AV Sensor Architectures for V2V Crash Reconstruction and Prediction

Mohammad Mahfuzul Haque

A thesis submitted to Auckland University of Technology

in fulfilment of the requirements for the degree of Doctor of Philosophy (PhD)

February 2025

School of Engineering, Computer and Mathematical Sciences

Abstract

The major challenge in Autonomous Vehicle (AV) crash reconstruction and prediction is the need for appropriate AV sensor data to accurately reconstruct a range of crash events and predict potential crashes across diverse driving scenarios. This thesis aims to develop methods for collecting such data for effective evidence-based crash reconstruction and prediction to address these limitations. This research introduces a novel Simulation Method for Performance Evaluation (SMTPE) to select a sensor architecture for collecting relevant AV sensor data. The crash data collected using the chosen architecture are utilised for crash reconstruction and, subsequently, for prediction with a proposed Vehicle Crash Reconstruction and Prediction model (VCRPM). By delving into pre-crash and crash scenarios through simulated Vehicle-to-Vehicle (V2V) crash events, this research offers a solution to time and resource constraints, pushing the boundaries of what is possible in the pursuit of safer, more intelligent AVs.

The findings indicate that AVs can effectively reconstruct crashes and predict potential accidents in real time by integrating data from Radio Detection and Ranging (RADAR), cameras, and light detection and ranging (LIDAR) sensors, utilising data fusion and machine learning techniques. This research presents a functional sensor architecture for AV crash studies, a new approach for crash reconstruction, and three ensemble-based machine learning models for real-time crash prediction.

***Keywords*—**Sensor architecture, sensor fusion, AV sensor data, V2V, crash reconstruction, crash events, crash prediction, machine learning, VCRPM.

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgements), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning. I also declare that AI tools, such as Grammarly Premium and ChatGPT 4o Mini, have been used solely to improve English grammar and sentence structure and not to generate any thesis content, including artwork or artefacts.

Signature of candidate: Date: 25/02/2025

Acknowledgment

I would like to express my deepest gratitude to everyone who supported me throughout my PhD journey. Their assistance and dedicated involvement have helped to accomplish this thesis.

I would like to thank my primary supervisor, Professor Ajit Narayanan, for his consistent support and motivation in my journey of PhD study. Especially during and after the COVID pandemic, he has provided continuous academic support with his heartiest empathy. His excellent supervision helped me to accomplish my research goals and kept me moving forward. He has been a great source of inspiration and guidance in my academic journey.

My special thanks go to my other supervisor, Dr. Akbar Ghobakhlou, for his intense supervision and inspiration. I am always grateful to him for his valuable support and for reviewing my thesis writing.

I extend my thanks to the authority of the ICT Division, Ministry of Post Telecommunication and Information Technology, Bangladesh, for the Higher Education and Research in ICT grant and the APC fund.

I am grateful to Professor Ajit Narayanan and Dr. Akbar Ghobakhlou for their contribution to the article processing charge of a paper in the Sensors journal. Moreover, I am thankful to the AUT for providing me with an excellent educational platform and research facilities, including the library, laboratories, and simulation software, which have played a crucial role in this thesis.

I am very much thankful to my friends and family members for their inspiration in my PhD journey. Lastly, and most importantly, I would like to give special thanks to my wife, Abida Sultana, for her patience, love, support, and encouragement throughout.

Table of Contents

List of Abbreviations and Acronyms

AI	Artificial Intelligence
ADS	Advance Driving Systems
ADAS	Advanced Driver Assistance Systems
ANN	Artificial Neural Networks
AV	Autonomous Vehicle
AVs	Autonomous Vehicles
AV-CRPRM	AV Crash Reconstruction and Prediction Research Method
CA-DMV	Californian Department of Motor Vehicles
CIREN	Crash Injury Research Engineering Network
CNN	Convolutional Neural Network
C-OSPA	Combined Optimal Sub-Pattern Assignment
DSRM	Design Science Research Methodology
EDR	Event Data Recorder
EDRs	Event Data Recorders
GCPs	Ground Control Points
GNSS	Global Navigation Satellite System
GOSPA	Generalised Optimal Sub-Pattern Assignment
GBM	Gradient Boosting Machines
HDV	Human-Driven Vehicle
JPDA	Joint Probabilistic Data Association
KNN	K-Nearest Neighbour
LIDAR	Light Detection and Ranging
MIO	Most Important Object
ML	Machine Learning
MLP	Multi-Layer Perceptron
MMW	Millimetre-Wave
MTT	Multi-Target Tracking
NHTSA	National Highway Traffic Safety Administration

NN	Neural Networks
OSPA	Optimal Sub-Pattern Assignment
PHD	Probability Hypothesis Density
RADAR	Radio Detection and Ranging
RF	Random Forest
RM	Research Methodology
ROC	Receiver Operating Characteristics
RQ	Research Question
SBB	Smart Black Box
SHAP	SHapley Additive exPlanations
SMOTE	Synthetic Minority Over-sampling Technique
SMTPE	Simulation Method for Tracking Performance Evaluation
SSM	Surrogate Safety Measures
SVM	Support Vector Machine
TA	Tracking Architecture
TTC	Time-To-Collision
TTE	Time-To-Escape
UAV	Unmanned Aerial Vehicle
V2V	Vehicle-To-Vehicle
VCRM	Vehicle Crash Reconstruction Method
VCRPM	Vehicle Crash Reconstruction and Prediction Model
WHO	World Health Organisation
XGBoost	Extreme Gradient Boosting

List of Figures

List of Tables

Chapter 1 Introduction

This chapter begins by providing the background of research on AV crash reconstruction and prediction using AV sensor data. It then presents the motivation behind this research, followed by a discussion of its rationale. The chapter highlights the main significance of the research before detailing the research questions and methodology. Finally, it concludes with an overview of the contributions of this research and the overall structure of the thesis.

1.1 Research Background

The adoption of Autonomous Vehicles (AVs) is increasingly reliant on Advanced Driving Systems (ADS), which offer a range of comfort features and artificial intelligence-based services. The development of AVs has advanced significantly, particularly with the introduction of Advanced Driver Assistance Systems (ADAS), which enable level 2 autonomous driving (partial automation with an attentive driver). AV technology has progressed to level 3, where drivers can take their eyes off the road in highly automated driving conditions. As ADS technology improves, AVs are advancing toward level 4, where the driver can focus on tasks other than driving. Eventually, at level 5, full automation will take over all driving functions (Domova et al., 2024; Moradloo et al., 2024; Nurliyana et al., 2023; Qayyum et al., 2020; SAE, 2021; Wang et al., 2020a).

Despite significant advancements in ADS, accidents involving AVs still occur, with some crashes that AVs are unable to avoid. The reasons for these crashes are being actively researched by analysing existing crash datasets (P. Liu et al., 2024; Q. Liu et al., 2024; Saravanarajan et al., 2024; Zhou et al., 2023). One such dataset, the autonomous vehicle collision reports, had collected 760 AV crash reports by November 2024 (DMV, 2024). When AV crashes happen, the data collection methods from the vehicle's sensors must be more specialised than traditional digital forensics (Feng et al., 2019). Accurate crash investigations, critical for determining liability, are often hindered by current methods that do not fully incorporate the digital data available from AV sensors (Répás et al., 2022). While Event Data Recorders (EDRs) provide crash data from vehicles, the data extracted is often of lower quality than the rich data generated by AV sensors, making the latter a more reliable source for crash reconstruction (Beck et al., 2023).

Recent research has made significant advancements in road safety by utilising sensor data and Artificial Intelligence (AI) techniques, such as Machine Learning (ML), to enhance safety models (Sohail et al., 2023). For instance, crash databases are

increasingly being employed to develop real-time crash prediction models, which depend on accurate pre-crash information. However, the lack of appropriate crash data can limit the effectiveness of existing crash reports, potentially leading to less reliable road safety initiatives (Imprialou & Quddus, 2019).

The use of in-vehicle sensors to monitor the surroundings and vehicle context has gained increasing attention in the past few years to ensure the safety of AVs. Data from these sensors aid in various autonomous driving functions, including object detection (Zhao et al., 2020), adaptive cruise control (Wei et al., 2024), decision-making (Ignatious et al., 2022; Modas et al., 2020), and monitoring the driver's behaviour and health (Qiu et al., 2018; Zeng et al., 2022). These sensor data are also invaluable for reconstructing crash events (Alexakos et al., 2021). Crash analysts use such data to examine vehicle behaviour during an accident, such as changes in the direction and speed over time (Alim et al., 2023).

When developing sensor architectures for Vehicle-to-Vehicle (V2V) crash reconstruction and prediction, it is important to consider several key elements contributing to success. One of the key elements is selecting appropriate sensor types and defining their roles in data collection, enhancing overall effectiveness (Pour et al., 2022). A second essential element is the seamless integration and fusion of sensor data, which is vital for achieving the accuracy and reliability necessary for meaningful insights (Li et al., 2020; Zazuli et al., 2024). Another important element is simulating various crash scenarios for thorough testing and evaluation (Ren et al., 2023; Y. Zhang et al., 2024), which will strengthen a sensor architecture. It is also noteworthy that incorporating redundancy among multiple sensors can significantly improve data reliability (Q. Liu et al., 2021). Implementing real-time data processing capabilities will facilitate timely interventions and enhance V2V crash prediction efforts (G. Zhang et al., 2024; Zheng & Sayed, 2020).

The availability of various sensor types for fusion, along with tracking algorithms and architectures, has opened up multiple avenues for AV development (Choi et al.,

2021; Hou et al., 2023; Shah et al., 2022; Wang et al., 2023; Zhang et al., 2023). However, there needs to be more literature concerning multi-sensor fusion for AV crash reconstruction, especially compared to the research on individual sensor architectures. This gap is particularly significant for forensic and investigative purposes in reconstructing AV crashes, as it is crucial to ensure that all relevant information is captured and that nothing important is overlooked (Haque et al., 2024).

Vehicle crash reconstruction is important for understanding the causes of accidents. This process often utilises Three-Dimensional (3D) models to visualise events and provide crucial data for legal proceedings (Law, 2024). These reconstructions are valuable for accident scientists, courts, and other stakeholders, particularly when pre-crash data are available for analysis and decision-making (Society, 2022).

Simulation-based testing is increasingly used to evaluate AV safety features (Zhou et al., 2024), especially in high-risk scenarios, as it offers a cost-effective and safe alternative to road testing (Arcaini et al., 2022). For instance, real-life AV accident reconstruction models have been developed to improve traffic safety and reduce accidents (Lengyel et al., 2023; Parseh & Asplund, 2022; Ren et al., 2023). Simulation-based crash analysis enhances the understanding of contributing factors and helps improve ADS safety. However, the potential of AV sensor data for crash investigation has yet to be explored (Beck et al., 2023).

AVs rely on sensor-based object detection for autonomous driving and collision avoidance. The limitations of single-sensor detection can be overcome by fusing multiple sensors, such as Radio Detection and Ranging (radar), cameras, and Light Detection and Ranging (LIDAR), to improve the accuracy and reliability of detection (Xiang et al., 2023; Yeong et al., 2021).

Real-time vehicle crash prediction relies on the classification of crash events (i.e., whether they are crashes or non-crashes) to train machine learning models (Ali et al., 2024). The differentiation between crash and non-crash occurrences can be achieved

through crash detection using in-vehicle sensor data (M. I. B. Ahmed et al., 2023; Kumar et al., 2021; Sohail et al., 2023). Developing an accurate crash detection model is essential for understanding crashes and their contributing factors (Ali et al., 2024).

Accurate crash detection, based on in-vehicle sensor data, enables rapid response by accident rescue teams and helps traffic authorities mitigate the consequences of crashes (Huang et al., 2020). Identifying contributing factors in crash detections is crucial for AV manufacturers to reduce crash severity in future models (Zhu et al., 2022).

Machine learning-based real-time crash detection heavily relies on the identification of near-crash events. Detecting near-crash or pre-crash events, such as potential conflicts between V2V, requires continuous monitoring of the driving environment using in-vehicle sensors (Ke et al., 2023).

1.2 Rationale, Motivation and Significance of the Study

Road traffic deaths and injuries have long been major global health and development challenges. Despite various safety initiatives aimed at reducing fatalities, progress in decreasing road traffic deaths has been slow. A recent report highlighted a modest 5% global reduction in road traffic deaths over the past decade. In 2010, the number of road traffic deaths was 1.25 million, which decreased to 1.19 million by 2021. However, the World Health Organisation (WHO) urged to achieve a target of a 50% reduction in road traffic deaths by 2030 (WHO, 2023).

Advancements in AVs and their sensor technology are helping improve driving safety, but the collection of accurate crash data using AV sensors remains a challenge, especially in the aftermath of accidents (Haque et al., 2024). Proper crash data, including pre-crash information, are crucial for accurate crash reconstruction and effective real-time crash prediction. Current crash reports often need more critical information, undermining the credibility of road safety initiatives (Imprialou & Quddus, 2019). AV sensors, however, have the potential to produce higher-quality data

than EDRs for crash reconstruction (Beck et al., 2023). Still, there is a need for a structured framework to test different sensor architectures rather than relying on arbitrary sensor fusion-based tracking configurations (Haque et al., 2024). The growing demand for publicly available AV crash datasets (Chen et al., 2024) highlights the challenge for ML-based crash prediction, as the lack of real-world AV accident data hampers safety improvements and crash analysis (Abdel-Aty & Ding, 2024). The primary challenge for ML in this context is the need for suitable AV sensor data to accurately reconstruct various crash scenarios, thereby enabling effective prediction of AV crashes.

It is important to develop a mechanism that can provide appropriate AV sensor data for crash reconstruction and prediction to fill the existing gap in this research area. Current AV sensor technology generates large volumes of data, and the need to collect these data through a functional sensor architecture for accurate crash reconstruction and prediction is the driving motivation behind this research. The goal of reducing road traffic deaths (WHO, 2023) further inspired the development of ML models for real-time crash prediction using simulated AV sensor data, allowing for timely intervention and crash avoidance measures. This study introduces a novel Vehicle Crash Reconstruction and Prediction Model (VCRPM) in this context. The significance of this model lies in its ability to analyse hundreds of simulated driving scenarios to gather relevant AV sensor data with pre-crash information from various crash types, including front, head-on, rear-end, and side-impact collisions. The proposed model effectively uses AV sensor data for evidence-based, data-driven crash reconstruction and prediction, contributing to crash forensics and prevention advances.

1.3 Research Questions

AV sensors generate high-quality data that can be used for accurate crash reconstruction (Haque et al., 2024). These sensor data, along with pre-crash information, can be used to train machine learning models for crash prediction. Although AV sensors produce vast amounts of data, it is important to collect precise and reliable evidence-based crash

data using a functional sensor architecture. This approach will support data-driven crash reconstruction and prediction. This thesis presents the following research questions in light of the identified research problem described in the previous section.

Research Question 1 (RQ1):

"What simulation method can be developed to evaluate architectures using multi-sensor fusion for AV sensor data collection?"

Research Question 2 (RQ2):

"How accurately can pre-crash and crash events be reconstructed using simulated AV sensor Data?"

Research Question 3 (RQ3):

"What machine learning models can be developed to predict real-time crashes using simulated AV sensor data?"

1.4 Methodology for Investigation

This thesis develops a model to extract relevant AV sensor data through evidence-based crash simulations. The goal is to enhance the processes of data-driven crash reconstruction and prediction. To achieve this, the primary objective of the thesis is to gather and analyse AV sensor data generated from a variety of simulated driving scenarios. These scenarios encompass different types of collisions, including front-end collisions, head-on crashes, rear-end impacts, and side-impact incidents.

In support of this objective, several key elements have been incorporated into the research framework. These include the selection of specific simulation parameters, the integration of diverse AV sensor technologies, and the documentation of crash dynamics under various conditions. By focusing on these elements, the study aims to create a dataset capable of providing insights into the intricacies of crash phenomena, ultimately contributing to more effective crash reconstruction techniques and crash

predictive models in the field of automotive safety. The following research objectives have been included to support the primary objective:

- Develop a simulation method to select a workable sensor architecture for AV sensor data preparation.
- Develop a mechanism to reconstruct evidence-based AV crashes accurately using simulated AV sensor data.
- Develop data-driven machine learning models to predict AV crashes effectively.

This research reviews existing literature focused on vehicle crashes to formulate a robust Research Methodology (RM) to address specific research objectives. The review encompasses a variety of studies examining important aspects related to vehicle crashes. For example, (Alrejjal et al., 2022) investigated the many factors contributing to vehicle crashes, shedding light on the complexities involved. In another study (Song et al., 2021), sequenced the occurrences of autonomous vehicle crashes to enhance safety testing protocols. In (Ashraf et al., 2021), decision tree analysis was employed to extract and identify contributing factors associated with AV crashes. Furthermore, (Stark et al., 2020) explored methodologies designed to validate the decision-making processes of AVs by utilising crash data, ensuring that safety measures can be effectively evaluated. (Sinha et al., 2021) took this further by investigating AV crashes to develop a crash injury model that could predict outcomes and inform safety improvements.

Despite the valuable insights provided by these studies, this research highlights a significant gap: the need for a methodology capable of tackling challenges related to sensor architecture and data collection techniques for both V2V crash reconstruction and real-time crash prediction. The rapid advancement of AV technology contributes to minimising this gap, underscoring the importance of addressing these evolving challenges. Therefore, the development of the research methodology for this thesis is grounded in a careful analysis of existing designs that have aimed to confront vehicle crash issues alongside established research methodologies from related fields that have

successfully addressed similar challenges.

(Kothari, 2004) described the research methodology, methods, and techniques and defined research as a search for knowledge to find a solution to a specific problem or to deliver a theory in a simplified way. The author classified the basic types of research as 1) descriptive vs analytical, 2) applied vs fundamental, 3) quantitative vs qualitative, 4) conceptual vs empirical, and 5) some other types of research such as one-time research or longitudinal research, field setting research or laboratory research or simulation research, and so on. Kothari also generalised all these research approaches as either quantitative or qualitative. According to the same author, the quantitative research approach was strict to strenuous quantitative analysis with quantitative data, and it could be categorised into an inferential, experimental, or simulation approach. On the other hand, a qualitative approach reflected the researcher's insights on an independent evaluation of attitudes, opinions, and behaviour. The author also defined an RM as systematically solving the research problem. Research method or technique involved performing research operations by analysing data from available sources or by creation.

The Design Science Research Methodology (DSRM), articulated in (Peffers et al., 2007), was a well-known RM for model design, implementation, and dissemination of solutions that had been rigorously evaluated. This methodology was notable for its iterative and quantitative approach, which unfolded through six distinct yet interconnected steps: first, it involved identifying the specific research problem and motivating factors; second, it required a clear definition of research objectives; third, it encompassed the design and development phases of crafting a viable solution; fourth, it entailed demonstrating the effectiveness of the developed solution in practical scenarios; fifth, it necessitated a thorough evaluation of the outcomes and impacts; and finally, it included the crucial step of publishing research findings to share insights and advancements with the broader community.

In this research endeavour, the DSRM framework is adopted to devise a systematic

method characterised by dynamism and quantitative rigour. This thesis proposes an innovative Autonomous Vehicle Crash Reconstruction and Prediction Research Method (AV-CRPRM) to achieve the research objectives. The structural framework and key components of the AV-CRPRM are illustrated in Figure 1.1, providing a visual representation of its methodology and anticipated impact in the field.

The AV-CRPRM is designed with three fundamental components: input, process, and output. At the heart of the AV-CRPRM lies the process component, characterised by a dynamic, iterative flow that enhances its effectiveness. Each iteration is comprised of five distinct stages: planning, construction, testing, analysis, and reconstruction.

The research method employed within this framework is systematic, involving continuous development, rigorous testing, affirmation of results, and revisitation of previous steps. This cycle is crucial for achieving the research objectives, as it facilitates both the construction and reconstruction phases effectively.

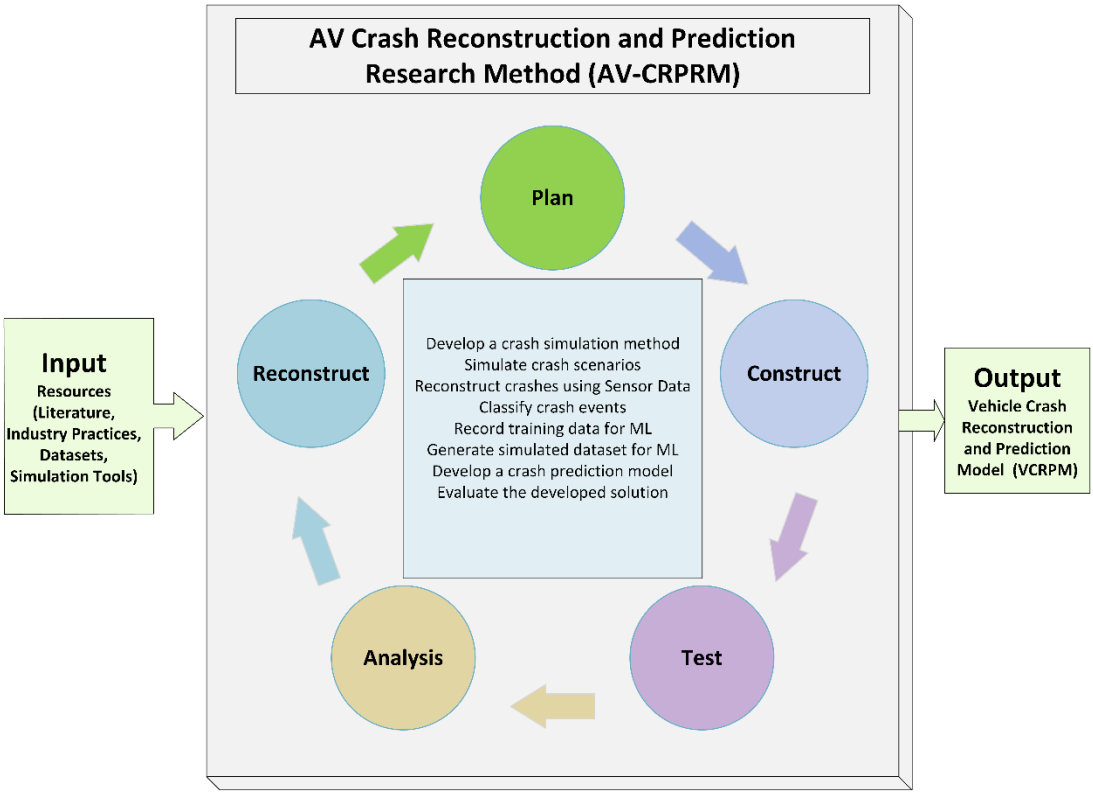


Figure 1.1: Block diagram of the developed research method (AV-CRPRM).

One of the notable features of this research method is its adaptability. Each iteration incorporates various inputs and encompasses a range of sub-processes, allowing it to accommodate different parameters as needed. This flexibility is important for executing diverse sub-processes tailored to specific aspects of the research.

As a result of its iterative nature, the method supports numerous cycles of development, which s enhances the precision of crash reconstructions and improves the detection of crash events. In doing so, it fulfils the aims and objectives of the research, paving the way for more accurate and reliable outcomes in the field of accident analysis.

1.5 Research Contributions

The automotive industry is poised for a significant shift as vehicles increasingly move towards a ‘driverless’ future, driven by remarkable advancements in sensor and control

technologies. These developments in AVs are groundbreaking and hold the promise of transforming transportation. However, a need exists to investigate and refine methodologies for accurately capturing crash data from various types of accidents, utilising the sensors integrated into AVs, which is important for effective crash reconstruction and prediction.

This research endeavours to tackle this challenge by implementing an advanced sensor architecture (Haque et al., 2024). Through empirical experimentation, this approach aims to harness data from AV sensors to enhance the processes of crash reconstruction and predictive analysis. By emphasising important elements in the design and development of AV sensor architectures for vehicle-to-vehicle crash reconstruction and prediction, this thesis offers several original contributions that seek to address the complexities and issues identified in crash data collection and utilisation. Below are three main contributions, providing an overview of the advancements and insights gained through this research.

1. The most important contribution of this thesis is the development of an innovative Vehicle Crash Reconstruction and Prediction Model (VCRPM). This model is based on evidence-derived simulated autonomous vehicle sensor data collected using a designed sensor architecture, as discussed in Chapter 4. A comprehensive evaluation of the model's performance has been conducted, and the findings indicate that it can accurately reconstruct vehicle crashes and predict future incidents with exceptional precision, as detailed in Chapter 5, and Chapter 6.

2. This thesis proposed a new method called Simulation Method for Tracking Performance Evaluation (SMTPE) to help select the best tracking system for analysing crashes involving AVs (Haque et al., 2024). The research reveals that a centralised multi-sensor-based vehicle tracking architecture, specifically sensor setup two combined with tracking architecture one, is the most effective configuration for reconstructing crashes involving AVs. The evaluation results

demonstrate that this setup can accurately reconstruct crashes using sensor data collected under ideal conditions. The findings also demonstrate that this sensor architecture included three key elements considered during the architecture design: the abilities of different sensors, sensor fusion, and data reliability using multi-sensors. It has been noted that developed sensor architecture can provide accurate crash data without extra road infrastructure or outside sensor data for reconstruction, making AV crash analysis more feasible in areas without such infrastructure (Chapter 4).

3. A key contribution of this research is the successful classification of crash events for training machine learning models using sensor data from autonomous vehicles collected in hundreds of simulated driving scenarios (Chapter 5). The study effectively integrates the two important elements of the sensor architecture by simulating crashes and processing data in real-time. Sixteen ML models have trained on a simulated crash dataset, among which three ensemble models—bagged trees, subspace K-nearest neighbours, and RUSBoosted trees—achieved perfect predictions of crash events. The evaluation results noted the potential of ML to predict crashes in real time using sensor data, facilitating the implementation of effective avoidance measures (Chapter 6).

This thesis makes several important contributions, outlined as follows:

- **Development of AV-CRPRM**: This thesis introduces the AV-CRPRM (Figure 1.1), a novel research method for AV crash reconstruction and prediction. This method can be adopted to enhance AV technology, especially for improving safety and advancing AV crash forensics research.

- **Pre-Crash and Crash Event Classification**: A mechanism for classifying pre-crash and crash events has been developed. The pre-crash includes categorising cut-in, conflict, and potential crash events using a new Most Important Object (MIO) detection algorithm (Chapter 3).This classification approach holds the potential for enhancing collision avoidance systems and driving safety, a key

aspect of AV safety improvement.

- **Evaluation of Multi-Tracking Architectures**: This research evaluates the performance of various multi-tracking architectures, revealing that a centralised tracking architecture offers the most accurate tracking results. This insight contributes to AV crash reconstruction and prediction and can be applied to improve trajectory planning and control in AV development (Chapter 4).

- **Guidelines for Tracking and Crash Detection**: This thesis provides recommendations for assessing tracking performance, outlines best practices for crash reconstruction, and suggests effective methods for detecting crash events. These guidelines are intended to support both academia and the AV industry in enhancing crash analysis and safety (Chapter 4).

- **Areas for Future Research**: This thesis identifies key areas for further exploration based on simulated crash reconstruction experiments. For instance, improving the MIO detection mechanism through 360-degree sensor fusion could enhance AV safety, planning, and control (Chapter 4).

- **Crash Dataset Generation**: The proposed Vehicle Crash Reconstruction Method (VCRM) method successfully collects data from AV sensors and generates a crash dataset, as discussed in Chapter 5. This dataset can be reused for further analysis and prediction of crashes, representing a novel approach to utilising AV sensor data for real-time crash prediction. This innovation opens up new avenues for research in AV safety, as highlighted in Chapter 6.

- **VCRPM for Real-Time Crash Prediction**: The Vehicle Crash Reconstruction and Prediction Model (VCRPM) gathers crash events and kinematic data from live driving using available AV sensors. This model can be applied to any road, helping to overcome the challenge of predicting crashes in real time for specific locations or areas (Chapter 5).

- **Crash Warning Dashboard**: The developed VCRPM includes a crash warning dashboard that displays real-time predictions of potential crashes, as well as detections of pre-crash and crash events. A key component of the dashboard is the crash prediction lamp, which can provide a warning up to five seconds before a possible crash occurs. The lamp employs a colour-coded system: green indicates low risk, blue and yellow signal medium risk, orange denotes high risk, and red represents very high risk. This system enhances safety for AV driving (Chapter 6).

- **Recommendations for Future Research**: This thesis also offers brief recommendations that could aid future research into real-time crash prediction (Chapter 6).

In summary, this research contributes to more accurate crash reconstruction, potentially leading to better attribution of responsibility and causes. The real-time crash prediction models, along with the crash warning dashboard, could enhance vehicle and road safety and improve driving instruction by offering deeper insights into crash avoidance in real-time.

1.6 Structure of This Thesis

The overall structure of the thesis is presented in Figure 1.2 and described as follows:

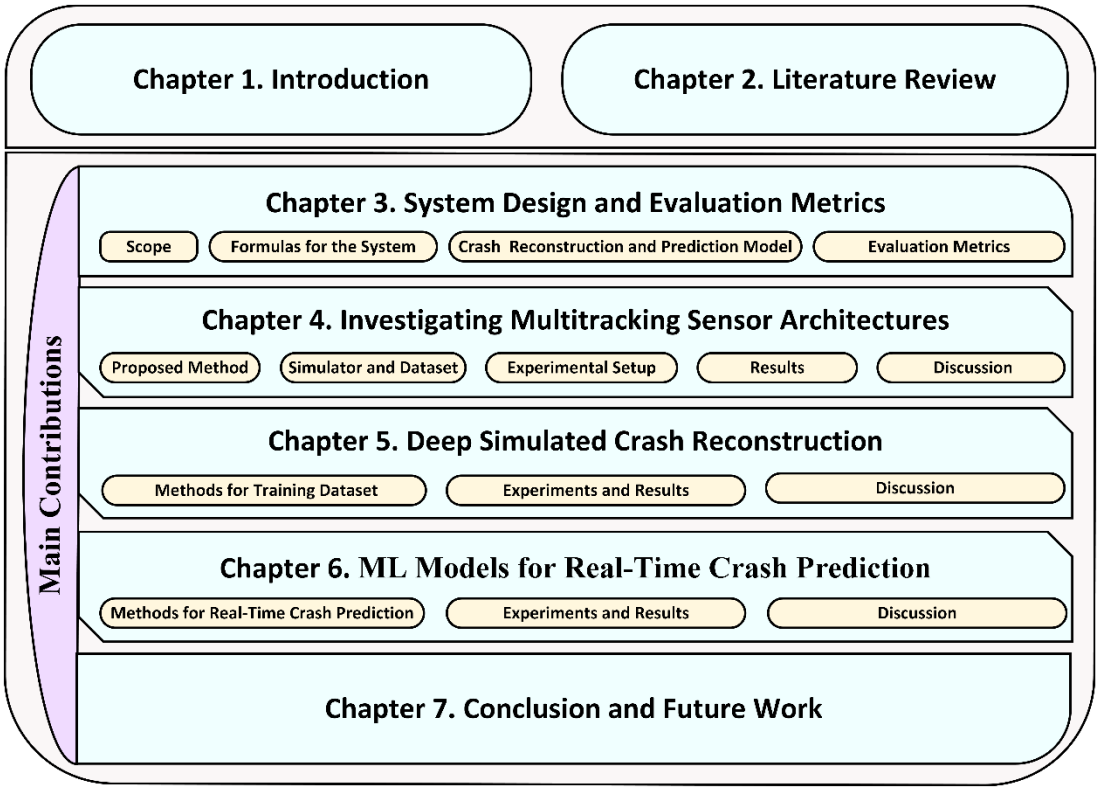


Figure 1.2: The structure of this thesis.

Chapter 2 offers a detailed exploration of research on AV crashes, providing insights into multiple aspects of the topic. It discusses current practices and provides an understanding of AV sensors used for object detection, vehicle tracking, crash reconstruction, crash event detection and classification, as well as ML-based crash prediction. This chapter establishes a foundation for the thesis and concludes with a summary of the key gaps identified in the reviewed literature.

Chapter 3, focuses on refining the thesis and provides a detailed explanation of the research methodology, including definitions and mathematical formulations. This chapter introduces new concepts for autonomous vehicle crash reconstruction and prediction through the design of the VCRPM model, establishing the foundation for simulation-based experiments. It also presents the evaluation metrics used in this research.

Chapter 4 examines multitracking sensor architectures for crash reconstruction using AV sensors. This chapter introduces a novel SMTPE to help select a suitable sensor architecture (Haque et al., 2024). It then presents a review of existing crash simulation tools and crash datasets. Following this, the experimental setup and results for multiple tracking architectures are discussed. The chapter concludes with research findings and brief guidelines. Finally, a summary is provided at the end of the chapter.

In Chapter 5, a simulation approach is introduced to accurately reconstruct data-driven AV crashes. The chapter begins with a brief introduction and then describes the methods used to prepare the training dataset. It presents evidence-based results from deep simulated experiments that classify crash events, detailing the setup of these experiments. This chapter describes the data processing steps to create a training dataset for ML using AV sensor data. Finally, a conclusion is drawn after evaluating the classification of crash events.

Chapter 6 introduces methods for predicting crashes in real time using data-driven approaches. It presents experiments and evaluation results for ML models based on simulated AV crash data. The chapter outlines the outcomes of ML-based real-time crash predictions and assesses their performances. This chapter also explains the ML model using the Shapley value analysis. The chapter concludes by discussing the research findings and offering guidelines for effective data-driven real-time crash prediction.

Chapter 7 provides an overview of the entire thesis, summarising key findings, contributions, and implications. It also outlines potential areas for future research.

1.7 Publications

Journal Publication:

Haque, M. M., Ghobakhlou, A., & Narayanan, A. (2024). Multi-Tracking Sensor Architectures for Reconstructing Autonomous Vehicle Crashes: An Exploratory Study. *Sensors*, *24*(13).

Author Contributions: Conceptualization, Haque, M. M.(M.M.H.) and Narayanan, A. (A.N.); methodology, M.M.H.; software, M.M.H.; validation, M.M.H., Ghobakhlou, A. (A.G.), and A.N.; formal analysis, M.M.H.; investigation, M.M.H.; resources, M.M.H., A.G., and A.N.; data curation, M.M.H.; writing—original draft preparation, M.M.H.; writing—review and editing, M.M.H., A.G., and A.N.; visualization, M.M.H., A.G., and A.N.; supervision, A.G. and A.N.; funding acquisition, M.M.H. and A.N.

Part of the Sections – Vehicle Tracking Using Sensor Fusion (2.3) of Chapter 2, Types of Vehicle Crashes (3.2.2), Object Tracking, State Update, and Measurement Models (3.3.1), Generalised Optimal Sub-Pattern Assignment Metric (3.5.1) of Chapter 3, Introduction (4.1), Proposed Method (4.2), Experimental Setup for Multi-Sensor-Based Surround Vehicle Sensor Fusion (4.4), Results (4.5), and Discussion (4.6) of Chapter 4 are drawn from this paper.

Presentation in Workshop:

Haque, M. M., Ghobakhlou, A., & Narayanan, A. (2023). Sensor Fusion-based Crash Detection Model for Autonomous Vehicles Accident Reconstruction [Workshop abstract]. IEEE Instrumentation and Measurement (IEEE I&M) Society New Zealand Chapter workshop on Sensing, Measurement, and Instrumentation for Healthcare, Food, Agriculture, Environment, and Security, Auckland University of Technology, Auckland, New Zealand. Date: 2–3 November 2023.

Presented by: Haque, M. M. on 02 November 2023. Available Online: https://wise.aut.ac.nz/__data/assets/pdf_file/0004/831901/Workshop-Programme-Booklet-final-website.pdf , p26.

Part of the Sections – The experiments and results in Chapter 5 are based on the presented research.

Chapter 2 Literature Review

Chapter 1 introduced this thesis by outlining the motivation and rationale behind the research, followed by a concise overview of the thesis structure and its main contributions. For an in-depth understanding of the context and potential outcomes of V2V crash reconstruction and prediction using AV sensor technologies, it is important to first review the existing body of research on these topics. Previous studies have investigated various aspects of AV sensor technology's role in autonomous driving, precisely its effectiveness in crash reconstruction and prediction. This chapter establishes the foundation of the thesis by reviewing the current literature in these areas, highlighting both their strengths and limitations, and offering a basis for the present study.

2.1 Introduction

This thesis aimed to develop a model that harnesses evidence-based crash simulation to provide accurate AV sensor data, paving the way for data-driven crash reconstruction and prediction. The focus of this chapter is to establish a foundation for achieving the objectives of this thesis, beginning with a thorough review of existing research.

Section 2.2 delves into the fundamental concepts of key AV sensors used in autonomous driving, laying a groundwork for understanding radar, camera, and LIDAR technologies alongside their innovative fusion approaches, as explored in the literature.

Section 2.3 highlights the latest advancements and uncovers research gaps in the field of vehicle tracking through sensor fusion, emphasising areas that still require further exploration.

In Section 2.4, this thesis conveys the attention to evidence-based crash reconstruction, examining the detection and classification of crash events in related works. This section also identifies important research gaps, offering a detailed overview of the challenges that need addressing.

Section 2.5 analyses available AV crash data, reports, and datasets, assessing their suitability for machine learning applications. This section also sheds light on the inherent limitations of publicly available AV crash data in the context of real-time crash prediction, which impact their reliability and use.

In Section 2.6, this study reviews the commonly employed classification algorithms in previous machine learning studies, laying the groundwork for real-time, data-driven crash prediction. Finally, the chapter concludes with a summary of key insights and future directions.

This chapter, as a whole, serves as a stepping stone toward achieving the thesis's objectives, addressing gaps, and advancing the field of AV crash prediction and reconstruction.

2.2 Typical AV Sensors for Object Detection

Research in AV sensor technology continues to advance, driving the development of ADS toward the goal of fully autonomous driving. As research into fully autonomous driving progresses, various foundational technologies are contributing to its evolution, with sensor technology playing a pivotal role (Khan et al., 2022). The effectiveness of ADS perception relies on the sensors integrated into AVs. These sensors are designed to gather important data, enabling the system to sense both the internal and external environments, provide precise localisation and kinematic information, and detect stationary or moving objects in the vicinity of the vehicle (Brummelen et al., 2018; Ignatious et al., 2023; Yahyaei et al., 2022; Yeong et al., 2021). The careful selection and arrangement of these sensors for ADS design are crucial in ensuring accurate perception and object detection (Marti et al., 2019). A comprehensive sensing strategy is required, where multi-sensor fusion plays a significant role in generating rich, in-depth information to achieve highly accurate object detection (Ignatious et al., 2023; Yeong et al., 2021). It is also important to understand the various types of AV sensors and their functionalities for the effective development of AV safety features and collision avoidance technologies (Ignatious et al., 2022).

AV sensors can generally be classified into two categories: proprioceptive (or introspective) sensors and perceptual (or exteroceptive) sensors (Marti et al., 2019; Yahyaei et al., 2022; Yeong et al., 2021). Proprioceptive sensors monitor the internal state of the vehicle, gathering vehicle dynamics and kinematic information such as speed, acceleration, angular rate, and direction. These sensors include Global Navigation Satellite System (GNSS) receivers, Inertial Measurement Units (IMUs), and gyroscopes. In contrast, exteroceptive sensors focus on the external environment, collecting crucial data for object detection. These sensors include a variety of cameras, radar, and LIDAR systems. Exteroceptive sensors can further be divided into active and passive sensors. Active sensors, such as radar or LIDAR, emit energy from their location and detect objects by analysing the reflected signals, while passive sensors, like cameras, solely rely on receiving energy reflected from objects in their

surroundings. The research (Yeong et al., 2021) illustrated the application of exteroceptive sensors by reproducing a diagram from (Wendt & Cook, 2018), showing their typical installation positions on AVs, sensor coverage, and primary functionalities (Figure 2.1). Further details on these perceptual sensors are explored in the following subsections.

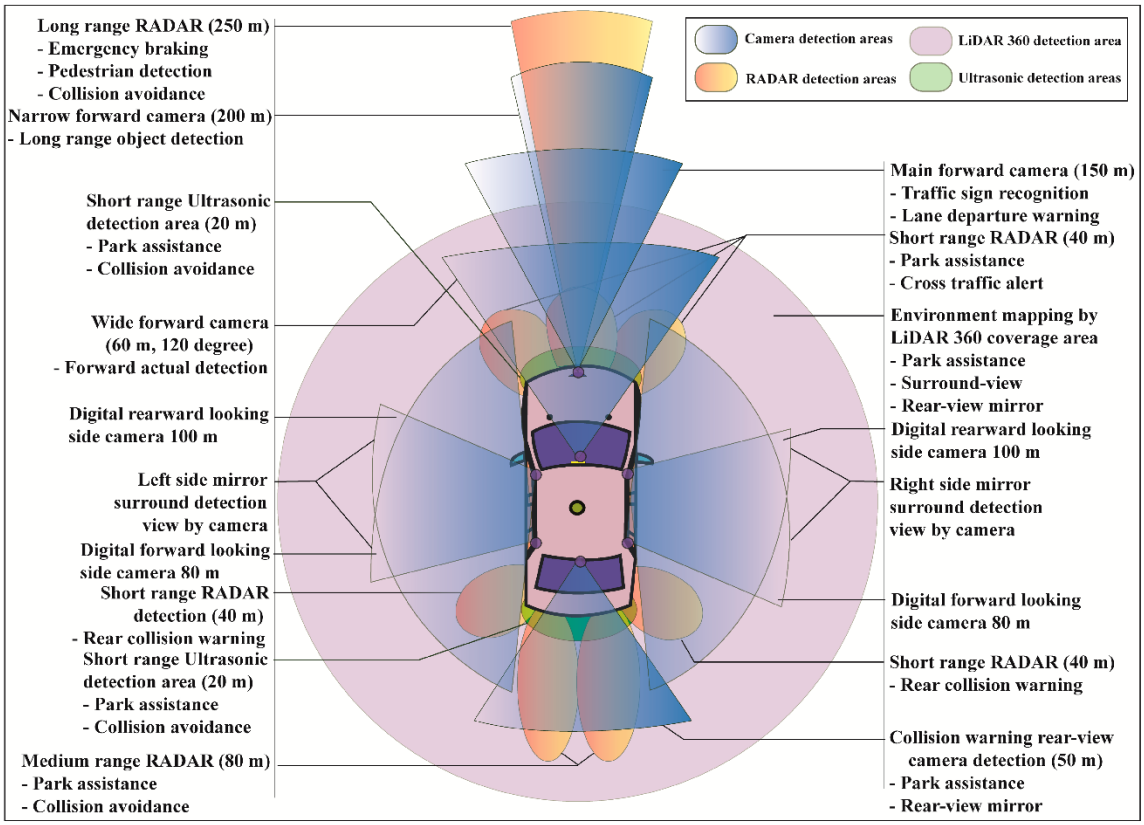


Figure 2.1: Typical AV sensors with their positions and coverages. Note: from (Hasanujjaman et al., 2023) In *Sensors.*

2.2.1 Radar

Radar sensors provide a way to gauge both the distance and velocity of objects by harnessing the power of radio waves. These sensors send out radio signals that bounce off surrounding objects, and by analysing the reflected waves, they sense the environment. The radar calculates an object's distance by measuring the time gap between the transmission and reception of the signals. Furthermore, it can assess

relative velocity by detecting changes in frequency, a phenomenon known as the Doppler effect, which captures electromagnetic wave frequency shifts between the outgoing and incoming signals. The most common type of radar used in AVs is the Millimetre-Wave (MMW) radar, operating in the 1–10mm wavelength range with frequencies spanning 76–81GHz. This Doppler effect allows for highly accurate distance and velocity measurements. While traditional radar systems can provide 3D information—offering insights into range, Doppler velocity, and azimuth angle—new advancements have introduced the Four-Dimensional (4D) radar. This cutting-edge technology incorporates an additional elevation angle, enabling even more detailed and high-resolution object detection (Yao et al., 2024). By utilising multiple transmitting and receiving antennas, 4D radar systems can deliver unparalleled accuracy in understanding the environment (Yahyaei et al., 2022).

2.2.2 Camera

A camera functions as an artificial vision sensor, making it an indispensable tool in various applications due to its affordability and versatile capabilities. Its remarkable ability to capture critical information—such as shape, size, and distance—plays a vital role in object detection and classification (Marti et al., 2019). A camera is designed to capture and process images seamlessly through its image signal processor, comprising a lens and an image sensor (Yao et al., 2024). Cameras primarily detect objects by capturing reflected light, making them adept at identifying both stationary and moving entities, such as road signs, pedestrians, and vehicles.

Within the realm of AV, three camera types dominate the landscape: monocular, stereo, and fish-eye cameras (Yeong et al., 2021). However, despite their versatility, camera sensors face challenges under low-light conditions or in environments with poor visibility, such as in adverse weather. Additionally, their performance diminishes in high dynamic range environments, like a tunnel's exit, where drastic lighting changes complicate accurate object detection (Marti et al., 2019).

2.2.3 LIDAR

As an exteroceptive and active sensor, LIDAR measures the distance to objects by calculating the round-trip time of a laser light pulse (Marti et al., 2019; Yahyaei et al., 2022). This technology enables LIDARs to capture high-density 3D data, providing detailed insights into the surrounding environment (Yusuf et al., 2024). With the collected data, LIDARs generate vivid 3D driving scenarios in the form of a point cloud (Campbell et al., 2018), offering a clear and precise representation of the surroundings.

LIDAR systems use either mechanical or solid-state light-beam steering mechanisms. Mechanical LIDARs can gather 360-degree data by rotating their laser arrangement, while solid-state LIDARs, which lack a rotating feature, require multiple units to achieve the same wide-angle view, making them a more cost-effective option for AV perception (Khader & Cherian, 2023).

What sets LIDARs apart is their ability to perceive the environment in high-resolution 3D, offering superior reconstruction of driving scenarios (MathWorks, 2024g). These capabilities are crucial for creating highly accurate HD maps, which play an important role in the AV's navigation system. Despite their remarkable strengths in many AV perception tasks, LIDARs do face specific challenges, such as reduced performance when dealing with reflective surfaces, low light conditions at dawn or dusk, and long-distance ranging (Warren, 2019).

Figure 2.2 highlights the strengths and weaknesses of radar, camera, and LIDAR technologies, offering a comparative view of depth and distance perception, velocity detection, performance under low light, adverse weather conditions, object classification, and lateral resolution. Each sensor type brings unique advantages and limitations to the table.

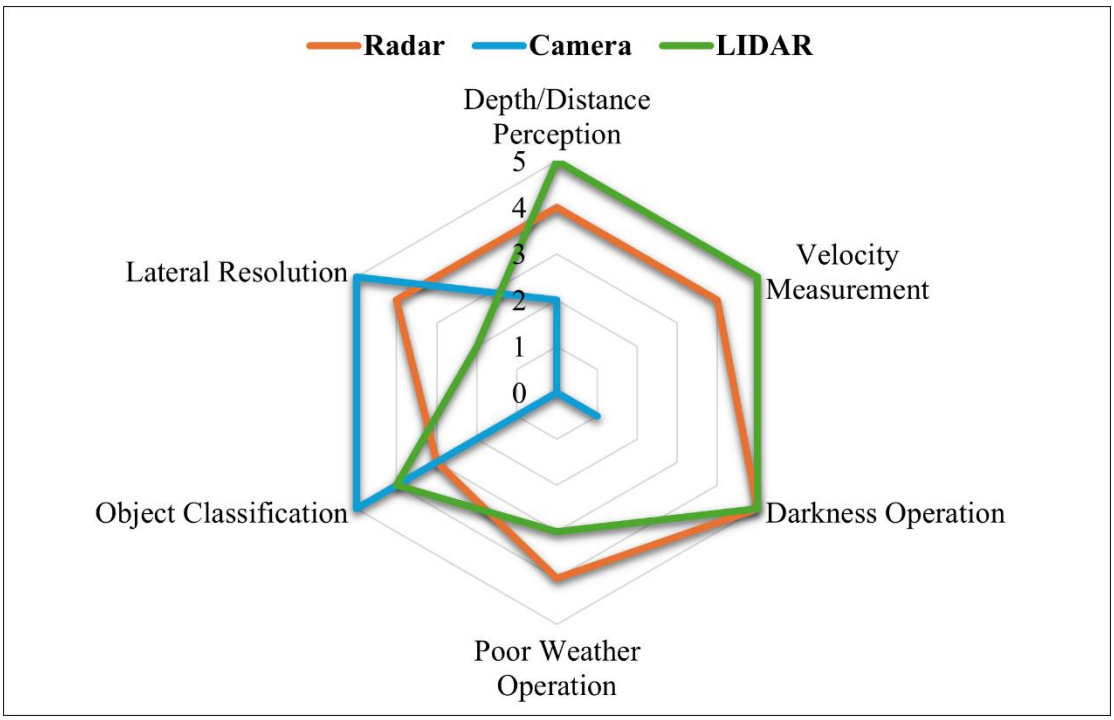


Figure 2.2: A comparison of radar, camera, and LIDAR sensors using six different features. The performance measurement value zero means not performing, and five refers to the highest performance value. Note: redrawn from (Yusuf et al., 2024) In *Transportation Research Interdisciplinary Perspectives*.

