deep learning: A review,” *Sensors*, vol. 22, no. 14, p. 5353, 2022.
- [226] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.
- [227] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, “Frustum pointnets for 3d object detection from rgb-d data,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 918–927.
- [228] W. Zhou, Y. Dai, and R. He, “Efficient depth estimation from single image,” in *2014 IEEE China Summit & International Conference on Signal and Information Processing (ChinaSIP)*. IEEE, 2014, pp. 296–300.
- [229] F. Faisal, M. A. Salam, M. B. Habib, M. S. Islam, and M. M. Nishat, “Depth estimation from video using computer vision and machine learning with hyperparameter optimization,” in *2022 4th International Conference on Smart Sensors and Application (ICSSA)*. IEEE, 2022, pp. 39–44.
- [230] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, “Robot operating system 2: Design, architecture, and uses in the wild,” *Science robotics*, vol. 7, no. 66, p. eabm6074, 2022.

- [231] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “Ros: an open-source robot operating system,” in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.
- [232] Y. Maruyama, S. Kato, and T. Azumi, “Exploring the performance of ros2,” in *Proceedings of the 13th international conference on embedded software*, 2016, pp. 1–10.
- [233] C. Plasberg, H. Nessau, D. Timmermann, C. Eichmann, A. Roennau, and R. Dillmann, “Towards distributed real-time capable robotic control using ros2,” in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2022, pp. 2205–2210.
- [234] G. Deng, G. Xu, Y. Zhou, T. Zhang, and Y. Liu, “On the (in) security of secure ros2,” in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 739–753.
- [235] Y. H. Jo, S. Y. Cho, and B. W. Choi, “Towards a ros2-based software architecture for service robots,” *Bulletin of Electrical Engineering and Informatics*, vol. 12, no. 5, pp. 3027–3038, 2023.
- [236] G. Stavrinos, “Ros2 for ros1 users,” *Robot Operating System (ROS) The Complete Reference (Volume 5)*, pp. 31–42, 2021.
- [237] K. Belsare, A. C. Rodriguez, P. G. Sánchez, J. Hierro, T. Kołcon, R. Lange, I. Lütkebohle, A. Malki, J. M. Losa, F. Melendez *et al.*, “Micro-ros,” in *Robot Operating System (ROS) The Complete Reference (Volume 7)*. Springer, 2023, pp. 3–55.
- [238] M. Zhang, X. Xu, Y. Chen, and M. Li, “A lightweight and accurate localization algorithm using multiple inertial measurement units,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1508–1515, 2020.

- [239] J. Kim, M. Song, J. Kim, and D. Lee, “Gps-free localization based on multiple imus and federated filter fusion,” *Journal of Institute of Control, Robotics and Systems*, vol. 26, no. 9, pp. 708–714, 2020.
- [240] B. Karlsen, “Humans: trouble, or treasure in the eyes of en 50126-1: 2017 railway applications-the specification and demonstration of reliability, availability, maintainability and safety (rams).” 2024.
- [241] E. Std, “Railway applications-communications, signalling and processing systems-safety related electronic systems for signalling,” *European Committee for Electrotechnical Standardisation (CENELEC)*, 2019.
- [242] D. Hopkins and T. Schwanen, “Talking about automated vehicles: What do levels of automation do?” *Technology in Society*, vol. 64, p. 101488, 2021.
- [243] S. Shao, Z. Li, T. Zhang, C. Peng, G. Yu, X. Zhang, J. Li, and J. Sun, “Objects365: A large-scale, high-quality dataset for object detection,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 8430–8439.
- [244] M. Rahman, H. Liu, I. D. Cardenas, A. Starr, A. Hall, and R. Anderson, “A review on the prospects of mobile manipulators for smart maintenance of railway track,” *Applied Sciences*, vol. 13, no. 11, p. 6484, 2023.
- [245] M. Chen, J. Fu, and H. Ling, “One-shot neural ensemble architecture search by diversity-guided search space shrinking,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 16 530–16 539.
- [246] A. Borji, “Complementary datasets to coco for object detection,” *arXiv preprint arXiv:2206.11473*, 2022.
- [247] M. Mohammadi, M. Eftekhari, and A. Hassani, “Image-text connection: Exploring the expansion of the diversity within joint feature space similarity scores,” *IEEE Access*, vol. 11, pp. 123 209–123 222, 2023.