2.2.4 Sensor Fusion for Object Detection

The current AV perception mechanism has played a pivotal role in advancing the integration of multi-sensor technology for autonomous driving. By accurately sensing the vehicle's environment, mainly through real-time object detection along its trajectory, an AV's performance hinges on its perceptual accuracy. Given the current advancements in sensor technology for autonomous vehicles, the perception system can be crafted using either a single sensor or an appropriate combination of multiple sensors. Notably, the fusion of multiple sensors enhances object detection performance, far surpassing the capabilities of relying on a single sensor alone (Haque et al., 2024).

Among the various detection approaches, the 3D object detection method, powered by sensor fusion, stands out by offering more thriving and more detailed information compared to 2D detection. This 3D object detection added depth in perception is crucial

for tasks like path planning and collision avoidance. The precision of 3D object detection can be further amplified by using multi-sensor fusion (Arnold et al., 2019).

As illustrated in Figure 2.3, the unique strengths of radar, camera, and LIDAR sensors are clearly shown, with their performance substantially enhanced when merged through the sensor fusion approach, as demonstrated in (Hasanujjaman et al., 2023). This synergistic fusion offers an unparalleled advantage in autonomous driving systems, where accuracy and real-time response are paramount.

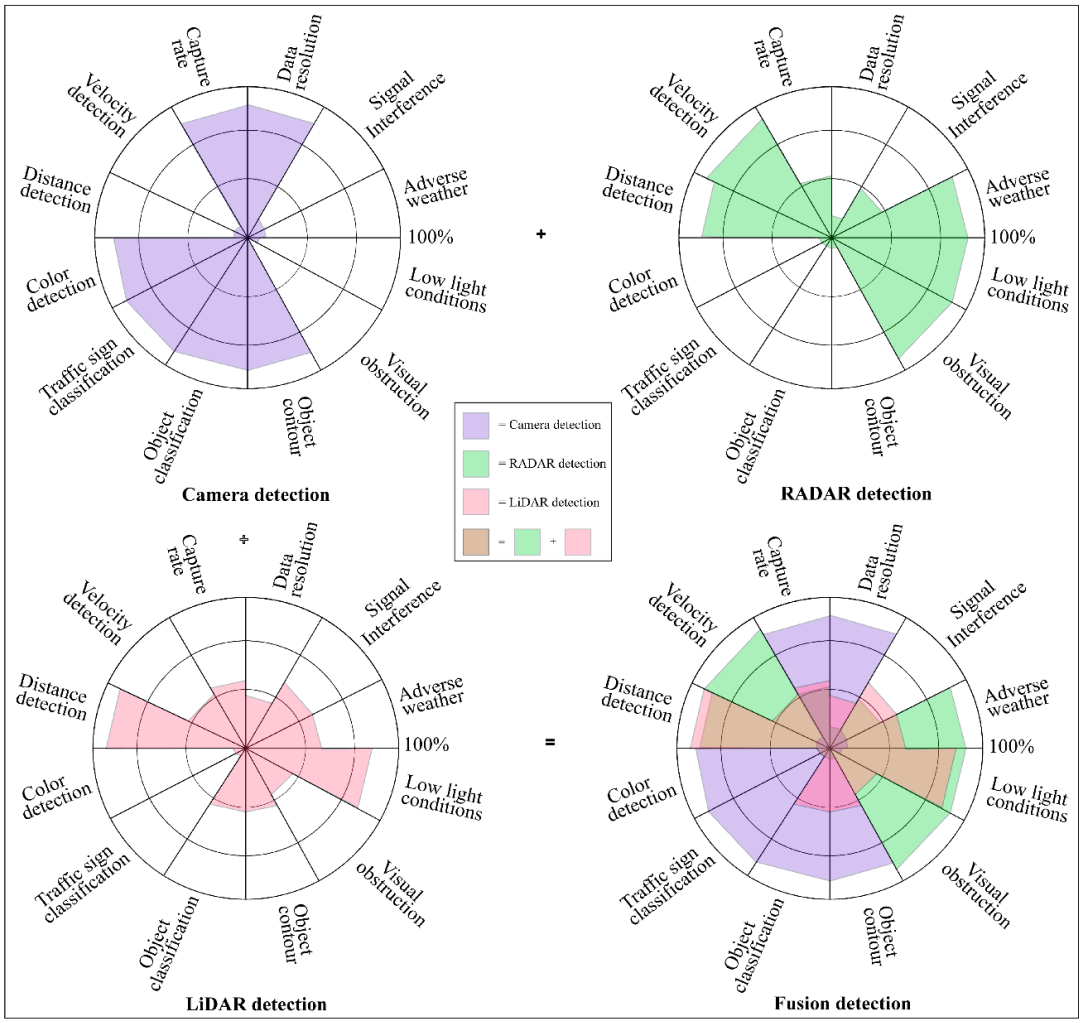


Figure 2.3: Capacity comparison of radar, camera, and LIDAR sensors and their sensor fusion detection approach. Note: from (Hasanujjaman et al., 2023) In *Sensors.*

The multi-sensor fusion approach is important in detecting, tracking, and classifying moving objects. Combining the strengths of radar, camera, and LIDAR sensors enhances frontal object perception, enabling the accurate detection of dynamic targets. Once detected, the fused data undergo further processing for precise object tracking, allowing seamless interaction with ADAS-installed vehicles. Notably, this fusion technique excels in detecting and tracking objects when any vehicle moves directly in front of the ego vehicle, providing safety and accuracy (Chavez-Garcia & Aycard, 2016).

A thorough review of radar-camera fusion methods for object detection unveils a wealth of insights, offering clear guidelines on how to optimally integrate these sensors for both object detection and semantic segmentation. The cost-effective radar-camera fusion approach is particularly advantageous, as it performs reliably across various lighting conditions and adverse weather environments. With the advent of advanced sensor technologies, such as 4D radar, integration with cameras has the potential to deliver highly detailed and valuable information, driving the future of AVs (Yao et al., 2024).

Sensor fusion stands as a cornerstone in enhancing object detection capabilities, paving the way for improved vehicle tracking to aid autonomous driving. Radar, camera, and LIDAR sensors are the mainstay of AV perception systems, each offering unique advantages and limitations. Since no single sensor provides flawless data, sensor fusion becomes crucial in ensuring the reliability and accuracy of environmental perception. By combining the strengths of multiple sensors, fusion significantly boosts detection accuracy, offering a robust solution to the challenges of autonomous driving (Butt et al., 2022; Kang & Kum, 2020; Natan & Miura, 2022; Yeong et al., 2021).

This section covered the strengths and limitations of radar, camera, and LIDAR sensors, emphasising their enhanced performance through sensor fusion. The following section focused on vehicle tracking using this approach.

2.3 Vehicle Tracking Using Sensor Fusion

In this thesis, one of the primary objectives is to develop an innovative simulation method for selecting an optimal sensor architecture for AV sensor data preparation. This section offers a background study focused on achieving this objective by analysing the latest advancements and identifying gaps in this research domain.

Xiaobin et al. introduced the Combined Optimal Sub-Pattern Assignment (C-OSPA) metric, a tool for evaluating Multi-Target Tracking (MTT) performance (Shi et al., 2017). Their work highlighted several areas for improvement to overcome the limitations of the optimal Sub-Pattern Assignment (OSPA), including the detection of labelling errors and the integration of tracking quality metrics for more accurate performance assessments. By addressing angular error issues within C-OSPA, they provided a refined approach for state estimation, enhancing existing OSPA methods. However, their study remains limited, as it primarily focused on mathematical simulations without incorporating real-world object detection or trajectory evaluations.

A tracking performance evaluation method was proposed to detect better centre locations of objects (Huang et al., 2022). The authors used image pixels to correct an object's centre location with the help of an adaptive threshold and background suppression. This image processing-based approach used data from the camera sensor to label object detections using an annotation box and bounding box with an improved centre location. While this method can track vehicles with a single sensor, it remains constrained by its lack of multi-sensor integration.

In 2023, Ghazali et al. explored intelligent trajectory tracking using three distinct algorithms, evaluating tracking performance, execution time, and steering errors (Ghazali et al., 2023). The authors considered lateral position and heading angle errors as steering errors. However, their study overlooked the perception mechanism necessary for real-world trajectory environments, leading to potential repeatability issues in practical applications.

Yoonsuk et al. introduced a model predictive control-based algorithm that harnessed the power of sampling time to incorporate crucial inputs such as lateral velocity and steering angle, enhancing path-tracking accuracy (Choi et al., 2021). In their MATLAB-based simulation experiments, they tested their model across four distinct sampling times, demonstrating superior performance with shorter sampling intervals. The authors applied their developed algorithm in a single-vehicle travel scenario, effectively tracking the vehicle's kinematic behaviour to extend the boundaries further. However, for AV development, real-world applications demand the detection of surrounding vehicles on the road to ensure safe and efficient path tracking. Thus, while their algorithm shows promise, further refinements are needed for practical deployment.

Rahmathullah et al. addressed the challenges of the Optimal Sub-Pattern Assignment (OSPA) metric by introducing the Generalised Optimal Sub-Pattern Assignment (GOSPA) metric, which optimised assignment calculations (Rahmathullah et al., 2017). The GOSPA metric detected estimated, missed, and false targets by penalising localisation errors. It also accommodated a Random Finite Set (RFS) of targets, allowing for the evaluation of MTT performance. Additionally, the authors expanded the GOSPA metric by incorporating penalised trajectory switching to measure the gap between two sets of trajectories (García-Fernández et al., 2020).

Another recent initiative to improve the GOSPA metric was presented in (Su et al., 2024). The authors introduced a multi-target tracking evaluation method incorporating temporal dimension specifics to assess predicted trajectories by comparing them with actual trajectories. This method evaluated tracking accuracy, continuity, and clarity by considering errors such as localisation, missed targets, trajectory switching, and false trajectories. These features provided an approach to selecting more effective MTT algorithms, leading to improved tracking performance.

In an overview study, the authors revisited visual tracking measures and evaluated the performance measures for tracking performance. The authors evaluated 16 trackers

using 25 video scenarios to present the performance results accurately and robustly. Minimising the number of performance measures from a list of enormous measurement variables for tracking evaluation was the main idea behind visual tracking evaluation reliability (Čehovin et al., 2016).

A vehicle tracking algorithm was developed based on the extended Kalman filter for tracking the trajectories of a platoon of vehicles in (Song & Hyun, 2024). The tracking performance of a vehicle platoon was evaluated using external numerical analysis, applying improved position estimation through an array of antennae and a transmitting infrastructure built into the road system. This vehicle-to-infrastructure-dependent tracking method requires data produced from external sources rather than relying on internal AV data alone.

Alai and Rajamani developed a sensor fusion-based, cost-effective vehicle tracking system that can detect obstacles in real time to avoid possible crashes (Alai & Rajamani, 2024). Their algorithm improved the target vehicle's corner detection through multiple sensor fusion. However, this solution is limited to micromobility vehicles such as e-scooters and deals only with front-end target tracking.

Many researchers utilised tracking performance thresholds to improve the results of their proposed solutions for vehicular developments. Such initiatives included vehicular platoon management (Guo & Li, 2019) and controlling the tracking performance of mobile robots (Ahmad, 2020).

A summary of the recent research advancement and gaps in vehicle tracking sensor architecture is presented in Table 2-1 from the analysis of the above literature review. The following section presents recent studies in crash reconstruction and crash event detection.

Table 2-1: Summary of the vehicle tracking sensor architecture related works. Note: from (Haque et al., 2024) In *Sensors.*

References	Key Contribution	Limitation/Efficacy
(Shi et al., 2017)	Proposed a metric (C-OSPA) to evaluate multi-target tracking performance	Performed mathematical simulations only
(Huang et al., 2022)	Developed a tracking performance evaluation method	Used a single sensor for detection
(Ghazali et al., 2023)	Algorithms for intelligent trajectory tracking performance evaluation	The perception mechanism was not explained
(Choi et al., 2021)	Improved path-tracking performance by the proposed algorithm	Absent other vehicles in the experiment scenario
(García-Fernández et al., 2020; Rahmathullah et al., 2017)	Solved the issues of the OSPA metric and proposed GOSPA metric	Useful for tracking performance evaluation
(Su et al., 2024)	Improved the GOSPA metric using temporal dimension specifics	A potential metric for tracking evaluation
(Čehovin et al., 2016)	Visual tracking measures for tracking performance	Considered only video data
(Song & Hyun, 2024)	Improved the position estimation for vehicle tracking	Only numerical evaluation was performed
(Alai & Rajamani, 2024)	Sensor fusion-based cost-effective vehicle tracking system	Limited to only front-end target tracking

2.4 Crash Reconstruction

The process of crash reconstruction plays an important role in unveiling the sequence of events leading up to an accident, offering insights for post-crash analysis and liability determination (Shen et al., 2023). By providing detailed descriptions of vehicle dynamics, as well as pre-crash and crash movements, this method allows for an in-depth examination of the crash's causes. Traditionally, crash reconstructions were based on physical accident artefacts and crash data collected from the scene, shedding light on vehicle speeds and crash severity (Struble & Struble, 2020). Nevertheless, with the

rise of advanced technologies, the landscape of crash reconstruction has evolved. Today, data from Event Data Recorders (EDRs) are increasingly used to replace the need for physical evidence at the scene, enabling a more streamlined and precise approach to analysing vehicle crashes. Given the rapid advancements in sensor technology, it is now more critical than ever to develop methodologies that leverage these innovations, providing an opportunity for even more accurate crash reconstructions with detailed crash data captured directly from vehicle sensors (Brach et al., 2022).

A cost-effective crash scene reconstruction system was developed using a GNSS receiver, an Unmanned Aerial Vehicle (UAV) equipped with a camera, and multiple Ground Control Points (GCPs) to gather crash data without disrupting traffic (Pérez et al., 2024). The accuracy of georeferencing depended on the placement of the GCPs, with optimal results achieved when the distance between two GCPs fell between 50 and 100 metres. This approach revolutionised traditional, time-intensive crash data-gathering methods, offering an effective alternative. However, advancements in camera-based image collection are still needed, and multi-sensor fusion holds promise in addressing challenges like poor lighting and inclement weather. While managing multiple GCPs poses its own set of challenges, specialised expertise is essential to operate and harness the full potential of UAV-based data collection for crash reconstruction.

A statistical analysis framework was developed to gather crash factors and establish correlations between Surrogate Safety Measures (SSMs) and crash statistics through traffic reconstruction (Qu & Wu, 2025). The research revealed a connection between SSMs—specifically time-to-collision and deceleration-rate-to-avoid-a-crash—and the occurrence of crashes. This study highlighted the need for enhanced data collection methods derived from simulated traffic scenarios to drive accurate advancements in AV safety. Noted that it uncovered a pressing need for more granular crash data in publicly accessible datasets, emphasising the importance of including trajectory information to reconstruct better and understand various types of crashes.

Liu et al. developed an intelligent crash reconstruction method, using simulated crash scenarios and multi-object optimisations (Yu Liu et al., 2022). Their crash reconstruction framework was validated against real-world collisions—specifically between an electric bicycle (e-bike) and a car—alongside simulated scenarios. The method incorporated six key initial variables: the velocities of both the car and e-bike, impact angles, the e-bike's posture, the car's angular velocity, impact positions, and friction coefficients. In their developed method, kinematic data from the reconstructed crash were captured using fixed surveillance video footage. While the reconstruction method excelled at animating the crash sequence, it faced limitations, such as its inability to automatically collect all output parameters where additional calculations were needed. Moreover, while it effectively visualised the crash, the method fell short in automatically detecting conflicts or crash events.

Vehicle crash reconstruction serves as a powerful tool for delivering insights into crash dynamics, enabling investigators to visualise animated crash scenarios for a more precise assessment of the incident (Ortiz & Dávila, 2023). Kolla et al. demonstrated the use of terrestrial 3D laser scanner point clouds combined with dashcam video to reconstruct traffic incidents (Kolla et al., 2022). Their method, using a monocular camera, provided detailed incident data through reconstructed crash scenarios, offering key kinematic details such as speed, travel distance, acceleration, and deceleration of vehicles. While they fused laser scanner point clouds and video data for crash reconstruction, the validation of their method was based on data from a single dashboard camera. Integrating multiple vehicle sensor types could enhance the reconstruction accuracy, an approach that remains unexplored in their study.

Crash reconstruction has transitioned from physical evidence to advanced technologies, enhancing data collection with sensor fusion. While simulation methods provide detailed reconstructions, further developments are required.

2.4.1 Crash Events Detection and Classification

Pour et al. proposed a machine learning-based framework to detect vehicle accidents

using in-vehicle sensor data (Pour et al., 2022). By using sensors such as gas-pedal position, acceleration, steering angle, and speed, they gathered comprehensive data from the vehicle's internal network and tested five feature extraction techniques. Their findings revealed that Convolutional Neural Network (CNN) and Support Vector Machine (SVM) features can be combined to realise comparatively better vehicle accident detection. While their developed automatic accident detection system showed great promise, it still falls short in capturing and analysing the sequence of events leading up to a crash.

Mateen et al. proposed an accident detection mechanism that integrates sensors like Infrared (IR), microphones, and smoke detectors to create a smart road system (Mateen et al., 2022). Their solution included a dynamic alert module with light and sound signals, ensuring swift notifications to rescue teams such as ambulances and fire brigades. The system was designed to alert nearby drivers, encouraging immediate rescue actions and minimising the risk of multi-vehicle collisions. Their solution focused solely on crash events and did not take into account the pre-crash events.

A great deal of previous research into AV crash analysis was focused on harnessing in-vehicle sensor data to uncover the underlying factors contributing to AV crashes (Beck et al., 2023). By simulating AV crashes using data from sensors like cameras, radar, and LIDAR, researchers can generate insights into crash dynamics. Beck et al. demonstrated that sensor-based simulated crash scenarios provide valuable data for in-depth AV crash investigations (Beck et al., 2023). Their work introduced an innovative data pipeline concept, showcasing the potential of AV sensor data to simulate realistic crash scenarios. However, the research overlooked the integration of sensor fusion techniques, which led to accuracy limitations in the crash simulations.

In the study by Krishnendhu & Mohandas, a Sensor-based Anomaly Detection (SAD) system was introduced to efficiently detect vehicle accidents and alert emergency responders (Krishnendhu & Mohandas, 2023). This system integrated anomaly detection and emergency status invocation modules to enhance its response

capabilities. The SAD detected unusual pressure generated during an accident, triggering a camera-based image processing module to visually confirm the incident utilising a piezoelectric vibration sensor. The authors emphasised that the system effectively identified accidents through post-collision kinematics, such as vehicle orientation changes or verifying the vehicle's stop status. However, the study noted a limitation: a single vibration sensor provided less accuracy than a multi-sensor fusion approach. Additionally, the system might fail to send a digital trigger signal to the detection module if the vibration sensor detects insufficient pressure due to a side-impact crash.

A recent study by Hasanujjaman et al. explored the power of sensor fusion, combining advanced AV sensors with traffic surveillance cameras to enhance object detection and localisation (Hasanujjaman et al., 2023). The researchers compared object detection performance using individual AV sensors—radar, camera, and LIDAR—against the performance achieved through sensor fusion. Twelve detection parameters were evaluated, including low light conditions, visual obstructions, object contours, classification of objects and traffic signs, colour and distance detection, velocity tracking, capture rate, data resolution, signal interference, and adverse weather conditions to assess the effectiveness of the fused system. Remarkably, the fusion of radar, camera, and LIDAR delivered accurate and reliable detection results. However, the combination of radar and camera alone also produced comparable performance, offering a cost-effective alternative. The study also highlighted the need for additional processing costs when integrating roadside cameras to ensure optimal accuracy in the proposed system.

The crash reconstruction approaches are continuously changing alongside technological developments. This section revealed from the analysis of the crash reconstruction-related studies that there is a demand for detailed crash data collection, and accurate reconstruction approaches still exist. The summary of the crash reconstruction-related work is presented in Table 2-2.

Table 2-2: summary of the crash reconstruction-related work.

References	Key Contribution	Limitation/Efficacy
(Pérez et al., 2024)	Developed a mechanism to reconstruct crashes using GNSS and cameral-equipped UAV	Sensor fusion was not considered, and specialised expertise is needed to operate the developed system
(Qu & Wu, 2025)	Developed a statistical analysis framework using crash reconstruction	Discovered the demand for detailed crash information in the publicly available crash datasets
(Yu Liu et al., 2022)	Developed an intelligent crash reconstruction method using simulated crash scenarios	Unable to detect conflict or crash events automatically
(Kolla et al., 2022)	Developed a method using dashcam video to reconstruct traffic incidents	The method used only a single camera to generate crash information
(Pour et al., 2022)	Proposed a framework to detect vehicle accidents	Unable to describe the sequences of pre-crash events
(Mateen et al., 2022)	Proposed an accident detection mechanism	Only crash events can be labelled, pre-crash events were not included
(Beck et al., 2023)	Developed a data pipeline concept to simulate crashes	Sensor fusion technique was not considered
(Krishnendhu & Mohandas, 2023)	Proposed a sensor-based anomaly detection	A single vibration sensor was used to detect accidents
(Hasanujjaman et al., 2023)	Improved object detection using AV sensors	Additional processing needed

2.5 AV Sensor Data for ML-Based Crash Prediction

In a recent study, Ali et al. reviewed papers published on ML-based vehicle crash modelling, categorising it into three key areas: crash occurrence or real-time crash prediction, crash frequency prediction, and crash injury severity prediction (Ali et al., 2024). They also identified research gaps and outlined future directions for each category. Binary classification models are primarily designed to predict crash or no-crash events, relying on pre-crash data. For real-time crash prediction, selecting appropriate non-crash events is crucial. This type of prediction can support traffic

management (Islam, Abdel-Aty, Wang, et al., 2024), assist in developing countermeasures, and aid in road safety decision-making (Ali et al., 2024). Notably, many researchers used pre-crash traffic data ranging from 5 to 10 mins before crashes (Islam, Abdel-Aty, Wang, et al., 2024) for real-time crash prediction, whereas some used 30 mins of traffic data before accidents (Ali et al., 2024; Islam, Abdel-Aty, Islam, et al., 2024). Early crash prediction can be a game-changer in traffic management, helping provide safe route recommendations to avoid accidents and congestion (Berhanu et al., 2024). However, using longer pre-crash data periods may lead to an overrepresentation of non-crash events, creating data imbalance issues that can affect the accuracy of predictions (Elassad et al., 2024). In the context of AVs, only a few seconds of pre-crash data are required for trajectory planning and crash avoidance, as they need fractions of a second to react. For example, a Tesla with full self-driving capability can react under 0.3 s to avoid potential hazards (Johnson, 2024).

The California Department of Motor Vehicles (CA-DMV) provides the most extensively analysed publicly available AV crash reports (DMV, 2024). The CA-DMV has compiled 747 AV collision reports covering the past ten years (from October 2014 to October 2024). These reports are divided into six sections, with a summary of the data collection in Table 2-3. Section 1 details the manufacturer information of the involved AV, while Section 2 provides time-related details of the accidents. Section 3 focuses on information about other vehicles involved in the collisions. Injury-related data are covered in Section 4. The most significant section, Section 5, offers a description of the crash scenario, along with additional details such as weather and lighting conditions, road surface status, pre-crash manoeuvres, and collision type. The final section includes the name and designation of the person who signed the report. The CA-DMV crash reports are frequently used to analyse factors contributing to AV crashes (P. Liu et al., 2024; Q. Liu et al., 2024). However, this study found very few papers utilising CA-DMV data for real-time crash prediction. Real-time crash prediction requires time series data, such as vehicle dynamics, speed, and pre-crash events, which are not included in the CA-DMV dataset.

Table 2-3: Summary of an AV crash report (CA-DMV) (DMV, 2024).

Section	Property Name (Value)
1. Manufacturer info.	Manufacturer name, business name, Address, and phone number
2. Accident info. (AV)	Date, time, year, make, model, license and VIN, accident location, movement, crashed with, driver information, damage description (major, moderate, minor, unknown, none)
3. Other party's info.	Year, make, model, license and VIN, movement, crashed with, driver information
4. Injury/death, property info.	Name, address (driver/passenger, injured, deceased, bicyclist, property information)
5. Accident details	Driving mode (autonomous, conventional), accident description, weather (clear, cloudy, raining, snowing, fog/visibility, wind, other) light (daylight, dusk/dawn, dark), roadway surface (dry, wet, snowy/icy, slippery), roadway condition (holes, loose material, obstruction, construction, reduced width, flooded, good, other), movement preceding collision (stopped, proceeding straight, ran off road, right turn, left turn, backing, slowing, passing, changing lanes, parking, entering traffic, unsafe turning, opposite lane, parked, merging, other) collision type (head-on, side swipe, rear end, broadside, hit object, overturned, vehicle/pedestrian, other) other factors (rule violated, vision obscurement, inattention, stop/go sign, ramp, previous collision, unfamiliar road, others)
6. Certification	Name of authorised person, signature, date

The Transport Research Centre (CDV) used the world map to visualise 801 AV crashes geographically distributed all over the world (CDV, 2024). Out of the presented active AV crashes, 771 crashes occurred in the USA because the CDV included the CA-DMV data with other limited sources. The map-based visualisation of AV crashes included fifteen variables: country, a brief description of the incident, vehicle information, damage details, injury description, vehicle movement status, accident cause, weather conditions, time of day, road conditions, whether the vehicle was driverless, collision type, and vehicle type. Though this map-based visual information helps to understand the present AV crash data available publicly, it demands detailed

crash information for real-time crash prediction.

The AV crash open dataset (AVOID) was prepared using CA-DMV data and other vehicle incident reports reported worldwide (Zheng et al., 2023). It was developed to overcome limitations in existing AV crash datasets, such as small sample sizes, data imbalance, and missing information. In this dataset, 37 types of crash data were arranged to present AV crash information into seven categories. These categories were vehicle conditions, pre-crash conditions, road and environment, crash outcome, and collided points of the ego vehicle and the other vehicle. While the dataset contains information on pre-crash manoeuvres, it does not classify the time series of crash occurrences, making it less appropriate for real-time crash prediction.

The NHTSA provided two different types of autonomous driving crash datasets. One of these datasets contained AV crash data up to level 2 autonomous driving, and the summary report on level 2 ADAS crashes can be found in (NHTSA, 2022c). The other dataset covered crash data for Level 3 and above ADS (NHTSA, 2024). The datasets provided by the NHTSA contained 136 attributes. These attributes can be classified in a list of information as follows: report identity, vehicle manufacturer, milage, ADAS or ADS status, report data source, crash location, road, weather, vehicle damage, vehicle movement type and speed (time series data not provided) during the crash, airbag and passenger belts status, crash data availability, investigation agency information, short crash narrative, and nine other attributes. These AV crash datasets do not include pre-crash kinematics and vehicles' dynamic information, which are crucial for real-time crash prediction.

Research on real-time AV crash prediction mainly generated data from the AV sensors (Hussain et al., 2023; Islam, Abdel-Aty, Islam, et al., 2024; Islam, Abdel-Aty, Wang, et al., 2024) to train ML models. Due to the current demand for appropriate AV crash datasets, available naturalistic non-crashed AV driving data were alternative sources of crash datasets for real-time crash prediction. For example, the study by Anis et al.(Anis et al., 2024) used non-crash AV data (Waymo Open Dataset (Sun et al.,

2019)) to estimate real-time crash risk for road safety. Their framework for estimating crash risk incorporated vehicle dynamics and factors such as speed, acceleration, steering, and heading based on near-miss crashes. While near-miss events are valuable for crash risk estimation and can aid in crash avoidance, the absence of actual crash data in the training sets may limit the accuracy of real-time predictions. Additionally, existing methods (Hussain et al., 2023; Islam, Abdel-Aty, Islam, et al., 2024; Islam, Abdel-Aty, Wang, et al., 2024) that rely on long-term sensor data collection for real-time crash prediction face challenges related to the imbalance between crash and non-crash data in the generated datasets.

Simulation-based crash estimation enhances the understanding of the factors contributing to collisions, which in turn helps improve the collision avoidance capabilities of ADS (Calvi, D'Amico, et al., 2020; Calvi, D'Amico, et al., 2020; Wang et al., 2020b), and AV driving safety (Ren et al., 2023; Wu et al., 2023). Analysing AV crashes requires considering the vehicle's perception and decision-making based on sensor data, as these help to identify the contributing crash factors (Yeong et al., 2021). Recent research has focused on the use of simulation-based driving data, which has gained popularity in high-risk scenarios due to its cost-effectiveness and the ability to evaluate AV performance in a virtual environment (Arcaini et al., 2022). AVs rely on multi-object trackers that use sensor fusion to gather tracking data about their surroundings (MathWorks, 2019b). The Most Important Object (MIO) is identified from the track data generated by the multi-object tracker, and its attributes are used to calculate the Forward Collision Warning (FCW) distance (MathWorks, 2024b). Haque et al. introduced a novel SMTPE (Haque et al., 2024), which aided in conducting high-accuracy AV crash data recording using available AV sensor data.

A summary of the current research advancement and gaps in AV sensor data for ML-based crash prediction is presented in Table 2-4.

Table 2-4: Summary of AV sensor data for ML-based crash prediction-related research work.

References	Key Contribution	Limitation/Efficacy
(DMV, 2024)	Publicly available AV crash reports	Useful for crash factors analysis and inappropriate for real-time crash prediction
(CDV, 2024)	World map-based visual AV crashes	Presented limited crash information, inappropriate for crash prediction
(Zheng et al., 2023)	AV crash open dataset (AVOID)	Did not classify time series values of crashes
(NHTSA, 2022c, 2024)	Publicly available ADAS and ADS crash datasets	Datasets did not include pre-crash kinematics and vehicles' dynamic information
(Anis et al., 2024)	Generated a secondary dataset for ML using Waymo Open Dataset	Near-miss crashes were used for crash risk estimation, but crash events were not included
(Haque et al., 2024)	Simulation-based sensor data for ML	Sensor fusion and vehicle tracking approaches for training data preparation

2.6 Real-Time Crash Prediction Using Machine Learning

In general, real-time crash risk prediction refers to the likelihood of a possible vehicle crash within a short time using traffic dynamics (Hossain et al., 2019; Hussain et al., 2023). Estimating potential crashes is important for triggering immediate crash avoidance measures and improving road safety (Abdel-Aty et al., 2024; Basso et al., 2021; Elassad et al., 2020; Lei et al., 2021). Over the past two decades, advancements in artificial intelligence and machine learning techniques have significantly contributed to the development of crash prediction models. ML-based models have demonstrated superior performance compared to traditional statistical models, driving their adoption in the research community (Ali et al., 2024). This thesis reviewed recent studies of ML-based crash predictions to uncover the current state of the development and challenges

in real-time crash prediction.

2.6.1 ML and Classification Algorithms for AV

AI is transforming AV technology, with the performance of ML applications in AVs being heavily influenced by the quality of training data generated by AV sensors (Ma et al., 2020). Recent advancements have highlighted the growing popularity of various ML techniques, including supervised learning. Supervised ML models typically involve classification tasks, where the training data must be labelled in order to predict outputs (Yuan et al., 2022). The effectiveness of real-time, data-driven crash prediction is directly tied to the choice of classification algorithms. Therefore, selecting appropriate classification methods is important for research related to crash prediction (Santos et al., 2022).

Over the past few years, ML techniques have been increasingly applied in road safety research to analyse various aspects of road crashes. Common ML algorithms used in this field include naive Bayes, decision trees, regression analysis, Artificial Neural Networks (ANN), and Support Vector Machines (SVM) (Santos et al., 2022; Sohail et al., 2023). A recent review of ML techniques for traffic analysis and prediction highlighted algorithms such as naive Bayes, Random Forest (RF), Gradient Boosting Machines (GBM), Linear Regression, SVM, and Multi-Layer Perceptron (MLP) (Behboudi et al., 2024). Similar algorithms, including RF, SVM, and Extreme Gradient Boosting (XGBoost), had been employed in real-time crash risk prediction (Z. Zhang et al., 2024). Ensemble-based classification algorithms have become a key focus in crash prediction research. For example, a recent study developed an ensemble fusion system and tested various classifiers, such as Adaptive Boosting (AdaBoost), XGBoost, RF, GBM, LightGBM, CatBoost, and K-Nearest Neighbours (KNN), for road crash event prediction (Elassad et al., 2024). Additionally, a summary of the most commonly used ML models for crash prediction identified decision tree-based models, neural network-based models, and others like SVM and KNN (Ali et al., 2024). A literature review examining the performance of machine learning algorithms for crash prediction,

which analysed 56 papers published between 2001 and 2021, found over 20 different ML approaches in use. The findings provided a comparative analysis, revealing the frequency of use and the best-performing algorithms (Santos et al., 2022). Table 2-5 presents a comparison of these algorithms based on their frequency and performance.

Table 2-5: ML algorithms were used in crash injury severity prediction papers, where the best-performing number and percentage are presented. Note: from (Santos et al., 2022) In *Safety Research*.

ML Algorithm	Number of Times used in Papers	Number of times performed the best	Best-Performing Percentage
Decision tree	26	8	30.76%
RF	23	16	69.56%
Logistic regression	19	2	10.52%
SVM	17	9	52.94
KNN	10	4	40%
ANN	10	3	30%
MLP	9	3	33.33%
Bayesian network	3	2	66.66%

2.6.2 Real-Time Crash Prediction

Islam et al. proposed a large-scale crash prediction framework utilising machine learning for crash occurrence prediction across an extensive geographic area (Islam, Abdel-Aty, Wang, et al., 2024). The authors incorporated spatial ensemble learning and knowledge distillation to tackle challenges related to pre-crash data, optimised data processing costs, and reduced false alarm rates. Their study was conducted in a specific region—Interstate-75 in Florida, USA—requiring further validation before extending it to other regions. In their previous study, the authors introduced a real-time crash likelihood and severity prediction framework, testing it on 625 crash events and 6,749,447 non-crash events (Islam, Abdel-Aty, Islam, et al., 2024). To address the data imbalance between crash and non-crash events, they applied the Synthetic Minority

Over-sampling Technique (SMOTE). However, the use of SMOTE in highly imbalanced datasets may lead to overfitting (Ali et al., 2024) and introduce bias in performance results.

The study experimented with four ML models, such as logistic regression, RF, SVM, and XGBoost, to predict real-time crashes using the HERE database (Z. Zhang et al., 2024). Among these, the RF model performed the best in predicting three types of crashes: single-vehicle, rear-end, and sideswipe crashes. The RF model's prediction accuracy for rear-end crashes was the highest, at 65.1%. However, traffic speed was the only contributing factor used for crash prediction in this study, which led to relatively low prediction accuracy. This limitation reduces the potential effectiveness of the developed ML model for real-time crash prediction.

The research developed an ensemble fusion system for real-time crash prediction using a simulated training dataset (Elassad et al., 2024). The dataset was generated by collecting vehicle kinematics, driver responses, and weather-related features from simulated driving scenarios. Seven ensemble-based machine learning classifiers were tested to evaluate crash events, employing an imbalance-learning approach for the training dataset. The study highlighted issues related to data imbalance between crash and non-crash events, addressing this with techniques such as over-sampling, under-sampling, and SMOTE-Tomek-Links. While the proposed crash prediction fusion framework showed promise, it did not account for pre-crash events, including conflicts and potential crashes.

Ahmed et al. conducted an analysis of a road crash dataset provided by the New Zealand Ministry of Transport, employing six machine learning models to highlight the key contributing factors to crashes and enhance the prediction of accident severity (S. Ahmed et al., 2023). Their explainable machine learning models used global Shapley additive explanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME) to offer insights into feature importance. The study identified several factors associated with severe crashes, including drug-impaired driving, road geometry, speed

limits, and elderly drivers. The findings could assist road transport agencies and relevant experts in improving road safety and supporting long-term safety planning. However, the research does not offer a real-time crash prediction model capable of preventing collisions.

The study assessed three Artificial Neural Network (ANN)-based machine learning models, training them on simulated crashes to predict fault rates in traffic accidents (Yilmaz et al., 2016). A key limitation of this approach was its inability to explain how objects and the road environment were detected during fault prediction for collisions. Additionally, the research focused solely on three simulated crash variables: deformation energy, velocity, and fault rate. The developed models could not provide an explanation of the events occurring during an accident.

Mo et al. introduced a model designed to predict dynamic short-term crash risks at toll plazas, combining a random effects logit model with a cutting-edge Long Short-Term Memory Convolutional Neural Network (LSTM-CNN) (Mo et al., 2024). Their comparison with two other prediction models revealed that the LSTM-CNN delivered significantly more accurate results. While this model excelled in predicting crashes at toll plazas, its applicability remains limited, and additional experimentation is necessary to test its effectiveness in broader transportation settings.

Wang et al. explored the ability of Surrogate Safety Measures (SSMs) as predictors for crash risk, offering a real-time glimpse into potential crashes (Wang, Xu, et al., 2024). Through the use of ridge regression and by analysing four distinct time series over a 4-second period, they evaluated the effectiveness of early crash warnings. Their findings revealed that the Time-To-Collision (TTC) metric outperformed other deceleration-based SSMs, providing a more accurate and reliable indication of crash occurrences. It is important to note the potential of TTC as a key feature in forecasting crashes and achieving optimal prediction outcomes.

A summary of the ML-based crash prediction-related works is presented in Table 2-6.

Table 2-6: Summary of the ML-based crash prediction-related research work.

References	Key Contribution	Limitation/Efficacy
(Islam, Abdel-Aty, Islam, et al., 2024; Islam, Abdel-Aty, Wang, et al., 2024)	Proposed a large-scale crash prediction framework	Data overfitting issues
(Z. Zhang et al., 2024)	Developed a model to predict single vehicle, rear-end, and sideswipe crashes	Only traffic speed was considered for crash prediction
(Elassad et al., 2024)	Developed an ensemble fusion system for real-time crash prediction	Pre-crash events were not considered
(S. Ahmed et al., 2023)	Developed explainable ML model	Model for crash severity prediction, not applicable for real-time crash prediction
(Yilmaz et al., 2016)	Developed neural network-based ML models to predict fault rates of traffic accidents	Used only three variables, and crash detection not explained
(Mo et al., 2024)	Developed a model to predict dynamic short-term crash risk at toll plazas	Transportability issue
(Wang, Xu, et al., 2024)	Assessed SSMs for crash occurrence prediction	Explained the implication of SSMs for crash prediction

2.7 Summary

This chapter highlighted the role of a functional sensor architecture in generating suitable AV sensor data for evidence-based, data-driven crash reconstruction and prediction. As demonstrated in the literature review, numerous advancements have been made in developing new metrics for tracking performance evaluation and enhancement. Researchers have already designed algorithms and evaluation criteria to determine the optimal simulation setups for assessing tracking performance. However, a clear gap exists in identifying a robust method for selecting experimental setups that can produce high-quality, evidence-based AV sensor data. Moreover, the need for an approach that automatically collects and utilises AV sensor data for precise crash

reconstruction, along with accurate pre-crash descriptions, is becoming increasingly urgent.

ML-based real-time crash prediction techniques are emerging as preferred solutions among researchers. Significant evidence has been provided regarding the effectiveness of multi-sensor fusion in crash event classification and prediction. Furthermore, recording the sequence of events leading up to a crash is important for real-time AV crash prediction. As reviewed in the literature, most real-time crash prediction systems leverage ML to improve traffic management and enhance driving safety. In these systems, data were typically collected 5 to 30 minutes before a crash. However, the collection of extended pre-crash data often results in a data imbalance, with non-crash events far outnumbering crash events. Techniques like SMOTE were frequently used to address this imbalance, though they can lead to data overfitting.

Research also suggested that region-specific crash prediction models—such as those focused on short regions like toll plazas or extended regions like Interstate-75 in Florida—tend to outperform area-independent models. Another key finding is the need for a suitable AV crash dataset that supports real-time crash prediction using various types of crash events derived from diverse driving scenarios.

Above all, it needs to be explored whether sensor data available in typical AVs can be utilised for evidence-based crash reconstruction and data-driven real-time AV crash prediction, as such a model has yet to be developed in the reviewed studies. To this end, this research is the first to collect and analyse AV sensor data using the Simulation Method for Tracking Performance Evaluation (SMTPE) —an innovative method for evaluating sensor architectures. This method is designed to identify the optimal sensor configuration by analysing various multi-sensor architectures. Building on this foundation, the research reconstructed evidence-based crash scenarios and classified both crash and pre-crash events using a new model called the Vehicle Crash Reconstruction Method (VCRM), which was driven by simulated AV sensor data. Furthermore, this is the first study to utilise the resulting dataset with a newly proposed

model known as the Vehicle Crash Reconstruction and Prediction Model (VCRPM). This model enables more accurate, data-driven crash prediction and analysis. A summary of the novelty of this thesis compared to previous works is presented in Table 2-7.

Table 2-7: Summary of the novelty of this thesis compared to previous works.

Research Area	Name of the Limitation	Limitation Summary of Existing Research	This Research Novelty and Contributions
Vehicle tracking using sensor	Lack of multi-sensor fusion	Tracking performance using a single sensor (Huang et al., 2022),	Developed a novel SMTPE and evaluated multi-sensor tracking architectures
	Perception mechanism	Perception mechanism unexplained (Ghazali et al., 2023)	Performed surround vehicle sensor fusion for perception
	Lack of real-world validation	Mathematical simulations only (Shi et al., 2017)	Real crash dataset utilised in simulation
	Infrastructure dependency	Applicability issues in non-equipped environments (Song & Hyun, 2024)	Used AV sensors, no additional infrastructure needed
	Limited scenario coverage	Single-vehicle scenario (Choi et al., 2021), only front-end target tracking (Alai & Rajamani, 2024)	Evaluated front-end, head-on, rear-end, and side impact V2V crash scenarios
Sensor data for crash reconstruction	Dependence on expert operation	Specialised expertise was needed to operate (Pérez et al., 2024)	The new VCRM can reconstruct crashes automatically
	Limited crash sequence	Only described crash moments (Pour et al., 2022), (Mateen et al., 2022)	Successfully described sequences of crashes (cut-in, conflict, potential crashes and crashes)
	Automatic parameter extraction	Manual calculations were needed (Yu Liu et al., 2022)	Automatically records data and can present 3-D crashes
	Data processing Cost	Cost of data processing of roadside cameras (Hasanujjaman et al., 2023)	Only AV sensor data are utilised, hence data processing is faster
Sensor data for ML-based AV crash prediction	Publicly available AV crash dataset-specific limitations	Lack of time-series pre-crash data (DMV, 2024), (NHTSA, 2022c, 2024), classification issue (Zheng et al., 2023)	Developed VCRPM is the first work to generate appropriate crash data using simulated data from AV sensors
	AV-specific gaps	AVs react in milliseconds, but many studies used 5–30 minutes of crash data (Islam, Abdel-Aty, Islam, et al., 2024; Islam, Abdel-Aty, Wang, et al., 2024)	Shapley analysis of developed ML models revealed that near-crash events are more important for effective prediction
	Geographic location-specific Models	Model tailored to toll plazas (Mo et al., 2024), area specific study (Islam, Abdel-Aty, Wang, et al., 2024)	Developed VCRPM can produce real-time crash data from any geographic location and successfully predict crashes
	Crash prediction features	Lack of appropriate feature (traffic speed only) for crash prediction (Z. Zhang et al., 2024)	This research experimented with fifteen features for real-time crash prediction
	Pre-crash events consideration	Did not consider pre-crash features such as conflicts or near-missed crashes (Elassad et al., 2024)	For the first time, this study included pre-crash features with 12 other features for prediction

The next chapter introduces the mathematical foundations and design principles that underpin the proposed model and methods.

Chapter 3 System Design and Evaluation Metrics

In Chapter 2 the background of object detection and tracking mechanisms using AV sensors was discussed. Building on this foundation, related works on sensor architectures, crash detection, reconstruction, and prediction were reviewed to understand the advancements and pinpoint the issues in these areas. This chapter introduces the key formulas, proposed methods, models, and evaluation criteria—all designed to tackle the issues outlined in the previous chapter.

3.1 Introduction

This chapter lays the important groundwork for this thesis, providing a structured foundation, an overview of the proposed model's design, and the evaluation metrics that will assess its methods. Section 3.2 defines the scope of this research, narrowing its focus to Vehicle-to-Vehicle (V2V) crash reconstruction and prediction, clearly delineating the types of V2V crashes considered for the study and setting the boundaries for the experiments discussed in subsequent chapters.

Section 3.3 delves into the core foundation of this research. A key objective is to collect real-time driving data from AV sensors, and the formulas in this section form the bedrock for achieving this objective. It covers the groundwork for object detection, tracking, and measuring vehicle movements, which is important for identifying MIOs. Furthermore, it provides the definitions and mathematical formulations for the detection of pre-crash and crash events, which are vital for data-driven crash reconstruction (Chapter 5) and prediction (Chapter 6).

In Section 3.4, a detailed component-level diagram of the proposed system is presented, offering a visual guide for the development and design of the model and methods explored in later chapters. Section 3.5 outlines the evaluation metrics used to assess the performance of tracking, classification of crash events, and prediction accuracy. Finally, Section 3.6 concludes with a summary, encapsulating the key points of the chapter.

3.2 Understanding the scope of the study

There are reports of collisions involving AVs and other entities such as vehicles, bicycles, and pedestrians. The California Department of Motor Vehicles (CA-DMV) in the United States is currently collecting traffic collision reports that involve AVs and other parties (DMV, 2024). In their reports, the CA-DMV classified AV-related crashes into distinct categories, which encompassed collisions with pedestrians, other automobiles, bicycles, and a diverse range of objects—such as skateboarders,

motorcyclists, animals, road dividers, and traffic islands. The growing trend of simulating vehicle crashes has become a critical tool for testing and analysing hazardous driving scenarios between AVs and other vehicles or obstacles, helping to understand better and prevent such incidents (Calvi, D'Amico, et al., 2020; Calvi, D'Amico, et al., 2020; Wang et al., 2020b; Wu et al., 2023).

The National Highway Traffic Safety Administration (NHTSA) has taken an important step in enhancing road safety by releasing a Standing General Order designed to gather crucial, evidence-based data on crashes involving vehicles equipped with ADS or Level 2 ADAS (NHTSA, 2022c). Drawing on this vital data, the NHTSA published a comprehensive summary report on Level 2 ADAS-related crashes, revealing key insights into the emerging landscape of AV interactions. The report highlighted a total of 392 crashes as of 15 May 2022, with 116 of these incidents involving V2V collisions between AVs and other vehicles. Of the V2V crashes, 62 involved AVs colliding with passenger cars, making up the largest group, while the second highest, 27 crashes, were between AVs and trucks. Notably rare, there was only a single crash involving an AV and a non-motorist on a bicycle, and two crashes between AVs and first responder vehicles. The report also recorded three crashes between AVs and non-motorist pedestrians.

For a deeper insight into the nature of AV crashes, the ADS crash dataset (covering the period from July 2021 to 15 September 2024) provided by the NHTSA was analysed (NHTSA, 2024). Among the 1,425 crashes involving ADS-equipped AVs, a significant 1,123 incidents occurred in collisions with other vehicles, representing a striking 78.80% of all AV crashes. As illustrated in Figure 3.1, passenger vehicles were involved in the majority of these crashes, accounting for 39.58%, the highest proportion among all categories. Following closely, 17.05% of AV crashes involved Support Utility Vehicles (SUVs), while trucks were the third most common collision type, contributing 14.53%. Non-motorist participants, such as bicycles, pedestrians, and other objects, were involved in 6.88% of incidents, while motorcycles accounted for 2.74%. Surprisingly, only 1.12% of crashes involved AVs and animals. An important

takeaway from this analysis is the noticeable fact that nearly 80% of all AV crashes involved another vehicle, with approximately 40% of these crashes occurring with passenger cars alone.

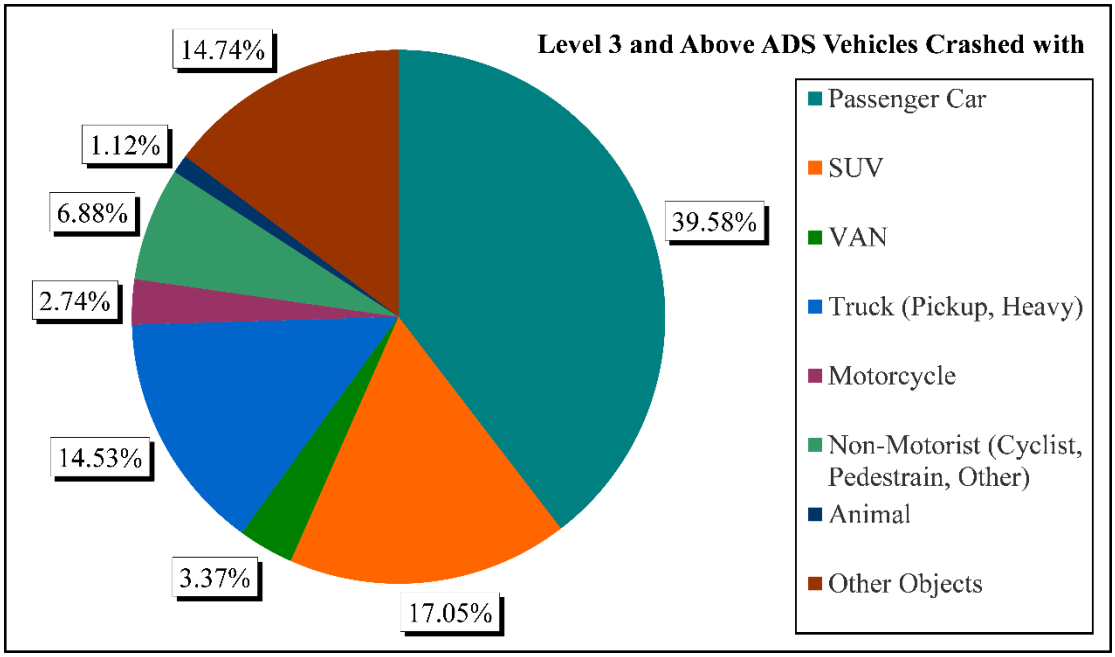


Figure 3.1: Level 3 and above ADS-installed AV crashes with another vehicle or object. The dataset source is (NHTSA, 2024), covering the period from July 2021 to 15 September 2024.