- [248] H. Chang, J. Han, C. Zhong, A. M. Snijders, and J.-H. Mao, “Unsupervised transfer learning via multi-scale convolutional sparse coding for biomedical applications,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 5, pp. 1182–1194, 2017.
- [249] W. Ying, Y. Zhang, J. Huang, and Q. Yang, “Transfer learning via learning to transfer,” in *International Conference on Machine Learning*. PMLR, 2018, pp. 5085–5094.
- [250] J. Talukdar, S. Gupta, P. Rajpura, and R. S. Hegde, “Transfer learning for object detection using state-of-the-art deep neural networks,” in *2018 5th international conference on signal processing and integrated networks (SPIN)*. IEEE, 2018, pp. 78–83.
- [251] X. Xu, H. Zhang, Y. Ma, K. Liu, H. Bao, and X. Qian, “Transdet: Toward effective transfer learning for small-object detection,” *Remote Sensing*, vol. 15, no. 14, p. 3525, 2023.
- [252] J. J. Lim, R. R. Salakhutdinov, and A. Torralba, “Transfer learning by borrowing examples for multiclass object detection,” *Advances in neural information processing systems*, vol. 24, 2011.
- [253] W. Yang, J. Zhang, Z. Chen, and Z. Xu, “An efficient semantic segmentation method based on transfer learning from object detection,” *IET Image Processing*, vol. 15, no. 1, pp. 57–64, 2021.
- [254] M. Long, H. Zhu, J. Wang, and M. I. Jordan, “Deep transfer learning with joint adaptation networks,” in *International conference on machine learning*. PMLR, 2017, pp. 2208–2217.
- [255] S. J. Pan, J. T. Kwok, Q. Yang *et al.*, “Transfer learning via dimensionality reduction,” in *AAAI*, vol. 8, 2008, pp. 677–682.

- [256] B. Cao, S. J. Pan, Y. Zhang, D.-Y. Yeung, and Q. Yang, “Adaptive transfer learning,” in *proceedings of the AAAI Conference on Artificial Intelligence*, vol. 24, no. 1, 2010, pp. 407–412.
- [257] Y. Yao and G. Doretto, “Boosting for transfer learning with multiple sources,” in *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 1855–1862.
- [258] J. Wang, Y. Chen, W. Feng, H. Yu, M. Huang, and Q. Yang, “Transfer learning with dynamic distribution adaptation,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 11, no. 1, pp. 1–25, 2020.
- [259] X. Bu, J. Peng, J. Yan, T. Tan, and Z. Zhang, “Gaia: A transfer learning system of object detection that fits your needs,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 274–283.
- [260] Z. Shi, P. Siva, and T. Xiang, “Transfer learning by ranking for weakly supervised object annotation,” *arXiv preprint arXiv:1705.00873*, 2017.
- [261] M. Woźniak, M. Wiecek, and J. Siłka, “Deep neural network with transfer learning in remote object detection from drone,” in *Proceedings of the 5th international ACM mobicom workshop on drone assisted wireless communications for 5G and beyond*, 2022, pp. 121–126.
- [262] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, “Bytetrack: Multi-object tracking by associating every detection box,” 2022.
- [263] N. Aharon, R. Orfaig, and B.-Z. Bobrovsky, “Bot-sort: Robust associations multi-pedestrian tracking,” *arXiv preprint arXiv:2206.14651*, 2022.
- [264] W. Wang, K. Sakurada, and N. Kawaguchi, “Reflectance intensity assisted automatic and accurate extrinsic calibration of 3d lidar and panoramic camera using a printed chessboard,” *Remote Sensing*, vol. 9, no. 8, p. 851, 2017.

- [265] Z. Lai, Y. Wang, S. Guo, X. Meng, J. Li, W. Li, and S. Han, “Laser reflectance feature assisted accurate extrinsic calibration for non-repetitive scanning lidar and camera systems,” *Optics Express*, vol. 30, no. 10, pp. 16 242–16 263, 2022.
- [266] P. An, T. Ma, K. Yu, B. Fang, J. Zhang, W. Fu, and J. Ma, “Geometric calibration for lidar-camera system fusing 3d-2d and 3d-3d point correspondences,” *Optics express*, vol. 28, no. 2, pp. 2122–2141, 2020.
- [267] E.-S. Kim and S.-Y. Park, “Extrinsic calibration of a camera-lidar multi sensor system using a planar chessboard,” in *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, 2019, pp. 89–91.
- [268] D. Tsai, S. Worrall, M. Shan, A. Lohr, and E. Nebot, “Optimising the selection of samples for robust lidar camera calibration,” in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2021, pp. 2631–2638.
- [269] S. K. Bhandary, R. Dhakal, V. Sanghavi, and P. K. Verkicharla, “Ambient light level varies with different locations and environmental conditions: Potential to impact myopia,” *Plos one*, vol. 16, no. 7, p. e0254027, 2021.
- [270] E. Bylow, J. Sturm, C. Kerl, F. Kahl, and D. Cremers, “Real-time camera tracking and 3d reconstruction using signed distance functions.” in *Robotics: Science and Systems*, vol. 2, 2013, p. 2.
- [271] B. Benfold and I. Reid, “Stable multi-target tracking in real-time surveillance video,” in *CVPR 2011*. IEEE, 2011, pp. 3457–3464.
- [272] C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, L. Jesus, R. Berriel, T. M. Paixao, F. Mutz *et al.*, “Self-driving cars: A survey,” *Expert systems with applications*, vol. 165, p. 113816, 2021.
- [273] K. Shahriari and M. Shahriari, “Ieee standard review—ethically aligned design: A vision for prioritizing human wellbeing with artificial intelligence and autonomous