3.2.1 Research Scope

A recent study by Abdel-Aty and Ding offered an insightful perspective on the comparative safety of AVs over Human-Driven Vehicles (HDVs), revealing that AVs were significantly safer, particularly in reducing crashes (Abdel-Aty & Ding, 2024). Despite advancements in ADS, accidents involving AVs are still occurring. For instance, a vehicle crash analysis highlighted that rear-end collisions were notably more frequent, accounting for a substantial 52.46% of AV-related crashes, while lane-change crashes made up 18.85% (Qian Liu et al., 2021).

Moreover, another study on AV crash data (Pamidimukkala et al., 2023) supported this trend, with rear-end crashes representing a staggering 61.8% of incidents. Sideswipe collisions followed closely behind, constituting 21.2%, while broadside and

head-on crashes comprised smaller portions, at 6.1% and 5.8%, respectively. These statistics emphasised the importance of studying different types of AV crashes for data-driven crash reconstruction and prediction.

Further supporting this, the NHTSA confirmed that most AV crashes involved interactions with other vehicles (NHTSA, 2024). This evidence underscores the need for focused research on V2V collisions, which promises deeper insights into the nature of AV crashes and aspires to enhance the understanding of their frequency and impact. This study narrows its focus to experimenting with V2V crashes, providing a targeted approach to analysing AV crash occurrences with other vehicles, thereby advancing the pursuit of safer roadways.

3.2.2 Types of Vehicle Crashes

This thesis delves into AV crashes, building on the previously outlined requirements. The causes behind crashes involving AVs and traditional vehicles were examined, drawing from a study that classified accidents into rear-end, sideswipe, and other types of crashes, such as head-on collisions or impacts with objects (Novat et al., 2023; Su et al., 2024). According to the Crash Injury Research Engineering Network (CIREN), vehicle damage patterns were categorised across multiple impact zones, including the front, rear, left, right, top, and undercarriage (NHTSA, 2017-2023). A more recent study further refined these types, grouping V2V crashes into four primary categories: broadside, head-on, sideswipe, and rear-end (Abdel-Aty & Ding, 2024). In this research, V2V crashes—the ego vehicle (which records and processes data) and the other vehicles—are categorised into four distinct types. These are as follows:

1. Front crash (the ego vehicle and the other vehicle are traveling in the same direction)
2. Head-on crash (the ego vehicle and the other vehicle are traveling in opposite directions)
3. Rear-end crash (the ego vehicle and the other vehicle are traveling in the same direction)

4. Side-impact crash (the other vehicle can hit from any direction the ego vehicle's left or right side).

Pre-crash (cut-in, conflict, potential crash) and crash events can be observed from the vehicle trajectory for each type of the above vehicle crashes. Figure 3.2 shows the front (Figure 3.2a), head-on (Figure 3.2b), rear-end (Figure 3.2c), and side-impact (Figure 3.2d) crashes.

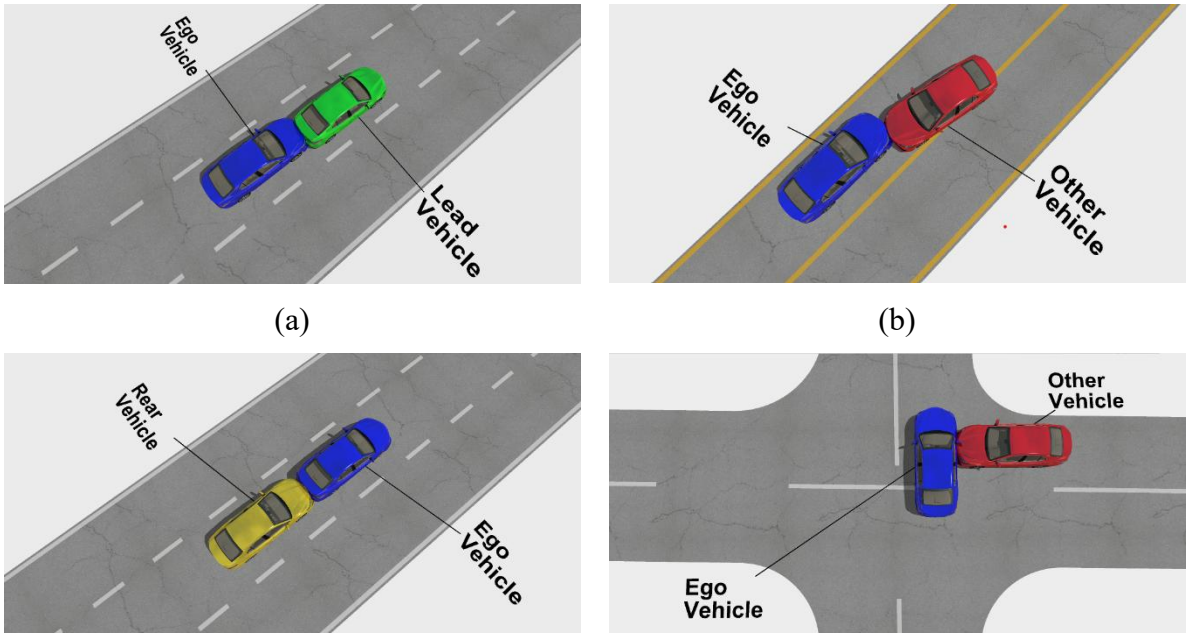


Figure 3.2: Types of vehicle crashes: (a) front crash; (b) head-on crash; (c) rear-end crash; (d) side-impact crash. Note: from (Haque et al., 2024) In *Sensors*.

This research considers the above four types of V2V crashes to generate AV sensor data for evidence-based data-driven crash reconstruction and prediction.

3.3 Formulas for the Proposed System

The previous section provided an overview and visual representation of the vehicle crash types examined in this research. The following section of this chapter presents the mathematical foundation underlying this thesis. It covers the fundamental concepts of sensor fusion-based tracking, labelling of crash and pre-crash events, and crash

prediction using AV sensor data. The mathematical representations of tracking performance evaluation were presented in (Haque et al., 2024). These formulas are utilised for this research and are given below with other required formulas.

3.3.1 Object Tracking, State Update, and Measurement Models

3.3.1.1 Tracking Objects

The Kalman filter uses a chain of detections or measurements from the vehicle sensors to estimate an object's state based on its motion model. An object's state contains position, velocity, acceleration, or turn rate in a motion model. An extended Kalman filter is a nonlinear version of the Kalman filter that linearises the state equation and measurement equation. Linearisation helps create the linear formatted state and state covariance, which needs Jacobians of the state and measurement equations (MathWorks, 2017).

In this study, the extended Kalman filter was initialised with a constant velocity motion model and a constant velocity measurement model for the sensor fusion-based object tracking. An object tracker, such as a joint probabilistic data association tracker, can utilise a filtering function to produce estimated states as vector values and state estimation error covariance as a matrix to track objects. As a filtering function, the extended Kalman filter uses the state transition function (a function returns the state vector at a time step using the state vector of the previous time step), measurement function (a function returns the output measurement for each time step), and initial states as input parameters in a tracker based on the data fused from the sensors (MathWorks, 2017). These input values are then processed for each time step by an extended Kalman filter algorithm to serve the state estimation of a target object. The three-dimensional (3-D) constant velocity model's state equation (MathWorks, 2018) is shown in (3.1). Here, $x_k, y_k,$ and z_k are the position coordinates of vehicle k, $v_{x,k}, v_{y,k},$ and $v_{z,k}$ are corresponding velocities, and T is the time step of the state transition.

$$\begin{bmatrix} x_{k+1} \\ v_{x,k+1} \\ y_{k+1} \\ v_{y,k+1} \\ z_{k+1} \\ v_{z,k+1} \end{bmatrix} = \begin{bmatrix} 1 & T & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ vx_k \\ y_k \\ vy_k \\ z_k \\ vz_k \end{bmatrix} \tag{3.1}$$

The simplified 3-D state vector structure is shown in (3.2).

$$\text{State} = [x;\ vx;\ y;\ vy;\ z;\ vz] \tag{3.2}$$

3.3.1.2 State Update Model

The extended Kalman filter maintains prediction and correction in a loop that uses the nonlinear state update and measurement functions in a time series cyclic order (MathWorks, 2017). A predicted state can be computed as a function of a previous state x_k, controls u_k, noise w_k, and time t, as shown in (3.3).

$$x_{k+1} = f(x_k, u_k, w_k, t) \tag{3.3}$$

The partial derivatives of the Jacobian with respect to the previous state can be expressed in (3.4).

$$F^{(x)} = \frac{\partial f}{\partial x} \tag{3.4}$$

Function (3.5) presents the Jacobian of the predicted state concerning the noise.

$$F^{(w)} = \frac{\partial f}{\partial w} \tag{3.5}$$

Functions (3.3)–(3.5) are more straightforward when the noise is additive in the state update equation as shown in (3.6). Here, $F^{(w)}$ is an identity matrix. The previous state x_k, controls u_k, and noise w_k, are used to compute the predicted state x_{k+1} at time t.

$$x_{k+1} = f(x_k, u_k, t) + w_k \tag{3.6}$$

3.3.1.3 Measurement Model

A measure can be calculated using the nonlinear function of the state and the measurement noise, as shown in Equation (3.7).

$$z_k = h(x_k, v_k, t) \tag{3.7}$$

The Jacobian of the measurement concerning the state is presented in Equation (3.8).

$$H^{(x)} = \frac{\partial h}{\partial x} \tag{3.8}$$

The Jacobian of the measurement concerning the measurement noise is shown in Equation (3.9).

$$H^{(v)} = \frac{\partial h}{\partial v} \tag{3.9}$$

Functions (3.7)–(3.9) are more straightforward when the noise is additive in the measurement equation as shown in (3.10). Here, $H^{(v)}$ is an identity matrix. The measurement z_k of the vehicle k is calculated using control x_k and measurement noise v_k at time t.

$$z_k = h(x_k, t) + v_k \tag{3.10}$$

3.3.2 Detecting Most Important Object

A multi-object tracker generates tracks from sensor detections, enabling the identification of the Most Important Object (MIO). The MIO is typically the closest track detected by the ego vehicle, which refers to the vehicle producing and storing the data from its designated travel lane. When any vehicle enters or travels within the ego vehicle's lane and reduces the gap between them, it should be recognised as the MIO. Potential MIOs are depicted in Figure 3.3, highlighting the dynamic interactions on the road that influence the ego vehicle's decision-making.

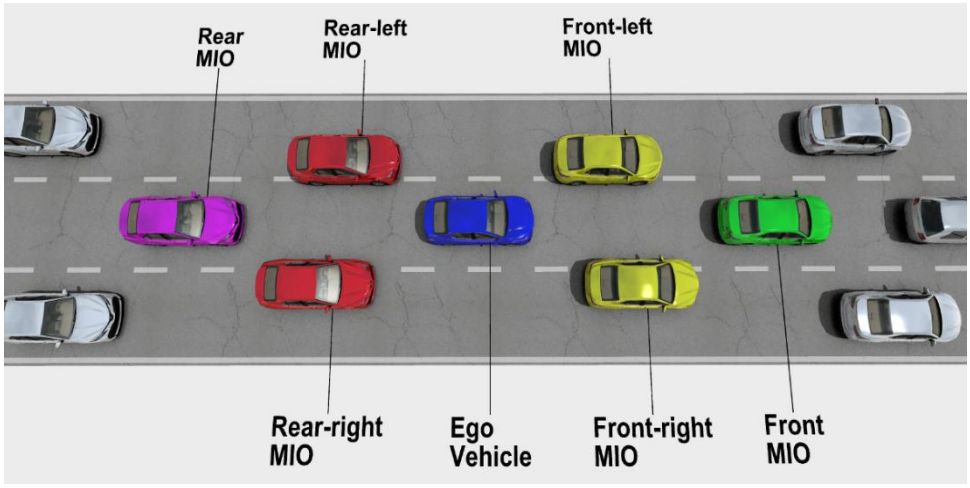


Figure 3.3: The ego (blue) vehicle detects the most important objects when green (front) and purple (rear) vehicles are decreasing distances and also when yellow vehicles (front-left, front-right) and red vehicles (rear-left, rear-right) are entering its lane.

MATLAB provides a polynomial evaluation function to measure the distance between the left and right lanes relative to the ego vehicle (MathWorks, 2024e). This function, as outlined in equation (3.11), plays an important role in detecting the MIOs. Measured left and right lane distances are then used to detect MIOs. The function returns a vector that matches the size of the query point x, providing an efficient evaluation of the polynomial p at each point in x. The polynomial equation is presented in equation (3.12), ensuring a robust and accurate measurement for lane distance detection.

$$y = polyval(p, x) \tag{3.11}$$

$$p(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_{n+1} \tag{3.12}$$

The ego vehicle uses a tracker to generate confirmed tracks, which serve as the primary input for identifying MIOs essential for self-driving. Additional inputs include the ego vehicle's position, lane width, and a state-based position selector, all of which are utilised to produce MIO detections, as outlined in Algorithm 3.1.

Alongside MIO detection, this algorithm provides information on longitudinal

distance, lateral distance, longitudinal velocity, lateral velocity, and the distances from the ego vehicle to the left and right lane boundaries. In this context, egoLat refers to the lateral position of the ego vehicle relative to the centre of the road, while x and vx denote the longitudinal distance and velocity, respectively. Similarly, y and vy represent the lateral distance and velocity, while leftLane and rightLane indicate the distances from the left and right lanes to the ego vehicle.

Algorithm 3.1: Find MIOs using sensor fusion-based tracking

Input: tracks, position, lane
Output: trackID, x, vx, y, vy, leftLane, rightLane

```
For each time step
    For i = 1 to number of confirmTracks
        Assign x, y, egoLat using (3.2)
        If x in rangeLimit then
            Case A: ego right side when egoLat < 0
                calculate leftLane, rightLane using (3.11)
            Case B: ego left side when egoLat > 0
                calculate leftLane, rightLane using (3.11)
            Case C: ego in lane centre when egoLat = 0
                calculate leftLane, rightLane using (3.11)
            If rightLane <= y <= leftLane then
                Update rangeLimit = x and trackID = i
            End if
        End if
    End for
    If trackID > 0 then
        Assign trackID, x, vx, y, vy, leftLane, rightLane
    Else
        Assign default values to state and lane variables
    End if
End for
```

3.3.3 Detection and Classification of Pre-Crash and Crash Events

Gaining a deeper understanding of vehicle pre-crash scenarios is important in shaping the future of vehicle safety and mobility. Simulating these high-stakes situations can unravel the complexities of vehicle movements, dynamics, and the critical moments leading up to an accident (Najm et al., 2013). using sensor fusion technology for vehicle tracking provides an effective way to pinpoint and label critical accident events. Table 3-1 showcases the terminology used to define pre-crash and crash events, clarifying this vital aspect of accident analysis.

Table 3-1: Description of terminologies used for pre-crash and crash events.

Terminology	Description
t	Time stamp
L_O	Length of the other vehicle
W_O	Half of the width of the other vehicle
W_b	Half of the width of both ego vehicle and other vehicle
Γ_{DLt}	Relative lateral distance between ego vehicle and other vehicle
Γ_{VLt}	Relative lateral velocity between ego vehicle and other vehicle
Γ_{DLn}	Relative longitudinal distance between ego vehicle and other vehicle
Γ_{VLn}	Relative longitudinal velocity between ego vehicle and other vehicle
λ_L	Left lane distance from the ego vehicle
λ_R	Right lane distance from the ego vehicle
K_L	The other vehicle's cut-in from left lane
K_R	The other vehicle's cut-in from right lane
ρ_{LnE}	Longitudinal position of the ego vehicle
ρ_{LnO}	Longitudinal position of the other vehicle
ρ_{LtE}	Latitudinal position of the ego vehicle
ρ_{LtO}	Latitudinal position of the other vehicle
δ_{prox}	Ego vehicle's proximity zone (front 30 feet, rear 4 feet)
α_{max}	Maximum deceleration (40% of the gravitation acceleration)

3.3.3.1 Time-to-Collision, Time-to-Escape, and Forward Collision Distance

Time-To-Collision (TTC) refers to the required time to collide between two vehicles continuously driving on the same road at the same speed. In order to prevent a collision, a driver needs a minimum amount of time for preventive actions, such as braking or changing lanes, which is known as the TTC threshold time. The existing TTC calculation method is presented in (Nadimi et al., 2020). The TTC is defined in equation (3.13) as:

$$TTC(t) = \left| \frac{\rho_{LnO} - \rho_{LnE} - L_O}{\Gamma_{VLn}} \right| \tag{3.13}$$

The authors defined the Time-To-Escape (TTE) as the time required to avoid a collision by taking driving actions, such as changing lanes, when braking alone is insufficient (Li et al., 2021). The TTE value can be used in conjunction with the safe proximity zone of the ego vehicle to detect potential conflicts between the ego vehicle and other vehicles. The TTE is defined in equation (3.14).

$$TTE(t) = \frac{distance\ to\ escape}{relative\ lateral\ speed} = \left| \frac{W_b - \Gamma_{DLt}}{\Gamma_{VLt}} \right| \tag{3.14}$$

Forward collision distance d_{FC} is calculated using the Euro NCAP AEB Test Protocol (MathWorks, 2024b) and it is presented as below:

$$d_{FC} = 1.2 * |\Gamma_{VLn}| + \frac{{\Gamma_{VLn}}^2}{2\alpha_{max}} \tag{3.15}$$

3.3.3.2 Cut-in Event

A cut-in event refers to a driving scenario where a vehicle enters the ego vehicle's lane from either the left or right lane. In situations where a cut-in event is risky, with a low Time-To-Collision (TTC) value, the driver of the ego vehicle must take immediate action to prevent a collision. This research introduces a new formula for detecting cut-in events based on sensor fusion data. In the proposed detection formula, the time range,

τ_{cut}, is between greater than zero and less than equal to twenty seconds, as researchers proposed storing data in the Event Data Recorder (EDR) for 20 seconds (NHTSA, 2022b). An unsafe cut-in event will be detected in time (t) when the MIO > 0 and $TTC_{cut} < 5$ seconds. The cut-in event has been formulated using the equations (3.13) and (3.16)–(3.18). Here, δ_L is the relative lateral distance between the ego vehicle and the other vehicle when a vehicle is entering from the left lane, whereas δ_R is the relative lateral distance from the right lane. A cut-in from the left lane (K_L) is presented in (3.19), and (3.20) presents a cut-in from the right lane (K_R). A cut-in event can be observed from the following three situations: cut-in from the opposite direction or front cut-in, cut-in from the same direction or rear cut-in, and side cut-in from another road. These three types of cut-in events can be detected by (3.19) and (3.20).

$$\tau_{cut} = \{TTC_{cut} \mid 0 < TTC_{cut} \leq 20\} \tag{3.16}$$

$$\delta_L = \{\Gamma_{DLt} \mid W_O \leq \Gamma_{DLt} \leq \lambda_L\} \tag{3.17}$$

$$\delta_R = \{\Gamma_{DLt} \mid \lambda_R \leq \Gamma_{DLt} \leq (-W_O)\} \tag{3.18}$$

$$f(K_L, t) = \begin{cases} K_L Front, if\ \tau_{cut} \wedge (MIO > 0) \wedge (\Gamma_{VLn} \leq 0) \wedge (\Gamma_{VLt} < 0) \wedge \delta_L \\ K_L Rear, if\ \tau_{cut} \wedge (MIO > 0) \wedge (\Gamma_{VLn} > 0) \wedge (\Gamma_{VLt} < 0) \wedge \delta_L \\ K_L Side, if\ \tau_{cut} \wedge (MIO > 0) \wedge (\Gamma_{VLt} < 0) \wedge \delta_L \end{cases} \tag{3.19}$$

$$f(K_R, t) = \begin{cases} K_R Front, if\ \tau_{cut} \wedge (MIO > 0) \wedge (\Gamma_{VLn} \leq 0) \wedge (\Gamma_{VLt} > 0) \wedge \delta_R \\ K_R Rear, if\ \tau_{cut} \wedge (MIO > 0) \wedge (\Gamma_{VLn} > 0) \wedge (\Gamma_{VLt} > 0) \wedge \delta_R \\ K_R Side, if\ \tau_{cut} \wedge (MIO > 0) \wedge (\Gamma_{VLt} > 0) \wedge \delta_R \end{cases} \tag{3.20}$$

3.3.3.3 Conflict Event

A conflict will arise when a vehicle attempts to enter the ego vehicle's proximity zone, thereby escalating the likelihood of a collision, which could result in an accident within a few seconds. The conflict between the colliding vehicles is formulated using Equations (3.14)–(3.15) and (3.21)–(3.23). The range of the conflicting time is τ_{con}, as presented in (21), and δ_{Cn} is the range of the longitudinal distance, as shown in (3.22). The TTE_{cn}, as presented in (3.23), is the driving situation when time-to-escape of the ego vehicle is logically checked with relative longitudinal distance between the ego vehicle and the other vehicle. A conflict event (Cn) in time t is presented in (3.24).

$$\tau_{con} = \{TTC_{con} \mid 0 < TTC_{con} \leq 5\} \tag{3.21}$$

$$\delta_{Cn} = \{\Gamma_{DLn} \mid d_{FC} > \Gamma_{DLn} \leq \delta_{prox}\} \wedge (\Gamma_{DLt} \leq 1.1 * W_b) \tag{3.22}$$

$$TTE_{cn} = (TTE(t)\ in\ \tau_{con} \wedge (\Gamma_{DLn} \leq \delta_{prox}) \wedge (\Gamma_{DLt} \leq 1.1 * W_b)) \tag{3.23}$$

$$f(Cn,t) = \begin{cases} CnFront, if\ \tau_{con} \wedge (MIO > 0) \wedge (\Gamma_{VLn} \leq 0) \wedge (TTE_{cn} \vee \delta_{Cn}) \\ CnRear, if\ \tau_{con} \wedge (MIO > 0) \wedge (\Gamma_{VLn} > 0) \wedge (TTE_{cn} \vee \delta_{Cn}) \\ CnSide, if\ \tau_{con} \wedge (MIO > 0) \wedge (TTE_{cn} \vee \delta_{Cn}) \end{cases} \tag{3.24}$$

3.3.3.4 Potential Crash Event

A potential crash refers to the moments just before a vehicle collision. Highly attentive and immediate driving reactions, such as hard braking or safe lane changing, can serve as preventive measures to avoid the collision; otherwise, this event will escalate into a confirmed crash. As part of safety driving assistance, this event will trigger a life-saving alarm to facilitate the last possible reactive measure to prevent a collision. The range of a potential crash time is τ_{pcr}, as presented in (3.25). The δ_{pcr} is the potential crash measuring conditions, as presented in (3.26), where the maximum relative longitudinal distance can be double the length of the other vehicle and the relative lateral distance can be up to W_b. The potential crash event is formulated using equations (3.25) and (3.26). A potential crash can be detected using Equations (3.27).

$$\tau_{pcr} = \{TTC_{pcr} \mid 0 < TTC_{pcr} \leq 2\} \tag{3.25}$$

$$\delta_{pcr} = (L_O < \Gamma_{DLn} \leq 2L_O \wedge \Gamma_{DLt} \leq W_b) \tag{3.26}$$

$$f(PCr,t) = \begin{cases} PCrFront, if\ \tau_{pcr} \wedge (MIO > 0) \wedge (\Gamma_{VLn} \leq 0) \wedge \delta_{pcr} \\ PCrRear, if\ \tau_{pcr} \wedge (MIO > 0) \wedge (\Gamma_{VLn} > 0) \wedge \delta_{pcr} \\ PCrSide, if\ \tau_{pcr} \wedge (MIO > 0) \wedge \delta_{pcr} \end{cases} \tag{3.27}$$

3.3.3.5 Crash Event

A crash event is a trajectory event that occurs when there is insufficient time for a driver to respond as reactive measures in order to avoid a crash. According to Čulík et al., the average reaction time for an attentive driver is approximately 0.8 seconds (Čulík et al., 2022). Consequently, in the case where the TTC is less than or equal to 1 second

($TTC_{cr} \leq 1$), no driver can reflect successfully to perform any countermeasure to avoid a collision. Based on this concept, τ_{cr} is defined as shown in (3.28). The δ_{cr} is the crash measuring conditions, as presented in (3.29), where the relative longitudinal distance Γ_{DLn} is less than or equal to the length of the other vehicle, and the relative lateral distance Γ_{DLt} can be up to half of the W_b. The crash event detection is constructed using (3.28) and (3.29), and it is presented in Equation (3.30).

$$\tau_{cr} = \{TTC_{pcr} \mid 0 \leq TTC_{cr} \leq 1\} \tag{3.28}$$

$$\delta_{cr} = (\Gamma_{DLn} \leq L_O \wedge \Gamma_{DLt} \leq 0.5 * W_b) \tag{3.29}$$

$$f(Cr,t) = \begin{cases} CrFront, if\ \tau_{cr} \wedge (MIO > 0) \wedge (\Gamma_{VLn} \leq 0) \wedge \delta_{cr} \\ CrRear, if\ \tau_{cr} \wedge (MIO > 0) \wedge (\Gamma_{VLn} > 0) \wedge \delta_{cr} \\ CrSide, if\ \tau_{cr} \wedge (MIO > 0) \wedge \delta_{cr} \end{cases} \tag{3.30}$$

The formulas presented in this section are the basis for developing models and methods for this thesis. The following section presents a component-level system design.

3.4 The Vehicle Crash Reconstruction and Prediction Model (VCRPM)

Vehicle trajectory refers to a vehicle's location, speed, acceleration, and jerk over time (Jacob & Violette, 2012). Trajectory-based crash events labelling involves classifying the occurrence (true or false) of a vehicle's specific manoeuvre at a given moment or shortly before or during a crash. This research proposes an evidence-based, data-driven Vehicle Crash Reconstruction and Prediction Model (VCRPM). The VCRPM is a crash reconstruction and real-time crash prediction approach that leverages sensor data from AVs. The proposed VCRPM has four primary objectives: (1) collecting AV sensor data, (2) reconstructing vehicle crashes using AV sensor data, (3) generating training datasets for machine learning, and (4) real-time crash prediction using the generated datasets. The component diagram of the proposed VCRPM is shown in Figure 3.4. This diagram consists of four components, each designed to achieve the objectives outlined above.

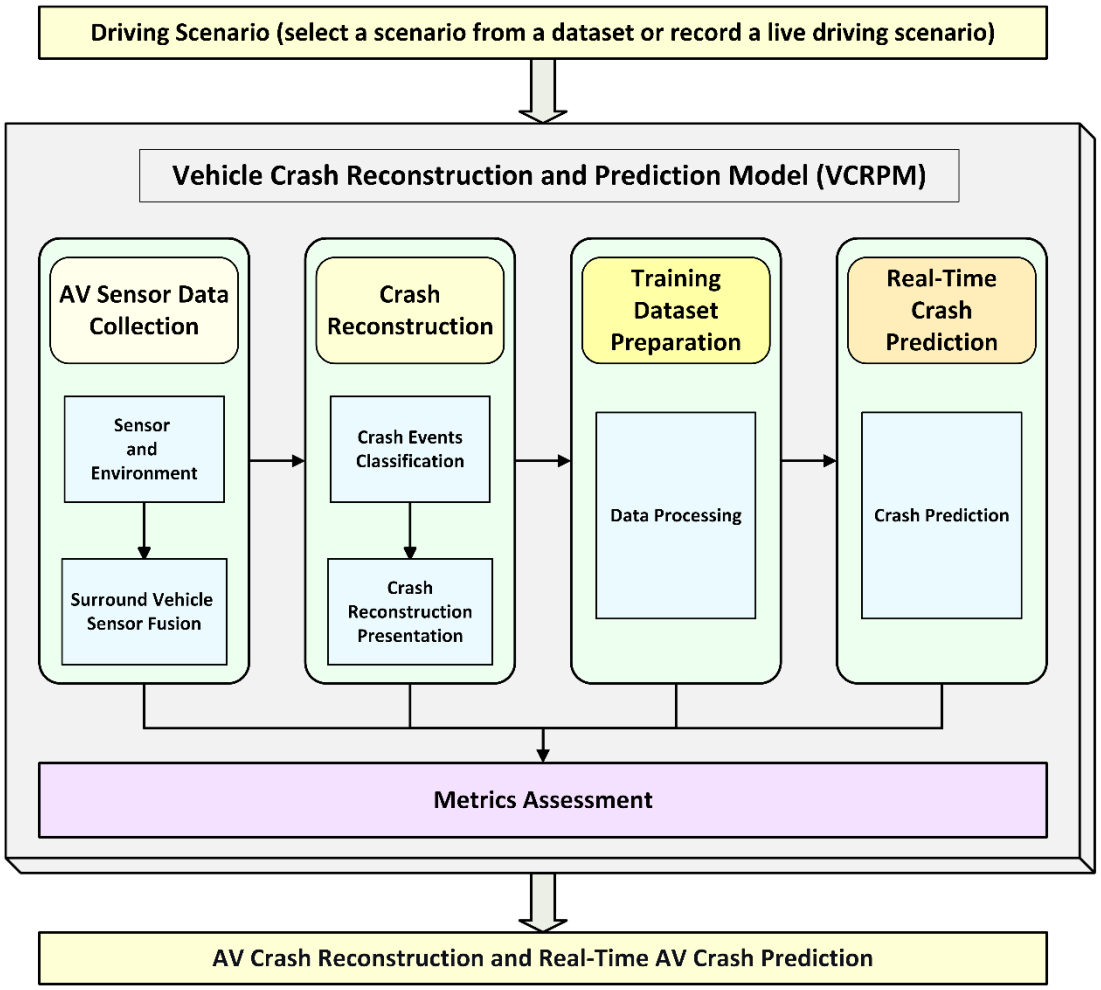


Figure 3.4: The component diagram of the Vehicle Crash Reconstruction and Prediction Model (VCRPM).

To begin with, the VCRPM receives a driving scenario from a simulated driving scenario dataset. It is also designed to record live driving scenarios, processing them in real-time within the model. The AV sensor data collection component kicks off by initialising the sensors and environment, effectively simulating the driving scenario. Within this component, a surround-vehicle-sensor-fusion subsystem integrates AV sensor data using an advanced multitracking sensor architecture, providing precise tracking information. These collected data, combined with the tracking insights, are then processed in the crash reconstruction component.

The primary role of the crash reconstruction component is to detect and classify both pre-crash and crash events, utilising the AV sensor data to reconstruct the event timeline. A key feature of this component is its ability to visually present the animated crash scenario, a valuable tool for thorough crash investigations. The third component of the VCRPM focuses on preparing the training dataset. Here, processed data from the AV sensors are recorded from each driving scenario and consolidated to form a training dataset for machine learning.

Finally, the real-time crash prediction component brings the system to life. The dataset prepared in the previous step serves as the foundation for data-driven crash prediction, allowing the model to forecast potential accidents in real-time. Further details of the VCRPM components are elaborated in Chapters 4 to 6.

All component results are evaluated in the metrics assessment block of the model. The experiment results of VCRPM are evaluated using the evaluation metrices presented in the next section.

3.5 Evaluation Metrics

Previous sections of this chapter described the foundation of the proposed system for this research and a component-level system design. This section is the basis for evaluating the proposed system.

3.5.1 Generalised Optimal Sub-Pattern Assignment Metric

The Generalised Optimal Sub-Pattern Assignment (GOSPA) metric, introduced as a Simulink block in MATLAB version R2021a, plays a pivotal role in assessing the performance of tracking algorithms. This powerful tool evaluates tracking effectiveness by taking as input both the tracks generated from object detections and the known truth values. Beyond delivering the GOSPA score, it also offers insightful metrics such as switching errors, localisation errors, missed target errors, and false track errors. The algorithms of the GOSPA metric calculation (MathWorks, 2021a) are provided below.

Equation (3.31) contains a set of truths at time t_k, whereas Equation (3.32) represents a list of tracks at time t_k. The GOSPA metric needs three input parameters to evaluate the tracking performance. These are confirmed tracks produced by a tracker, converted coordinates (positions) of the target vehicle (from the vehicle's coordinates to world coordinates), and ego vehicle's positions (world coordinates). The truth data, such as the coordinates of the target vehicle and ego vehicle, are extracted from the trajectory scenario by the vehicles, whereas a tracker produces the list of tracks in the sensor fusion system. So, the truth date of Equation (3.31) and the list of tracks of Equation (3.32) are provided by the above methods. These tracks and truths are used as the input to calculate the GOSPA as shown in Equation (3.33).

$$X = (x_1, x_2, \dots, x_m) \tag{3.31}$$

$$Y = (y_1, y_2, \dots, y_n) \tag{3.32}$$

$$GOSPA = [\sum_{i=1}^{m} d_c^p \left(x_i, y_{\pi(i)}\right) + \frac{c^p}{\alpha}(n-m)]^{1/p} \tag{3.33}$$

Here, $m \leq n$, d_c is the cutoff-based distance, $y_{\pi(i)}$ represents the track assigned to truth x_i, and α denotes the error due to cardinality mismatch. The cutoff-based distance d_c is defined in Equation (3.34) as:

$$d_c(x, y) = \min [d_b(x, y), c] \tag{3.34}$$

Here, c is the cutoff distance threshold, p is the order of the metric, and $d_b(x, y)$ is the base distance between the track and truth. When α = 2, the GOSPA metric can simplified as shown in Equation (3.35):

$$GOSPA = [loc^p + miss^p + false^p]^{1/p} \tag{3.35}$$

Here, localisation error (loc^p), missed targets ($miss^p$), and false targets ($false^p$) are calculated using Equations (3.36)–(3.38).

$$loc = [\sum_{i=1}^{h} d_b^p \left(x_i, y_{\pi(i)}\right)]^{1/p} \tag{3.36}$$

$$miss = \frac{c}{2^{1/p}} (n_{miss})^{1/p} \tag{3.37}$$

$$false = \frac{c}{2^{1/p}} (n_{false})^{1/p} \tag{3.38}$$

Here, h is the number of nontrivial assignments, and n_{miss} and n_{false} are the numbers of missed targets and false tracks, p is the order of the metric, c is the cutoff distance, d_b is the base distance between the track and truth, $y_{\pi(i)}$, and represents the track assigned to truth x_i.

3.5.2 Crash Events Classification and Prediction Metrics

Assessing the performance of the developed model is crucial in determining how effectively the crash reconstruction and prediction model can forecast real-time vehicle crashes. This research introduced a mechanism for processing sensor fusion data (Haque et al., 2024), enabling AV crash event labelling and real-time crash predictions using a dataset derived from crash events and pre-crash information. The classification of real-time crashes into crash or non-crash events relies on binary classification models, which are trained to predict crashes with high accuracy.

This study compared the results from the MATLAB-based model with actual crash data from the CIREN dataset (NHTSA, 2017-2023) and EDR data to validate the classification of pre-crash and crash events. Key performance metrics such as detection accuracy, timing errors, and deviations in kinematic variables (for example, velocity and distance) were analysed. Additionally, the simulated crash outcomes from the MATLAB-based model were compared to those generated by virtual CRASH software (CRASH, 2024), providing a clear picture of the model's crash event classification capabilities.

To evaluate the performance of the machine learning-based supervised learning models, this research utilised well-established metrics, including accuracy, precision, recall, and F1 score. These metrics offer valuable insights into the model's accuracy, its generalisation ability with new data, and its optimisation needs for effective crash

predictions. A comprehensive evaluation was conducted using tools like the confusion matrix, Receiver Operating Characteristic (ROC) curve, precision-recall curve, and self-explanatory Shapley value figures. This detailed performance analysis and feature importance evaluation are important for selecting the best model for predicting crash occurrences. This research evaluated the performance of the ML-based models using equations, as shown in (3.39)–(3.42).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.39}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.40}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.41}$$

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3.42}$$

The accuracy of the developed model was evaluated using (3.39), where TP and TN refer to true positive and true negative crash predictions. In contrast, FP and FN denote false positive and false negative predictions of crash occurrence. A model's precision and recall values can be calculated using (3.40) and (3.41), respectively. The recall value explains how correctly a model can detect all positive instances, whereas a precision value refers to a model's ability to accurately detect only relevant occurrences. The F1–score is a harmonised mean value of recall and precision, which helps to understand the crash occurrence prediction performance by combining precision and recall values. The F1–score can be calculated using the precision and recall values presented in (3.42).

The investigation of multitracking sensor architectures is described in Chapter 4, based on the formulas provided in subsection 3.3.1. Tracking performance evaluation is performed based on the GOSPA metric described in subsection 3.5.1.

3.6 Summary

The research focus of this thesis was narrowed to address four specific types of V2V crashes: front, head-on, rear-end, and side-impact crashes. The mathematical foundation for this research was presented by defining object tracking and developing an algorithm for MIO detection. Crash event detection and classification processes were defined to generate appropriate data using AV sensors, enabling data-driven crash reconstruction and prediction. The component diagram of the proposed model for VCRPM was presented, showcasing the model's architecture.

Lastly, this chapter outlined the evaluation metrics that will be applied in the subsequent chapters of this thesis. The object tracking formulas presented in this chapter serve as the foundational basis for selecting the optimal sensor architecture, which is further explored in Chapter 4 to enhance the accuracy of crash reconstruction.

Chapter 4 Investigating Multitracking Sensor Architectures for Crash Reconstruction Using AV Sensors

Chapter 3 constructed a foundation for the thesis by introducing the key formulas that form the core of the proposed methods and model. It then presented the evaluation metrics important for assessing the system's overall performance. Using these principles helps collect sensor data for AV crash investigations. The aim of this chapter is to present a novel approach for selecting optimal sensor arrangements for crash investigations. A performance evaluation of the tracking system follows, accompanied by an analysis that provides insights for future improvements. In conclusion, the proposed sensor architecture successfully demonstrates its ability to provide accurate and reliable vehicle tracking data, setting a foundation for AV crash reconstruction and prediction that are described later in the thesis.

4.1 Introduction

This chapter focuses on sensor fusion-based tracking performance, using multiple sensor arrangements to generate accurate AV sensor data important for crash reconstruction and the creation of robust training datasets. The proposed method for simulation-based tracking performance (SMTPE) is presented in Section 4.2, designed to identify the optimal sensor fusion and tracking architecture arrangements. This section also presents the process flow of SMTPE, providing an overview of the experimental sequence.

Section 4.3 highlights the criteria for selecting the most suitable simulation software and vehicle crash datasets, laying the groundwork for the subsequent analyses. A description of the experiment setup for multi-sensor-based surround vehicle sensor fusion is presented in Section 4.4. subsection 4.4.1 concentrates on the arrangement of commonly used AV sensors such as radar, cameras, and LIDAR, while subsection 4.4.2 delves into the tracking architectures used to simulate diverse driving scenarios, enabling an investigation of tracking performance with AV sensor data.

This chapter performed various experiments on multitracking sensor architectures to uncover evidence-based tracking performances. The results of these experiments are showcased in Section 4.5. subsection 4.5.1 highlights the object detection outcomes from multiple sensor configurations, providing a comparison of sensor-based vehicle detection performances. In subsection 4.5.2, this research explores vehicle tracking performances across various multitracking setups.

Section 4.6 offers a synthesis of the key research findings, presenting a set of best practices for evaluating tracking performance in multi-sensor fusion studies. The chapter concludes with a summary of key insights in Section 4.7.

4.2 The Proposed Method

4.2.1 Background Study

Perception stands as one of the five key components of AV navigation and autopiloting, alongside localisation, planning, decision-making, and dynamic vehicle control (Brummelen et al., 2018; Cheng, 2011). Much like the way human drivers sense their surroundings, an AV's perception system relies on sensors to detect objects within its environment. Yet, like any system, sensor perception has its boundaries. These limitations can be overcome by fusing data from multiple sensors, boosting overall performance (Xiang et al., 2023). It is also noted that combining radar, camera, and lidar sensors can significantly improve object detection capabilities (Yeong et al., 2021).

The data gathered by sensors are processed in trackers, which produce motion states that enable the system to monitor and predict the movements of surrounding vehicles. These motion states include crucial parameters such as position, velocity, and orientation, all of which are necessary for the AV to make informed decisions (Brummelen et al., 2018; Cheng, 2011).

Researchers are consistently advancing tracking algorithms to enhance AV performance. For instance, Wang et al. proposed an improved target-tracking approach by incorporating optimal geometry and motion coordination in multi-sensor target tracking (Wang et al., 2023). This innovation boosted the accuracy of tracking in complex environments. Similarly, Zhang et al. developed a lane detection algorithm utilising multi-sensor fusion to enhance lane boundary tracking, particularly in curvy road scenarios (Zhang et al., 2023). Furthermore, Hou et al. tackled the challenges of crowded scenes (Hou et al., 2023), improving tracking performance by addressing issues such as lag smoothing. Meanwhile, Shah et al. made strides in tracking turning targets (Shah et al., 2022), and Choi et al. focused on optimising vehicle path tracking, further enhancing overall AV navigation capabilities (Choi et al., 2021).

MATLAB and Simulink provide AV researchers with options for testing and simulating tracking algorithms. These platforms offer two primary architecture options for implementing tracking systems: central-level tracking and track-to-track fusion. Central-level tracking is more efficient, as it receives raw object detection information directly from the vehicle's sensors, enabling faster and more accurate processing. In contrast, track-to-track fusion, part of a distributed architecture, relies on inputs from sensor-level tracks, which are then fused at a higher level. While this distributed approach is more resource-efficient, as it produces smaller data sizes compared to unprocessed sensor data, it is particularly beneficial in situations with limited computational resources, such as restricted transmission bandwidth or computation power (MathWorks, 2020, 2021b).

Given the increasing number of AV sensors and tracking architectures, AV development research can be commenced by arranging them in any combination. However, unlike papers dealing with sensors and their architectures, published research on AV crash reconstruction using multi-sensor-based sensor fusion needs an agreed-upon sensor architecture. Therefore, developing a simulation method for evaluating different architectures for AV crash reconstruction is important.

4.2.2 The Proposed Method (SMTPE)

In order to achieve a repeatable solution for tracking performance evaluation, this research proposed a Simulation Method for Tracking Performance Evaluation (SMTPE) to select the appropriate sensor fusion and tracking architecture arrangements (Haque et al., 2024), as shown in Figure 4.1. The SMTPE is derived from the research method presented in Section 1.4 of this thesis. The purpose of the SMTPE is to obtain the best experimental setup for evidence-based data-driven vehicle crash reconstruction research.

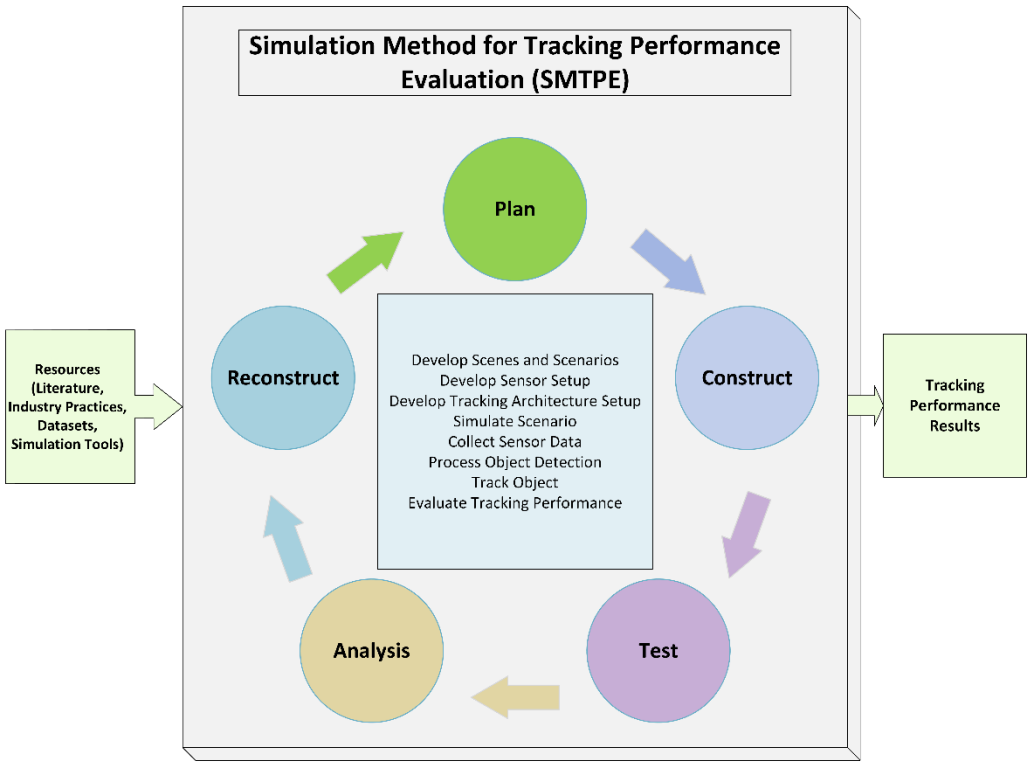


Figure 4.1: Proposed Simulation Method for Tracking Performance Evaluation (SMTPE): a repeatable solution for object tracking performance evaluation in AV development. Note: from (Haque et al., 2024) In *Sensors*.

The primary process of the SMTPE is iterative, continuing until a decision can be made based on the evaluated performance of object tracking. This cyclical process includes key stages such as planning, constructing, testing, analysing, and reconstructing, each contributing to the overall efficiency and accuracy of the simulation.

The planning stage marks the start of this iterative journey, where the initial design of the simulation is crafted. This phase involves analysing resources, such as selecting simulation tools and datasets, which will form the foundation for the simulation's structure. In the construction phase, the design is transformed into a functional model. This phase includes developing trajectory scenarios, creating the tracking architecture, and setting up sensors—all aligned with the plan using the chosen simulators.

Once the model is built, the testing phase follows, where the simulations are executed, and any errors are identified and rectified. This testing is crucial in ensuring the reliability of the simulation setup. Afterwards, the simulation results are analysed for performance evaluation, determining how well the object tracking functions in different scenarios.

The reconstruction stage forms the final element of this cycle. It ensures that the simulation can be reliably repeated under similar conditions and helps to enhance understanding of complex scenarios, such as AV crash scenarios. The purpose here is to validate that the design works as expected and can produce consistent, repeatable results.

The iterative process concludes if an optimal tracking solution is found during the analysis. However, if the desired performance is not yet achieved, the process loops back to earlier stages to refine and improve the design. Through these continuous iterations, SMTPE performs the tasks of developing simulation scenes, scenarios, sensor setups, and tracking architectures, as well as running simulations, collecting sensor data, processing object detection, and evaluating tracking performance. The following subsection will delve deeper into the specifics of each phase within the SMTPE process.

4.2.3 Process Flow of SMTPE

Figure 4.2 illustrates the process flow for SMTPE. The process begins by gathering the relevant datasets or recording data from a real-world scenario. Once the trajectory data are analysed, the next step involves planning the creation of the required scenes or scenarios. In this phase, the ego vehicle, other vehicles, roads, and additional objects (such as pedestrians and cyclists, if necessary) are modelled consistently with the acquired dataset. A trajectory scenario includes multiple objects, and it is important to assign the correct properties to these objects to ensure more accurate and reliable experimental outcomes.

Precise knowledge is needed to deal with sensing the surroundings in the different environments of the trajectories. Indeed, expertise on the advantages and limitations of sensors is also important to dealing with real-world environments such as perception at night, rain, fog, or driving on a mountainous road. Similarly, experimenting with different sensors (such as radar, camera, and LIDAR) with multiple orientation-based setups can help select the best experimental setup for planned research. Concurring with the above statements, SMTPE's step of developing multiple sets of sensor setups generates data to feed in for the next step.

Two common types of Tracking Architecture (TA) are used in multi-sensor tracking systems: central-level TA uses sensor detections directly, while track-to-track fusion TA performs fusion at the track level (MathWorks, 2021b). These TAs help to experiment with different trackers (a tracker tracks target objects by following their movements) and algorithms that produce target tracking. The core processing occurs within the sensor fusion subsystem during the TA step, where the outputs of different trackers and the performance of tracking algorithms are compared. Therefore, organising a set of TAs for experiments is more effective than testing with a single TA, as it helps in selecting a better tracking architecture overall.

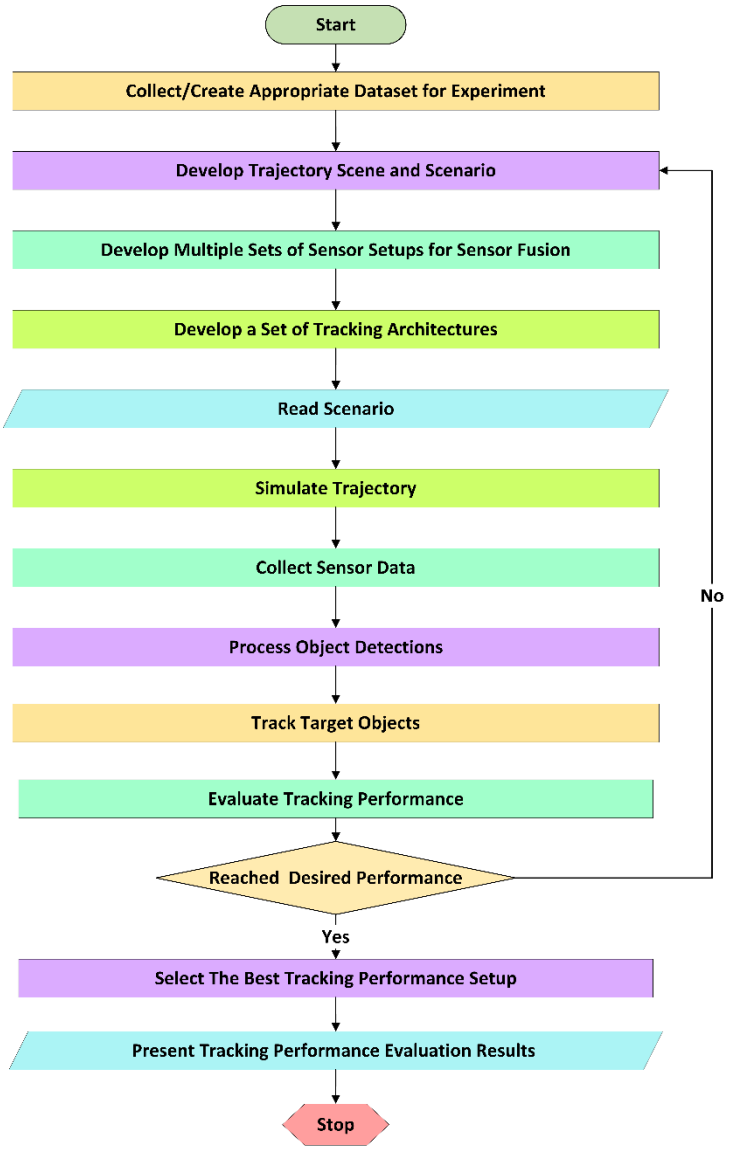


Figure 4.2: The process flow of SMTPE: selecting the best tracking performance setup for AV crash reconstruction development. Note: from (Haque et al., 2024) In *Sensors*.

A trajectory scenario reading is important for operating a simulator. As a result, the scenario reading and simulation steps need to be carried out sequentially to gather simulated data from the reconstructed scenario.

The primary function of a sensor is to detect and identify objects. These object detections require further processing before being input into a tracker. This processing includes aggregating detections from multiple sensors and incorporating localisation

information. In SMTPE, this processing is handled during the object detection step.

Selecting the appropriate tracking algorithm and object tracker requires insight into the tracking algorithm and tracker functionalities. Experimenting with different combinations of tracking algorithms and trackers allows for the selection of a more effective TA. The track-target-object step of the SMTPE offers this flexibility. In addition, the simulation performance evaluation of object tracking needs to be iterative to reach a desired performance so that there is confidence that enough information has been obtained from the scenario and trajectory scenes for setup evaluation to proceed. A tracking metric is important for this performance assessment and needs to be understood and researched to achieve the desired performance outcomes. For instance, the GOSPA metric is commonly used in various studies, and researchers are continuously working to enhance its features (García-Fernández et al., 2020; Rahmathullah et al., 2017; Su et al., 2024).

The optimal tracking setup is determined by evaluating various experimental configurations against a specific performance threshold. This research has set a target tracking accuracy of 90% for all experiments. The configurations are systematically ranked to identify the most effective setup. When the evaluation cycle meets the required performance level, the results from SMTPE's evaluation and ranking phases are presented to the researcher. If the performance criteria are not met, the pursuit of the optimal architecture will return to the scenario development phase of SMTPE. This iterative process continues until optimal results are achieved.

4.3 Selecting Simulator and Vehicle Crash Dataset

The development of ADS is progressing rapidly, highlighting the need for experimentation and evaluation to ensure compliance with safety standards before deploying AVs. While naturalistic field operation tests are valuable during the early stages of AV development, they are often time-consuming and expensive, particularly when dealing with infrequent crash events (Alghodhaifi & Lakshmanan, 2021).

Simulation-based testing is important for ensuring AV safety, as it helps reduce the time required for testing, minimises driving risks (Ghafarian et al., 2023), lowers costs, decreases insecurity (Zhou et al., 2022), and cuts down on labour needed (Li et al., 2024). With the growing number of simulators available for designing models and collecting data for AV development, it is essential to select the most effective simulator for replicating real-world scenarios (Rosique et al., 2019).

A recent review of simulators for autonomous driving decisively categorised them into four main types: traffic flow simulators, sensor data simulators, driving policy simulators, and vehicle dynamics simulators (Li et al., 2024). According to Li et al., traffic flow simulators were important for the development of transportation systems, while the other three types were crucial for advancing ADS. A robust simulator effectively integrated two or more of these functionalities. The overall effectiveness of ADS simulation was significantly enhanced by combining perception (through sensor data simulators), planning (via driving policy simulators), and control (with vehicle dynamics simulators).

A primary objective of this research is to collect AV sensor data from various crash scenarios, requiring a simulator to generate trajectory-based sensor data and simulate vehicle dynamics. To this end, this research analysed recent review papers that compare AV driving simulators. The findings of this analysis are summarised in Table 4-1.

This study reviewed six popular autonomous driving simulators, three of which are open-source: CARLA (Dosovitskiy et al., 2017), LGSVL (Rong et al., 2020), and AirSim (Shah et al., 2018). The other three simulators require commercial licenses: MATLAB-Simulink, PreScan, and CarMaker (Alghodhaifi & Lakshmanan, 2021; Kaur et al., 2021; Li et al., 2024; Zhou et al., 2022). This study also reviewed the features of prebuilt tools available in these simulators, which facilitate rapid simulation development, such as the navigation toolbox (MathWorks, 2024c) and automated driving toolbox (MathWorks, 2024a).

Table 4-1: Functionality comparison between six AV simulators.

License	Simulator	Supported Sensors	Model Design	Planning	Vehicle Dynamics	ADS Toolbox
Open source	CARLA	Radar, camera, LIDAR, GPS, IMU	N	Y	N	N
	LGSVL	Radar, camera, LIDAR, GPS, IMU	Y	N	Y	N
	AirSim	Camera, LIDAR, GPS, IMU	Y	N	N	N
Commercial	MATLAB -Simulink	Radar, camera, LIDAR, GPS, IMU, ultrasonic	Y	Y	Y	Y
	PreScan	Radar, camera, LIDAR, GPS	Y	N	Y	N
	CarMaker	Radar, camera, LIDAR, GPS, ultrasonic	Y	N	Y	N

Note: The term 'Y' refers to supported functionality by the simulator and 'N' means that the simulator does not support this function(MathWorks, 2024a) (Alghodhaifi & Lakshmanan, 2021; Kaur et al., 2021; Li et al., 2024; MathWorks, 2024c; Zhou et al., 2022).

This research selected a suitable crash dataset to evaluate the performance of evidence-based multitracking sensor architectures based on two main criteria. First, the dataset must include a detailed description of the crash scene, including a narrative accident and a map of the crash area with a depiction of the vehicle's movements. Second, it should capture pre-crash events and vehicle dynamics, allowing for an in-depth analysis of the sequence leading to the crash. Datasets incorporating Event Data Recorder (EDR) information are particularly valuable, as they can deliver insights into pre-crash dynamics.

In this endeavour, several datasets were scrutinised, including the CA-DMV (DMV, 2024), the ADAS crashes (NHTSA, 2022c), the ADS crash dataset (NHTSA, 2024), and the CIREN dataset (NHTSA, 2017-2023). Notably, the CIREN dataset stands out as it fully satisfies both criteria, making it the ideal choice for the evidence-based experiments conducted in this thesis. By utilising this dataset, a robust foundation for our research can be ensured, leading to more effective safety improvements in AV technology.

4.4 Experimental Setup for Multi-Sensor-Based Surround Vehicle Sensor Fusion

This research developed a proof of concept for the proposed SMTPE using MATLAB (R2023b), Simulink (R2023b), and various associated tools, including the Driving Scenario Designer app, Navigation Toolbox, and Automated Driving Toolbox. The experiments were conducted on a machine running Windows 11 Pro (64-bit operating system, version 23H2), equipped with an 11th Gen Intel Core i7 processor and 16 GB of RAM. The primary objective of the simulated experiments in this study was to evaluate tracking performance in different crash scenarios. The Driving Scenario Designer app in MATLAB was utilised to simulate these crash scenarios using the CIREN dataset (NHTSA, 2017-2023).

All the sensor arrangements are first tested using the Driving Scenario Designer app and then simulated using the Simulink software. Three types of crash scenarios were simulated from the CIREN dataset. These were:

- Head-on crash (CIREN accident no 664 or CIREN-664);
- Rear-end crash (CIREN accident no 816 or CIREN-816);
- Side-impact crash (CIREN accident no 226 or CIREN-226).

The experimental setup subsection was split into sensor setup and tracking architecture setup.

4.4.1 Sensor Setup

AVs typically rely on an array of sensors to perceive their environment, with radar, cameras, and LIDAR being the most prevalent technologies employed to scan the surroundings. Each sensor brings unique strengths to the table but also comes with its own set of limitations. These challenges can be mitigated through the fusion of multiple sensors, enhancing overall performance. By combining radar, cameras, and LIDAR, sensor fusion can significantly elevate the driving capabilities of AVs. This study

explored various configurations of these sensors, creating two distinct setups for multi-sensor-based surround vehicle sensor fusion. The specific setups created for this multi-sensor-based surround vehicle sensor fusion were as follows:

Sensor Setup 1 (S1): The sensor setup 1 or S1 consisted of three radar sensors, five cameras, and one LIDAR sensor. The camera sensing fully covered the ego vehicle's surroundings, which was achieved by positioning five cameras in the ego vehicle. Three radars were positioned: one at the front, one at the front left side of the vehicle, and another at the front right side of the vehicle. The LIDAR was placed at the centre of the ego vehicle. These sensor setups are shown in Table 4-2.

Table 4-2: Sensor setup 1 (S1): the ego vehicle's sensor arrangements for the surround vehicle sensor fusion. Note: from (Haque et al., 2024) In *Sensors.*

Sensor	Location	Position (m) [x, y, z]	Rotation (°) [Roll, Pitch, Yaw]	Max. Range (m)
Radar	Front	[3.7, 0, 0.2]	[0, 0, 0]	160
	Front-left	[2.8, 0.9, 0.2]	[0, 0, 45]	30
	Front-right	[2.8, −0.9, 0.2]	[0, 0, −45]	30
Camera	Front	[2.95, 0, 1.1]	[0, 1, 0]	250
	Front-left	[2, 0.9, 0.7]	[0, 1, 65]	80
	Front-right	[2, −0.9, 0.7]	[0, 1, −65]	80
	Rear-left	[2.8, 0.9, 0.7]	[0, 1, 140]	100
	Rear-right	[2.8, −0.9, 0.7]	[0, 1, −140]	100
LIDAR	Centre	[1.5, 0, 1.6]	[0 0 0]	120

The Driving Scenario Designer's view of the developed S1 is shown in Figure 4.3.

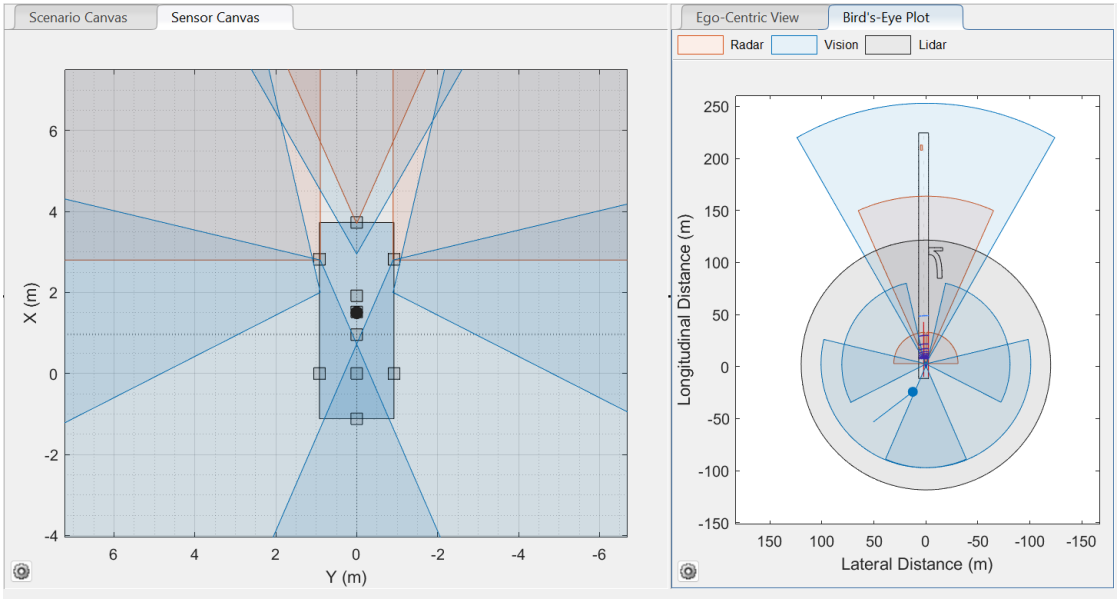


Figure 4.3: Sensor setup 1 (S1): three radars, five cameras, and one LIDAR. Note: from (Haque et al., 2024) In *Sensors*.

Sensor Setup 2 (S2): The sensor setup 2 or S2 contained six radar sensors, two cameras, and one LIDAR sensor. These sensor setups are shown in Table 4-3.

Table 4-3: Sensor setup 2: the ego vehicle's sensor arrangements for the surround vehicle sensor fusion. Note: from (Haque et al., 2024) In *Sensors*.

Sensor	Location	Position (m) [x, y, z]	Rotation (°) [Roll, Pitch, Yaw]	Max. Range (m)
Radar	Front	[1.9, 0, 0.2]	[0, 0, 0]	160
	Front-left	[2.8, 0.9, 0.2]	[0, 0, 60]	30
	Front-right	[2.8, −0.9, 0.2]	[0, 0, −60]	30
	Rear-left	[0, 0.9, 0.2]	[0, 0, 120]	30
	Rear-right	[0, −0.9, 0.2]	[0, 0, −120]	30
	Rear	[0.95, 0, 0.2]	[0, 0, −180]	160
Camera	Front	[2.1, 0, 1.1]	[0, 1, 0]	150
	Rear	[0.56, −0.9, 1.1]	[0, 1, −180]	150
LIDAR	Centre	[1.5, 0, 1.6]	[0 0 0]	120

The radars spanned the ego vehicle's full surroundings for the S2. The ego vehicle had one camera at the front and another at the rear, mounted to fuse with the other sensors. The LIDAR was placed at the centre of the ego vehicle.

The Driving Scenario Designer's view of the developed S2 is shown in Figure 4.4.

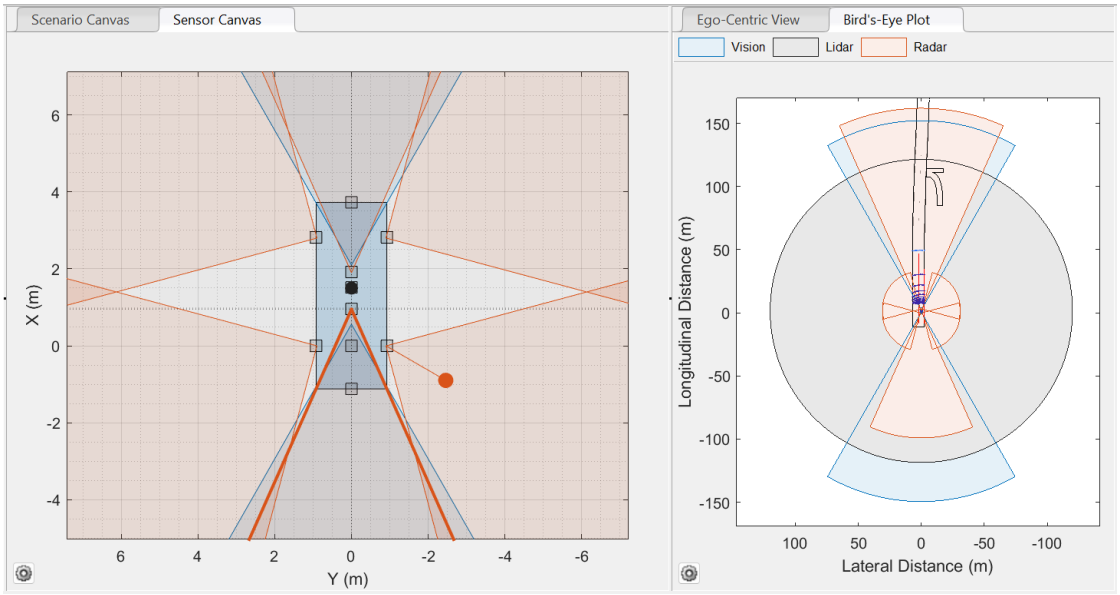


Figure 4.4: Sensor setup 2 (S2): five radars, two cameras, and one LIDAR. Note: from (Haque et al., 2024) In *Sensors*.

4.4.2 Tracking Architecture (TA) Setup

The multi-object tracker precisely follows target vehicles to deliver accurate and reliable confirmed tracks. These confirmed tracks are generated through the tracker, which includes estimated states and state covariances for each vehicle. The estimated states capture the kinematic details of the target vehicles, such as position, velocity, and orientation, while the state covariances reflect the uncertainty inherent in the filtered states. A filtering algorithm plays an important role in constructing these confirmed tracks.

When it comes to tracking architectures, a central-level tracking system unequivocally outperforms track-to-track fusion architectures. This superiority stems

from the central-level tracker's ability to directly integrate object detection data from vehicle sensors, resulting in a more efficient and precise tracking process. In contrast, a decentralised tracking architecture operates on sensor-level tracking inputs, which are later fused at the track level through a track-to-track fusion process.

The commonly used tracking architectures are illustrated in Figure 4.5, where Figure 4.5a presents the centralised tracking architecture and Figure 4.5b showcases the decentralised track-to-track fusion architecture. To further explore these two approaches, this research has developed three distinct Tracking Architectures (TAs)—two centralised and one decentralised—as detailed in Table 4-4, with variations in the sensors utilised.

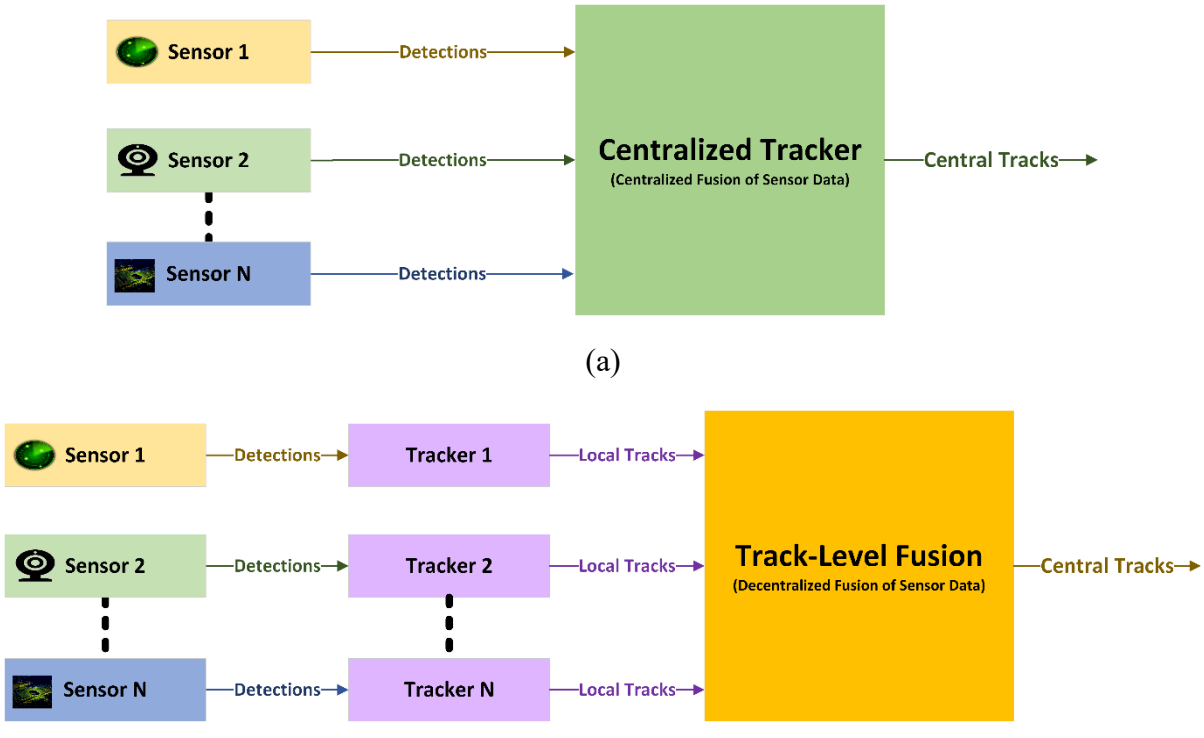


Figure 4.5: Tracking architectures: (a) centralised tracking architecture; (b) decentralised tracking architecture. Note: from (Haque et al., 2024) In *Sensors*.

Table 4-4: Tracking Architecture (TA) for the sensors' data fusion. Note: from (Haque et al., 2024) In *Sensors*

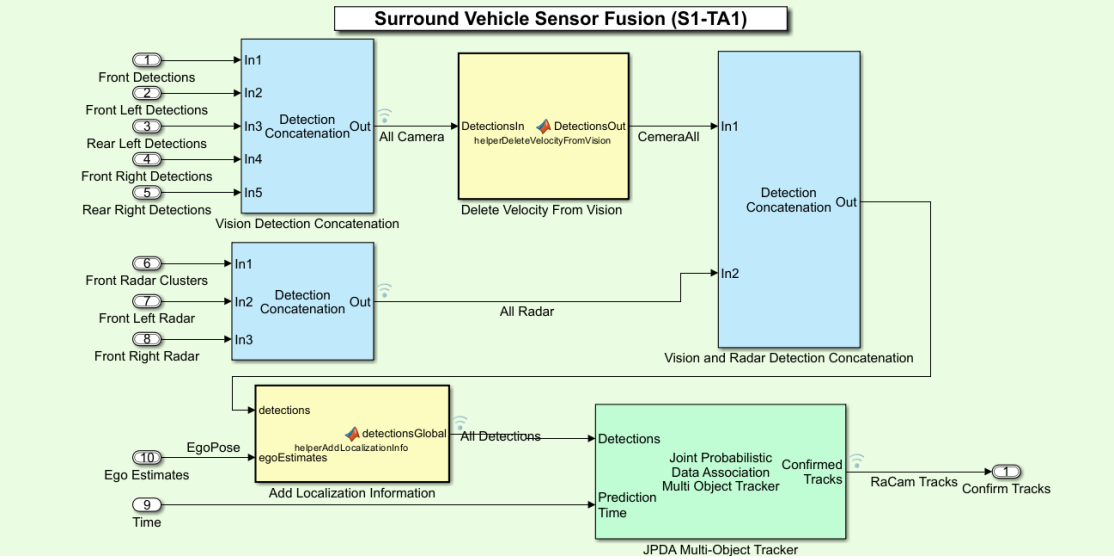
TA	Fusion Type	Tracker	Filtering Function	Sensors
TA-1	Centralised	JPDA Tracker	helperInitialiseCVEKFFilter	Radar, Camera
TA-2	Centralised	JPDA Tracker	helperInitLidarCameraFusionFilter	LIDAR, Camera
TA-3	Decentralised	JPDA, Track-To-Track Fuser	central2sensor, sensor2central	Radar, Camera, Lidar

The above three architectures and two sensor setups provided a unique opportunity for this research to explore six combinations of sensor fusion and tracking architectures. The two Simulink-based tracking architectures are showcased in Figure 4.6.

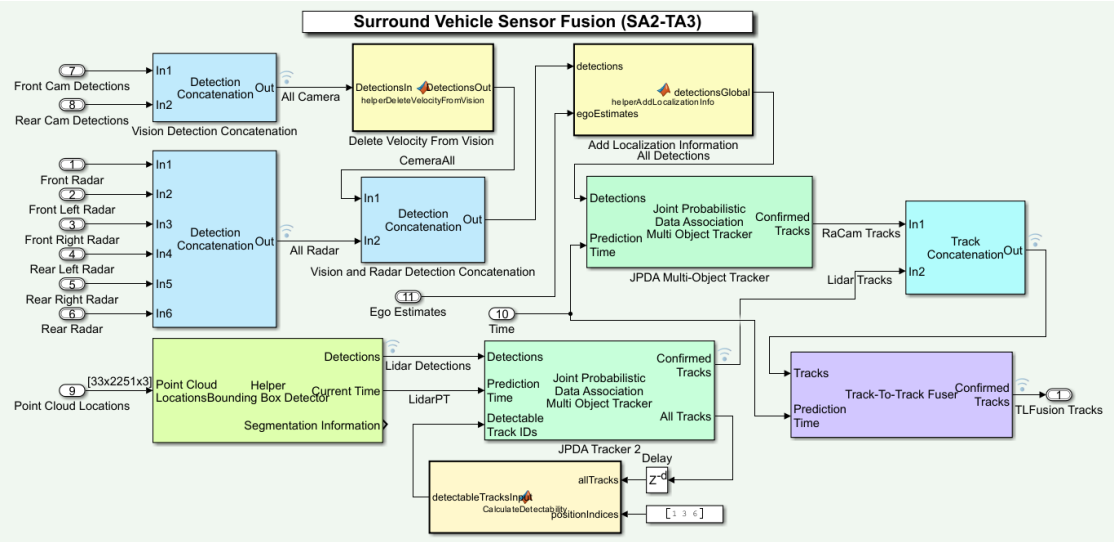
For illustrative purposes, this research highlights two TAs from a total of six combinations. The centralised TA is depicted in Figure 4.6a, while the track-to-track-level TA is presented in Figure 4.6b. In Figure 4.6a, the Joint Probabilistic Data Association (JPDA) tracker plays a crucial role in generating the confirmed tracks during a trajectory simulation.

The process begins with the surround vehicle sensor fusion system, which receives sensor data as input from the driving scenario. The system then accumulates and forwards various sensor data signals through a signal bus. Using Simulink's built-in concatenation blocks, five camera detections and three radar sensor readings are integrated using two detection concatenation blocks. Prior to fusing the camera and radar detections, a custom function ensures that the camera data are processed without velocity information.

Once the data fusion is complete, the fused information—enhanced with localisation details via a custom function—is sent to the JPDA tracker. This tracker then applies the formulas (3.1)−(3.10) outlined in subsection 3.3.1 of this thesis to produce the confirmed tracks at each simulation step.



(a)



(b)

Figure 4.6: Simulink-based tracking architecture design: (a) centralised tracking architecture using radar and camera (S1-TA1); (b) decentralised tracking architecture using radar, cameras, and LIDAR (S2-TA3). Note: from (Haque et al., 2024) In *Sensors.*

A similar data flow process with three additional steps is outlined in Figure 4.6b. The first step generates LIDAR detections from the LIDAR point cloud data. The second step uses another JPDA tracker to generate confirmed tracks. In the final step, the LIDAR and radar-camera tracks are merged using a track concatenation block, and the fused tracks are passed to a track-to-track fuser to create the central tracks.

In the Simulink model, the signal from a detected object is forwarded to a multi-object tracker designed to track target vehicles surrounding the ego vehicle. Simulink offers an array of multi-sensor-based multi-object trackers (MathWorks, 2019b), each serving distinct tracking needs. Among these, the JPDA tracker and the Probability Hypothesis Density (PHD) tracker stand out as centralised trackers, ideal for monitoring the movement of vehicles using multiple sensors. Meanwhile, the track-to-track fuser facilitates decentralised sensor fusion, ensuring smooth integration of tracking data.

This study focused on conducting experiments with the JPDA tracker for centralised tracking and the track-to-track fuser for track-level fusion. The JPDA tracker leverages the helperInitialiseCVEKFFilter algorithm for radar and camera fusion, while LIDAR and camera fusion use the helperInitLidarCameraFusionFilter algorithm for precise object detection filtering. The helperInitialiseCVEKFFilter is a refined version of the constant-velocity extended Kalman filter (initcvekf) function provided by MATLAB, specifically adapted to handle the data from fast-moving vehicles by adjusting the velocity covariance for more accurate tracking. A deeper dive into the extended Kalman filter can be found in subsection 3.3.1. To complement the fusion process, two tracking algorithms—central2sensor and sensor2central—were implemented to optimise track-level fusion and enhance overall system performance.

The following section delves into the results of the comprehensive examinations, offering further insight into the performance and accuracy of the proposed methods.

4.5 Results

The proposed SMTPE was validated through experimental simulations of three distinct crash scenarios. .Using the CIREN (NHTSA, 2017-2023) dataset, this research simulated three different vehicle crash types: head-on crash (CIREN-664), rear-end crash (CIREN-816), and side-impact crash (CIREN-226). These simulations were designed to assess the tracking performance of SMTPE under real-world conditions. This study simulated these crash scenarios using tools such as MATLAB, Simulink,

and the Driving Scenario Designer app, along with the Navigation Toolbox and Automated Driving Toolbox. The simulations were run with three different sensor update rates—200 ms, 100 ms, and 50 ms—to examine the impact on multi-sensor fusion data sizes to provide a comprehensive analysis. The resulting data, summarised in Table 4-5, provide valuable insights into the tracking performance across varying sensor frequencies. Key findings from this experiment are as follows:

Table 4-5: Generated sensor-based data (data size in KB) from simulations of vehicle crashes using the CIREN dataset (NHTSA, 2017-2023). Note: from (Haque et al., 2024) In *Sensors.*

Sensor Update Rate (ms)	Sensor Setup	TA	Sensor Data (KB) CIREN-664	Sensor Data (KB) CIREN-816	Sensor Data (KB) CIREN-226
200	S1	TA-1	584.63	326.90	1264.81
		TA-2	179.57	283.05	237.63
		TA-3	567.09	339.10	1228.31
	S2	TA-1	421.79	883.29	1374.46
		TA-2	136.50	196.90	136.31
		TA-3	436.51	875.04	1370.06
100	S1	TA-1	929.62	654.08	2551.38
		TA-2	346.61	540.92	456.40
		TA-3	958.03	610.84	2457.92
	S2	TA-1	852.00	1683.90	2800.71
		TA-2	263.99	380.89	264.44
		TA-3	897.13	1692.71	2827.99
50	S1	TA-1	1861.59	1300.67	4980.17
		TA-2	698.62	1077.75	887.63
		TA-3	1877.60	1212.71	4993.81
	S2	TA-1	1664.29	3422.20	5491.92
		TA-2	533.18	757.50	515.96
		TA-3	1705.84	3389.04	5580.13

- The 200 ms sensor update rate yielded the smallest data size and provided the fastest processing across all planned sensor setups, but it resulted in a few missed object detections.
- In contrast, the 50 ms sensor update rate generated the largest data size, allowing for more detailed detections of the simulated environment scenarios.
- Among the three tracking architectures, TA3 (track-to-track fusion using radar, cameras, and LIDAR) required the highest processing time when compared to TA1 (centralised fusion using radar and cameras) and TA2 (centralised fusion using cameras and LIDAR).

Overall, the 100 ms update rate delivered the necessary object detections while maintaining optimal data sizes.

The data generated from the 100 ms sensor update rate are analysed and showcased in Figure 4.7. Across all scenarios, S2-TA2 (centralised fusion utilising sensor setup 2, cameras, and LIDAR) consistently produced the smallest data volumes, while TA3 (decentralised fusion with radar, cameras, and LIDAR) resulted in the largest data outputs. A key takeaway from this research is that the data sizes from the simulated scenarios varied significantly and were influenced by each scenario's specific environment and the trajectory time duration.

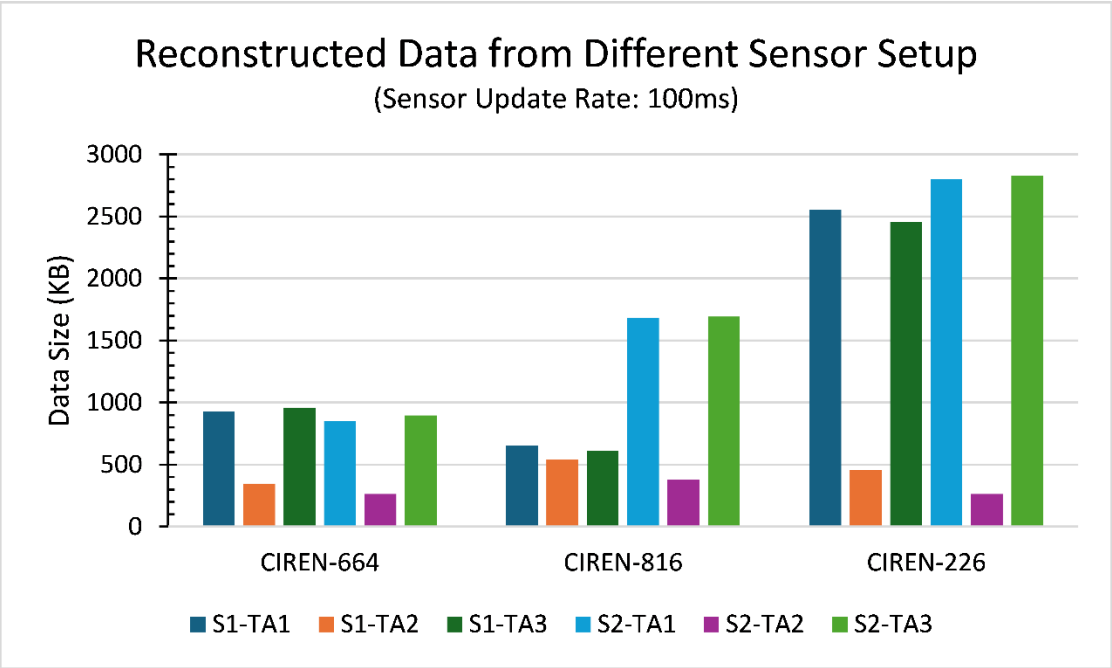


Figure 4.7: Simulated data from multiple sensor arrangements with a 100 ms sensor update rate for CIREN accidents ID 664, 816, and 226. Note: from (Haque et al., 2024) In *Sensors*.

4.5.1 Multi-Sensor-Based Object Detection and Evaluation

Object detection using sensor data plays a pivotal role in the success of ADS. In today's advanced AV technology, sensors are employed to achieve robust environmental perception. Camera sensors are utilised to classify various objects, such as vehicles, road obstacles, and lane boundaries. Radar excels in generating kinematic information about surrounding vehicles, while LIDAR provides high-density Three-Dimensional (3D) point clouds, enabling precise 3D object detections. Object detection is significantly enhanced by fusing these three sensor types, offering a more comprehensive view of the environment. Before proceeding with sensor fusion, this research first compared the object detection capabilities of each sensor in different configurations across three crash scenarios. The object detection results for CIREN-664 (head-on crash) are illustrated in Figure 4.8.

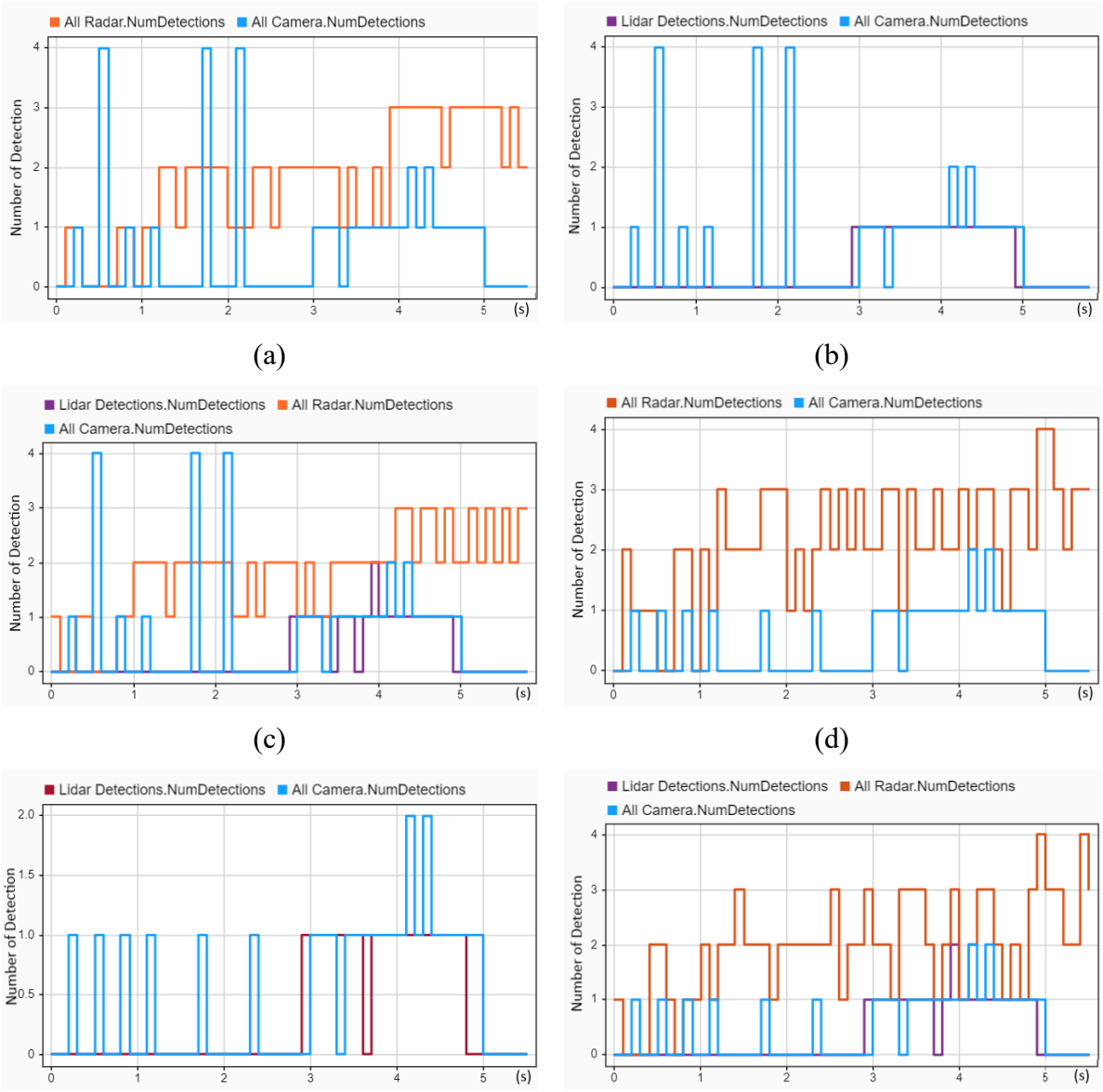


Figure 4.8: Object detections of the CIREN-664 (head-on crash) scenario using different sensors and tracking architectures setups: (a) used S1-TA1; radars have few zero object detections from 0 to 1 s, and cameras have higher zero object detections between 0 and 3 s. (b) used S1-TA2; LIDAR has object detections only from 2.9 to 4.9 s. (c) used S1-TA3; camera object detections remain the same as (a, b), whereas radars and LIDAR object detections decrease. (d) used S2-TA1; object detection gaps are decreased by radars, and object detection gaps are increased by cameras. (e) used S2-TA2; cameras have only a few object detections up to 3 s, and LIDAR can detect objects from 2.9 to 4.8 s. (f) used S2-TA3; radar object detections decrease, whereas LIDAR and camera object detections remain the same as (d, e). Note: from (Haque et al., 2024) In *Sensors.*

In Figure 4.8a–c, the object detections for TA1, TA2, and TA3 using sensor setup one are illustrated, while Figure 4.8d–f present the object detections of the same targets using sensor setup 2. This comparison highlights intriguing patterns in detection performance across different sensor configurations. It was found that radar sensors generally exhibited fewer detection gaps compared to camera sensors, which had more noticeable gaps in the data. Multiple detections from the same sensor type were also observed, such as in Figure 4.8a, where camera detections reached as many as four due to the five cameras in sensor setup S1. Likewise, radar detections in Figure 4.8d showed several instances of four-object detections stemming from the six radar units in sensor setup S2. Another noteworthy finding in the data was the behaviour of LIDAR, where the object detection results were comparable for both setups, S1 and S2, but the detection range was shorter than that of the other sensors.

Further, in the simulation of the CIREN-816 rear-end crash scenario (Figure 4.9), radars demonstrated more frequent gaps in object detection due to their positioning at the front, front-left, and front-right sides of the ego vehicle, as shown in Figure 4.9a. In contrast, when three radars were positioned at the rear of the ego vehicle, as depicted in Figure 4.9d, there were no gaps in detection throughout the trajectory. Lastly, an interesting observation emerges from the simulated CIREN-226 side-impact crash scenario in Figure 4.10, where LIDAR outperformed other sensors, showing more frequent object detections, further emphasising its unique capabilities in specific crash scenarios.

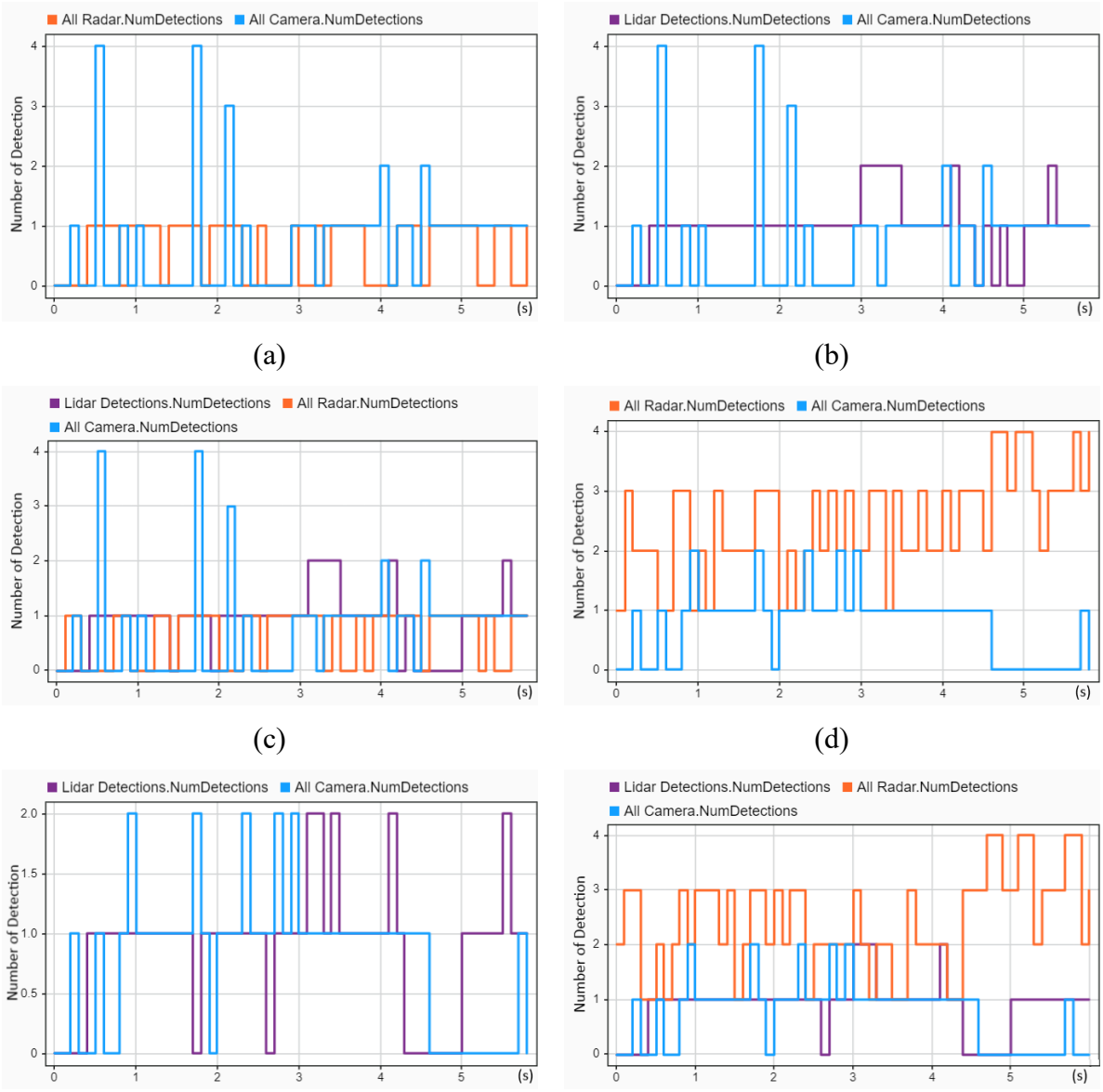


Figure 4.9: Object detections of the CIREN-816 (read end crash) scenario using different sensors and tracking architectures setups: (a) used S1-TA1; front-side radars have many zero object detections from 2 to 4 s, whereas cameras have zero object detections between 1.2 and 2.8 s. (b) used S1-TA2; LIDAR has only two zero object detections at 4.6 and 4.8 s, whereas cameras have more zero object detections between 1 and 3 s. (c) used S1-TA3; object detections from radars and cameras are almost the same as (a, b), but LIDAR object detections decrease. (d) used S2-TA1; radars have no zero object detections, whereas cameras have zero object detections from 4.5 to 5.8 s. (e) used S2-TA2; cameras have zero object detections from 4.5 to 5.8 s, whereas LIDAR is unable to detect objects between 4.3 and 5 s. (f) used S2-TA3; radar and camera

object detections are almost the same as (d, e), whereas LIDAR has fewer zero object detections than (e). Note: from (Haque et al., 2024) In *Sensors.*

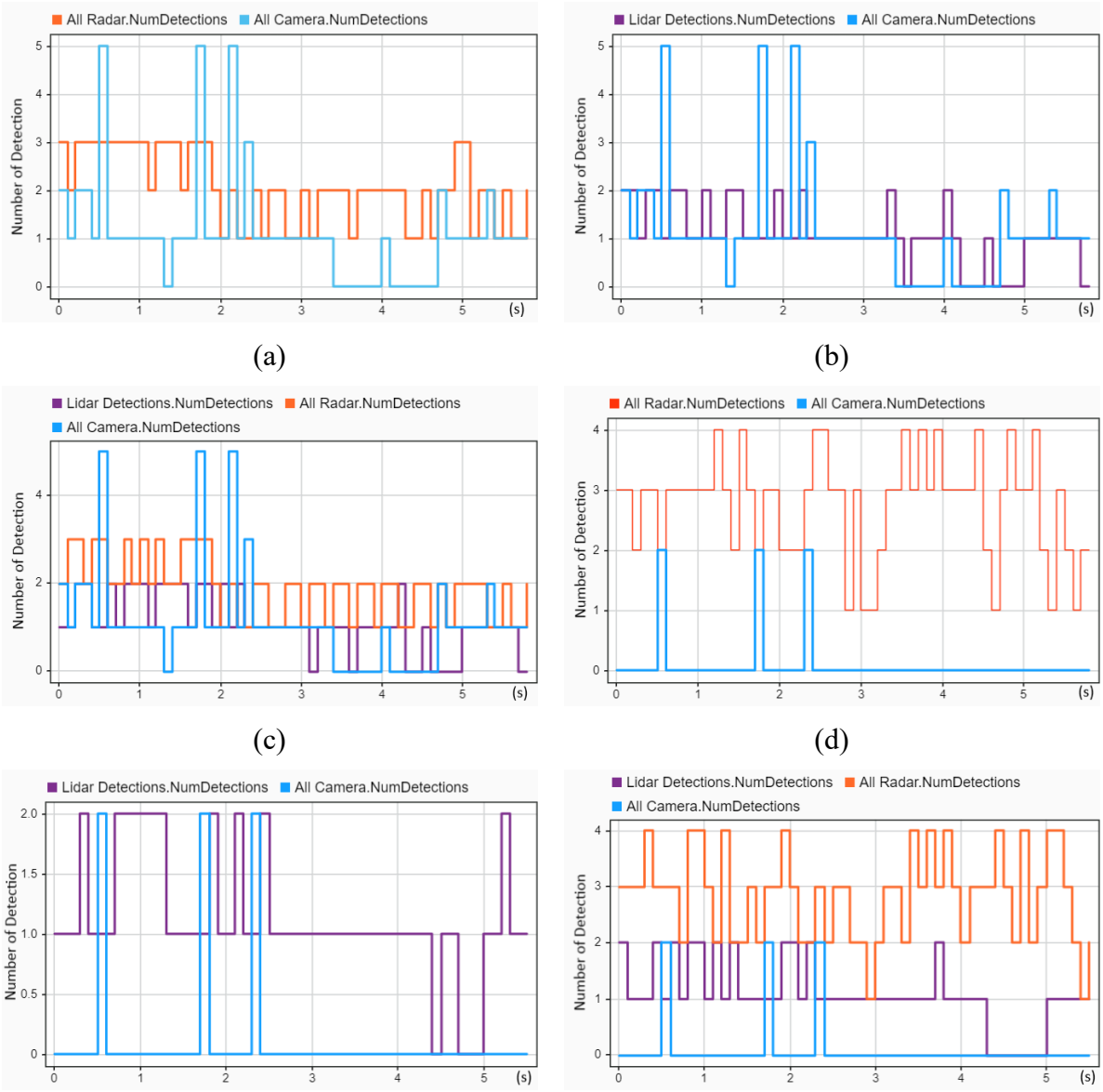


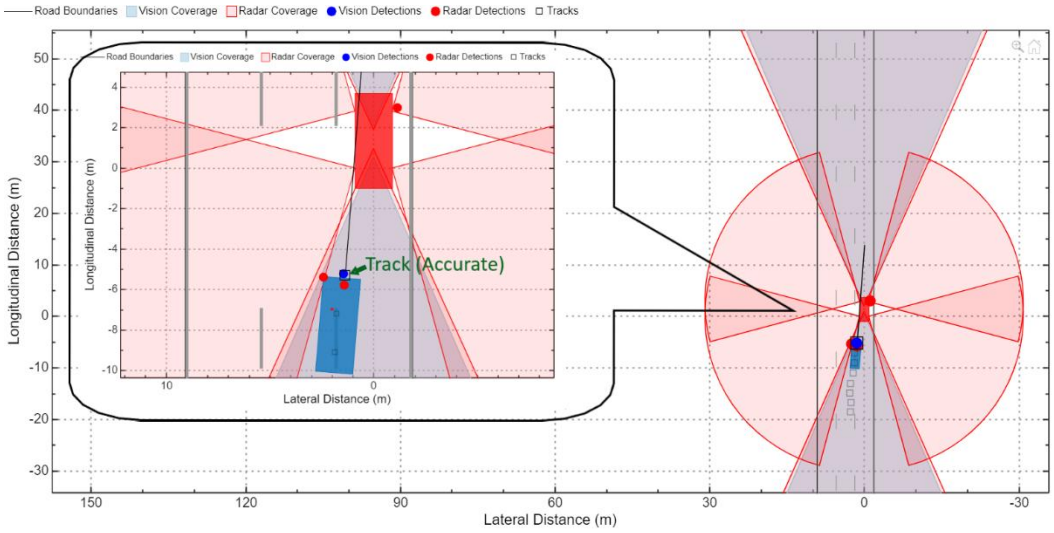
Figure 4.10: Object detections of the CIREN-226 (side impact crash) scenario using different sensor and tracking architecture setups: (a) used S1-TA1; radars have object detections for the complete trajectory duration, but cameras have three-zero object detections starting at 1.3 s, 3.4, and 4.7 s. (b) used S1-TA2; cameras have higher object detection gaps than LIDAR from 3.4 to 5 s. (c) used S1-TA3; LIDAR, radars, and cameras have almost the same object detections as (a,b). (d) used S2-TA1; radars have no zero detections, whereas cameras have three-time object detections at 0.5, 1.7, and 2.3 s. (e) used S2-TA2; LIDAR detects objects for the entire duration of the trajectory

but it has zero object detections at 4.4 and 4.7 s. (f) used S2-TA3; LIDAR object detections decrease, whereas radar and camera object detections remain the same as (d,e). Note: from (Haque et al., 2024) In *Sensors.*

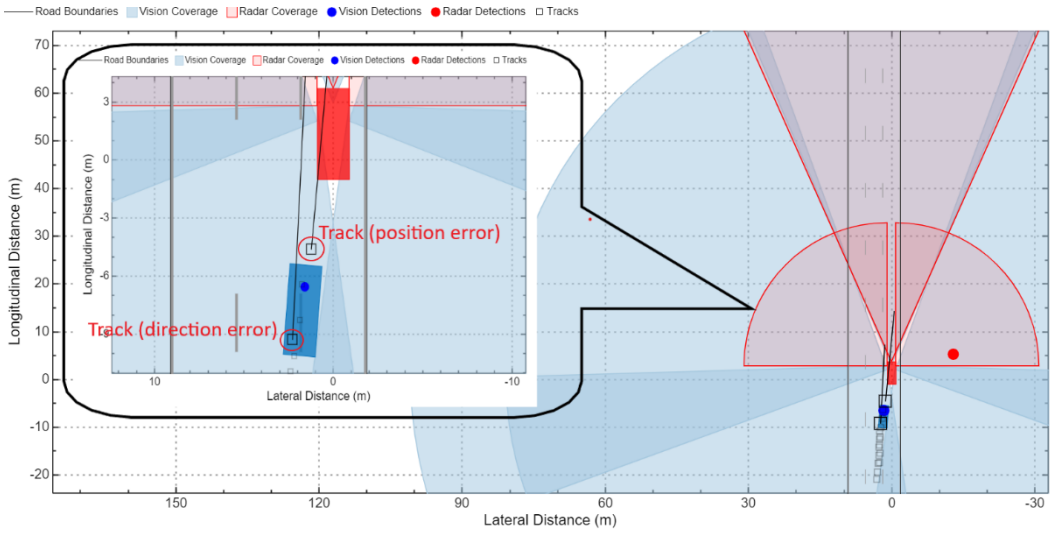
4.5.2 Evaluation of the Tracking Performance

Figure 4.11 presents a visual representation of the target vehicle's tracking results during the simulated CIREN-816, rear-end crash scenario. In this crash simulation, the ego vehicle (depicted in orange) utilised a multi-sensor-based surround vehicle sensor fusion to precisely track the target object (shown in blue). Notably, the tracker performed well when using the radar and camera fusion setup S2-TA1 (centralised tracking architecture), as illustrated in Figure 4.11a, where the tracking was accurate and seamless.