- systems,” in *2017 IEEE Canada International Humanitarian Technology Conference (IHTC)*. IEEE, 2017, pp. 197–201.
- [274] N. A. Smuha, “The eu approach to ethics guidelines for trustworthy artificial intelligence,” *Computer Law Review International*, vol. 20, no. 4, pp. 97–106, 2019.

Appendix A

Appendix

A.1 RailSem19 variation results

A.1.1 RailSem19 instance table

Class Name	RS19_v0	RS19_v1	RS19_v2	RS19_v3	RS19_v4
Switch-left	1963	-	-	-	-
Switch-indicator	2005	2005	2005	2005	-
Switch-right	2080	-	-	-	-
Switch-unknown	2490	-	-	-	-
Switch (combined)	-	6533	6533	6533	6533
Switch-static	2516	2516	2516	2516	-
track-signal-front	5761	-	-	-	-
track-sign-front	7396	7396	7396	7396	-
track-signal-back	3446	-	-	-	-
track-signal (combined)	-	9207	9207	9207	9207
pole	10858	10858	10858	-	-
platform	2482	2482	-	-	-
crossing	2885	2885	2885	2885	2885
train-car	1949	1949	1949	-	-
rail-occluder	3919	3919	3919	-	-
fence	291	291	-	-	-
buffer-stop	218	218	218	-	-
person	234	234	234	234	234
car	172	172	-	-	-
person-group	62	62	62	62	62
truck	11	11	-	-	-

Table A.1: Comparison of instances in different versions of RailSem19 dataset

A.1.2 Results: one-stage pre-trained with RailSem19

RS_v0:

Class	P	R	F1	mAP50	mAP95
All-class	0.562	0.581	0.571	0.618	0.337
Pedestrian	0.558	0.813	0.662	0.816	0.429
Worker	0.833	0.767	0.799	0.818	0.467
Warthog	0.749	1.000	0.856	0.992	0.842
Signal	0.411	0.364	0.386	0.437	0.141
Switch	0.427	0.294	0.348	0.305	0.064
Ped-cross	0.393	0.250	0.306	0.342	0.0785
speed (ms/image)	2.2				

Table A.2: Results: RS_v0 one-stage pre-trained

RS_v1:

Class	P	R	F1	mAP50	mAP95
All-class	0.630	0.588	0.608	0.611	0.333
Pedestrian	0.846	0.857	0.851	0.934	0.485
Worker	0.739	0.656	0.695	0.775	0.375
Warthog	0.867	1.000	0.929	0.992	0.800
Signal	0.304	0.364	0.331	0.297	0.104
Switch	0.417	0.235	0.301	0.235	0.0545
Ped-cross	0.610	0.417	0.495	0.435	0.183
speed (ms/image)	2.1				

Table A.3: Results: RS_v1 one-stage pre-trained

RS_v2:

Class	P	R	F1	mAP50	mAP95
All-class	0.691	0.568	0.623	0.607	0.325
Pedestrian	0.922	0.841	0.880	0.878	0.477
Worker	0.861	0.769	0.812	0.765	0.366
Warthog	0.854	1.000	0.921	0.988	0.818
Signal	0.546	0.273	0.364	0.342	0.079
Switch	0.314	0.189	0.236	0.247	0.0529
Ped-cross	0.649	0.333	0.440	0.425	0.158
speed (ms/image)	2.1				

Table A.4: Results: RS_v2 one-stage pre-trained

RS_v3:

Class	P	R	F1	mAP50	mAP95
All-class	0.670	0.483	0.561	0.552	0.300
Pedestrian	0.825	0.571	0.675	0.760	0.397
Worker	1.000	0.569	0.725	0.814	0.386
Warthog	0.906	1.000	0.951	0.995	0.799
Signal	0.393	0.364	0.378	0.323	0.107
Switch	0.192	0.0588	0.090	0.095	0.0191
Ped-cross	0.702	0.333	0.452	0.326	0.0918
speed (ms/image)	2.1				