In contrast, when the exact moment of trajectory was applied but with the sensor setup S1-TA3 (decentralised tracking architecture), the tracking accuracy decreased. As shown in Figure 4.11b, erroneous tracking detections appear, with two tracks marked in red circles. One of these tracks, positioned in front of the target vehicle, exhibits a significant position error, while the other, incorrectly estimated, veers off in the wrong direction (depicted by the straight line extending from the erroneous track to the next estimated position of the target vehicle). This stark visual discrepancy highlights the limitations of decentralised tracking and serves as a pivotal moment, prompting this research to dive deeper into tracking performance evaluations through multiple iterations using the proposed SMTPE methodology. The following sections delve into these tracking performance assessments across three crash scenarios, offering important insights into sensor fusion and tracking accuracy.



(a)



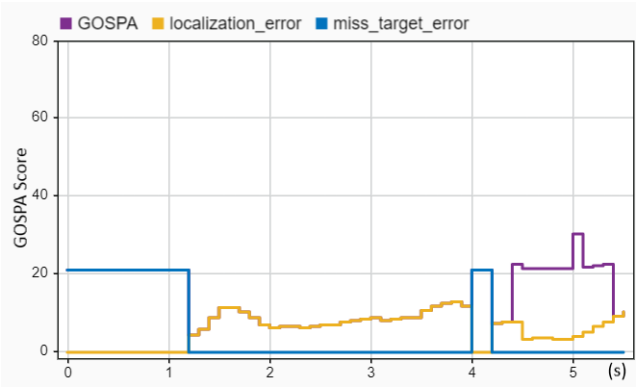
(b)

Figure 4.11: Simulink-based target tracking visualisation of the crash scenario CIREN-816: (a) accurate target tracked by using the setup S2-TA1; (b) error tracks observed by using the setup S1-TA3 and marked red circles with error types. Note: from (Haque et al., 2024) In *Sensors.*

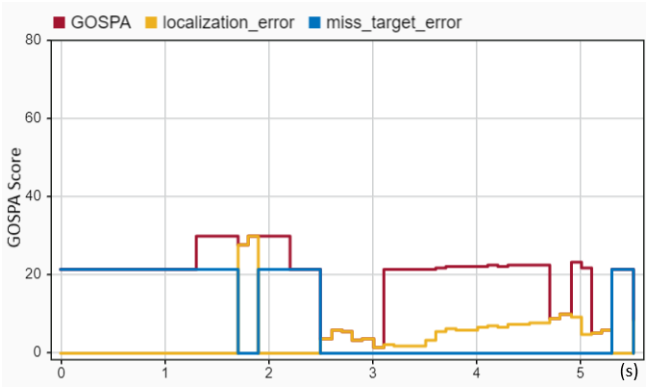
MATLAB offers a robust suite of system objects designed to streamline the tracking performance evaluation process for researchers. Among these, the OSPA-based trackOSPAMetric and the GOSPA-based trackGOSPAMetric stand out for their ability to assess tracking accuracy. The GOSPA metric, in particular, offers a more comprehensive evaluation compared to OSPA, incorporating additional features such

as localisation errors, missed targets, and false track components (MathWorks, 2019a). These extended capabilities have made the GOSPA metric increasingly attractive to researchers, driving further advancements in the field (García-Fernández et al., 2020; Su et al., 2024). Recognising the GOSPA metric's superiority, this research has chosen to evaluate the tracking performance in AV crash reconstruction using this metric, as detailed in subsection 3.5.1 of this thesis.

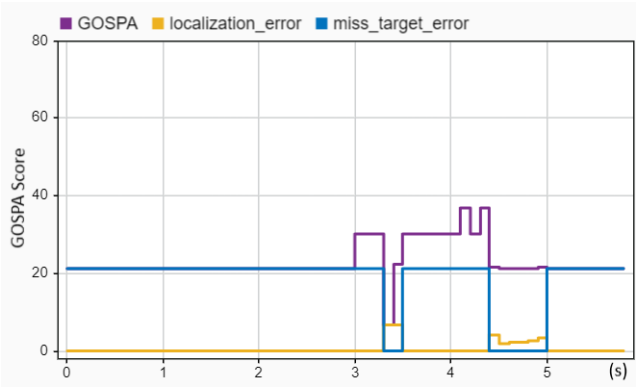
The tracking performance for the simulated head-on crash scenario (CIREN-664) is showcased in Figure 4.12. In Figure 4.12a, localisation errors range from 1.2 to 4.4 seconds, with error values fluctuating between 8 and 15—substantially higher than those observed in Figure 4.12b, where localisation errors remain below ten throughout the entire trajectory. Interestingly, Figure 4.12a exhibits superior GOSPA scores (above 20 after 4.4 seconds) and missed target errors (only 21 between 4.0 and 4.2 seconds) compared to the results in Figure 4.12b, which shows GOSPA scores reaching 30 between 1.3 and 3.2 seconds, and 21 from 3.1 to 5.1 seconds. A similar example emerges when comparing Figure 4.12e and Figure 4.12f, reinforcing the consistency of these findings. Notably, Figure 4.12b and Figure 4.12c are nearly identical, highlighting the stability of the results across different tracking setups. Most importantly, throughout all three tracking architectures, TA1 consistently outperformed the others in terms of tracking accuracy for the CIREN-664 crash scenario, establishing it as the most effective solution.



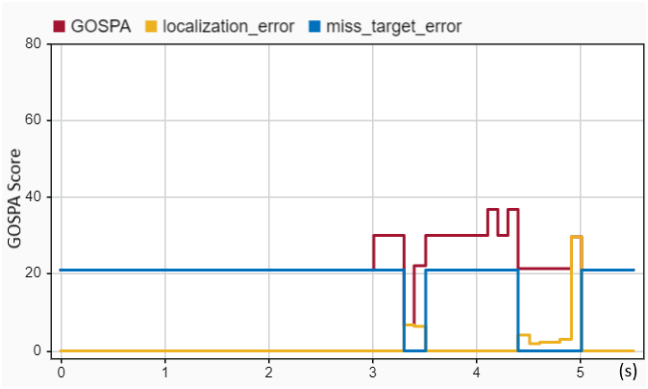
(a) sensor setup S1, tracking architecture TA1



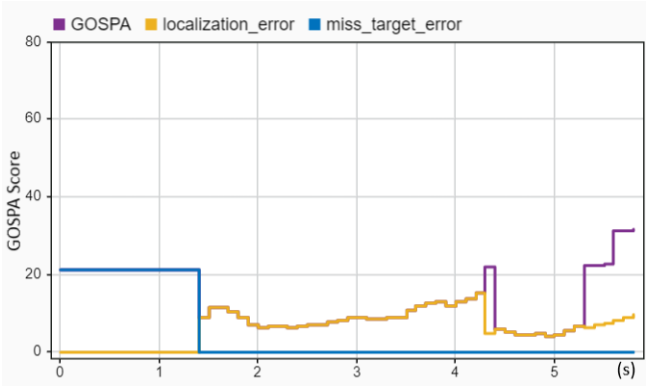
(b) sensor setup S2, tracking architecture TA1



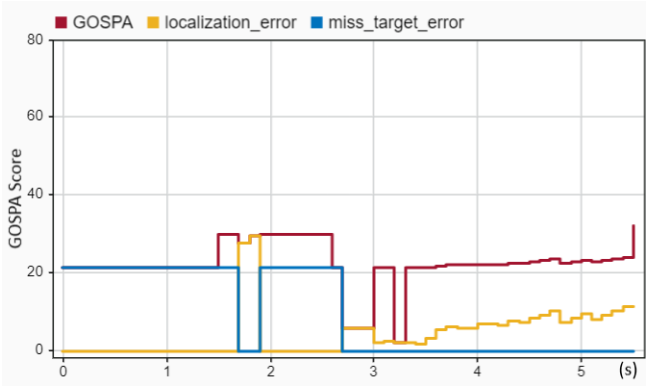
(c) sensor setup S1, tracking architecture TA2



(d) sensor setup S2, tracking architecture TA2



(e) sensor setup S1, tracking architecture TA3



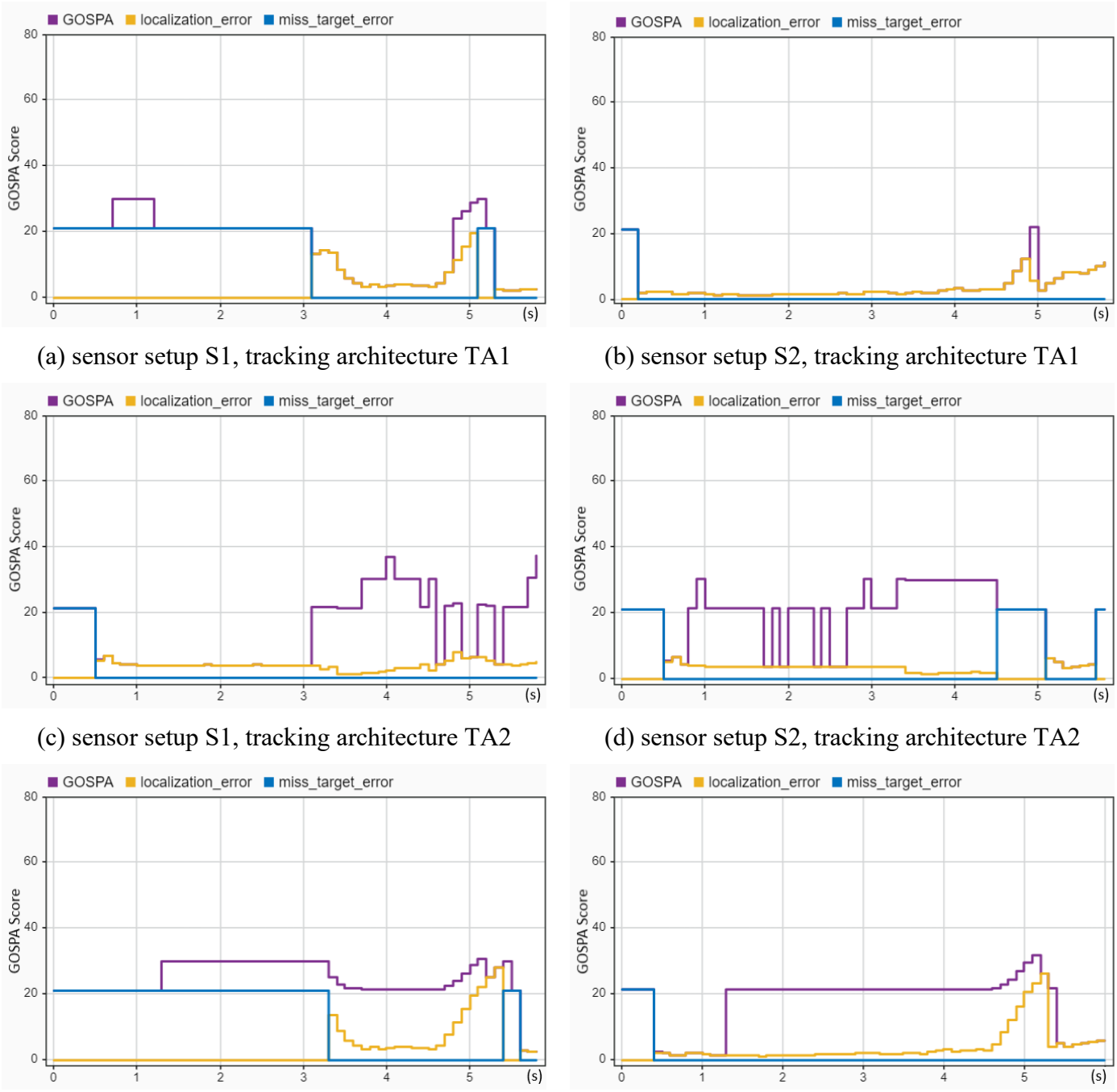
(f) sensor setup S2, tracking architecture TA3

Figure 4.12: Sensor fusion-based tracking performance assessment of the CIREN-664 (head-on crash) scenario using different sensors and tracking architectures setups: (a) used S1-TA1; GOSPA and localisation errors are the same from 1.2 to 4.4 s with error values between 8 and 15, but GOSPA errors are more than 20 after 4.4 s; missed target errors are found at 4 to 4.2 s. (b) used S2-TA1; localisation errors always remain below ten; the GOSPA errors are about 30 from 1.3 to 2.2 s and 23 from 3.2 to 5.2 s; missed target error scores are 21 from 0 to 1.7 s and 1.9 to 2.5 s. (c) used S1-TA2; localisation errors are zero except for 3.3 and 4.4 s; GOSPA score is above 20, and more than 30

from 3 to 4.4 s. (d) used S2-TA2; GOSPA, localisation, and missed target errors are almost the same as (c). (e) used S1-TA3; GOSPA and localisation errors are the same from 1.4 to 5.3 s with error values between 5 and 14, and missed target errors are zero from 1.3 s. (f) used S2-TA3; GOSPA scores are above 20, whereas localisation errors are below 10 after 1 s. See Equation (15) and association equations for formal definitions of GOSPA error, missed targets, and false targets. Note: from (Haque et al., 2024) In *Sensors.*

In the simulation of a rear-end crash scenario (CIREN-816), using the same tracking architectures (TA1, TA2, and TA3), the tracking performance was evaluated, and the results are presented in Figure 4.13 and Figure 4.14. A direct comparison of the tracking performance in Figure 4.13a and Figure 4.13b reveals significant differences in the metrics. In Figure 4.13b, the GOSPA score (30 at 0.7 seconds and 5 seconds), localisation error (reaching 20 at 5 seconds), and missed target error (20.5 between 0 to 3.1 seconds) are considerably higher than those in Figure 4.13a, where localisation errors remain under ten and no missed target errors occur throughout the entire time. Moreover, the GOSPA score in Figure 4.13a peaks at only 21 at 4.9 seconds. A similar trend can be observed in the comparison between Figure 4.13e and Figure 4.13f, with the latter showing superior tracking metrics.

Further analysis of TA2, presented in Figure 4.13c and Figure 4.13d, reveals that the GOSPA score and missed target errors are more favourable in Figure 4.13c, underscoring the benefits of this tracking configuration. However, it is worth noting that across all six combinations of tracking architectures and sensor setups, the S2-TA1 configuration delivered the most robust performance. As illustrated in Figure 4.13, S2-TA1 stood out as the top-performing combination, achieving the highest accuracy and efficiency in tracking the target vehicle throughout the rear-end crash simulation.



(a) sensor setup S1, tracking architecture TA1

(b) sensor setup S2, tracking architecture TA1

(c) sensor setup S1, tracking architecture TA2

(d) sensor setup S2, tracking architecture TA2

(e) sensor setup S1, tracking architecture TA3

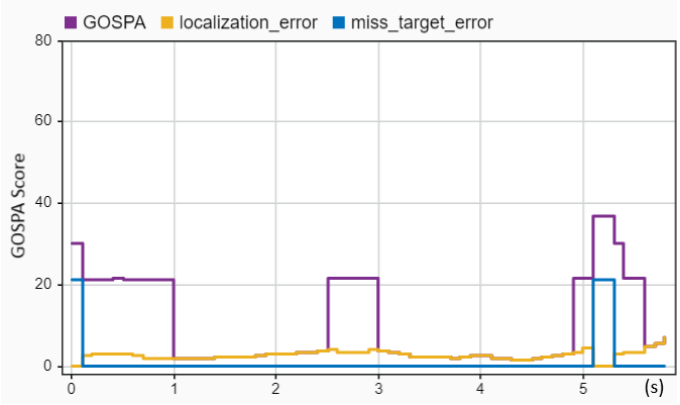
(f) sensor setup S2, tracking architecture TA3

Figure 4.13: Sensor fusion-based tracking performance assessment of the CIREN-816 (rear-end crash) scenario using different sensors and tracking architectures setups: (a) used S1-TA1; missed target errors are 21.5, and localisation errors are 0 from 0 to 3.1 s; GOSPA and localisation errors are the same from 3.1 to 4.7 s with error values between 5 and 12. (b) used S2-TA1; GOSPA and localisation errors are the same from 0.2 to 4.9 s with error values below ten, whereas missed target errors remain zero from

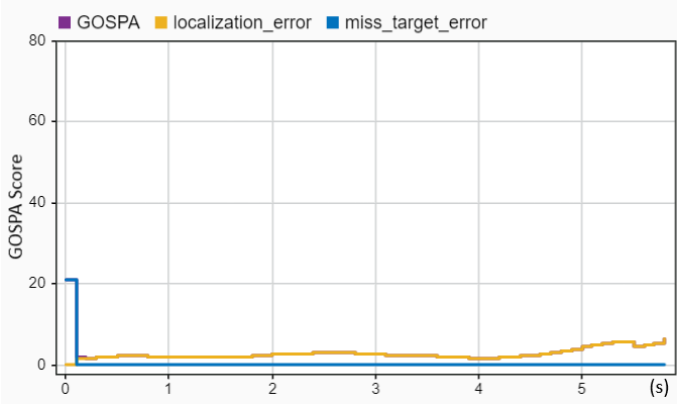
0.2 s to the end of the trajectory. (c) used S1-TA2; GOSPA and localisation errors are the same from 0.5 to 3.1 s and then the GOSPA reaches its highest value of 38 at 4 s. (d) used S2-TA2; GOSPA errors are between 20 and 30 from 1.8 to 5.1 s, whereas localisation errors are below 8, and missed target errors are zero for the same duration. (e) used S1-TA3; GOSPA errors increase to 30 from 20.5 at 1.3 s, remain the same up to 3.3 s, and then fluctuate from 21 to 30 from 3.3 to 5.1 s. (f) used S2-TA3; GOSPA errors are 21 from 1.3 to 4.6 s, then rise to 30 at 5.1 s, whereas localisation errors remain below 5 from 0.5 to 4.6 s and missed target errors remain zero from 0.5 s. Note: from (Haque et al., 2024) In *Sensors.*

An improvement in tracking performance is evident when comparing all six sensor setups and tracking architectures for the simulated CIREN-226 side-impact crash scenario, with the S2-TA1 configuration standing out, as depicted in Figure 4.14. In Figure 4.14b, using the S2-TA1 setup, the GOSPA scores and localisation errors remain low—below five between 0.1 and 5 seconds (leading up to the crash moment). Furthermore, missed target errors are absent during this entire period, showcasing the precision of this configuration. In contrast, Figure 4.14a exhibits sharp fluctuations in GOSPA values, with peaks of 20.5 at 0.1–1 seconds, 21 at 2.5–3 seconds, and 21 at 4.9 seconds, highlighting the inconsistencies in tracking performance.

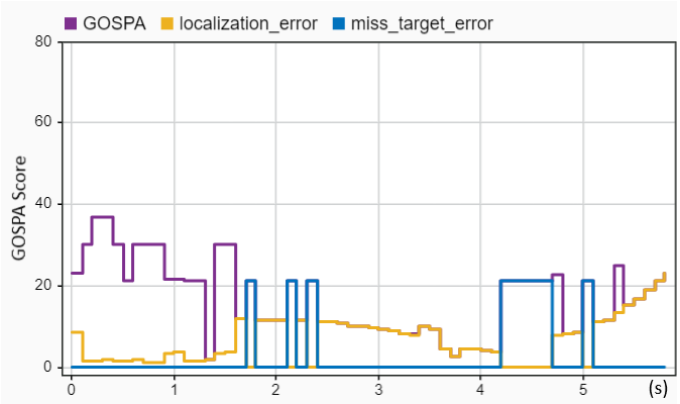
Looking at Figure 4.14c, all three evaluation variables—GOSPA scores, localisation errors, and missed target errors—are noticeably higher compared to Figure 4.14d, where the GOSPA scores are relatively stable at 21 at 0.8 seconds and 20.5 at 1.9 seconds. This example of performance disparities is similarly reflected in Figure 4.14e and Figure 4.14f, supporting the superiority of the S2-TA1 configuration in ensuring accurate and reliable tracking. These observations underscore the outstanding tracking capabilities of the S2-TA1 setup, making it the most suitable TA for crash simulation.



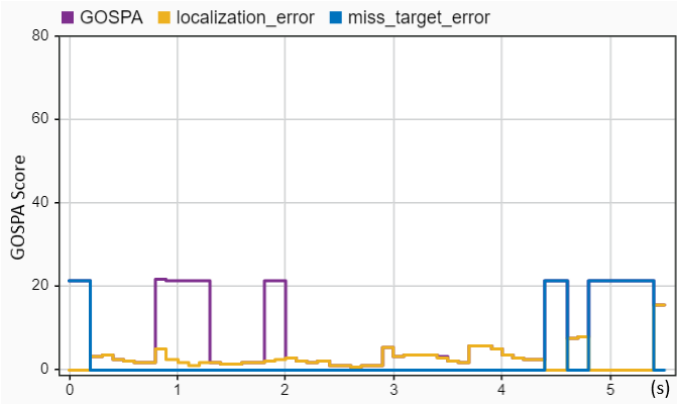
(a) sensor setup S1, tracking architecture TA1



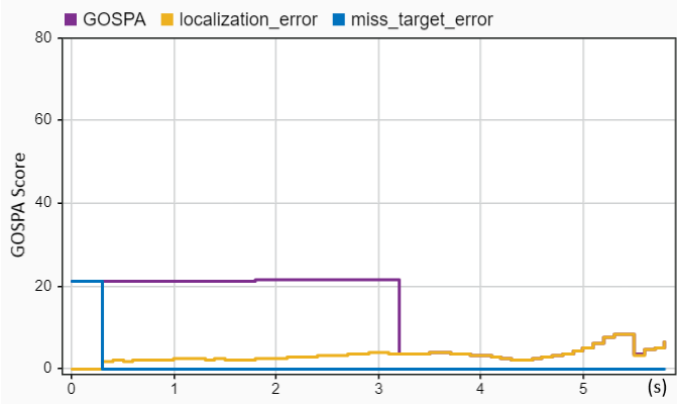
(b) sensor setup S2, tracking architecture TA1



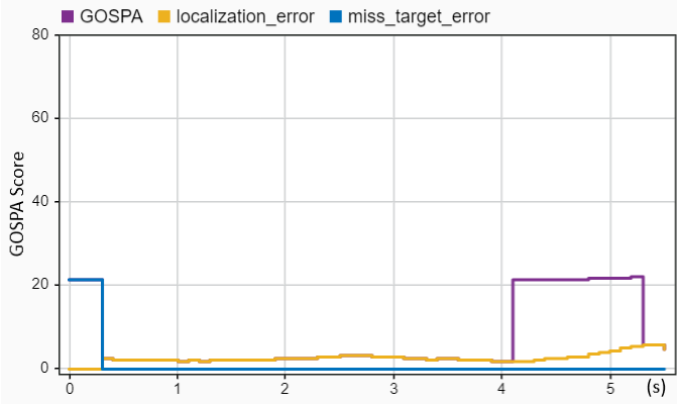
(c) sensor setup S1, tracking architecture TA2



(d) sensor setup S2, tracking architecture TA2



(e) sensor setup S1, tracking architecture TA3



(f) sensor setup S2, tracking architecture TA3

Figure 4.14: Sensor fusion-based tracking performance assessment of the CIREN-226 (side impact crash) scenario using different sensors and tracking architectures setups: (a) used S1-TA1; GOPSA scores are 21 or more from 0 to 1 s, 2.5 to 3 s, and 4.9 to 5.5 s; localisation errors are below ten, and missed target errors are zero for the entire trajectory time. (b) S2-TA1 has an excellent performance result; missed target errors are zero, and GOSPA and localisation errors are below five for the entire trajectory time.

(c) used S1-TA2; a few instances of GOSPA error scores of more than 30 can be observed from 0 to 1.6 s; localisation errors are more than ten from 1.6 to 3.6 s; the missed target error value is 21 from 4.2 to 4.7 s. (d) used S2-TA2; multiple instances of an error value of 21 can be observed for GOSPA (0.8–2.4 s, and 1.8 to 2 s) and missed target errors (4.5–4.6 s, 4.8–5.5 s), whereas localisation error scores are below ten for the entire trajectory time. (e) used S1-TA3; from 0 to 3.2 s, the GOSPA error scores are more than 20, whereas localisation errors are less than ten, and missed target errors are zero for all the trajectory time. (f) used S2-TA3; GOSPA error scores are more than 20 from 4.1 to 5.3 s, whereas localisation and missed target errors are below ten and zero for the entire trajectory time, accordingly. Note: from (Haque et al., 2024) In *Sensors.*

The SMTPE-based evaluation of tracking architecture performance setups provides a comprehensive overview, highlighting the highest and lowest-performing setups, with their respective ranks presented in Table 4-6. This research employed two measurement scales and four performance variables to assess these tracking setups. These variables—measurement errors, localisation errors, false target errors, and GOSPA scores—were instrumental in ranking the tracking architectures. The evaluation process involved applying error ranges and averaging the results over the total trajectory time of the simulated scenario, emphasising that lower error values signify superior tracking performance.

Table 4-6: Evaluation summary of the highest and lowest Tracking Architecture (TA) setups' performance achieved by multi-sensor fusion. The average values of the performance measurement variables (GOSPA score, localisation error, and missed target error) are used to rank the experimented tracking architectures. Note: from (Haque et al., 2024) In *Sensors.*

Rank	Tracking Architecture	Scenario	Average Localisation Errors	Average Missed Target Errors	Average False Target Errors	Average GOSPA Score
Best	S2-TA1(Centralised TA)	CIREN-226	2.54	0	0	2.54
Worst	S1-TA2 (Centralised TA)	CIREN-226	6.22	10.6	10.24	15.91

The findings from this analysis reveal a striking contrast in performance: the S2-TA1 tracking architecture emerged as the best-performing setup, achieving a GOSPA score of 2.54, while the S1-TA2 configuration lagged as the worst, with a GOSPA score of 15.91. This highlights the accuracy and reliability of S2-TA1, underscoring its superiority in delivering optimal tracking results across the simulated crash scenarios.

4.6 Discussion

This study introduced an innovative simulation method (SMTPE) designed to optimise the selection of a tracking architecture using multi-sensor fusion for AV crash reconstruction, and sheds light on RQ1 of this thesis. The results demonstrate that SMTPE excels at pinpointing the most effective tracking setup. Among the three tracking architectures tested, the S2-TA1—a cutting-edge radar-camera-based centralised architecture for multi-sensor surround vehicle fusion—emerged as the top performer.

While experimenting with the developed SMTPE, this research made many findings that could be worthwhile for further AV developments. The key findings are presented below.

- The size of sensor-based data is influenced by the specific driving scenario and the duration of the vehicle's trajectory. It is important to achieve an optimal data size, as it plays a key role in vehicle control and trajectory planning. In this research conducted for SMTPE, the use of sampling time was adopted to enhance tracking results (Choi et al., 2021). The study assessed three different sensor update rates—200 ms, 100 ms, and 50 ms—to identify the most effective data generation method from vehicle sensors. The findings revealed that a 100 ms update rate was optimal, as it provided the necessary object detections while maintaining an effective data size. By leveraging optimal sensor data, tracking performance is improved, leading to more precise and reliable vehicle navigation outcomes.

- Vehicle tracking performance relies on sensor-based object detection, which is vital in ensuring safety and efficiency in ADS. It is important to accurately position the vehicle's sensors for effective sensor fusion to achieve optimal detection coverage. A comparison of object detection capabilities reveals important differences among the three primary sensor types: radar, cameras, and LIDAR. Research indicates that cameras typically offer lower detection performance compared to radar in simulated crash scenarios. This finding underscores the limitations of individual sensors, reinforcing the need for a multi-sensor approach to enhance detection reliability (Xiang et al., 2023). Combining radar, cameras, or LIDAR through sensor fusion compensates for the weaknesses of each sensor and significantly improves overall detection performance (Yeong et al., 2021).
- The SMTPE unveils important insights, demonstrating that a lower error value from sensor fusion-based object tracking signifies a more sophisticated tracking architecture. This research explored three tracking architectures to evaluate various crash scenarios, including head-on collisions, rear-end crashes, and side-swipe collisions. By employing the GOSPA metric (García-Fernández et al., 2020; Rahmathullah et al., 2017; Su et al., 2024) to assess tracking performance in AV crash reconstruction, this study discovered that the centralised architecture using multi-sensor fusion—integrating both radar and cameras—delivered the most outstanding results. The innovative SMTPE stands out as a repeatable simulation method, offering a reliable pathway to identify the optimal tracking architecture for future vehicle crash reconstructions and advancing research to enhance AV technology.

Guidelines

This research offers recommendations designed to elevate the scientific rigour of multi-sensor-based sensor fusion in AV research, ensuring that developments are robust and reliable.

- For the upcoming experiment, selecting the right datasets and recordings from real-world scenarios is important to ensure a solid foundation.
- A meticulously crafted initial setup is important for unlocking valuable insights through multi-sensor fusion.
- Understanding how to navigate diverse environments is key, especially when it comes to analysing different trajectories. Knowledge of the strengths and limitations of various sensors is indispensable for tackling the unique challenges of real-world conditions—be it driving under the cloak of night, manoeuvring through rain or fog, or tackling the twists and turns of mountainous roads. Furthermore, delving into different sensors—such as radar, cameras, and LIDAR—while employing innovative orientation-based setups can yield rich and comprehensive results. This exploration will pave the way for identifying the most effective experimental design for the research endeavours.
- Utilising a diverse array of tracking architectures can enhance performance compared to relying on a single TA. Thus, exploring tracking performance through multiple TA setups for optimal results is highly beneficial.
- Selecting the ideal tracking algorithm and object tracker demands a deep understanding of their intricate functionalities. Experimenting with a variety of tracking algorithms and trackers can uncover the most effective TA. Furthermore, implementing well-regarded tracking metrics for performance evaluation is crucial for achieving reliable results.
- Establishing a suitable tracking threshold value is paramount for success. For example, setting a performance threshold—such as a 95% confidence interval—can pave the way for creating the finest tracking architecture. One researcher might aim for a minimum of 80% accuracy in tracking performance for AV obstacle detection, while another may strive for at least 90% target tracking accuracy in a collision avoidance scenario. In this study, threshold values were established to allow a maximum of 10% error, ensuring that factors such as GOSPA error, localisation error, missed target error, and false detection error remained at 10% or lower. The tracking performance outcomes validate these

predefined confidence levels for all factors being below 5, as illustrated in Table 4-6.

Ultimately, conducting an evaluation of object tracking performance is strongly recommended before embarking on any sensor fusion-based AV development, as it boosts confidence in the final results.

Future Directions

The development of a multi-tracking sensor architecture requires further advancements in the following areas:

Sensor Selection and Arrangement: It is crucial to choose the appropriate sensors, determine their quantity, establish their positions on the vehicle, and meet calibration requirements.

Data Management: Effective management of the substantial volume of data generated by sensors is important. This includes ensuring that no data are lost while capturing critical information.

Algorithm and Architecture Design: Selecting the right algorithms, filters, object trackers, and tracking architectures necessitates a high level of expertise.

360-Degree Sensor Fusion: Achieving comprehensive 360-degree perception is important, but it should be done without generating excessive sensor data that could increase processing and storage costs. This also involves finding a balance when using limited sensors with impractical viewing angles.

Sensor Overlap: Overlapping sensor coverage can result in inaccurate object detection, which is a challenge to address.

Overall, by focusing on these areas, the effectiveness of multi-tracking sensor architectures can be improved.

4.7 Conclusions

The realm of Autonomous Vehicle (AV) technology is rapidly evolving, driven sensors and tracking algorithms that improve target-following. This chapter investigates three dynamic tracking architectures that use multi-sensor fusion, combining radar, cameras, and LIDAR to create detailed tracks from crash scenarios, applying the Crash Injury Research Engineering Network (CIREN) crash dataset. Notably, this research constructed two centralised tracking architectures alongside one decentralised architecture to broaden the analysis.

The experiments were conducted with two multi-sensor-based surround vehicle sensor fusion setups to test the effectiveness of these tracking architectures. In the first setup, cameras played a pivotal role in capturing the surrounding environment, while radars took centre stage in the second setup. The simulation-generated tracks produced by these architectures underwent a thorough evaluation, aiming to unveil the best-performing configuration. The findings illustrated that a centralised tracking architecture, which fused data from radar and cameras in sensor setup 2, provided the most precise tracking performance. This multi-tracking sensor architecture will be used for AV crash reconstruction and prediction. The key contributions of this chapter can be encapsulated as follows:

- The standout contribution of this research lies in the innovative development of a centralised multi-sensor-based vehicle fusion tracking architecture (SA2, TA1), which excels in reconstructing crashes involving AVs. The evaluations reveal that this setup can accurately reconstruct crash scenarios using data collected under ideal conditions. Notably, no extra road infrastructure or external sensor data are required for such reconstruction, thereby adding to the notion of autonomous vehicle crash reconstruction in situations and contexts where such infrastructure is unavailable.

- This research proposed a simulation method to select a good multi-tracking sensor architecture. The proposed method is expected to provide better input for future AV development.
- A further implication of this study is that it will be helpful when reconstructing an AV crash for forensic and investigative purposes, where there is a need for assurance that the sensors have captured all the required information, and that no important information has been lost.
- Finally, in addition to the research findings, brief guidelines are provided for repeating and reusing the proposed method in a similar research domain.

The aim of this chapter was to evaluate how different sensor architectures could be useful for simulating sensor data in AVs instead of experimenting with these architectures in real crash scenarios. Such experiments may lead to further changes in sensor architecture and infrastructure due to issues encountered in real data acquisition, such as noise, corruption, and data loss. Based on the tracking performance evaluated, the future scope of this study will involve using the best multi-tracking sensor architecture for AV crash reconstruction and prediction. A notable feature of the SMTPE is its repeatability for other AV development tracking performance assessments. This methodology may also benefit future research in related sectors, such as Unmanned Aerial Vehicles (UAVs) and Autonomous Underwater Vehicles (AUVs).

This chapter investigated various sensor architectures and identified a suitable one: a centralised tracking architecture that integrates radar and cameras using sensor setup 2. This configuration can provide more accurate tracking information than the other five architectures considered. This selected sensor architecture will be utilised in Chapter 5 for precise crash reconstruction, including detecting and classifying crash events.

Chapter 5 Deep Simulated Crash Reconstruction for Training Data Preparation

Chapter 4 investigated multitracking sensor architectures using a proposed method and recommended a radar-camera-based centralised sensor architecture. This innovative sensor architecture has demonstrated potential for generating accurate data for AVs, leading to the question of whether it can effectively be used to reconstruct V2V crash events and provide reliable crash data. Answering this question may provide insights into future crash reconstructions and could pave the way for automated AV crash forensics. To this end, collected AV sensor data with pre-crash and crash information are processed using a proposed Vehicle Crash Reconstruction Method (VCRM) for reconstructing crashes and preparing training data. The evaluation results show that the proposed method can accurately reconstruct AV crashes and classify pre-crash and crash events using the simulated AV sensor data. Finally, this chapter prepared a training dataset using simulations of hundreds of crash scenarios for machine learning-based crash prediction in the final part of this thesis.

5.1 Introduction

This chapter delves into the field of V2V crash reconstruction using simulated AV sensor data. In Section 5.2, a method for vehicle crash reconstruction is proposed, delineated by a flowchart that encapsulates the sequential processes involved in both crash reconstruction and the generation of a training dataset derived from simulated sensor data.

Section 5.3 commences with the simulation setup for V2V crash experiments, setting the stage for applying algorithms designed to classify pre-crash and crash events. Building on this foundation, this research presents the results of deep-simulated crash reconstructions, showcasing the effectiveness of these classification algorithms in action. These simulations provide a rich, evidence-based collection of data drawn from a diverse set of hundred driving scenarios, ultimately creating a training dataset for data-driven crash prediction models.

Furthermore, the accuracy of the crash reconstructions is evaluated with a comparative analysis between the VCRM results, actual crash data from EDR, and findings from another leading simulator, VirtualCRASH. The outcomes confirm the reliability and precision of the reconstruction methodology.

Section 5.4 offers a discussion of the key findings, setting the stage for a summary in Section 5.5, where the chapter's contributions are wrapped up, providing readers with an understanding of the potential of simulated data in advancing crash reconstruction techniques.

5.2 The Method for Crash Reconstruction

5.2.1 Background Study

Traffic accident reconstruction is gaining traction as a tool for investigators to visually analyse vehicle crashes through animated reconstructions, enabling assessments based on reconstructed crash data (Ortiz & Dávila, 2023). Recent advancements in traffic

accident research and vehicle crash forensics have elevated crash scene reconstruction as one of the most sought-after methods for in-depth analysis (Kolla et al., 2022). With cutting-edge technology, a crash reconstructionist leverages reconstructed 3D crash scenes to conduct accurate crash analyses (Kamnik et al., 2020), often presenting these visual models in court to support legal cases (FARO, 2024).

Data for these reconstructions can be gathered directly from vehicle sensors (Ortiz & Dávila, 2023), with insights provided by the vehicle's EDR, which captures five seconds of pre-crash and crash data (Katarína et al., 2021). With most modern vehicles now equipped with EDRs, it was projected that 99.5% of new lightweight vehicles in the USA will include this technology by 2021 (NHTSA, 2022a). However, despite its widespread adoption, the National Highway Traffic Safety Administration (NHTSA) has noted limitations in the current EDRs, particularly in capturing comprehensive pre-crash data (NHTSA, 2022b).

Crash investigators face obstacles such as retrieving digital evidence from a damaged EDR following a post-crash fire, the potential tampering or removal of the EDR by a guilty party, and technical difficulties in data collection. The current crash reconstruction process relies on data collected from prior vehicle accidents, where accurate information is crucial for a reconstruction. In the absence of a driver's contribution to accidents in fully autonomous vehicles, there is a need to ensure accurate reconstruction of events immediately before the critical event for forensic purposes as well as insurance and legal purposes. Therefore, vehicle crash reconstruction must identify the leading cause of these accidents from data alone. To date, relatively little work has been carried out on how such accident reconstruction can take place from AV sensor data alone and be helpful to forensic experts. So, developing a mechanism to reconstruct crashes using AV sensor data could contribute to this research area.

5.2.2 The Proposed Method (VCRM)

This chapter introduces an innovative Vehicle Crash Reconstruction Method (VCRM)

designed to facilitate evidence-based crash reconstruction. Building on the AV-CRPRM outlined in Section 1.4, this proposed VCRM provides a robust framework for accurately reconstructing crash events. As depicted in Figure 5.1, the VCRM drives the reconstruction process and plays a pivotal role in generating the training datasets. The method used the most effective sensor architecture—specifically, a radar-camera-based centralised sensor system—identified through the investigations presented in Chapter 4., ensuring optimal data input and reconstruction precision.

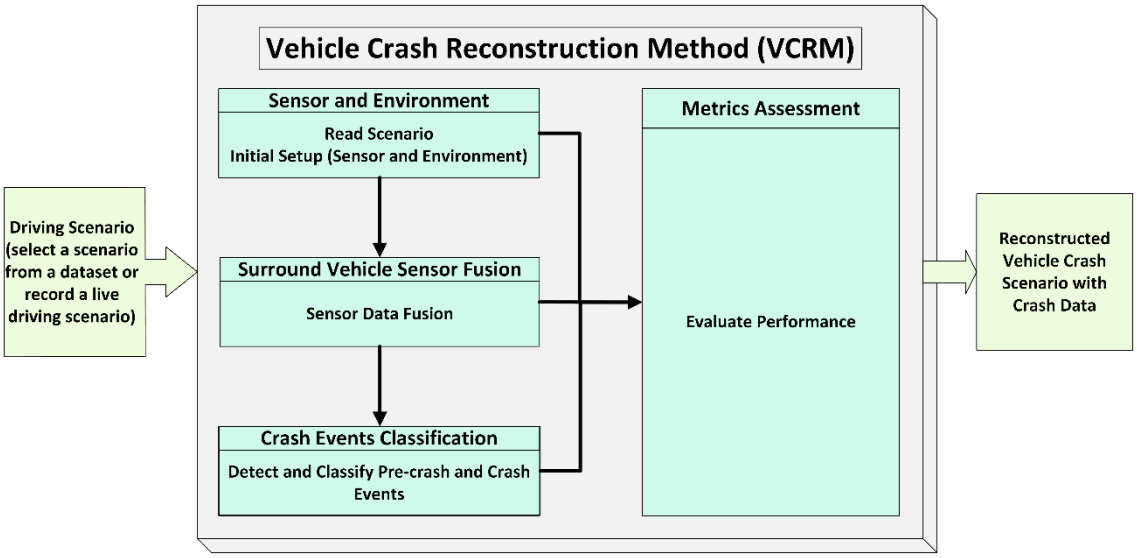


Figure 5.1: The proposed Vehicle Crash Reconstruction Method (VCRM).

The vehicle crash detection method of the VCRM is designed to recognise and classify cut-in, conflict, potential crash and crash events using Equations (3.16)–(3.30) presented in Section 3.3 of Chapter 3. In this chapter, the sensor and environment subsystem of the VCRM are arranged using the recommended sensor architecture presented in Chapter 4. These sensor data are then fused in the surround-vehicle-sensor-fusion subsystem to detect other vehicles and produce MIO tracking using the developed algorithm presented in subsection 3.3.2. Subsequently, the crash event classification subsystem processes both simulated tracking data and AV sensor information to detect pre-crash events, including cut-ins, conflicts, potential crashes, and actual crashes. Simultaneously, these events are accurately classified to generate

robust training datasets. The metric assessment subsystem developed on the formulas presented in Section 3.5 evaluates the performance of the VCRM's subsystems.

Ultimately, the VCRM delivers precise crash reconstruction data that are invaluable for crash analysis and the creation of training datasets. The following subsection delves into the streamlined process flow of the proposed VCRM, illustrating how it generates simulated AV sensor data for crash dataset development.

5.2.3 Process Flow of the VCRM

The process flowchart of the VCRM contains methods for generating a training dataset for ML, as shown in Figure 5.2.

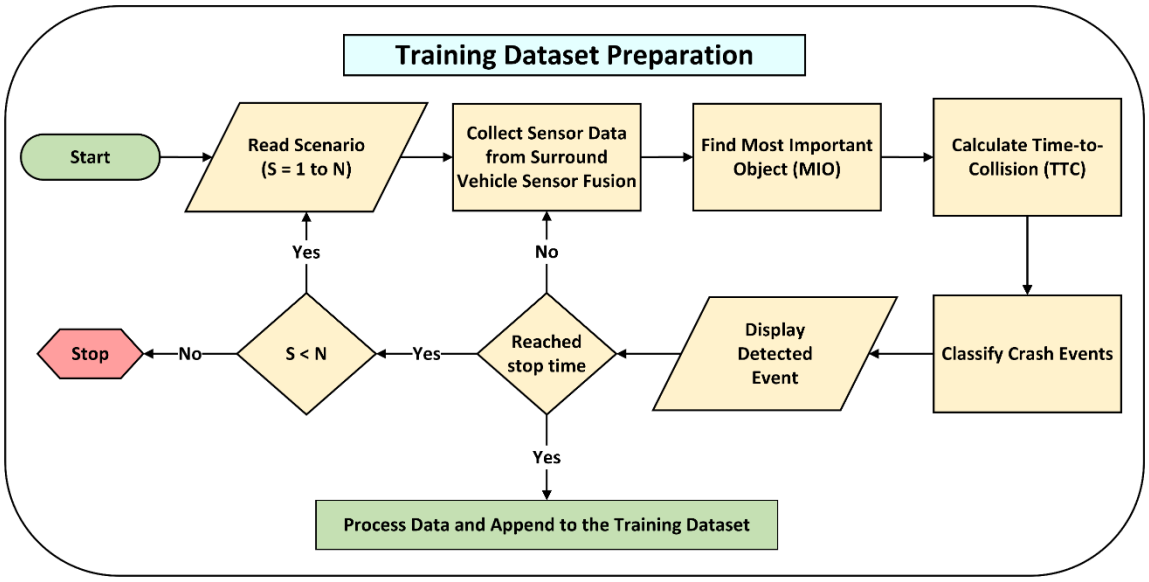


Figure 5.2: Process flow of the VCRM for training dataset preparation.

The flowchart begins by reading a driving scenario to generate sensor data from the simulation, setting the stage for detailed crash analysis. Next, it collects and fuses sensor data at every time step of the scenario, processing this information through the MIO detection algorithm introduced in subsection 3.3.2. The resulting MIO tracks are then used to calculate the TTC, providing important insights into the dynamics of the scenario.

The processed AV sensor data are subsequently employed to classify pre-crash and crash events based on the formulas developed in Section 3.3. The VCRM then presents the detected crash events via a dashboard, as detailed in Chapter 6. These event detections are further refined and automatically classified into pre-crash types (cut-in, conflict, potential crash) and crash events.

Once classified, the pre-crash events—along with other sensor data—are recorded and forwarded to the data processing component of the model, as outlined in Section 3.4. All the recorded data from the simulated driving scenarios are processed to generate a crash dataset. These data are automatically recorded and exported into Excel files, with each driving scenario being looped until the final scenario is reconstructed. The data from these files are then manually processed to concatenate the information from all scenarios, ultimately creating a robust training dataset for crash prediction.

5.3 Experiments and Results

5.3.1 Experiment Setup

This research developed and evaluated the VCRM using specialised vehicle crash reconstruction software such as Virtual CRASH 5 (CRASH, 2024), MATLAB (R2024b), Simulink (R2024b), and their associated tools, including the Driving Scenario Designer app, Navigation Toolbox, and Automated Driving Toolbox. The experiment was executed on a machine running Windows 11 Pro (64-bit, version 23H2), powered by an 11th Gen Intel Core i7 processor with 16 GB of RAM. The primary goal of this research was to reconstruct vehicle crashes using simulated sensor data, providing an approach to accident analysis. This chapter leverages the sensor architecture outlined in Chapter 4, focusing on the radar-camera-based centralised tracking system to experiment with the innovative VCRM model. The sensor configuration is detailed in Table 5-1.

Table 5-1: Sensor arrangements of the ego vehicle for crash data collection.

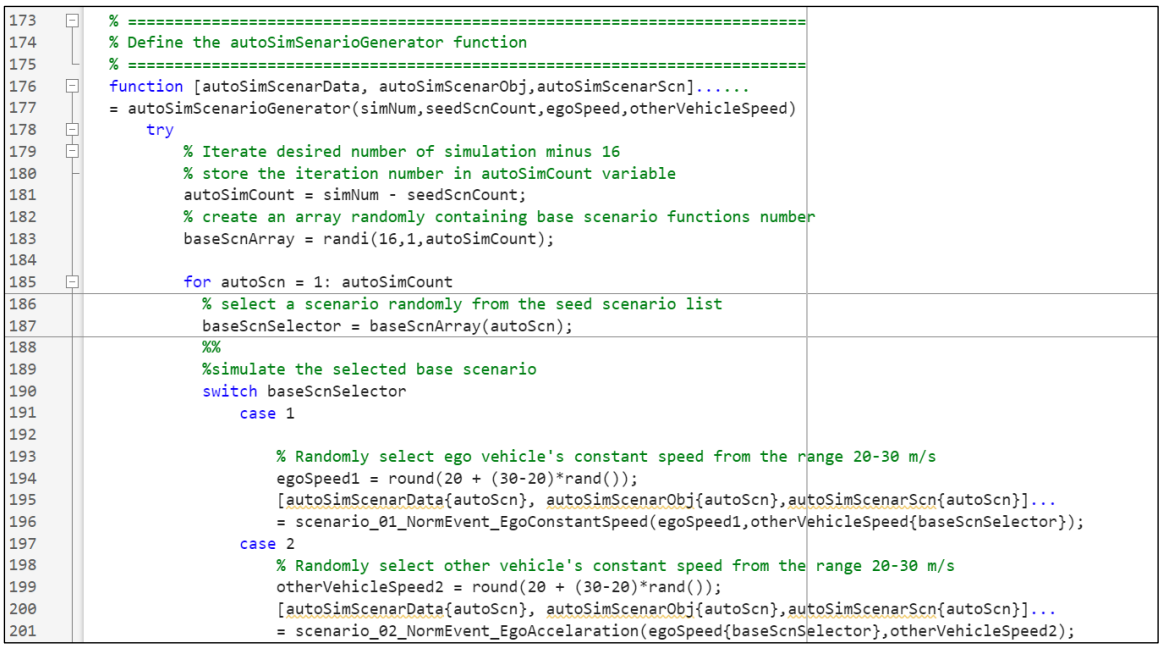
Sensor	Location	Position (m) [x, y, z]	Rotation (°) [Roll, Pitch, Yaw]	Max. Range (m)
Radar	Front	[1.9, 0, 0.2]	[0, 0, 0]	160
	Front-left	[2.8, 0.9, 0.2]	[0, 0, 60]	30
	Front-right	[2.8, −0.9, 0.2]	[0, 0, −60]	30
	Rear-left	[0, 0.9, 0.2]	[0, 0, 120]	30
	Rear-right	[0, −0.9, 0.2]	[0, 0, −120]	30
	Rear	[0.95, 0, 0.2]	[0, 0, −180]	160
Camera	Front	[2.1, 0, 1.1]	[0, 1, 0]	150
	Rear	[0.56, −0.9, 1.1]	[0, 1, −180]	150

This research boarded on an iterative development journey, following the AV-CRPRM methodology. In its second iteration, the AV-CRPRM framework led to the creation of the VCRM, a cornerstone tool for simulation-based vehicle crash experiments. Within this chapter, four distinct crash types—front-end, head-on, rear-end, and side-impact V2V crashes—were simulated using the CIREN dataset (NHTSA, 2017-2023), providing a robust foundation for evidence-based crash reconstruction. The study then expanded into experimentation by simulating 100 crash and non-crash driving scenarios to explore pre-crash and crash dynamics in-depth. Sixteen seed scenarios were crafted, each varying in vehicle speeds, to generate trajectory-based events such as cut-ins, conflicts, potential crashes, and actual crashes. The driving scenarios setup and typical vehicle properties for these simulations are illustrated in Table 5-2. These seed scenarios served as the basis for simulating 84 unique driving scenarios, compelling the boundaries of crash scenario reconstructions.

Table 5-2: Basic parameters of a driving scenario.

Trajectory Object	Parameter Name	Value(m)
Road (Motorway)	Width	10.95
	Lanes Number	3
	Lanes Width	3.6
	Lane Type	Driving
	Lane Marking	Solid (Yellow)
	Road Centres in meter (x, y)	[-50 0 0; 0 0 0; 100 0 0; 200 0 0; 300 0 0; 400 0 0; 500 0 0; 550 0 0; 600 0 0; 650 0 0; 700 0 0]
Actors: Ego Vehicle, Other Vehicle	Length	4.7
	Width	1.
	Front Overhang	0.
	Rear Overhang	1
	3D Display Type	Sedan
	Colour	Blue (Host), Orange (Other)
	Waypoints	Varies for each trajectory or crash scenarios

This research delved into the intricacies of the developed formulas for classifying pre-crash and crash events, using 100 trajectories and evidence-based crash data sourced from the CIREN dataset. A snapshot of the customised function, showcased in Figure 5.3 (with the full MATLAB script available in Appendix A), illustrates the construction of one hundred driving scenarios based on 16 foundational seed scenarios. These simulations were pivotal in reconstructing pre-crash and crash events, effectively preparing the data for training models. In the following subsection, these deep simulated driving scenarios are harnessed to ensure precise crash reconstruction and accurate classification of crash events, marking a step forward in vehicle accident analysis.



```
% ========================================================================
% Define the autoSimSenarioGenerator function
% ========================================================================
function [autoSimScenarData, autoSimScenarObj,autoSimScenarScn]......
= autoSimScenarioGenerator(simNum,seedScnCount,egoSpeed,otherVehicleSpeed)
    try
        % Iterate desired number of simulation minus 16
        % store the iteration number in autoSimCount variable
        autoSimCount = simNum - seedScnCount;
        % create an array randomly containing base scenario functions number
        baseScnArray = randi(16,1,autoSimCount);

        for autoScn = 1: autoSimCount
          % select a scenario randomly from the seed scenario list
          baseScnSelector = baseScnArray(autoScn);
          %%
          %simulate the selected base scenario
          switch baseScnSelector
              case 1

                  % Randomly select ego vehicle's constant speed from the range 20-30 m/s
                  egoSpeed1 = round(20 + (30-20)*rand());
                  [autoSimScenarData{autoScn}, autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}]...
                  = scenario_01_NormEvent_EgoConstantSpeed(egoSpeed1,otherVehicleSpeed{baseScnSelector});
              case 2
                  % Randomly select other vehicle's constant speed from the range 20-30 m/s
                  otherVehicleSpeed2 = round(20 + (30-20)*rand());
                  [autoSimScenarData{autoScn}, autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}]...
                  = scenario_02_NormEvent_EgoAccelaration(egoSpeed{baseScnSelector},otherVehicleSpeed2);
```

Figure 5.3: Snapshot of the MATLAB script for hundred driving scenario simulation.

5.3.2 Pre-crash and Crash Events Detection and Classification

5.3.2.1 Algorithms for Crash Reconstruction

This research utilises AV sensor data from the simulations of 100 driving scenarios. The ego vehicle, equipped with a tracker, generates confirmed tracks that serve as the cornerstone input for identifying MIOs through the algorithm detailed in subsection 3.3.2. Beyond detecting MIOs, this algorithm also computes parameters such as longitudinal and lateral distances, longitudinal and lateral velocities, and the distances to both the left and right lane boundaries from the ego vehicle. These key metrics, along with additional parameters like Time-To-Collision (TTC) and vehicle direction (VehicleDir), form the foundation for accurately detecting and classifying pre-crash and crash events. For trajectory-based cut-in event classification, the algorithm takes into account longitudinal distance (x) and velocity (vx), lateral distance (y) and velocity (vy), and the distances to the left and right lane boundaries (leftLane, rightLane), as illustrated in Algorithm 5.1. Other pre-crash events, including conflicts and potential

crashes, along with crash events, are methodically calculated in Algorithm 5.2, ensuring a comprehensive method for V2V crash event detection and classification.

Algorithm 5.1: Trajectory-based cut-in Event

Input: TTC, x, vx, y, vy, MIO, laneDistanceLeft, laneDistanceRight, VehicleDir
Output: statusCut-in

1: **For each** time step
2: Initialise statusCut-in as zero, τ_{con} (3.16), δ_L (3.17), and δ_R (3.18)
3: **Case A:** Cut-in from left lane detection and classification
4: Calculate cut-in using (3.19)
5: **Case B:** Cut-in from right lane detection and classification
6: Calculate cut-in using (3.20)
7: **End for**

Algorithm 5.2: Trajectory-based conflict, potential crash, and crash Events

Input: TTC, x, vx, y, vy, MIO, VehicleDir
Output: statusConflict, statusP_Crash, statusCrash

1: **For each** time step
2: Initialise statusConflict, statusP_Crash, statusCrash, τ_{cut} (3.21), δ_{Cn} (3.22), TTE_{cn} (3.23), τ_{pcr} (3.25), δ_{pcr} (3.26), τ_{cr} (3.28), δ_{cr} (3.29)
3: Calculate conflict event using (3.24)
4: Calculate potential crash event using (3.27)
5: Calculate crash event using (3.30)
6: **End for**

5.3.2.2 Crash Events Detection and Classification

The study adopted the recommended tracking configuration from Chapter 4, utilising six radars and two cameras within a centralised sensor architecture, as outlined in Table 5-1, for the crash reconstruction experiments. To illustrate the tracking performance, Figure 5.4 presents the results from the replicated simulation using the CIREN-226 crash scenario. The analysis reveals that the Generalised Optimal Sub-Pattern

Assignment (GOSPA) errors, localisation errors, and missed target errors remain impressively low, all within three seconds leading up to the crash moment (4.7s). Specifically, missed target errors are non-existent after 0.2 seconds from the start of the trajectory, while the GOSPA and localisation errors at 1, 2, 3, and 4 seconds are 1.9, 2.5, 2.5, and 1.5, respectively. The reproduced tracking performance is identical to the tracking performance evaluated in Section 4.5.2, demonstrating the accuracy and reliability of the tracking setup for AV crash reconstruction.

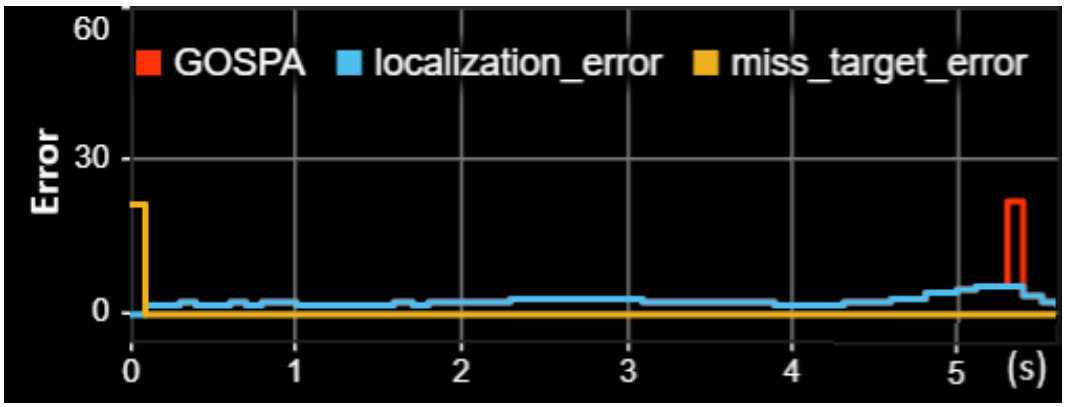


Figure 5.4: Evidence-based (CIREN-226) tracking performance measurement: GOSPA errors, localisation errors, and missed target errors are illustrated for the total trajectory time in seconds.

The proposed Vehicle Crash Reconstruction Method (VCRM) integrates crash event classification algorithms to accurately reconstruct evidence-based crash scenarios, including front or head-on, rear-end, and side-impact collisions using the CIREN-664, CIREN-816, and CIREN-226 datasets, respectively. In addition to these specific crash scenarios, one hundred diverse driving scenarios were simulated to facilitate sensor-based crash reconstructions for AVs. The analysis of these 103 driving scenarios revealed a total travel time of 36.55 minutes, covering 64,000 metres by the ego vehicle. The simulation process took 534.38 seconds, generating a data size of 4919.98 MB. These 103 driving scenarios were employed to test sensor fusion-based pre-crash and crash event classification techniques. The VCRM demonstrated capabilities in detecting and classifying four key events: cut-in, conflict, potential crash, and crash events, using the algorithms detailed in 5.1 and 5.2. The classification results,

shown in Figure 5.5, reveal that out of 103 simulated scenarios, only 12 featured cut-in events. The conflict event emerged as the most frequent, occurring 49 times, while potential crashes were identified in 42 scenarios. The VCRM successfully labelled a total of 31 confirmed crashes. This detailed classification of pre-crash and crash events and sensor data from each driving scenario form the foundation for preparing training data for machine learning. The following subsection outlines the data preparation process for this next experimentation phase.

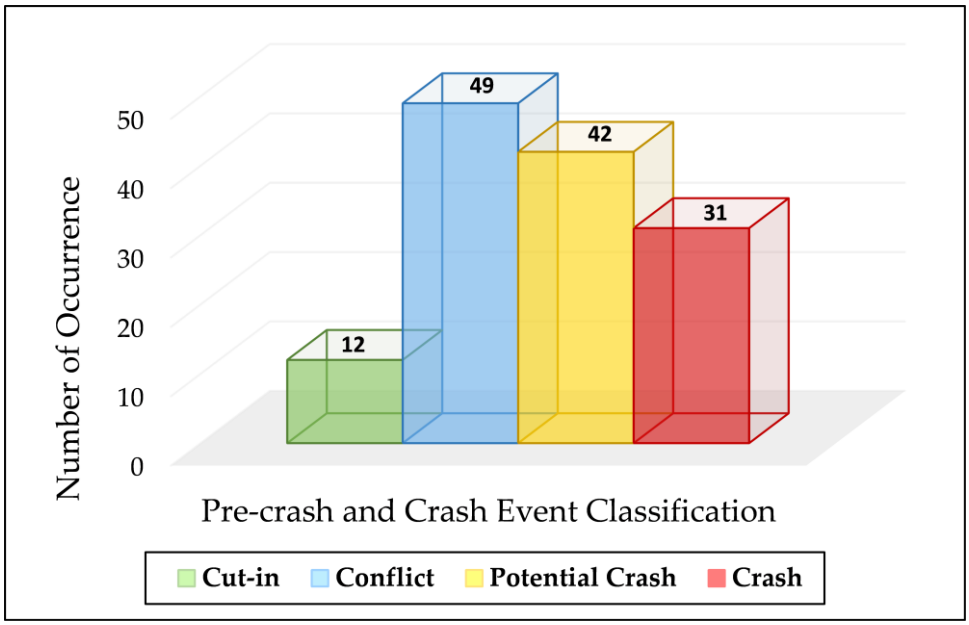
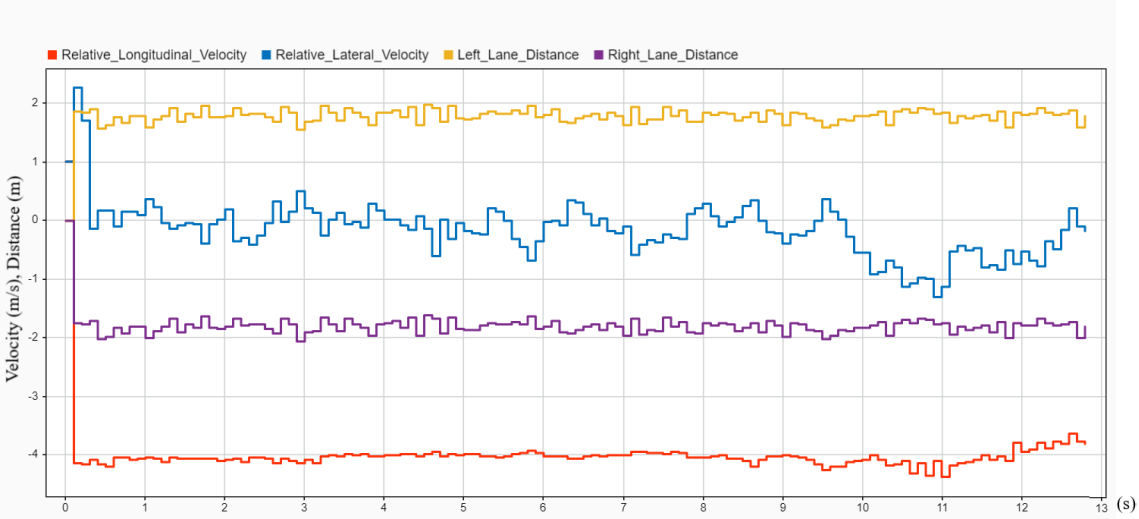


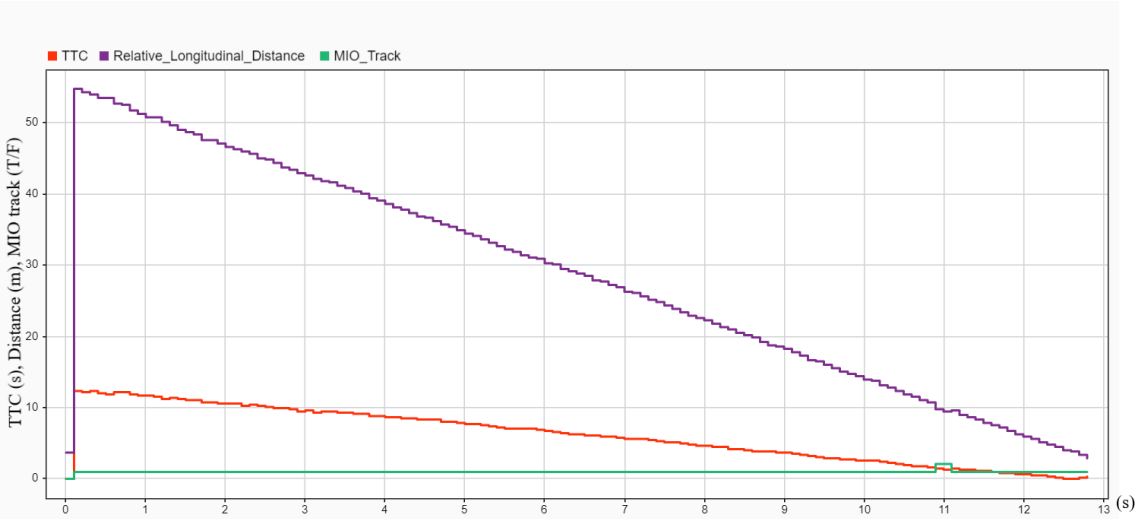
Figure 5.5: The number of driving scenarios classified as pre-crash and crash events by the VCRM.

5.3.3 Sensor Data Processing for Machine Learning

Using MATLAB-Simulink's live data recording and analysis capabilities, the study processed and examined the collected data to capture key kinematic variables across each crash simulation. These variables included the relative longitudinal and lateral distances, relative longitudinal and lateral velocities between the ego vehicle and other vehicles, all obtained through MIO detection via sensor fusion. The VCRM efficiently auto-labelled key events such as cut-ins, conflicts, potential crashes, and actual crashes, ensuring an accurate classification process. Figure 5.6 presents a snapshot of the processed AV sensor data from crash scenario 93. In Figure 5.6a, the relative longitudinal and lateral velocities between the ego vehicle and the other vehicle are displayed alongside the calculated left and right lane distances. Meanwhile, Figure 5.6b illustrates the calculated TTC, relative longitudinal distance, and MIO tracks for the same driving scenario, offering a view of the sensor data in action.



(a)



(b)

Figure 5.6: Simulated AV sensor data analysis using crash scenario 93, the x-axis presents the simulation time (s). The y-axis presents velocity (m/s), distance (m), and MIO track (s): (a) presents relative longitudinal velocity and relative lateral velocity between the ego vehicle and another vehicle, left lane distance and right lane distance of the ego vehicle; (b) presents TTC, relative longitudinal distance, and MIO tracks over the simulation time.

The classification of cut-in, conflict, potential crash, and crash events from a simulated crash scenario is presented in Figure 5.7. The measured variables from these simulations were processed through the VCRM's crash event classification subsystem, enabling the auto-labelling of pre-crash and crash events and generating precise crash

information. As each event—cut-in, conflict, potential crash, and crash—is detected, it is automatically recorded and classified, facilitating the training of machine learning models with these simulated AV sensor data. These recorded data were subsequently exported into Excel files and compiled from the reconstructed crash scenarios. The AV sensor-processed data, accumulated in an Excel file, formed the basis for creating a training dataset for crash prediction. The dataset is accessible via the link provided in (Haque, 2024). The generated dataset, which includes data from 103 driving scenarios, contains a total of 15,820 crash and non-crash records, with 31 crash scenarios contributing 3,666 crash and non-crash records. A partial preview of this extensive dataset is showcased in Figure 5.8.

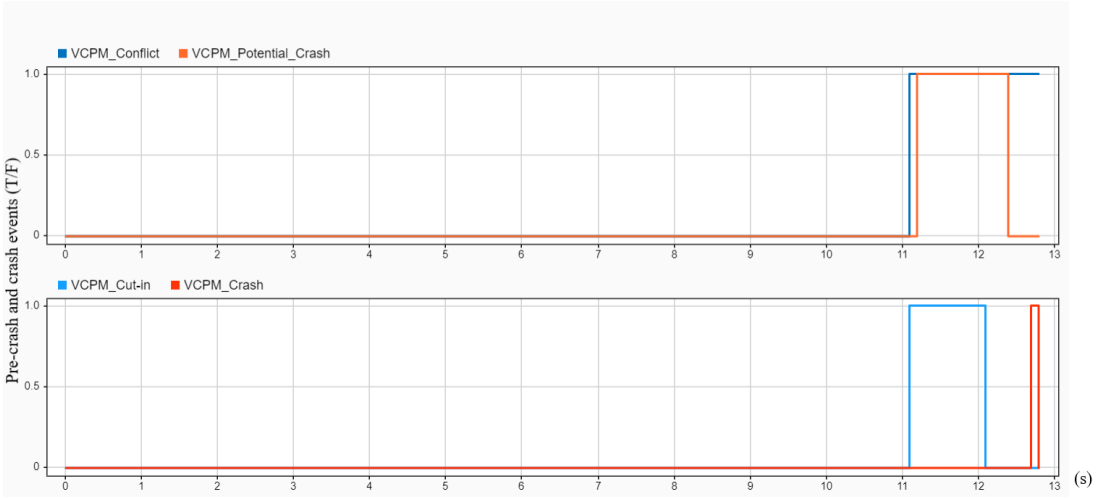
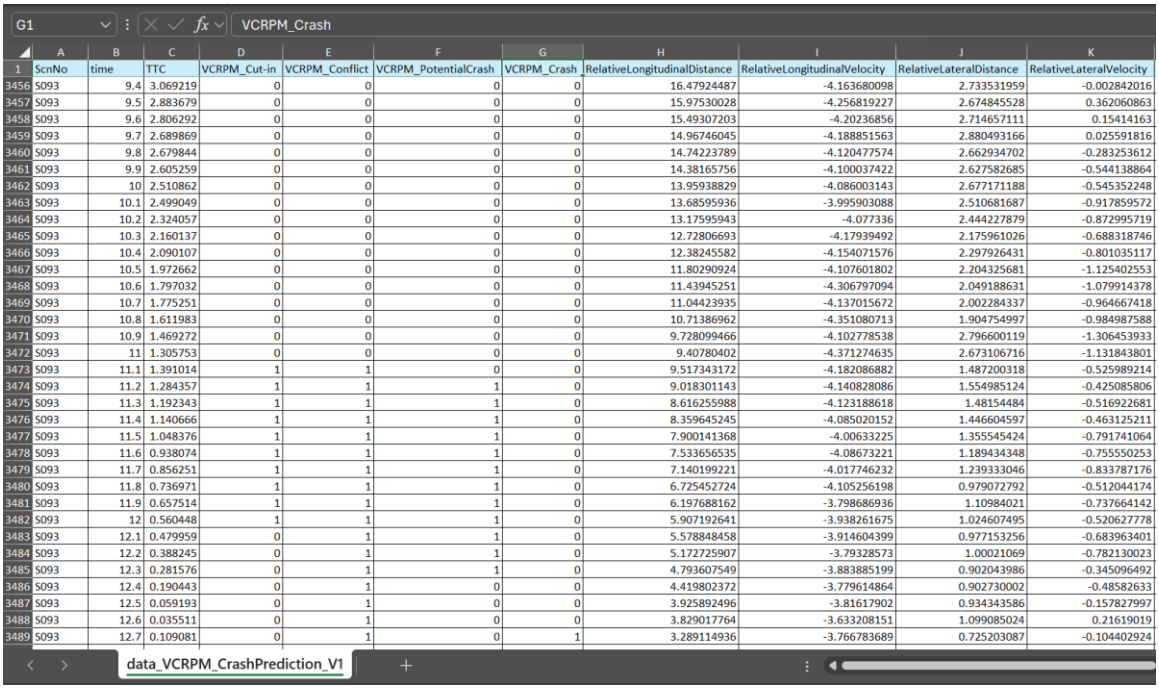


Figure 5.7: The VCRM classified pre-crash and crash events from the simulated scenario (scenario number 93). The x-axis presents the simulation time (s), and the y-axis presents crash events (T/F). Pre-crash events (cut-in, conflict, potential crash) and crash events were auto-labelled during the simulated crash scenario. At 12.7 s of the simulation, the crash event was detected as having a TTC value of 0.36 s.

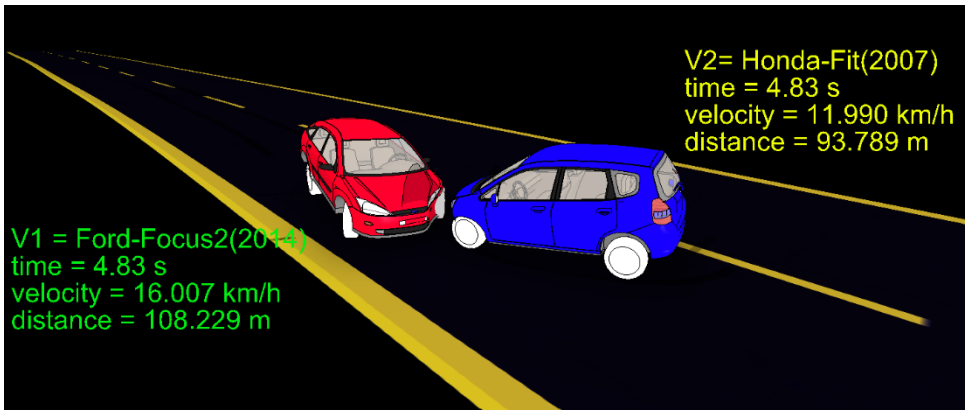


	A	B	C	D	E	F	G	H	I	J	K
1	ScnNo	time	TTC	VCRPM_Cut-in	VCRPM_Conflict	VCRPM_PotentialCrash	VCRPM_Crash	RelativeLongitudinalDistance	RelativeLongitudinalVelocity	RelativeLateralDistance	RelativeLateralVelocity
3456	S093	9.4	3.069219	0	0	0	0	16.47924487	-4.163680098	2.733531959	-0.002842016
3457	S093	9.5	2.883679	0	0	0	0	15.97530028	-4.256819227	2.674845528	0.362060863
3458	S093	9.6	2.806292	0	0	0	0	15.49307203	-4.20236856	2.714657111	0.15414163
3459	S093	9.7	2.689869	0	0	0	0	14.96746045	-4.188851563	2.880493166	0.025591816
3460	S093	9.8	2.679844	0	0	0	0	14.74223789	-4.120477574	2.662934702	-0.283253612
3461	S093	9.9	2.605259	0	0	0	0	14.38165756	-4.100037422	2.627582685	-0.544138864
3462	S093	10	2.510862	0	0	0	0	13.95938829	-4.086003143	2.677171188	-0.545352248
3463	S093	10.1	2.499049	0	0	0	0	13.68595936	-3.995903088	2.510681687	-0.917859572
3464	S093	10.2	2.324057	0	0	0	0	13.17595943	-4.077336	2.444227879	-0.872995719
3465	S093	10.3	2.160137	0	0	0	0	12.72806693	-4.17939492	2.175961026	-0.688318746
3466	S093	10.4	2.090107	0	0	0	0	12.38245582	-4.154071576	2.297926431	-0.801035117
3467	S093	10.5	1.972662	0	0	0	0	11.80290924	-4.107601802	2.204325681	-1.125402553
3468	S093	10.6	1.797032	0	0	0	0	11.43945251	-4.306797094	2.049188631	-1.079914378
3469	S093	10.7	1.775251	0	0	0	0	11.04423935	-4.137015672	2.002284337	-0.964667418
3470	S093	10.8	1.611983	0	0	0	0	10.71386962	-4.351080713	1.904754997	-0.984987588
3471	S093	10.9	1.469272	0	0	0	0	9.728099466	-4.102778538	2.796600119	-1.306453933
3472	S093	11	1.305753	0	0	0	0	9.40780402	-4.371274635	2.673106716	-1.131843801
3473	S093	11.1	1.391014	1	1	0	0	9.517343172	-4.182086882	1.487200318	-0.525989214
3474	S093	11.2	1.284357	1	1	1	0	9.018301143	-4.140828086	1.554985124	-0.425085806
3475	S093	11.3	1.192343	1	1	1	0	8.616255988	-4.123188618	1.48154484	-0.516922681
3476	S093	11.4	1.140666	1	1	1	0	8.359645245	-4.085020152	1.446604597	-0.463125211
3477	S093	11.5	1.048376	1	1	1	0	7.900141368	-4.00633225	1.355545424	-0.791741064
3478	S093	11.6	0.938074	1	1	1	0	7.533656535	-4.08673221	1.189434348	-0.755550253
3479	S093	11.7	0.856251	1	1	1	0	7.140199221	-4.017746232	1.239333046	-0.833787176
3480	S093	11.8	0.736971	1	1	1	0	6.725452724	-4.105256198	0.979072792	-0.512044174
3481	S093	11.9	0.657514	1	1	1	0	6.197688162	-3.798686936	1.10984021	-0.737664142
3482	S093	12	0.560448	1	1	1	0	5.907192641	-3.938261675	1.024607495	-0.520627778
3483	S093	12.1	0.479959	0	1	1	0	5.578848458	-3.914604399	0.977153256	-0.683963401
3484	S093	12.2	0.388245	0	1	1	0	5.172725907	-3.79328573	1.00021069	-0.782130023
3485	S093	12.3	0.281576	0	1	1	0	4.793607549	-3.883885199	0.902043986	-0.345096492
3486	S093	12.4	0.190443	0	1	0	0	4.419802372	-3.779614864	0.902730002	-0.48582633
3487	S093	12.5	0.059193	0	1	0	0	3.925892496	-3.81617902	0.934343586	-0.157827997
3488	S093	12.6	0.035511	0	1	0	0	3.829017764	-3.633208151	1.099085024	0.21619019
3489	S093	12.7	0.109081	0	1	0	1	3.289114936	-3.766783689	0.725203087	-0.104402924

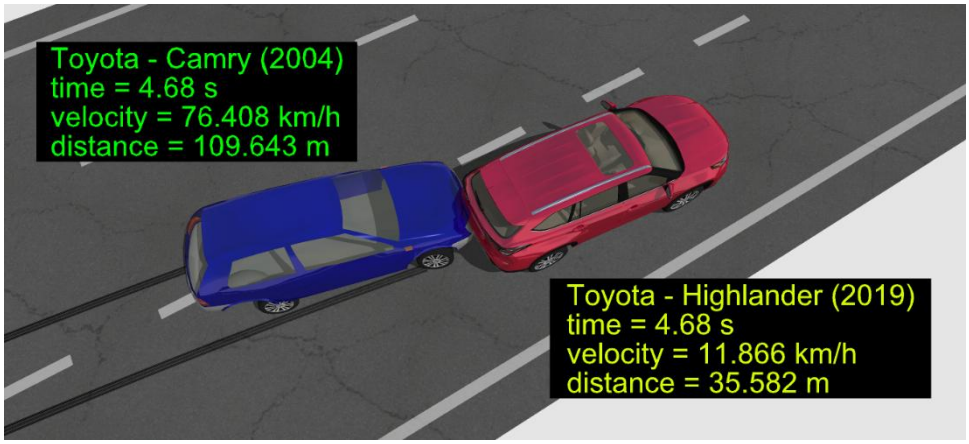
Figure 5.8: A snapshot of the training dataset.

5.3.4 Evaluation

This research analysed simulated crashes by comparing them with actual crash data. The study evaluated the outcomes of crashes simulated using the Virtual CRASH software (CRASH, 2024) alongside the proposed VCRM. Additionally, the research compared the crash event detections made by the VCRM with actual EDR data, providing an assessment of the model's performance. Evidence-based crashes were simulated using the CIREN dataset to gain insights into the crash sequences through animation and evaluate the accuracy of crash event detection by the VCRM. For instance, the front/head-on crash simulation is depicted in Figure 5.9a, while the simulated rear-end crash is shown in Figure 5.9b. These figures display key crash-related variables such as crash occurrence time, the velocity at the moment of impact, and the distances travelled by both the ego and the other vehicles throughout the simulation. The analysis revealed that the average crash moment occurred within less than one second, highlighting the precision of the simulated crash duration in the study.



(a)

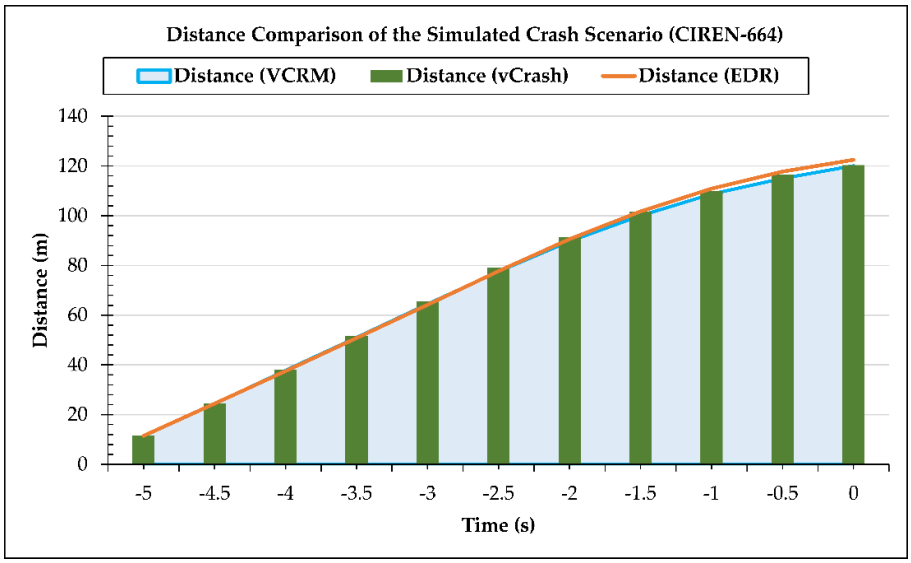


(b)

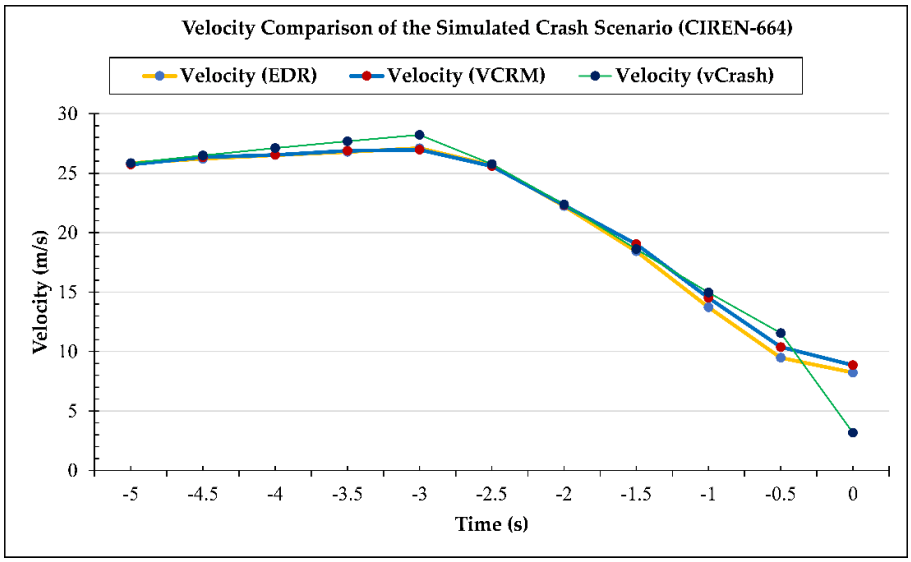
Figure 5.9: Simulated evidence-based vehicle crashes using Virtual CRASH software: (a) CIREN-664, front/head-on crash; (b) CIREN-816, rear-end crash.

The comparison between the actual EDR data and the simulated data from both Virtual CRASH (vCrash) and the VCRM are presented using the CIREN-664 crash scenario, as shown in Figure 5.10. The distance travelled by the ego vehicle over the simulation time was nearly identical across all three sources, as illustrated in Figure 5.10a. However, in Figure 5.10b, the velocity data over the simulation time reveals notable differences between the actual crash data (EDR data) and vCrash data. At –3 seconds in the simulation, the vCrash velocity was 28.21 m/s, while the actual EDR recorded velocity was 27 m/s. At the crash moment (0 seconds), the difference between the velocity detected by vCrash and the actual EDR velocity was significant, exceeding five m/s. In contrast, the ego vehicle's velocity, as measured by both the EDR and

VCRM, was almost identical throughout the simulation. The only notable deviation occurred at –0.5 seconds, where the VCRM-detected velocity was 0.8 m/s higher than the EDR-recorded velocity, emphasising the subtle variations in simulation accuracy and real-world data alignment.



(a)



(b)

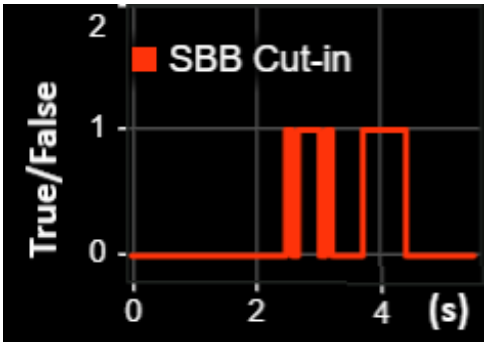
Figure 5.10: Comparing crash data of the simulated crash scenario of CIREN-664: (a) comparing travelled distance over the simulation time between the EDR (actual), the vCrash, and the VCRM; (b) comparing velocity over the simulation time between the EDR (actual), the vCrash, and the VCRM.

This research evaluated the classification of pre-crash and crash events using simulated driving scenarios. The trajectory-based algorithms for detecting cut-in and crash events demonstrated notable accuracy, identifying all cut-in events with only minor errors and detecting all crash events with zero errors, as detailed in Table 5-3. For instance, the VCRM successfully reconstructed the CIREN-664 crash scenario, detecting cut-in events just 0.2 seconds after the actual event and the CIREN-816 scenario with a mere 0.1-second delay. In contrast, the developed method achieved perfect accuracy in detecting cut-in events for the CIREN-226 crash scenario. Overall, the VCRM-based crash reconstruction enhances AV crash detection capabilities. The algorithms developed for classifying pre-crash and crash events offer precision, allowing for the accurate reconstruction of all crash types and thereby advancing the reliability of AV crash analysis.

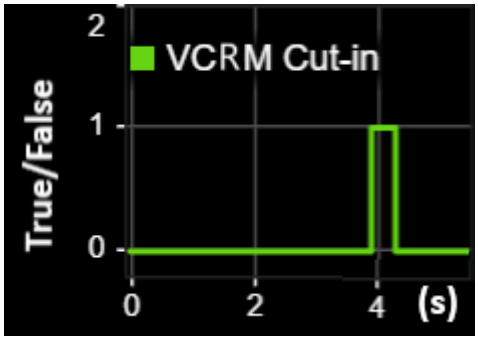
Table 5-3: Cut-in and crash event detections by the VCRM.

Scenario	Cut-in Detection	Crash Detection
CIREN-664	Detected (timing errors, +0.2s)	Accurate (no error)
CIREN-816	Detected (timing error, +0.1s)	Accurate (no error)
CIREN-226	Accurate (no error)	Accurate (no error)

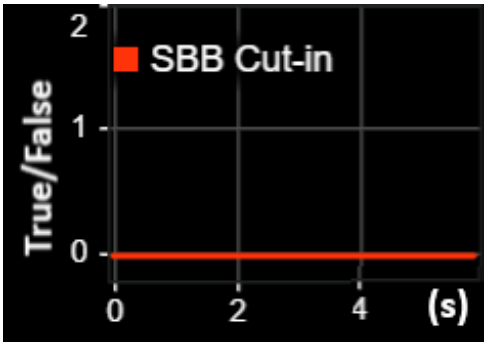
This research delved into the analysis of pre-crash (cut-in) and crash event detections from simulated crash scenarios to further assess the performance of crash reconstruction. A comparative evaluation was conducted between the recently proposed Smart Black Box (SBB) approach (Yao & Atkins, 2018, 2021) and the developed VCRM. The results demonstrate that the proposed algorithms outperform the SBB in detecting pre-crash (cut-in) and crash events, as shown in Figure 5.11 and Figure 5.12. The VCRM detects cut-in events with minimal timing errors, as seen in Figure 5.11b, where a mere 0.2-second delay occurs from the actual cut-in at 3.6 seconds into the trajectory. Similarly, Figure 5.11d illustrates a cut-in detection just 0.1 seconds after the actual event. In Figure 5.11f, the VCRM accurately identifies the cut-in event at 4.8 seconds of the trajectory time.



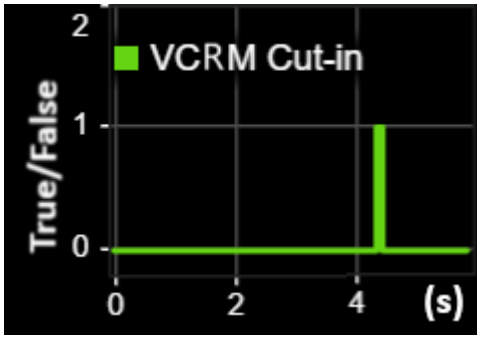
(a) SBB-CIREN-664



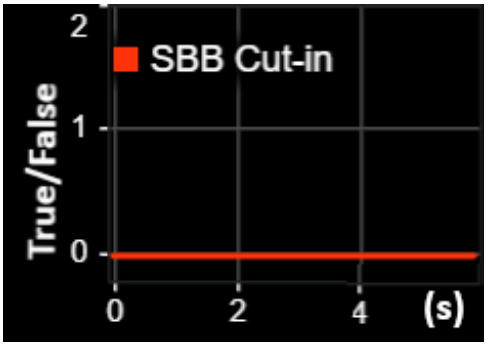
(b) VCRM-CIREN-664



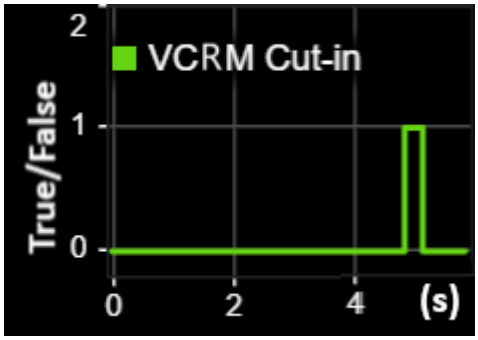
(c) SBB-CIREN-816



(d) VCRM-CIREN-816



(e) SBB-CIREN-226



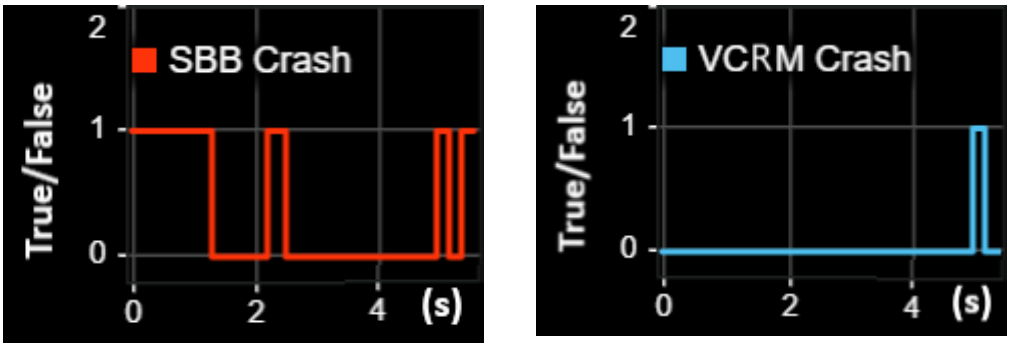
(f) VCRM-CIREN-226

Figure 5.11: Performance evaluation of cut-in event detection: The VCRM detected better cut-in events, as shown in (b), (d), and (f), than the SBB cut-in event detections as presented in (a), (c), and (e).

In contrast, the SBB shows a prolonged detection duration, from 3.7 to 4.4 seconds, as demonstrated in Figure 5.11a. Additionally, it erroneously detects a false positive cut-in between 2.5 and 3.2 seconds. Furthermore, the SBB fails to detect cut-in events

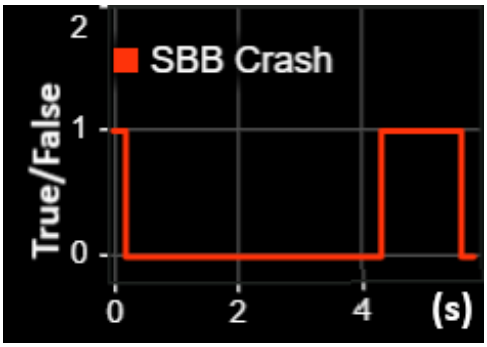
in the rear-end (CIREN-816) and side-impact (CIREN-226) crash scenarios, as evidenced in Figure 5.11c and Figure 5.11e, respectively. These findings underscore the accuracy and reliability of the VCRM in detecting pre-crash and crash events, setting it apart from approaches like the SBB.

Figure 5.12 highlights the capability of the VCRM in accurately detecting crashes across a range of scenarios, including head-on, rear-end, and side-impact collisions. The crash event detection duration is precise, falling between 4.9 and 5.1 seconds, as shown in Figure 5.12b. In Figure 5.12d, the detection time ranges from 4.3 to 4.9 seconds, and in Figure 5.12f, it spans from 4.8 to 5.1 seconds, demonstrating the VCRM's reliability in reconstructing crash events with minimal error. In contrast, while the SBB can detect crash events, it struggles with false positives and inaccuracies. For instance, in Figure 5.12a, a false positive crash is detected from 2.2 to 2.5 seconds. Figure 5.12c illustrates the SBB's inability to distinguish between a false positive crash (from 0 to 0.2 seconds) and an inaccurate true positive detection (from 4.3 to 5.6 seconds). Similarly, Figure 5.12e shows an accurate, true positive crash detection, yet it includes a false positive event at the beginning of the trajectory. Overall, the VCRM stands out for its ability to reconstruct all types of crash scenarios with precision and reliability, making it a superior tool for detecting both cut-in and crash events.