Table A.5: Results: RS_v3 one-stage pre-trained

RS_v4:

Class	P	R	F1	mAP50	mAP95
All-class	0.624	0.551	0.585	0.546	0.278
Pedestrian	0.583	0.929	0.716	0.894	0.436
Worker	0.595	0.385	0.468	0.481	0.170
Warthog	0.932	1.000	0.965	0.995	0.780
Signal	0.570	0.364	0.444	0.338	0.115
Switch	0.449	0.294	0.355	0.215	0.0549
Ped-cross	0.613	0.333	0.432	0.352	0.112
speed (ms/image)	2.1				

Table A.6: Results: RS_v4 one-stage pre-trained

A.1.3 Results: two-stage pre-trained with RailSem19

RS_v0:

Class	P	R	F1	mAP50	mAP95
All-class	0.629	0.624	0.626	0.652	0.394
Pedestrian	0.814	0.936	0.871	0.983	0.598
Worker	1.000	0.815	0.898	0.910	0.566
Warthog	0.830	1.000	0.907	0.995	0.842
Signal	0.376	0.364	0.370	0.337	0.119
Switch	0.387	0.294	0.334	0.300	0.0714
Ped-cross	0.368	0.333	0.350	0.387	0.169
speed (ms/image)	2.1				

Table A.7: Results: RS_v0 two-stage pre-trained

RS_v1:

Class	P	R	F1	mAP50	mAP95
All-class	0.736	0.618	0.672	0.657	0.388
Pedestrian	0.805	1.000	0.892	0.990	0.630
Worker	1.000	0.708	0.829	0.819	0.442
Warthog	0.866	1.000	0.928	0.995	0.861
Signal	0.797	0.358	0.494	0.446	0.162
Switch	0.458	0.235	0.311	0.313	0.0715
Ped-cross	0.493	0.406	0.445	0.377	0.162
speed (ms/image)	2.2				

Table A.8: Results: RS_v1 two-stage pre-trained

RS_v2:

Class	P	R	F1	mAP50	mAP95
All-class	0.682	0.606	0.642	0.624	0.372
Pedestrian	0.884	0.929	0.906	0.979	0.587
Worker	1.000	0.677	0.807	0.810	0.461
Warthog	0.900	1.000	0.947	0.990	0.858
Signal	0.611	0.455	0.522	0.442	0.115
Switch	0.206	0.176	0.190	0.217	0.051
Ped-cross	0.490	0.401	0.441	0.307	0.160
speed (ms/image)	2.2				

Table A.9: Results: RS_v2 two-stage pre-trained

RS_v3:

Class	P	R	F1	mAP50	mAP95
All-class	0.702	0.565	0.626	0.615	0.377
Pedestrian	0.917	1.000	0.957	0.990	0.612
Worker	1.000	0.691	0.817	0.859	0.474
Warthog	0.935	1.000	0.966	0.995	0.844
Signal	0.484	0.273	0.349	0.385	0.128
Switch	0.401	0.176	0.245	0.136	0.0258
Ped-cross	0.475	0.250	0.328	0.327	0.180
speed (ms/image)	2.2				

Table A.10: Results: RS_v3 two-stage pre-trained

RS_v4:

Class	P	R	F1	mAP50	mAP95
All-class	0.703	0.624	0.661	0.638	0.357
Pedestrian	0.633	1.000	0.775	0.960	0.550
Worker	0.947	0.615	0.746	0.795	0.409
Warthog	0.870	1.000	0.930	0.995	0.831
Signal	0.526	0.364	0.430	0.342	0.116
Switch	0.529	0.353	0.423	0.377	0.0649
Ped-cross	0.713	0.414	0.524	0.358	0.173
speed (ms/image)	2.1				

Table A.11: Results: RS_v4 two-stage pre-trained

A.1.4 Best performing dataset for each class

Class	Results (F1 / mAP50)	Dataset and Pre-training
Overall	0.672 / 0.657	RS19_v1, two-stage
Pedestrian	0.957 / 0.990	RS19_v3, two-stage
Worker	0.898 / 0.910	RS19_v0, two-stage
Warthog	0.966 / 0.995	RS19_v4, one-stage; RS19_v0, RS19_v1, RS19_v3, RS19_v4, two-stage
Signal	0.494 / 0.446	RS19_v1, two-stage
Switch	0.423 / 0.377	RS19_v4, two-stage
Pedestrian Crossing	0.524 / 0.435	RS19_v4, two-stage; RS19_v1, one-stage

Table A.12: Best overall performance metrics and corresponding dataset versions