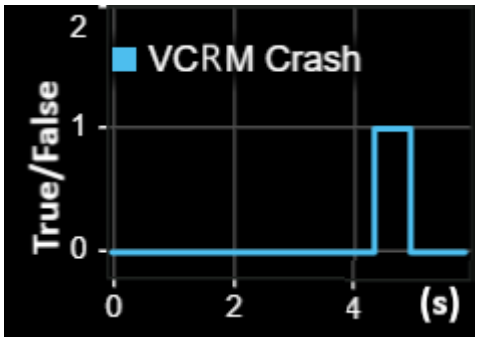


(a) SBB-CIREN-664

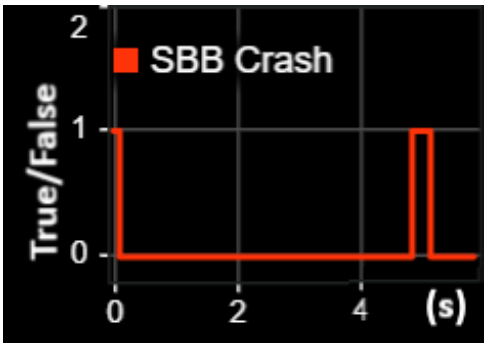
(b) VCRM-CIREN-664



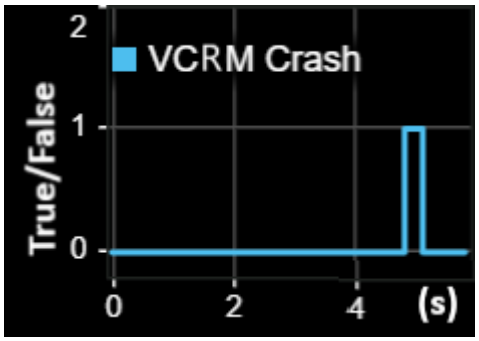
(c) SBB-CIREN-816



(d) VCRM-CIREN-816



(e) SBB-CIREN-226



(f) VCRM-CIREN-226

Figure 5.12: Performance evaluation of crash event detection: The VCRM crash detections are accurate, as shown in (b), (d), and (f), whereas the SBB includes false positive detections and errors in crash detections as depicted in (a), (c), and (e).

5.4 Discussion

This research developed a Vehicle Crash Reconstruction Method (VCRM) featuring innovative algorithms for pre-crash and crash event detection, explicitly tailored for AV crash reconstructions, and sheds light on RQ2 of this thesis. Using trajectory-based algorithms—ranging from cut-in and conflict detection to potential crash and crash

event identification—this method accurately reconstructs AV crash scenarios, providing detailed insights into both pre-crash and crash events. Through a detailed analysis of over a deep summated 100 crash scenarios, this study uncovers key findings that enrich the understanding of vehicle crash dynamics, as summarised in Table 5-4.

Table 5-4: Crash reconstruction information classification using AV sensor data.

Information	AV Sensor Data	Crash Period
Vehicle	size, weight, model, year	pre-crash
Crash Type	front, rear-end, side-impact crashes, and rollovers	in-crash
Scenario and environment	vehicle positions, local and world coordinate, road map	pre-crash, in-crash, post-crash
Vehicle Dynamics	vehicle speed, distance, direction, driver input	pre-crash, in-crash, post-crash
Crash data	collision points, angle of collision, post-impact movements	in-crash, post-crash
Event information	time series crash event detection and classification	pre-crash, in-crash, post-crash

The key findings of this study are outlined below:

- The sensors used in AV perception, object detection, and autonomous driving can also be harnessed to generate precise crash reconstruction data. This research validates and expands on previous work (Ortiz & Dávila, 2023), reinforcing the growing potential of AV sensor utilisation for vehicle crash analysis.
- The VCRM method developed in this study can autonomously label both pre-crash and crash events. This system can produce 3D reconstructions of crashes, offering important pre-crash and crash information—making it an invaluable tool for vehicle crash forensics (Kamnik et al., 2020).
- This research identifies vehicles in real-time during crash events by applying sensor fusion data to the developed MIO detection algorithm. The performance

of fusion-based tracking and object detection—compared to traditional single-sensor systems—proves to be superior in detecting MIOs, as noted in (Haque et al., 2024).

- The VCRM's ability to gather live pre-crash data—including crash events and kinematic information—enables the reconstruction of real-time AV crashes on any roadway. This study also introduces an approach to generating a crash dataset exclusively from sensor data, involving cut-in, conflict, potential crash, and crash event information—an innovative step toward real-time crash prediction and a leap in AV safety.

Finally, this research provides the following recommendations for advancing AV sensor data-driven crash reconstruction, setting the stage for future innovations in autonomous vehicle safety.

- A well-designed multi-sensor arrangement—such as the one recommended by (Haque et al., 2024)—is important for generating high-accuracy sensor data, ensuring the precision and reliability needed for effective crash reconstructions.
- It is crucial to generate comprehensive pre-crash data to provide the foundation for evidence-based crash reconstruction, ensuring that all critical events leading up to an accident are captured and analysed with precision.

Future Directions

Simulating different driving conditions, such as multiple-vehicle crashes and road environment variations, can include diverse types of crashes in the training data, which are out of this research scope.

Crash reconstruction is crucial in crash forensics, helping to analyse accidents and determine their causes and contributing factors. It helps to clarify the causes and dynamics of crashes, supports legal proceedings, and offers insights for enhancing safety. The development of automated crash forensics techniques, based on the crash reconstruction method presented in this thesis, will contribute to the future of crash

forensics.

5.5 Summary

This research addressed an issue of crash reconstruction using AV sensor data. The newly developed Vehicle Crash Reconstruction Method (VCRM) stands as a robust tool capable of accurately reconstructing various types of collisions, such as head-on and rear-end crashes, offering immense value for creating training datasets for AV crash-related research. The VCRM integrated four subsystems—sensor and environment, surround vehicle sensor fusion, crash event classification, and metrics assessment—to collect real-time AV sensor data, ensuring detailed and precise crash information presentation. The proposed crash event classification algorithms excelled in detecting and classifying key crash events, including pre-crash scenarios (cut-in, conflict, and potential crash) as well as actual crash occurrences.

A key achievement of this study is the generation of an evidence-based training dataset using simulated AV sensor data, making it suitable for AV crash prediction. This dataset can support future AV safety advancements and improve predictive capabilities. The main contributions of this research are:

- This research developed an innovative approach for crash reconstruction powered by simulated AV sensor data.
- The VCRM is reproducible, paving the way for creating valuable training datasets for future AV safety innovations.
- The developed method can provide appropriate data from vehicle crashes, which could play an important role in future autonomous vehicle accident forensics and preventing accidents.

This chapter delivers a training dataset derived from simulated AV sensor data, which will be used in Chapter 6 for real-time crash prediction, further establishing its significance in the evolution of AV safety systems.

Chapter 6 Machine Learning Models for Real-Time Crash Prediction Using Simulated AV Sensor Data

Chapter 5 developed a vehicle crash reconstruction method to classify crash events and to prepare a training dataset using simulated AV sensor data. Now that it has been shown that crash events can be reconstructed from sensor data, the question arises as to whether the same data can be used effectively for predicting crash events before they occur or whether additional data may be required to make these predictions more effective. In other words, the investigation is into whether data generated from crash simulations can also be utilised to train AVs to predict impending crashes, thereby enabling appropriate and timely actions to prevent such accidents. Answering this question may reveal important information on how far ahead of the crash a prediction can be made to allow avoidance action to be taken. To this end, the main contribution of this chapter is the proposed ML models for crash prediction. This chapter has utilised the crash dataset prepared in the previous chapter to train ML models for data-driven real-time crash prediction. This research has experimented with sixteen ML models and found that three ensemble models can produce perfect crash predictions. These self-explanatory ML models could contribute to AV driving safety, crash warning, and crash avoidance.

6.1 Introduction

The focus of this chapter is to develop a new approach to predict real-time AV crashes in different simulated driving scenarios using data generated from the AV sensors. Section 6.2 demonstrates the real-time crash prediction methods, followed by a brief background study.

Section 6.3 starts with the experiment setup for this chapter and then presents a real-time crash warning dashboard based on the simulated AV sensor data reconstructed from the evidence-based crashes. This dashboard can display real-time pre-crash events and provides crash prediction warnings. This research has conducted an evaluation of sixteen distinct classification algorithms to develop three separate sets of ML models for data-driven real-time crash predictions in a supervised learning context. The main objective was to identify the most effective machine learning models by analysing the crash prediction performances across a range of learning algorithms, ultimately contributing to developing more accurate and reliable systems for predicting crash events. Experiment results reveal that the three ML models can predict data-driven real-time crashes perfectly by using a simulated training dataset produced in Chapter 5. Chapter 6 thoroughly evaluates the performance of the ML models and presents a self-explanatory Shapley analysis to help readers better understand them and their crash-predicting features.

The findings of this research are discussed and correlated with existing studies in Section 6.4. Finally, this chapter concludes in Section 6.5.

6.2 Methods for Real-Time Crash Prediction

6.2.1 Background Study

Present-day AVs are advancing toward full autonomy, facilitated by AI, with multi-sensor technology providing critical data important for autonomous driving capabilities. The data generated from AV sensors undergoes processing to enable key

autonomous driving features, such as object detection (Zhao et al., 2020), adaptive cruise control (Wei et al., 2024), and decision-making (Ignatious et al., 2022; Modas et al., 2020). While rule-based methods are effective for low-level automation, data-driven approaches using AI, including machine learning, deep learning, and reinforcement learning, are gaining prominence for supporting complex decision-making in high-level AVs (Wang, Han, et al., 2024). Noteworthy datasets developed to support these data-driven approaches include the ML-based NuPlan, a planning benchmark for autonomous driving (Caesar et al., 2021), the Lyft Level 5 self-driving dataset for motion prediction (Houston et al., 2020), and the Waymo Open Dataset (Sun et al., 2019), all of which provide valuable resources for the research community to address real-world self-driving challenges.

While autonomous driving datasets are currently available, AV crash datasets remain relatively scarce, highlighting the necessity of incorporating relevant AV sensor data for ML-based real-time crash prediction. Researchers have emphasised the importance of having publicly available AV crash datasets for conducting investigations related to crashes (Chen et al., 2024). One of the most frequently used AV crash datasets comes from the autonomous vehicle collision reports maintained by the California Department of Motor Vehicles (CA-DMV) (DMV, 2024). It is also noted that the National Highway Traffic Safety Administration (NHTSA) offers AV crash datasets, including a summary report on Level 2 ADAS crashes (NHTSA, 2022c) and the ADS crash dataset (NHTSA, 2024).

In the above datasets, there is a need for time series pre-crash information, which is one of the important elements of real-time crash prediction. These datasets are primarily employed in vehicle crash research, including the analysis of contributing factors to AV crashes (P. Liu et al., 2024; Q. Liu et al., 2024), and crash injury severity analysis (Kuo et al., 2024). It is also worth mentioning that current real-time crash predictions involve data collection using sensors in a specific area over an extended period to generate a comprehensive crash dataset, which may lead to data imbalance issues. Studies addressing this challenge include those by (Hussain et al., 2023; Islam,

Abdel-Aty, Islam, et al., 2024; Islam, Abdel-Aty, Wang, et al., 2024). A major challenge for ML applications in crash prediction is the need for suitable AV sensor data that can facilitate accurate predictions across diverse driving scenarios. Therefore, developing an effective mechanism for accurate crash prediction using ML is important, as it would enable real-time crash warnings and allow for timely corrective actions to prevent potential collisions.

6.2.2 Process Flow of the Crash Prediction

The primary objective of this research was to create a robust, data-driven crash prediction model using simulated AV sensor data. To this end, the real-time crash prediction component of the proposed Vehicle Crash Reconstruction and Prediction Model (VCRPM), as detailed in Section 3.4, takes the lead in predicting V2V crashes.

The ML-based real-time crash prediction component is built around a crash prediction subsystem, which operates through a series of processes in a cyclic sequence. This setup allows experimentation with various ML models and their corresponding learning algorithms. The occurrence of a crash event (the target factor) serves as the basis for real-time crash prediction using ML. This target factor is classified as a binary value, either the crash or non-crash (true or false). As such, this research focuses on binary classification supervised ML models to predict the likelihood of crashes and non-crashes.

The ML techniques selected for real-time crash prediction were picked based on their prominence and proven effectiveness in the existing literature, as highlighted in subsection 2.6.1. These techniques include ensemble learning, Neural Networks (NN), and Support Vector Machines (SVMs). MATLAB, with its Classification Learner app, provides a platform for conducting experiments and analysing the performance of various ML algorithms. By leveraging such a wide array of learning algorithms, it becomes possible to pinpoint the most effective models for predicting crashes, enhancing the development of more accurate and reliable prediction systems.

In this research, to identify the most effective models for crash prediction, five ensemble learners (boosted trees, bagged trees, subspace discriminant, subspace KNN, and RUSBoosted trees), five feedforward neural networks (narrow NN, medium NN, wide NN, bilayered NN, and trilayered NN), and six support vector machines (linear SVM, quadratic SVM, cubic SVM, fine Gaussian SVM, medium Gaussian SVM, and coarse Gaussian SVM) were put to the test using the Classification Learner app. The training dataset, crafted from fifteen contributing crash factors as outlined in subsection 6.3.4, served as the primary input for training all these ML algorithms.

For further information on the default parameters utilised by MATLAB's Classification Learner App for each of the aforementioned machine learning methods, see Appendix B.

The metrics assessment subsystem within the VCRPM plays a pivotal role in evaluating training and prediction performance, ensuring the system runs at full throttle. This subsystem facilitates multiple rounds of performance evaluation, providing a continuous feedback loop that allows for fine-tuning and adjustment to hit the bullseye with the best possible crash prediction outcomes. In addition, the crash prediction component is like having a watchful eye on the road, delivering real-time crash warnings through a dashboard. This empowers stakeholders to take timely actions, steering clear of potential accidents. The overall process flow for this data-driven, real-time crash prediction system is laid out in Figure 6.1.

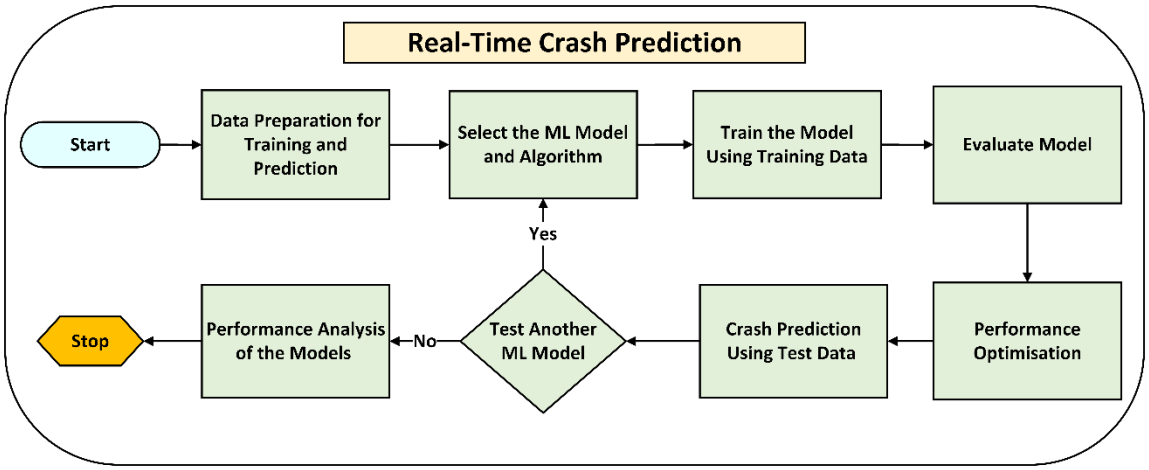


Figure 6.1: Process flow of the ML-based data-driven real-time crash prediction.

The crash prediction process begins with the important task of preparing the training data. This foundational step involves breaking the training dataset into two or three parts—training, evaluation, and test data—to ensure the successful training of ML models, assess how well they have been trained, and gauge their prediction accuracy. The most commonly approach is to allocate 70% or 80% of the data for training, leaving the remaining 20% or 30% for evaluation and testing. In this research, 80% of the data were dedicated to training, and 20% was reserved for testing the models.

The initial plan was to predict real-time crashes using ensemble learning, Artificial Neural Networks (ANN), and Support Vector Machine (SVM) models. The experiment starts by considering all 103 crash and non-crash scenarios. To tackle the issue of data imbalance—where non-crash events outnumber crash events—the models are trained on 31 crash scenarios, thus minimising the effect of the imbalance between crash and non-crash data.

When it comes to validating the models' performance, two key methods come into play: holdout and cross-validation. Cross-validation is particularly pivotal, as it tests the model's ability to predict accurately on unseen data. This approach divides the main dataset into several subsets, where some parts are used for training, while others are set aside for testing. Using multiple folds ensures the dataset is randomly split into several

chunks, with errors averaged across each fold to smooth out any randomness (MathWorks, 2024f).

Cross-validation also tackles issues like overfitting and underfitting, which can also help manage data imbalance. The models undergo multiple rounds of training and testing with various algorithms to identify the most fitting ML model for real-time crash prediction. Cross-validation can be used for hyperparameter optimisation (MathWorks, 2024d), and optimising hyperparameters can enhance the performance of ML models (Filion, 2019). This step-by-step process ensures the final model is as robust and accurate as possible.

The last stage of the crash prediction process involves comparing the performance of the evaluated models. This comparison is made using metrics like accuracy, precision, and recall, calculated with formulas (3.39), (3.40), and (3.41), respectively. The F1-score, which provides a balanced mean of recall and precision, is computed using the formula (3.42). The sequential process described here lays the groundwork for the crash prediction experiments and their evaluation, which are detailed in the following section.

6.3 Experiments, Results and Evaluation

6.3.1 Experiment Setup

This study presents real-time crash prediction methodology utilising MATLAB (R2024b), Simulink (R2024b), and the Classification Learner App (R2024b). The experiments were conducted using the same system employed for the evidence-based crash reconstruction discussed in Chapter 5.

This investigation involved sixteen classification algorithms to train three sets of ML-based supervised models for predicting real-time crashes. Detailed information regarding the 16 ML models is provided in Table 6-1. The models underwent testing through multiple iterations of ML-based supervised learning using an evidence-based

dataset derived from simulated AV sensor data, as outlined in Chapter 5. The initial dataset consisted of 103 scenarios, including crash and non-crash events, with a crash-to-non-crash ratio of 3:1, which was utilised for training and evaluating the models.

The research isolated 31 crash scenarios to create a refined dataset for addressing the data imbalance between crash and non-crash events. The same set of 16 ML models was subsequently applied to this new dataset. For ML model evaluation, a holdout method was used to assess the performance of ML algorithms during the test cycle, allocating 80% of the data for training and 20% for validation. Additionally, cross-validation techniques were implemented to evaluate the variability of results across different models. The performance of the crash predictions was compared using the original dataset of 103 scenarios and a dataset with 31 crash examples. Additionally, differences in results between the holdout and cross-validation methods were evaluated to identify the most effective ML models.

Following this analysis, the top 10 performing models were selected for performance optimisation. The cross-validation approach with an increased number of folds was used to enhance these models further and identify the most effective models for real-time crash prediction. Furthermore, a crash warning dashboard was introduced based on evidence-derived reconstructed pre-crash events, as detailed in Chapter 5.

Table 6-1: Supervised machine learning models and their properties for data-driven real-time crash prediction.

ML Technique, (No. of Models)	Algorithms	No. of Scenarios (Crash Scenario)	Validation
Ensemble, (5)	Boosted Trees, Bagged Trees, Subspace Discriminant, Subspace KNN, RUSBoosted Trees	103 (31)	Holdout, Cross-Validation
Neural Network (NN), (5)	Narrow NN, Medium NN, Wide NN, Bilayered NN, Trilayered NN	103 (31)	Holdout, Cross-Validation
Support Vector Machine (SVM), (6)	Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, Coarse Gaussian SVM	103 (31)	Holdout, Cross-Validation

6.3.2 Crash Warning Dashboard

A vehicle crash warning serves as an alert designed to inform drivers or ADS of potential or imminent hazards associated with a collision. The primary objective of a crash warning dashboard is to provide real-time crash alerts and information pertaining to the vehicle's surroundings. These real-time notifications are instrumental in anticipating and avoiding potential accidents. In contemporary vehicles, these dashboards may encompass features such as collision alerts and lane-change warnings, thereby enhancing safety measures for AVs. To this end, the proposed VCRPM is designed to display real-time pre-crash events—such as cut-ins, conflicts, and potential crash scenarios—as well as crash event detections through a crash prediction warning dashboard. When the developed model identifies a pre-crash event, it pairs this with the calculated TTC value to predict a crash at least 5 seconds in advance, with the prediction displayed on the dashboard, as shown in Figure 6.2. It is important to note that pre-crash events detected automatically trigger a potential crash warning, and incorporating ML-based crash prediction warnings could further enhance the effectiveness of this system. If a pre-crash event with a TTC of 5 seconds or less is detected, the dashboard activates the crash prediction lamp.

The final five seconds of the TTC are divided into five equal intervals (each lasting 1 second) to produce five distinct levels of crash prediction risk. Figure 6.2a illustrates the colours assigned to these five levels. In this system, crash warning state 1 signals a low risk of a crash with a green-coloured crash prediction lamp. Meanwhile, crash warning state 5 indicates a very high risk of crash, with the lamp turning red. States 2 and 3, representing medium crash risk, suggest that prompt driving interventions can prevent a collision. Specifically, crash warning state 4, marked by a yellow lamp, indicates a high risk of a crash, requiring immediate attention to avoid an accident.

Figure 6.2: This figure presents a crash warning dashboard. This dashboard displays real-time crash prediction of the simulated driving scenario (scenario number 43) using pre-crash events (cut-in, conflict, potential crash): (a) displays assigned colours (crash prediction states are 1 to 5, and the representing colours for these states are green, blue, yellow, orange, and red, accordingly) using detected pre-crash events and the calculated TTC of the crash prediction lamp; (b) displays a normal event (TTC 7 s) at the simulation time of 9.3 s; (c) displays a cut-in event detection (TTC > 5.5 s) at the simulation time of 10.2 s; (d) crash prediction lamp turned on (green, low crash risk) by the cut-in event when the TTC < 5 s at the simulation time of 10.7 s; (e) crash prediction lamp changed to blue (medium crash risk) by the cut-in event when the TTC < 4 s at the simulation time of 11.5 s; (f) crash prediction lamp changed to orange (high crash risk) by the conflict event when the TTC < 2 s at the simulation time of 13.7 s; (g) crash prediction lamp changed to red (very high crash risk) by the potential crash and conflict events when the TTC < 2 s at the simulation time of 14.2 s; (h) crash occurrence detected at the simulation time 15.0 s with the TTC value almost 0 s.

Figure 6.2b shows normal driving conditions, where no abnormal events were detected at a simulation time of 9.3 seconds. A cut-in event detection is displayed in Figure 6.2c, but the crash prediction lamp remains off as the TTC value exceeds 5 seconds at the simulation time of 10.2 seconds. Figure 6.2d shows a crash prediction scenario with the crash prediction lamp turning green following a cut-in event detection when the TTC drops below 5 seconds at the simulation time of 10.7 seconds. By simulation time 11.5 seconds, as shown in Figure 6.2e, the crash prediction lamp changes to blue when the TTC falls below 4 seconds, triggered by the cut-in event.

A high-risk crash prediction is presented in Figure 6.2f when a conflict event with a TTC value below 2 seconds is detected. The most critical scenario is demonstrated in Figure 6.2g, where a very high risk of a crash is detected due to a potential crash event with a TTC below 1 second at a simulation time of 14.2 seconds. Finally, Figure 6.2h depicts the crash event itself, detected at the simulation time of 15.0 seconds, with the TTC nearly 0 seconds.

This subsection highlights a real-time crash warning dashboard that plays a pivotal role in enhancing driving safety by triggering crash avoidance actions in potential collision scenarios. The following section will delve into the data-driven approach for real-time crash prediction.

6.3.3 ML-based Real-time Crash Prediction

This chapter used the training dataset from the previous chapter. Sixteen ML models were trained using this dataset to predict real-time crashes and determine which models perform best for the proposed VCRPM. The models' effectiveness was evaluated using the metrics described in subsection 3.5.2.

In the iterative evaluation process, all crash and non-crash driving scenarios, totalling 15,820 records, were included in the first iteration. The dataset was split so that 12,658 observations were used for training, while the remaining 3,164 records were set aside for testing the models. In this iterative phase, a holdout validation technique

was used, with 20% of the data held out for testing. Fifteen key crash-related factors were utilised to predict two response classes: 0 (non-crash event) and 1 (crash event). These crash-related factors or predictors are analysed to evaluate their impact on the model output, as presented in subsection 6.3.4. The performance results of the sixteen evaluated models are summarised in Table 6-2.

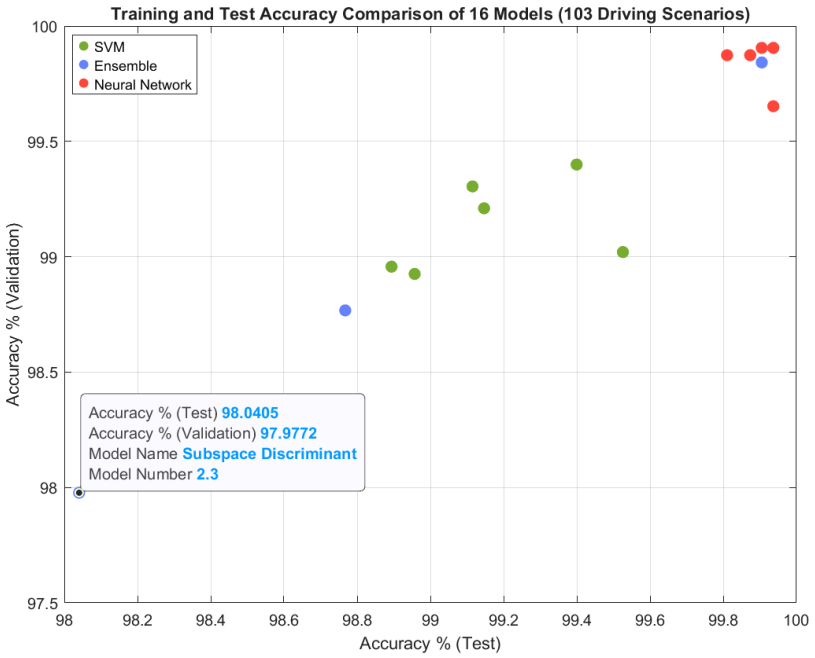
Table 6-2: Performance of the sixteen ML models for the real-time crash prediction using 103 scenarios (20% held out).

Model	Accuracy	Precision	Recall	F1-Score
Wide NN	99.94	99.94	99.94	99.94
Trilayered NN	99.94	99.94	99.94	99.94
Ensemble (Bagged Trees)	99.91	99.91	99.91	99.90
Ensemble (RUSBoosted Trees)	99.91	99.91	99.91	99.91
Bilayered NN	99.91	99.91	99.91	99.90
Ensemble (Subspace KNN)	99.87	99.87	99.87	99.87
Medium NN	99.87	99.87	99.87	99.87
Narrow NN	99.81	99.82	99.81	99.81
Quadratic SVM	99.53	99.50	99.53	99.49
Fine Gaussian SVM	99.40	99.35	99.40	99.34
Medium Gaussian SVM	99.15	99.16	99.15	99.15
Cubic SVM	99.12	99.14	99.12	99.13
Coarse Gaussian SVM	98.96	99.05	98.96	99.00
Linear SVM	98.89	99.02	98.89	98.95
Ensemble (Boosted Trees)	98.77	NaN	NaN	NaN
Ensemble (Subspace Discriminant)	98.04	98.78	98.04	98.34

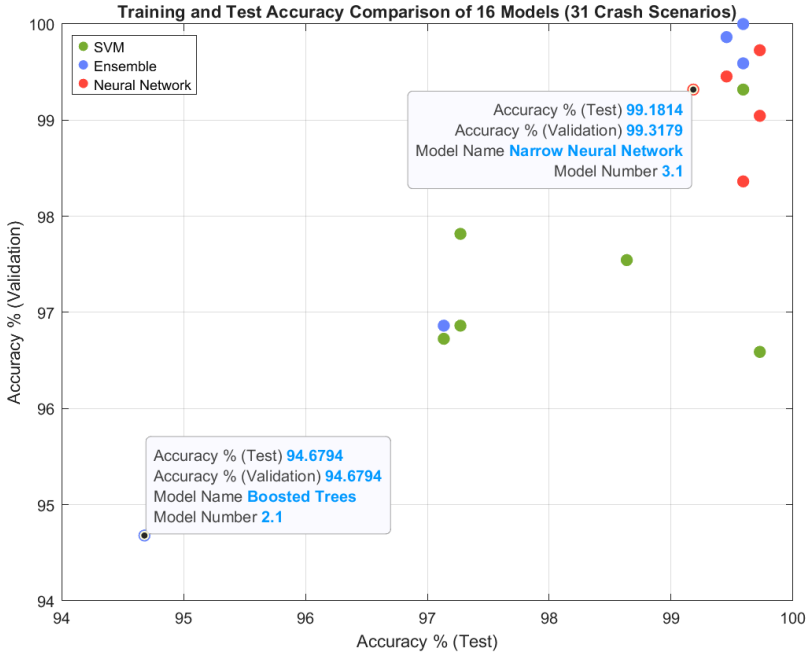
Table 6-2 is organised from the highest to the lowest accuracy metric, showcasing the weighted values for precision, recall, and F1-score. Among the sixteen machine learning models, the wide NN and the trilayered NN stood out, delivering exceptional performance with an accuracy of 99.94%. Notably, both models achieved identical values for precision, recall, and F1-score. The runner-up spot, with an accuracy of

99.91%, was achieved by three models: the ensemble (bagged trees), the ensemble (RUSBoosted trees), and the bilayered NN. The third-highest accuracy, 99.87%, was accomplished by the ensemble (subspace KNN) and medium NN models. At the bottom of the scale, the ensemble (subspace discriminant) model attained the lowest accuracy of 98.04%. The quadratic SVM and fine Gaussian SVM models made their way into the top ten, with accuracies of 99.53% and 99.15%, respectively. It is also noted that the ensemble (boosted trees) model, with an accuracy of 98.77%, failed to produce precision, recall, and F1-score metrics.

These sixteen models were tested on a dataset consisting of 103 driving scenarios, including 72 non-crash events. This research separated 31 crash scenarios to create a secondary dataset with 3,666 observations to address the data imbalance issue, particularly between crash and non-crash scenarios. In the second round of evaluations, the models were tested on the secondary dataset, applying a 20% holdout validation method to minimise performance biases. When comparing the results from the first and second iterations, it was observed that the accuracy of all sixteen models decreased in the second iteration. A comparative analysis of these results from both datasets (103 scenarios and 31 crash scenarios) is illustrated in Figure 6.3.



(a)



(b)

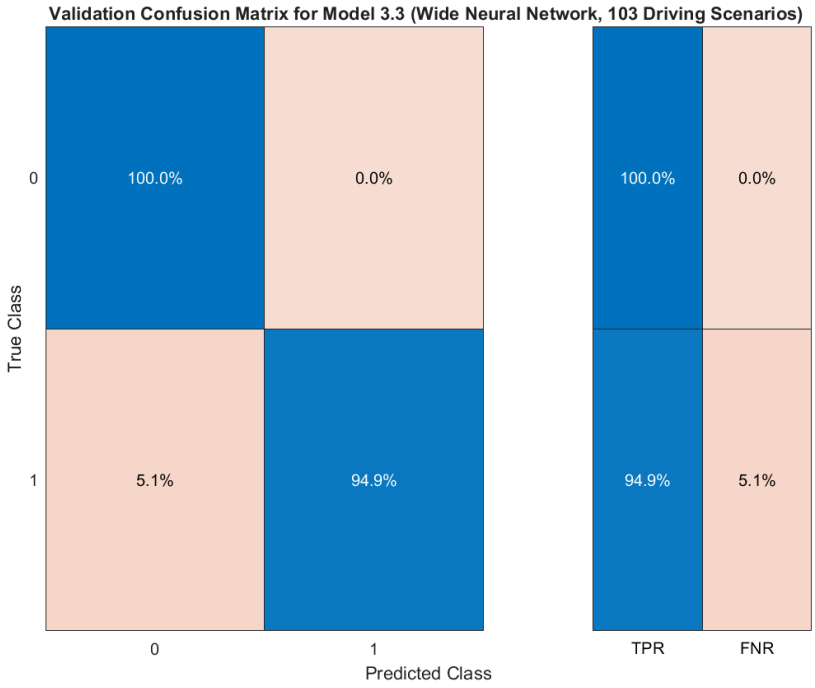
Figure 6.3: Performance comparison of the 16 models using 103 driving scenarios and 31 crash scenarios: (a) models performance using 103 driving scenarios; (b) models performance using 31 crash scenarios.

From this comparison, the study found that no lower-performing model made it into the top ten of Table 6-2. However, there were variations in the accuracy performance of the models, as shown in Table 6-3.

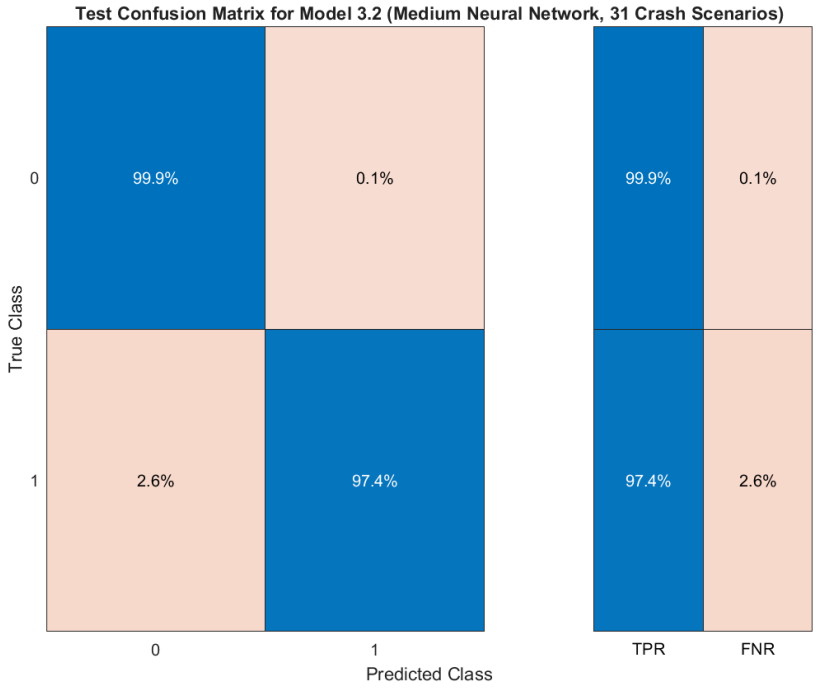
Table 6-3: Performance of the sixteen ML models for the real-time crash prediction using 31 crash scenarios (20% held out).

Model	Accuracy	Precision	Recall	F1-Score
Medium NN	99.73	99.73	99.73	99.73
Wide NN	99.73	99.73	99.73	99.73
Quadratic SVM	99.73	99.73	99.73	99.73
Ensemble (RUSBoosted Trees)	99.59	99.62	99.59	99.60
Ensemble (Subspace KNN)	99.59	99.59	99.59	99.59
Fine Gaussian SVM	99.59	99.59	99.59	99.59
Trilayered NN	99.59	99.59	99.59	99.59
Ensemble (Bagged Trees)	99.45	99.47	99.45	99.46
Bilayered NN	99.45	99.45	99.45	99.45
Narrow NN	99.18	99.18	99.18	99.18
Cubic SVM	98.64	98.59	98.64	98.60
Linear SVM	97.27	97.07	97.27	97.08
Coarse Gaussian SVM	97.27	97.07	97.27	97.08
Ensemble (Subspace Discriminant)	97.14	96.92	97.14	96.96
Medium Gaussian SVM	97.14	96.90	97.14	96.91
Ensemble (Boosted Trees)	94.68	NaN	NaN	NaN

This study went on to examine the best-performing models based on the two sets of evaluation results. A comparison of the confusion matrices for these top models is presented in Figure 6.4. The wide NN emerged as the best model in the first iteration of the evaluation (103 driving scenarios), while the medium NN was the top performer in the second phase (31 crash scenarios).



(a)



(b)

Figure 6.4: Confusion matrices of the best models of two evaluation cycles: (a) prediction performance of the wide NN model using driving 103 scenarios; (b) prediction performance of the medium NN model using 31 crash scenarios.

The wide NN model achieved 100% accuracy in predicting true positive non-crash events (class 0), as presented in Figure 6.4a, whereas the prediction accuracy of true positive non-crash events decreased a little for the medium NN model, which was 99.9%, as shown in Figure 6.4b. However, the true positive crash events (class 1) prediction improved by the medium NN model of the second evaluation results compared to the wide NN model of the first evaluation results. The true positive crash events (class 1) prediction accuracy was 97.4% by the medium NN model and 94.9% by the wide NN model.

The holdout method, a one-fold validation approach, is preferred when experimenting with large datasets to build an initial model. In contrast, cross-validation approach effectively reduces prediction errors by using multiple train-test splits and averaging the results across multiple folds (MathWorks, 2024f). The first two evaluation cycles of the ML models were conducted using holdout validation. In the third evaluation cycle, this research applied cross-validation (with 20% of observations used for testing) to predict real-time crashes using a dataset of crash-only scenarios (31 crash scenarios). For this round of experiments, the top ten models were selected from Table 6-3. Their performance evaluation is presented in Table 6-4. This evaluation began with 2-fold validation, progressively increasing the number of folds to optimise the models' performance. All ten models in Table 6-4 reached their optimal performance for crash occurrence prediction using 5-fold cross-validation. The evaluation results for 2-fold, 3-fold, 4-fold, and 6-fold validation are provided in Appendix C.

Among the three sets of ML techniques used in this research, the ensemble model set delivered the best performance, achieving perfect predictions for crash occurrences. The ensemble models included bagged trees, subspace KNN, and RUSBoosted trees. The second-best performing models were the narrow NN and medium NN, both achieving a crash prediction accuracy of 99.86%. In this evaluation cycle, all models showed improved performance compared to those presented in Table 6-3, except for the quadratic SVM model. The performance of this model decreased from 99.73% to

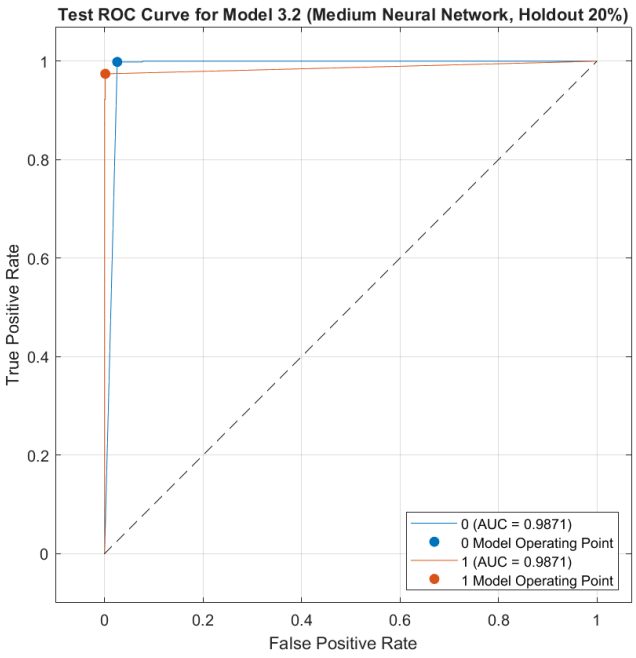
97.54%.

Table 6-4: Performance of the evaluated models (5-fold cross-validation approach) for the real-time crash prediction using 31 crash scenarios.

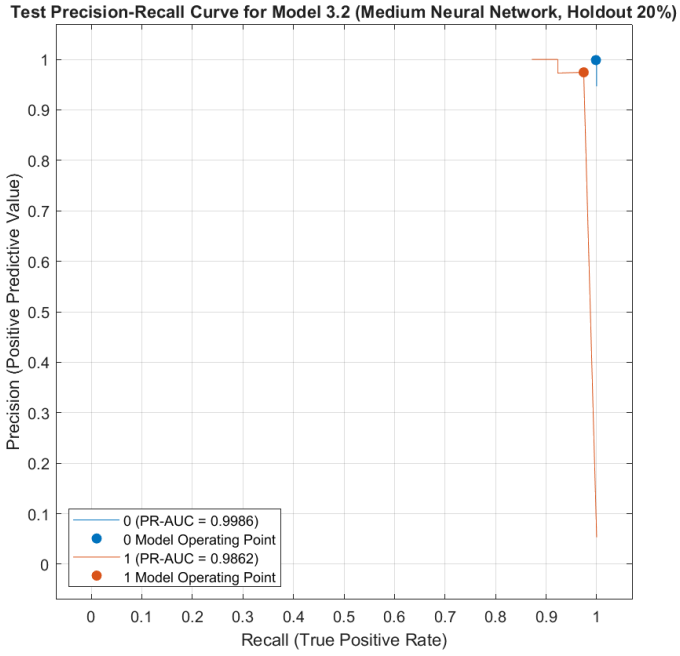
Model	Accuracy	Precision	Recall	F1-Score
Ensemble (Bagged Trees)	100.00	100.00	100.00	100.00
Ensemble (Subspace KNN)	100.00	100.00	100.00	100.00
Ensemble (RUSBoosted Trees)	100.00	100.00	100.00	100.00
Narrow NN	99.86	99.86	99.86	99.86
Medium NN	99.86	99.86	99.86	99.86
Fine Gaussian SVM	99.86	99.87	99.86	99.86
Wide NN	99.73	99.73	99.73	99.72
Bilayered NN	99.73	99.73	99.73	99.72
Trilayered NN	99.73	99.73	99.73	99.73
Quadratic SVM	97.54	97.49	97.54	97.51

In this performance evaluation, the research compared the ROC and Precision-Recall (PR) curves of the three models, as shown in Figure 6.5, Figure 6.6, and Figure 6.7.

Initially, the ROC and PR curves were generated for the medium NN model, which was the best-performing model in the second evaluation cycle, using holdout validation. These curves are shown in Figure 6.5a and Figure 6.5b. Next, the ROC and PR curves for the same medium NN model were created using 5-fold cross-validation, representing the second-best model in the third evaluation cycle. These results are displayed in in Figure 6.6a and Figure 6.6b. Finally, the subspace KNN-based ensemble model, the top model in the third evaluation cycle, was evaluated using 5-fold cross-validation, with its ROC and PR curves presented in Figure 6.7a and Figure 6.7b.

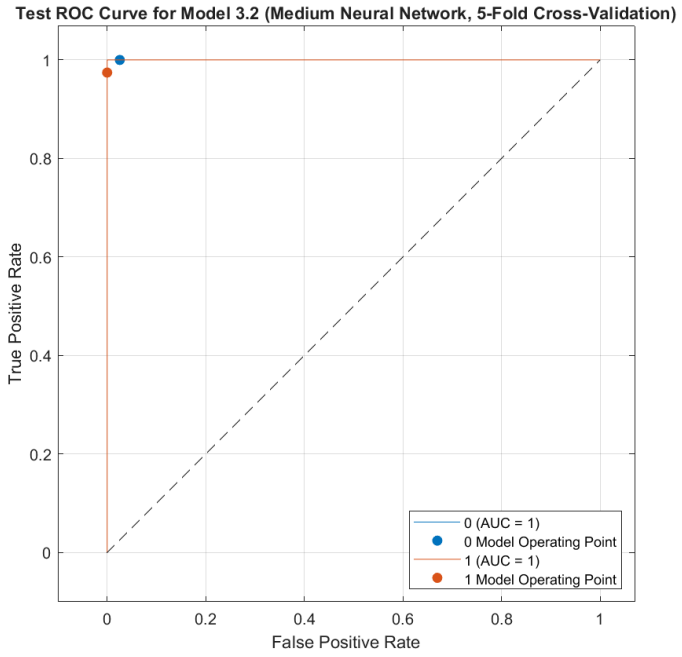


(a)

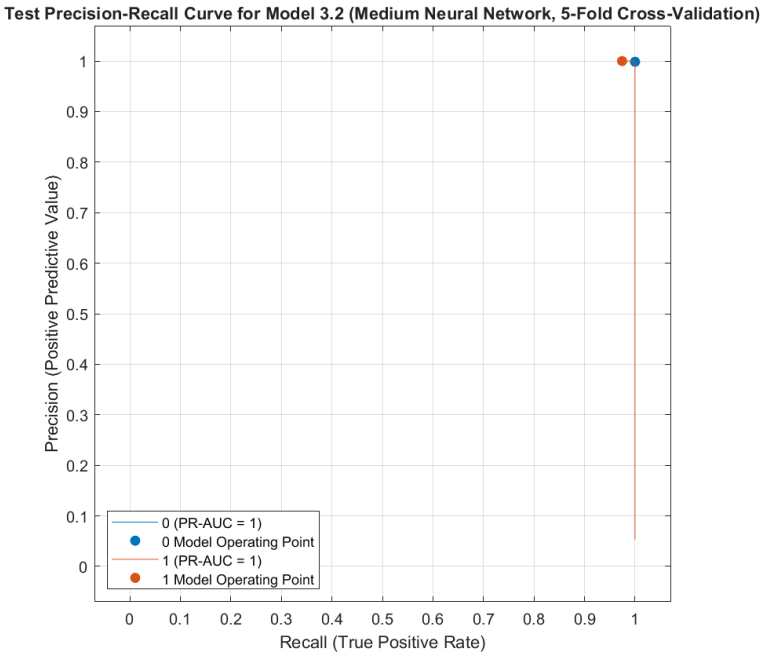


(b)

Figure 6.5: Presents ROC curve and Precision-Recall (PR) curve of medium NN models using 31 crash scenarios: (a) ROC curve of the medium NN model (holdout validation); (b) PR curve of the medium NN model (holdout validation).

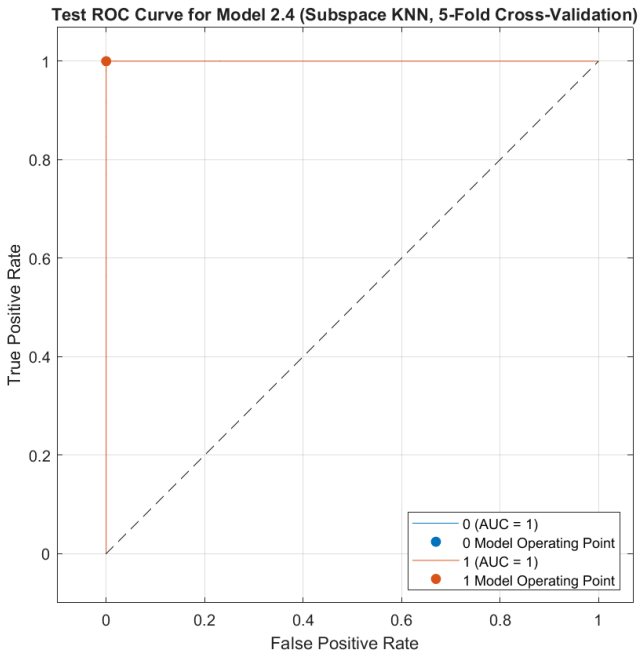


(a)

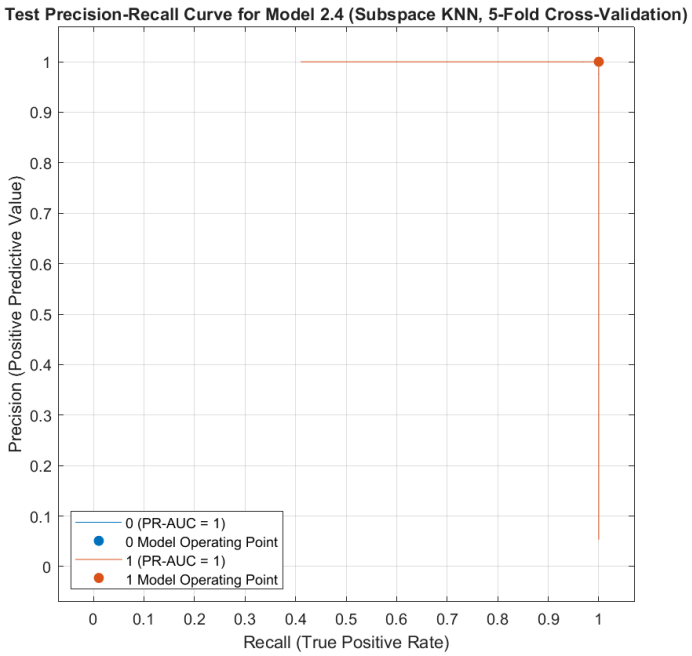


(b)

Figure 6.6: Presents ROC curve and Precision-Recall (PR) curve of medium NN models using 31 crash scenarios: (a) ROC curve of the medium NN model (5-fold cross-validation); (b) PR curve of the medium NN model (5-fold cross-validation).



(a)



(b)

Figure 6.7: Presents ROC curve and Precision-Recall (PR) curve of subspace KNN models using 31 crash scenarios: (a) ROC curve of the subspace KNN model (5-fold cross-validation); (b) PR curve of the subspace KNN model (5-fold cross-validation).

The subspace KNN-based model outperformed the others by covering the largest possible area under both the ROC and PR curves. In contrast, the ROC (Figure 6.5a) and PR (Figure 6.5b) curves for the holdout validation-based medium NN model show a smaller area under the curve compared to the medium NN model (Figure 6.6a and Figure 6.6b) evaluated using the cross-validation approach. The following subsection explains the subspace KNN model, offering a deeper understanding of the crash prediction features utilised in this research.

6.3.4 Self-Explanatory ML Model

This research applied the best-performing subspace KNN-based ensemble model to analyse the ML model's features (variables used to predict the target variable). Lundberg and Lee introduced the SHapley Additive exPlanations (SHAP) (Lundberg & Lee, 2017), a method used to explain an ML model's output by attributing each feature's contributions to a prediction (Lee et al., 2023). The SHAP method is based on the Shapley value interpretation, a tool for elucidating the workings of self-explanatory ML models (Ma et al., 2025; Ullah et al., 2023). Analysing Shapley values offers detailed insights, enabling both practitioners and non-technical stakeholders to understand how individual features influence the model's predictions (Yuchun Liu et al., 2022). The Shapley value analysis makes the model's behaviour even more transparent, which is crucial in high-stakes applications where explainability is important (Huang et al., 2024).

The use of the Shapley value to interpret ML models has gained attention in recent research (S. Ahmed et al., 2023; Ali et al., 2024). This study adopted the Shapley value interpretation, utilising 15 crash prediction features to develop a real-time crash prediction model. The influence of these features on the model's output is summarised and displayed in Figure 6.8. The Shapley value here illustrates the contribution of each feature to crash prediction. A dataset of 200 data points from observations was used to generate this Shapley summary, with feature importance arranged in descending order. Feature importance is indicated by both colour and score, where yellow (1) denotes

high importance and blue (0) represents low importance on the y-axis. The x-axis reflects the impact on the model output based on the Shapley value. Among the 15 features, the relative longitudinal distance is the most important for crash prediction, followed by TTC and relative longitudinal velocity, which are ranked second and third, respectively. Features such as potential crashes and conflicts hold the fourth and fifth positions in terms of impact on the prediction. In contrast, the cut-in feature ranks lowest in importance, with the MIO track feature being the second least important.

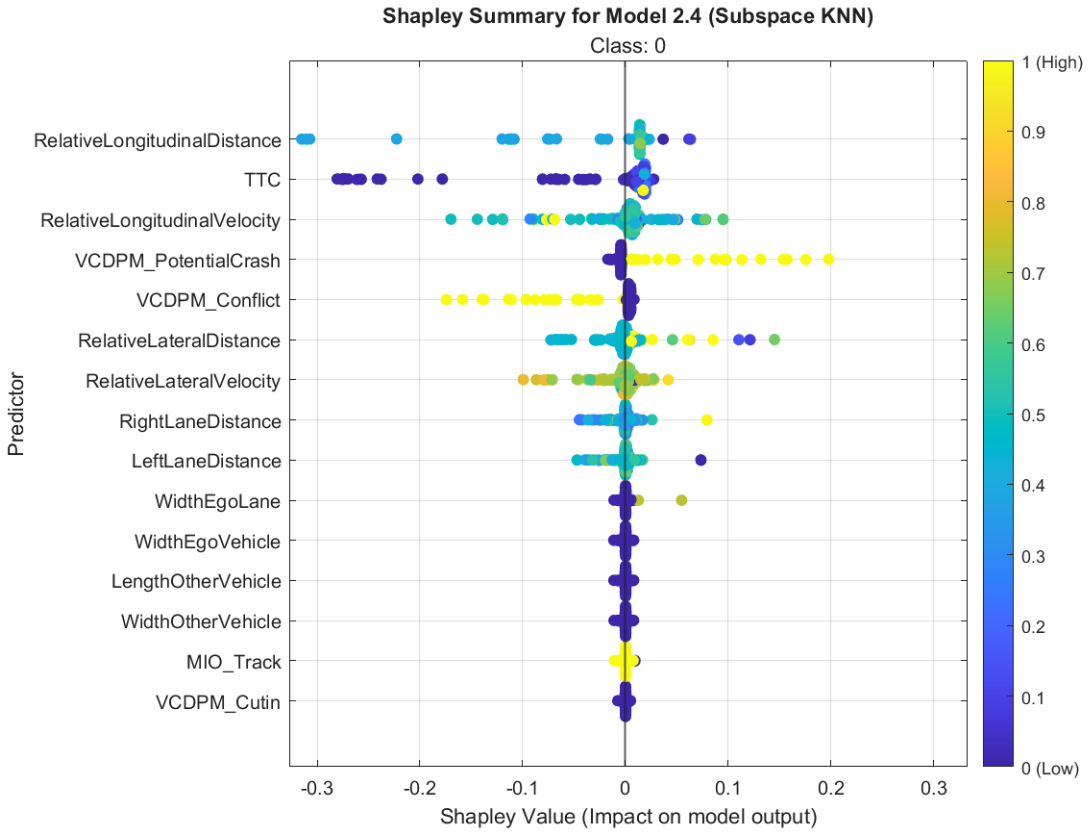


Figure 6.8: The feature importance and impact evaluation using the global Shapley summary of the best-performing ensemble (subspace KNN) model.

Figure 6.9 illustrates the importance of global Shapley using the mean absolute Shapley value for crash prediction. The top ten features are highlighted, with higher

mean absolute Shapley values indicating greater importance for predicting crashes. The non-crash class (0) is represented in blue, while the crash class (1) is shown in orange. The feature with the highest mean absolute Shapley value (0.085) for the crash class is relative longitudinal distance. Following this, the second and third most important features for the crash class are TTC (0.072) and longitudinal velocity (0.034), respectively. The fourth and fifth key features are potential crash and conflict. The remaining important features are presented in descending order of their mean values: relative lateral distance, relative lateral velocity, right lane distance, and left lane distance. The other five features, which have very low mean absolute Shapley values, are grouped together in Figure 6.9.

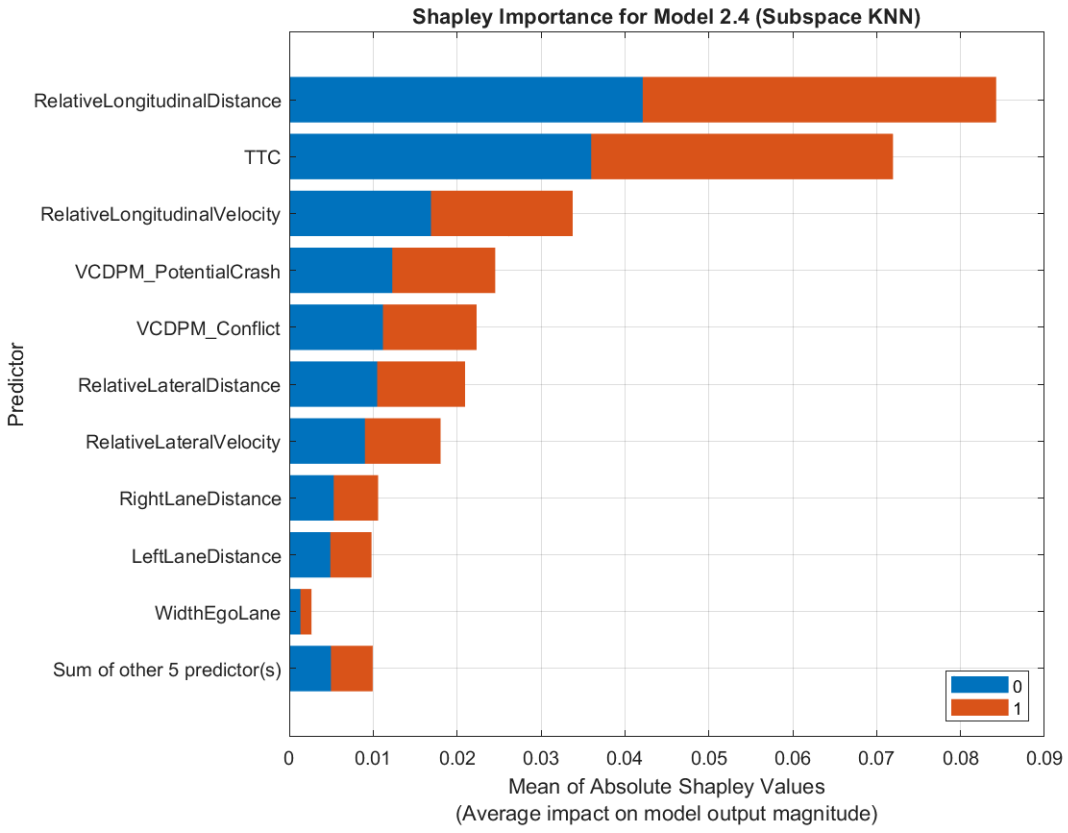
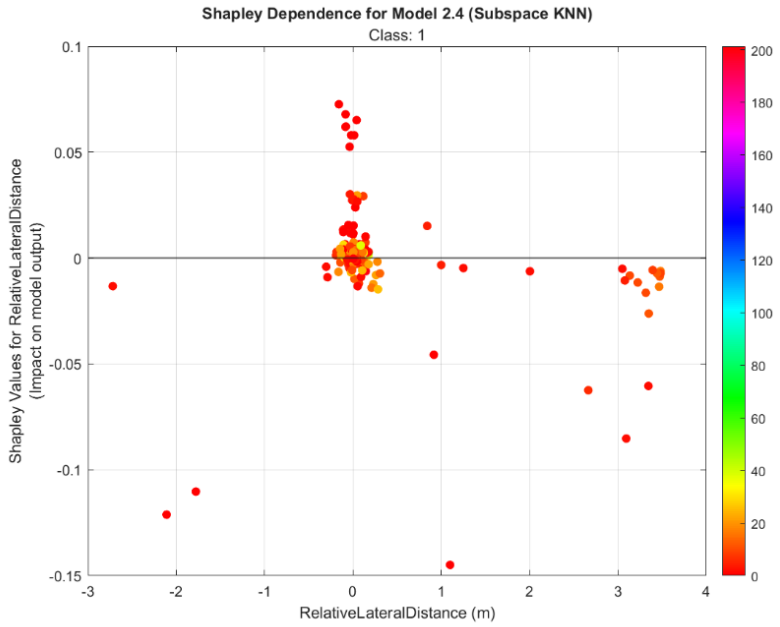
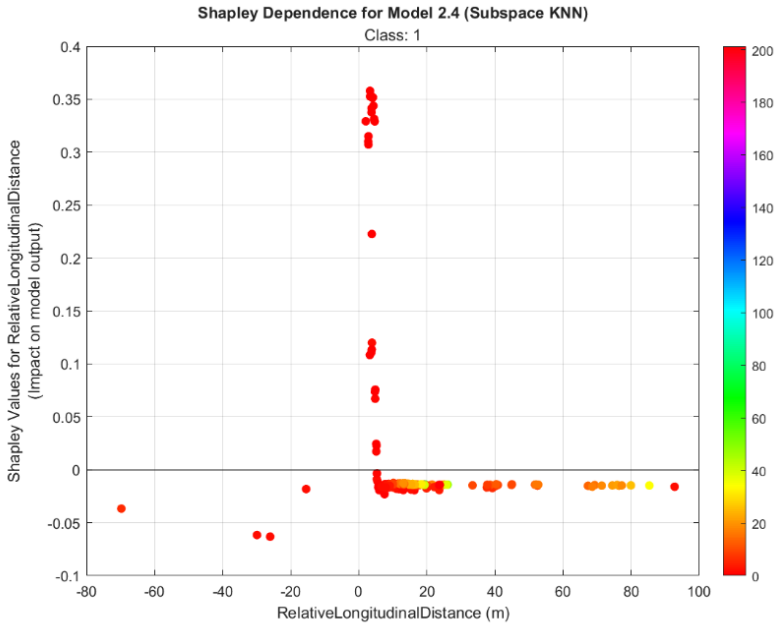


Figure 6.9: The global Shapley importance of the features using the mean of absolute Shapley values of the best-performing Ensemble (subspace KNN) model.

This study examined the relationship between crash occurrence (the target variable) and various crash prediction features to gain a deeper understanding of the factors influencing crash predictions. The analysis of feature dependence highlights how predictor variables affect crash prediction using the variation and distribution of Shapley values (S. Ahmed et al., 2023). Figure 6.10 displays the Shapley dependence values for crash occurrence (class 1) in the subspace KNN model. In Figure 6.10a, the influence of relative lateral distance on crash prediction is shown, while Figure 6.10b illustrates the effect of relative longitudinal distance on the model's output. Both features exhibit higher Shapley values when the relative distances between the ego vehicle and another vehicle approach zero metres, indicating an imminent crash event.



(a)

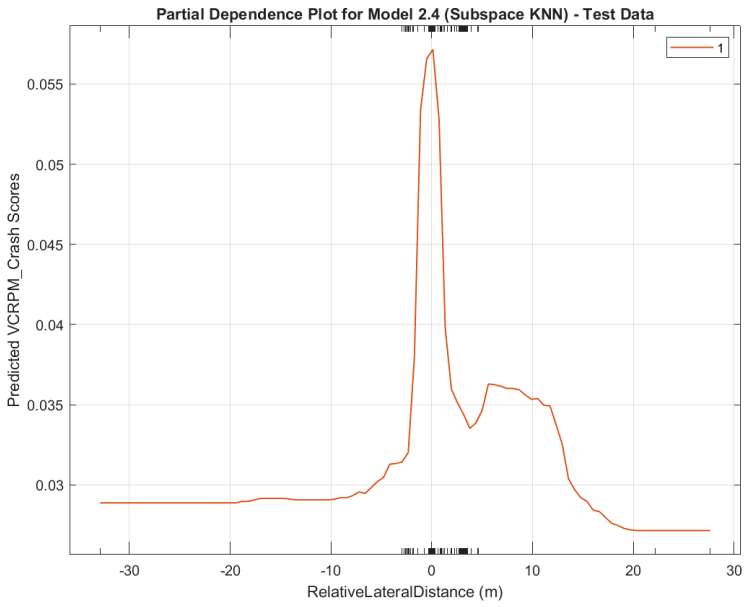


(b)

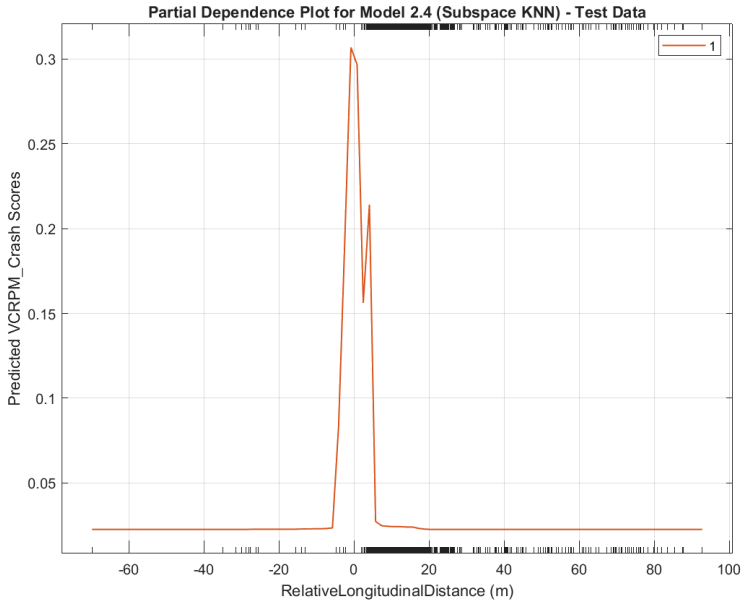
Figure 6.10: Shapley dependence of the crash occurrence (class 1) for subspace KNN model: (a) Shapley values for relative lateral distance; (b) Shapley values for relative longitudinal distance.

For a deep understanding of crash prediction features, this research presents partial

dependence plots in Figure 6.11.



(a)



(b)

Figure 6.11: Partial dependence of the crash occurrence (class 1) for subspace KNN model: (a) predicted crash scores for relative lateral distance; (b) predicted crash scores for relative longitudinal distance.

Figure 6.11a illustrates that the predicted crash scores increase as relative lateral distances decrease. The highest crash prediction score of 0.057 occurred when the relative lateral distance was 0.11m. A similar tendency is observed in the crash prediction scores based on relative longitudinal distance, as shown in Figure 6.11b. This plot reveals a higher crash prediction score near zero relative longitudinal distance compared to Figure 6.11a. The highest crash prediction score based on relative longitudinal distance was 0.30 when the ego vehicle was 0.80m away from another vehicle. This research presents more dependence analysis plots in Appendix D.

6.4 Discussion

This chapter focused on an ML-based approach integrated into the developed VCRPM, enabling real-time crash predictions, and sheds light on RQ3 of this thesis. The experimental results emphasise the accuracy of the model, demonstrating its ability to predict crashes with precision. This chapter also presented a crash prediction dashboard that utilises reconstructed pre-crash event detections from the simulated AV sensor data. While this dashboard can display real-time crash warnings based on pre-crash events, the integration of ML-based crash warnings could further enhance the effectiveness of the proposed model and contribute to improving AV driving safety.

Throughout the iterative development and evaluation of the ML models, this research yielded several findings that could advance AV crash prediction. These findings may assist AV researchers in developing further crash avoidance strategies. The key findings are summarised below.

- The sensors used in AVs can also be applied to generate data for machine learning-based crash prediction. This finding is consistent with similar results in previous studies on crash prediction (Ali et al., 2024; Wang, Xu, et al., 2024).
- This research uncovered a challenge, as it found very few publicly available AV crash datasets generated from AV sensors that could be utilised for real-time AV crash prediction. Therefore, generating publicly available appropriate datasets could contribute to advancing AV-related research. This research also

summarised the limitations of existing publicly available AV crash data/reports (DMV, 2024; NHTSA, 2022c, 2024), which may assist in preparing an appropriate training dataset for ML models focused on real-time AV crash prediction.

- Existing practices for crash occurrence predictions generally rely on using traffic data spanning extensive durations (ranging from 5 to 40 minutes, with 10 minutes being most commonly used) (Islam, Abdel-Aty, Wang, et al., 2024). While this is appropriate for real-time traffic management and road safety planning, a crash event may occur in less than a second or up to a few seconds. Therefore, recording longer durations of driving events before a crash could lead to data imbalances in the crash dataset (with an imbalance between crash and non-crash events), which may result in unreliable crash predictions. Although SMOTE is commonly used to address such imbalances, this method can lead to overfitting issues in the model (Ali et al., 2024). This research experimented with two datasets: one containing both crash and non-crash scenarios and another containing only crash scenarios. The findings suggest that the dataset with only crash scenarios produced more accurate crash occurrence predictions, with a higher true positive crash estimation. However, further exploration is needed to determine the optimal duration for recording sensor data for real-time AV crash prediction, especially considering AV reaction time is typically less than 0.3 seconds to avoid potential hazards (Johnson, 2024).
- This chapter developed a real-time crash warning dashboard, successfully detecting pre-crash events, such as cut-ins, conflicts, and potential crashes. It can warn five seconds before a crash using its crash prediction lamp. This lamp lights up in different colours to indicate varying levels of crash risk: green for low risk, blue and yellow for medium risk, orange for high risk, and red for very high risk. This dashboard can enhance safety for AV driving. This dashboard was developed to deliver potential crash alerts effectively by utilising real-time pre-crash events from simulated AV sensors. While integrating machine learning-based crash predictions with pre-crash event alerts could enhance the

autonomy and efficiency of the system, this aspect falls outside the scope of the current research. Displaying these real-time pre-crash events and signalling crash prediction warnings (Mateen et al., 2022) in an AV system could enhance AV driving safety.

- This chapter evaluated the performance of sixteen ML models using an iterative approach, concluding that cross-validation outperforms the holdout validation approach. This finding reinforces the importance of using the cross-validation approach to evaluate the performance of ML algorithms in making predictions (MathWorks, 2024f). Moreover, the tuning of model parameters can improve real-time crash prediction accuracy (Filion, 2019).
- It is also important to properly split the dataset when training and testing an ML-based model. A single-fold or holdout validation approach splits the dataset randomly, which could introduce a bias toward the class with more observations. Therefore, minimising data imbalance is crucial. Multi-fold cross-validation is the simplest and most effective method to address this issue.
- The Shapley value interpretation is a valuable approach for explaining a model's performance (S. Ahmed et al., 2023) and is particularly beneficial for crash analysis. This Shapley analysis can be integrated with commonly used evaluation metrics (such as accuracy, precision, recall, and F1-score) for ML models. This research applied the global Shapley value to summarise model performance and analyse the features (predictor variables) used for real-time crash predictions. The findings indicate that the Shapley analysis offers a highly effective method for interpreting models and provides another self-explanatory approach to represent crash analysis figures.
- Analysing crash features helps understand the extent to which each predictor (variable) contributes to the model's output and impact on crash prediction. This research revealed that potential crash events and conflict events have an influence on crash prediction. Notably, features such as relative longitudinal distance, relative longitudinal velocity, and TTC play important roles in real-time crash prediction. The dependence analysis based on Shapley values further

provided an in-depth understanding of the contributing features, allowing for the identification and removal of unnecessary features to develop a more effective ML model.

Finally, this research offers the following recommendations for improving real-time crash predictions:

- The dataset should be as balanced as possible to avoid overfitting while maintaining data integrity.
- The driving data duration for machine learning should reflect realistic conditions for real-time crash prediction.
- An optimal fold value for cross-validation should be chosen to maximise prediction accuracy.
- Hyperparameter tuning should be performed to enhance the model's performance.
- Shapley value analysis can be employed to interpret model behaviour and provide insights into crash prediction.

Future Directions

The study focused on V2V crash scenarios for data collection, leaving more complex situations, such as multi-vehicle collisions, curved roads, merging lanes, or hilly terrain, unexplored. Future research could investigate these more intricate driving conditions. This study did not account for adverse weather conditions (e.g., rain or snow), which could influence crash predictions. Assessing the model's performance under different weather scenarios would be a valuable next step. While the research focused on machine learning for real-time crash predictions, integrating deep learning techniques could enhance prediction accuracy, offering another avenue for improvement.

6.5 Conclusions

This research focused on the challenge of selecting effective ML models using appropriate training data for ML to predict crashes in AVs accurately. At the heart of

the study was the introduction of a novel Vehicle Crash Reconstruction and Prediction Model (VCRPM) that strived at real-time crash prediction using sensor data available in AVs. This research enhanced crash predictions through an iterative process within the VCRPM, evaluating multiple ML models to pinpoint the most effective ones for accurate crash prediction. Sixteen supervised crash prediction models based on machine learning were trained on a simulated dataset within the real-time crash prediction subsystem to identify the most effective ML models. This research used the advanced metrics assessment block of the VCRPM, which ensured precise evaluation of crash prediction performance. A key outcome of the study was the development of a crash warning dashboard capable of providing real-time alerts based on detected crash events. Beyond real-time crash prediction, the developed model holds potential for driving safety enhancements, crash warnings, and even crash avoidance in AVs. The main contributions of this paper are outlined as follows:

- The VCRPM enables effective real-time crash prediction and has the potential to enhance the safety of AVs and aid in collision avoidance.
- The real-time crash prediction component of the VCRPM includes a crash warning dashboard that displays real-time crash prediction alerts. It also shows real-time detections of pre-crash and crash events.
- This study showcased a real-time crash prediction method that is adaptable for optimising the performance of ML models and can be easily replicated for real-time crash prediction.
- In this research, three ensemble models (bagged trees, subspace KNN, and RUSBoosted trees) achieved optimal crash prediction results. The crash prediction evaluation method is straightforward and can be reused for future AV advancements.
- Lastly, alongside the findings of this chapter, the brief recommendations offered can be valuable for further studies on real-time crash prediction.

The next chapter concludes the thesis by summarising the research achievements and outlining directions for future research.

Chapter 7 Conclusion and Future Work

This concluding chapter of the thesis summarises the important findings, key contributions, and implications. This chapter also listed future research possibilities.

7.1 Conclusion

This thesis has focused on creating methods for V2V crash reconstruction and prediction using data from AV sensors. Given the vast volume of data produced by AV sensor technology, it becomes clear that a robust sensor architecture is important to enable both precise crash reconstruction and reliable predictive capabilities.

For the first time, this thesis presented a system developed using an evaluated sensor architecture for crash reconstruction and prediction. The system demonstrated its effectiveness by providing examples where AV sensor data were accurately prepared using the proposed sensor architecture. This research presented that the training dataset created from AV sensors is suitable for crash reconstruction and important for enabling effective, machine learning-based, real-time crash prediction. This thesis advances the field by conducting simulated experiments on various V2V crash scenarios, including front, head-on, rear-end, and side-impact collisions. The pivotal contributions of this thesis are clearly articulated in Chapters 3 through 6, with key findings summarised below.

The most important contribution of this thesis is that this research has designed and developed a novel Vehicle Crash Reconstruction and Prediction Model (VCRPM). Chapter 3 presented the system design and essential formulas for the proposed methods and model of this thesis. The subsequent chapters, specifically Chapters 4, 5, and 6, demonstrate the model's evaluation, showcasing its effectiveness in reconstructing and predicting V2V crashes.

In Chapter 4, this research introduced a Simulation Method for Tracking Performance Evaluation (SMTPE) to tackle the challenge of selecting the optimal tracking architecture. The tracking performance evaluation revealed that a centralised tracking architecture, which integrated radar and camera data using sensor setup two, produced the most accurate tracking results. This chapter has outlined the experimental setup and methodology used to ensure repeatability in tracking performance evaluations for AV crash analysis. The experiments were conducted using MATLAB, Simulink,

and related tools. Three tracking architectures based on multi-sensor fusion (which includes radar, camera, and lidar) were developed to reconstruct evidence-based crash scenarios from the CIREN crash dataset. Two centralised tracking architectures and one decentralised architecture were tested, utilising two different sensor fusion arrangements. The performance of these tracking architectures was evaluated using the metrics described in Chapter 3. The results indicated that a centralised radar-camera fusion setup provided the best tracking performance. Key findings included the optimal sensor data size for various driving scenarios, with a sensor update rate of 100 ms, which produced the most efficient data for tracking. The placement of sensors within the vehicle was important for achieving adequate detection coverage. Although cameras exhibited lower object detection rates than radars, sensor fusion enhanced overall detection performance. The guidelines presented in Chapter 4 are valuable for academia and the AV industry, contributing to AV crash analysis and safety advancements.

Chapter 5 of this thesis addressed the challenge of event data recording, emphasising the need for accurate descriptions of the leading causes of V2V accidents for crash analysis. This chapter has introduced a Vehicle Crash Reconstruction Method (VCRM) and used evidence-based simulated AV sensor data for experimental validation. Simulated AV sensor data from multiple crash scenarios were processed through the VCRM's crash events classification subsystem to reconstruct the incidents and generate detailed crash information. The chapter also proposed crash event classification algorithms that detected and categorised cut-ins, conflicts, potential crashes, and actual crash events. The reconstructed crash events were evaluated, and the accuracy of the reconstructions was found to be impressive. This chapter processed crash event descriptions using AV sensor data from hundreds of simulated driving scenarios, creating an evidence-based training dataset. This dataset served as the foundation for real-time crash prediction, as discussed in Chapter 6.

This research has addressed the challenge of selecting suitable training data for ML by auto-classifying pre-crash and crash events using typical AV sensors across various

V2V crash scenarios, with the goal of effectively predicting crashes (Chapter 5). The proposed VCRPM, detailed in Chapter 3. The real-time crash prediction feature of the VCRPM underwent several iterations to evaluate and identify the most effective ML models for this purpose. (Chapter 6). The crash prediction component used a simulated training dataset to train sixteen supervised ML models. The performance of these models was assessed using the metrics described in Chapter 3. Chapter 6 presented a crash warning dashboard that provides real-time alerts based on detected crash events, which could enhance driving safety for AVs. The evaluation of the sixteen ML models revealed that three ensemble models—bagged trees, subspace KNN, and RUSBoosted trees—were effective in producing accurate crash predictions. Beyond real-time crash prediction, the VCRPM could play a vital role in improving AV safety, crash warnings, and crash avoidance strategies. The findings and recommendations presented in Chapter 6 may contribute to further advancements in research in this field.

Overall, this thesis presented functional sensor architectures, collected appropriate sensor data from simulated V2V crashes, and then enhanced crash reconstruction and prediction, paving the way for future AV safety.

7.2 Out of the Scope

This thesis has narrowed its research boundaries by focusing on V2V crash occurrences, as detailed in Section 3.2. The scope included the importance of achieving 360-degree perception with sensors for autonomous driving but excluded the effects of sensor overlap. The algorithm developed for detecting MIOs performs effectively with lane markings, though further research is needed for MIO detection without them. The crash warning dashboard used real-time pre-crash events, and integrating ML-based crash predictions could enhance the dashboard, but it was not included in the scope. This research did not include simulating various driving conditions for crash reconstruction and prediction, which is acknowledged as a limitation.

7.3 Future Work

Crash reconstruction is important in crash forensics, helping to analyse accidents and determine their causes and contributing factors. It helps to clarify the causes and dynamics of crashes, supports legal proceedings, and offers insights for enhancing safety. The development of automated crash forensics techniques, based on the crash reconstruction method presented in this thesis, will contribute to the future of crash forensics.

Crash prediction and warning systems are important for improving road safety and preventing accidents. These systems provide timely alerts about potential hazards that could lead to collisions, allowing ADS to take preventative actions. Such actions may include braking, changing lanes, adjusting speed, or a combination of these measures to avoid crashes. Developing collision avoidance strategies based on the models and methods proposed in this thesis could improve the safety of future AVs. Further research in this area could focus on advancements in 360-degree sensor fusion for AVs, enhancing most important object detection and creating more comprehensive training datasets.

Appendices

Appendix A

MATLAB script for simulating 100 driving scenarios for Chapter 5.

```
% ========================================================================
% Start of the script
% ========================================================================
% Enter number of simulations you want
simNum = 84;
fprintf('The number of total desired simulation is: %d \n', simNum);
% Assign the number of seed scenarios in seedSinCount variable
seedScnCount = 16;
% Calling function to import ego speed seed data from excel file
egoSpeed = setSeedEgoSpeeds('seedEgoParamData.xlsx');
% Calling function to import other vehicle speed seed data from excel file
otherVehicleSpeed = setSeedOVSpeeds('seedOVParamData.xlsx');
% ========================================================================
% MAIN BODY OF THE SIMULATION
% ========================================================================
try
        % Call seedScenarioGenerator local function to simulate 16 base trajectories
        tic; % tic will statr a stop watch to count the time of simulation
        [seedTrajectoriesData,seedTrajectoriesObj,seedTrajectoriesScn]...
        = seedScenarioGenerator(seedScnCount,egoSpeed,otherVehicleSpeed);
        % automatically simulate trajectories using base secenarios
        [autoSimTrajectoriesData,autoSimTrajectoriesObj,autoSimTrajectoriesScn]...
        = autoSimScenarioGenerator(simNum,seedScnCount,egoSpeed,otherVehicleSpeed);
        tEnd = toc; %tEnd specifies the elapsed time since the call to the tic function

fprintf('========================================================================\n');
        fprintf('The number of successful simulations is: %d \n',simNum);
        fprintf('The execution time for the simulations is: %d seconds.\n',tEnd);
        % Adding all trajectories data, secenario objects and Sensor Fusion
         total100TrajectoriesData = horzcat(seedTrajectoriesData,
autoSimTrajectoriesData);
         total100TrajectoriesObj = horzcat(seedTrajectoriesObj, autoSimTrajectoriesObj);
         total100TrajectoriesScn = horzcat(seedTrajectoriesScn, autoSimTrajectoriesScn);
        % Save the seed simulated trajectory dataset in a MATLAB file. Maximium capacity
        % of Version '-v7.3' ≥ 2 GB on 64-bit computers for each variable.
         save('seedSim_16_AllData.mat', "seedTrajectoriesData", '-v7.3');
         save('seedSim_16_AllObjlect.mat', "seedTrajectoriesObj", '-v7.3');
         save('seedSim_16_AllSensor.mat', "seedTrajectoriesScn", '-v7.3');
        % Save the randomly selected simulated trajectory dataset in a MATLAB file.
        % Maximium capacity of Version '-v7.3' ≥ 2 GB on 64-bit computers for each
variable.
          save('AutoSim_84_AllData.mat', "autoSimTrajectoriesData", '-v7.3');
          save('AutoSim_84_AllObjlect.mat', "autoSimTrajectoriesObj", '-v7.3');
          save('AutoSim_84_AllSensor.mat', "autoSimTrajectoriesScn", '-v7.3');

        % Reconstruct a rendomly selected scenario from the simulations
          drivingScenarioDesigner(totalTrajectoriesObj{randi(simNum)});
catch ME
     rethrow(ME)
end
% ========================================================================
% END OF THE MAIN BODY
% ========================================================================
```

```
%%
% ========================================================================
% Define the seedSenarioGenerator function
% ========================================================================
function [seedScenarData,seedScenarObj,seedScenarScn]...
    = seedScenarioGenerator(seedScnCount,egoSpeed,otherVehicleSpeed)
    try
        % Iterate 16 times to genereate 16 base trajectories using 16 scenario functions
        for baseScn = 1:seedScnCount
          switch baseScn
            case 1
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_01_NormEvent_EgoConstantSpeed(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 2
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_02_NormEvent_EgoAccelaration(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 3
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_03_NormEvent_EgoDecelaration(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 4
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_04_AccelEvent_OV(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 5
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_05_DecelEvent_OV(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 6
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_06_RCEvent_Ego_OV_ConstantSpeed(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 7
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_07_OELeft_Ego_OV_ConstantSpeed(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 8
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_08_PRCEvent_Ego_OV_ConstantSpeed(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 9
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_09_LCEvent_EgoConstantSpeed(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 10
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_10_PLCEvent_EgoConstantSpeed(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 11
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_11_OERight_EgoConstantSpeed(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 12
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_12_PRCEvent_Ego_OV_VarSpeed(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 13
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_13_ConflictEvent_Ego_Accelarate(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 14
             [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_14_ConflictEvent_OV_Decelarate(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 15
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_15_PotentialcrashEvent(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            case 16
              [seedScenarData{baseScn}, seedScenarObj{baseScn},seedScenarScn{baseScn}] =
scenario_16_CrashEvent(egoSpeed{baseScn},otherVehicleSpeed{baseScn});
            otherwise
                  fprintf('Not a valid function!');
          end
          fprintf('Automatically simulated base scenario number: %d \n',baseScn);
        end
    catch ME
       rethrow(ME)
    end
%     fprintf('\n Simulated base scenario number is: %d \n', baseScn);
```

```
end
%%
% ========================================================================
% Define the autoSimSenarioGenerator function
% ========================================================================
function [autoSimScenarData, autoSimScenarObj,autoSimScenarScn]......
= autoSimScenarioGenerator(simNum,seedScnCount,egoSpeed,otherVehicleSpeed)
    try
        autoSimCount = simNum - seedScnCount;
        baseScnArray = randi(16,1,autoSimCount);

        for autoScn = 1: autoSimCount
         baseScnSelector = baseScnArray(autoScn);
         switch baseScnSelector
           case 1
             egoSpeed1 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_01_NormEvent_EgoConstantSpeed(egoSpeed1,otherVehicleSpeed{baseScnSelector});
           case 2
             otherVehicleSpeed2 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_02_NormEvent_EgoAccelaration(egoSpeed{baseScnSelector},otherVehicleSpeed2);
           case 3
             otherVehicleSpeed3 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_03_NormEvent_EgoDecelaration(egoSpeed{baseScnSelector},otherVehicleSpeed3);
           case 4
             egoSpeed4 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_04_AccelEvent_OV(egoSpeed4,otherVehicleSpeed{baseScnSelector});
           case 5
             egoSpeed5 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_05_DecelEvent_OV(egoSpeed5,otherVehicleSpeed{baseScnSelector});
           case 6
             egoSpeed6 = round(20 + (30-20)*rand());
             otherVehicleSpeed6 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_06_RCEvent_Ego_OV_ConstantSpeed(egoSpeed6,otherVehicleSpeed6);
           case 7
             egoSpeed7 = round(20 + (30-20)*rand());
             otherVehicleSpeed7 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_07_OELeft_Ego_OV_ConstantSpeed(egoSpeed7,otherVehicleSpeed7);
           case 8
             egoSpeed8 = round(20 + (30-20)*rand());
             otherVehicleSpeed8 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_08_PRCEvent_Ego_OV_ConstantSpeed(egoSpeed8,otherVehicleSpeed8);
           case 9
             egoSpeed9 = round(20 + (30-20)*rand());
             otherVehicleSpeed9 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_09_LCEvent_EgoConstantSpeed(egoSpeed9,otherVehicleSpeed9);
           case 10
             egoSpeed10 = round(20 + (30-20)*rand());
             otherVehicleSpeed10 = round(20 + (30-20)*rand());
```

```
            [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_10_PLCEvent_EgoConstantSpeed(egoSpeed10,otherVehicleSpeed10);
           case 11
             egoSpeed11 = round(20 + (30-20)*rand());
             otherVehicleSpeed11 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_11_OERight_EgoConstantSpeed(egoSpeed11,otherVehicleSpeed11);
           case 12
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_12_PRCEvent_Ego_OV_VarSpeed(egoSpeed{baseScnSelector},otherVehicleSpeed{baseScn
Selector});
           case 13
             otherVehicleSpeed13 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}]. =
scenario_13_ConflictEvent_Ego_Accelarate(egoSpeed{baseScnSelector},otherVehicleSpeed13);
           case 14
             egoSpeed14 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_14_ConflictEvent_OV_Decelarate(egoSpeed14,otherVehicleSpeed{baseScnSelector});
           case 15
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_15_PotentialcrashEvent(egoSpeed{baseScnSelector},otherVehicleSpeed{baseScnSelec
tor});
           case 16
             egoSpeed16 = round(20 + (30-20)*rand());
             otherVehicleSpeed16 = round(20 + (30-20)*rand());
             [autoSimScenarData{autoScn},
autoSimScenarObj{autoScn},autoSimScenarScn{autoScn}] =
scenario_16_CrashEvent(egoSpeed16,otherVehicleSpeed16);
           otherwise
             fprintf('\n Not a valid function! \n');
         end
         fprintf('Automatically simulated scenario number: %d using base scenario: %d
\n',autoScn, baseScnSelector);
       end
    catch ME
       rethrow(ME)
    end
end
%%
% ========================================================================
% End of the script
% ========================================================================
```

Appendix B

This appendix provides detailed information on the machine learning techniques used in this research, facilitating reproducibility by other researchers.

Hyperparameter optimisation is an important process in machine learning, involving the selection of predefined settings—called hyperparameters tuning—that influence model training. Due to the non-convex nature of most machine learning problems, various hyperparameter values can result in different model performances. Traditional tuning methods, such as grid search and random search, are straightforward but computationally intensive. Random search often performs better when only a few hyperparameters have a significant impact on performance. More advanced techniques, such as Bayesian optimisation, build surrogate models to guide the search more efficiently, considering both accuracy and training time (Filion, 2019). Bayesian optimisation aims to find a point that minimises an objective function. In hyperparameter tuning using the Classification Learner app, a point refers to a specific combination of hyperparameter values, and the objective function represents the loss function or classification error (MathWorks, 2025). Hyperparameter optimisation is a computationally expensive process, whereas strategies like feature engineering generally require less computational effort (Filion, 2019). Given this computational advantage, this research emphasises crash-related feature engineering, as detailed in Subsection 6.3.4, and utilises Bayesian optimisation-based default hyperparameter tuning provided by the Classification Learner app for the machine learning techniques presented in Table B.1.

Table B.1: Classification Learner app-based hyperparameter optimisation options and their default values (MathWorks, 2025).

Hyperparameter Optimisation Option	Parameter Value and Description
Optimiser	`Bayesopt` (default) function. This function performs a Bayesian optimisation.
Acquisition function	`Expected improvement per second plus`. When the Classification Learner app performs Bayesian optimisation for hyperparameter tuning, it uses the acquisition function to determine the next set of hyperparameter values to try.
Iterations	The default value is `30`. Each iteration corresponds to a combination of hyperparameter values that the app tries.
Training time limit	By default, the app does not have a training time limit. To set a training time limit, the maximum training time option needs to be selected.
Maximum training time in seconds	The default value is `300`. Set the training time limit in seconds as a positive real number. The runtime can go beyond the training time limit because this limit does not interrupt an iteration evaluation.

This research used a splitting of the dataset into 80% for training and 20% for testing for holdout validation and k-fold cross-validation. The holdout validation was initially used to evaluate the performance of 16 machine learning models presented in Table B.2. The accuracy of the models in predicting crashes was tested using both the full dataset, which included 103 scenarios, and a reduced dataset containing 31 crash cases. This research then compared the crash prediction results of holdout validation from these two datasets and selected the top 10 performing models for further optimisation as presented in Table 6-4. Subsequently, k-fold cross-validation was employed to assess the consistency and variability of model performance across different subsets. The top 10 performing models underwent iterative refinement through hyperparameter tuning, utilising k-fold cross-validation with an increased number of folds—from 2-fold to 6-fold—to enhance their robustness and identify the best ML models for real-time crash prediction.

More detailed information on the default parameters of the machine learning techniques is presented in Table B.2 to ensure reproducibility. The abbreviations used in Table B.2 are defined as follows: Fully Connected Layers Number (FCLN), First Layer Size (FLS), Second Layer Size (SLS), Regularisation Strength (RS), Standardised Data (SD), Learner Type (LT), Number of Learners (NoL), Subspace Dimension (SubDim), Maximum Number of Splits (MNoS), Number of Predictors (NoP), Kernel Function (KF), Kernel Scale (KS), Box Constraint Level (BCL), and Multiclass Coding (MC).

Table B.-2: Detailed information on the default parameters of the machine learning techniques utilised.

Model	Model Hyperparameters (Name: Value)
Narrow NN	FCLN: 1, FLS: 10, Activation: ReLU, Iteration: 1000, RS: 0, SD: Yes
Medium NN	FCLN: 1, FLS: 25, Activation: ReLU, Iteration: 1000, RS: 0, SD: Yes
Wide NN	FCLN: 1, FLS: 100, Activation: ReLU, Iteration: 1000, RS: 0, SD: Yes
Bilayered NN	FCLN: 2, FLS: 10, SLS: 10, Activation: ReLU, Iteration: 1000, RS: 0, SD: Yes
Trilayered NN	FCLN: 3, FLS: 10, SLS: 10, Activation: ReLU, Iteration: 1000, RS: 0, SD: Yes
Ensemble (Subspace KNN)	Method: Subspace, LT: Nearest Neighbours, NoL: 30, SubDim: 8
Ensemble (Subspace Discriminant)	Method: Subspace, LT: Discriminant, NoL: 30, SubDim: 8
Ensemble (Bagged Trees)	Method: Bag, LT: Decision Tree, MNoS: 2932, NoL: 30, NoP: Select All
Ensemble (Boosted Trees)	Method: AdaBoost, LT: Decision Tree, MNoS: 20, NoL: 30, Learning Rate: 0.1 NoP: Select All
Ensemble (RUSBoosted Trees)	Method: RUSBoost, LT: Decision Tree, MNoS: 20, NoL: 30, Learning Rate: 0.1 NoP: Select All
Fine Gaussian SVM	KF: Gaussian, KS: 0.97, BCL: 1, MC: 1-vs-1, SD: Yes
Medium Gaussian SVM	KF: Gaussian, KS: 3.9, BCL: 1, MC: 1-vs-1, SD: Yes
Cubic SVM	KF: Cubic, KS: Autometic, BCL: 1, MC: 1-vs-1, SD: Yes
Coarse Gaussian SVM	KF: Gaussian, KS: 15, BCL: 1, MC: 1-vs-1, SD: Yes
Linear SVM	KF: Linear, KS: Autometic, BCL: 1, MC: 1-vs-1, SD: Yes
Quadratic SVM	KF: Quadratic, KS: Autometic, BCL: 1, MC: 1-vs-1, SD: Yes

Appendix C

Additional model's performance results for Chapter 6.

This appendix presents the performance of the evaluated models using 2-fold (Table C.1), 3-fold (Table C.2), 4-fold (Table C.3), and 6-fold (Table C.4) cross-validation approach for the real-time crash prediction using 31 crash scenarios. The performance results of these tables are not sorted. After 5-fold cross-validation, the performance of the models is downgraded, and this research stopped the performance optimisation at 6-fold cross-validation.

Table C.1: Performance of the evaluated models (2-fold cross-validation approach) for the real-time crash prediction using 31 crash scenarios.

Model	Accuracy	Precision	Recall	F1-Score
Ensemble (Bagged Trees)	99.86	99.87	99.86	99.86
Ensemble (Subspace KNN)	99.45	99.47	99.45	99.46
Ensemble (RUSBoosted Trees)	99.73	99.73	99.73	99.73
Narrow NN	99.05	99.06	99.05	99.05
Medium NN	99.45	99.45	99.45	99.45
Fine Gaussian SVM	99.32	99.33	99.32	99.32
Wide NN	99.45	99.47	99.45	99.46
Bilayered NN	99.45	99.45	99.45	99.45
Trilayered NN	99.18	99.18	99.18	99.18
Quadratic SVM	97.68	97.64	97.68	97.42

Table C.2: Performance of the evaluated models (3-fold cross-validation approach) for the real-time crash prediction using 31 crash scenarios.

Model	Accuracy	Precision	Recall	F1-Score
Ensemble (Bagged Trees)	99.73	99.73	99.73	99.73
Ensemble (Subspace KNN)	100	100	100	100
Ensemble (RUSBoosted Trees)	99.45	99.51	99.45	99.47
Narrow NN	99.86	99.87	99.86	99.86
Medium NN	99.73	99.73	99.73	99.73
Fine Gaussian SVM	99.45	99.45	99.45	99.45
Wide NN	99.73	99.73	99.73	99.73
Bilayered NN	99.86	99.87	99.86	99.86
Trilayered NN	99.86	99.87	99.86	99.86
Quadratic SVM	99.59	99.62	99.50	99.60

Table C.3: Performance of the evaluated models (4-fold cross-validation approach) for the real-time crash prediction using 31 crash scenarios.

Model	Accuracy	Precision	Recall	F1-Score
Ensemble (Bagged Trees)	100	100	100	100
Ensemble (Subspace KNN)	100	100	100	100
Ensemble (RUSBoosted Trees)	99.59	99.62	99.59	99.60
Narrow NN	99.05	99.03	99.05	99.03
Medium NN	99.86	99.86	99.86	99.86
Fine Gaussian SVM	98.64	98.59	98.64	98.60
Wide NN	99.59	99.59	99.59	99.58
Bilayered NN	99.45	99.45	99.45	99.45
Trilayered NN	99.59	99.59	99.59	99.58
Quadratic SVM	96.45	96.07	96.45	96.15

Table C.4: Performance of the evaluated models (6-fold cross-validation approach) for the real-time crash prediction using 31 crash scenarios.

Model	Accuracy	Precision	Recall	F1-Score
Ensemble (Bagged Trees)	99.32	99.31	99.32	99.30
Ensemble (Subspace KNN)	99.45	99.45	99.45	99.45
Ensemble (RUSBoosted Trees)	99.73	99.74	99.73	99.73
Narrow NN	99.45	99.45	99.45	99.45
Medium NN	99.05	99.03	99.05	99.01
Fine Gaussian SVM	99.32	99.31	99.32	99.30
Wide NN	99.18	99.17	99.18	99.16
Bilayered NN	99.59	99.59	99.59	99.58
Trilayered NN	99.59	99.59	99.59	99.59
Quadratic SVM	98.09	98.13	98.09	97.89

Appendix D

Additional self-explanatory ML model results for Chapter 6.

The self-explanatory ML model's (subspace KNN) plots are presented in this appendix. These plots present the impact of the model results based on the features used for crash prediction. The impact on model results from conflict and potential crash events are presented in Figure D.1 and Figure D.2. The other two plots present relative longitudinal velocity and TTC impact on the model's output, as shown in Figure D.3 and Figure D.4, respectively.

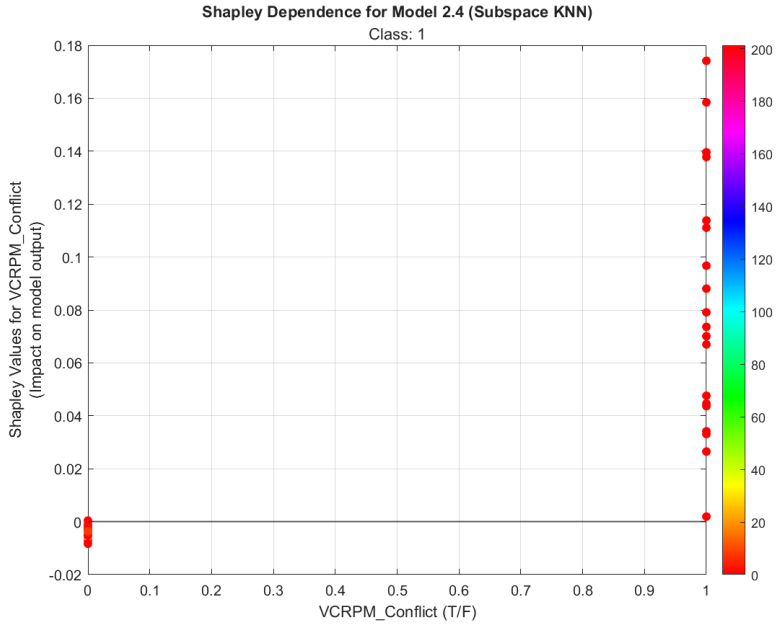


Figure D.1: Shapley dependence of the crash occurrence (class 1) for subspace KNN model using conflict events between the ego vehicle and other vehicle.

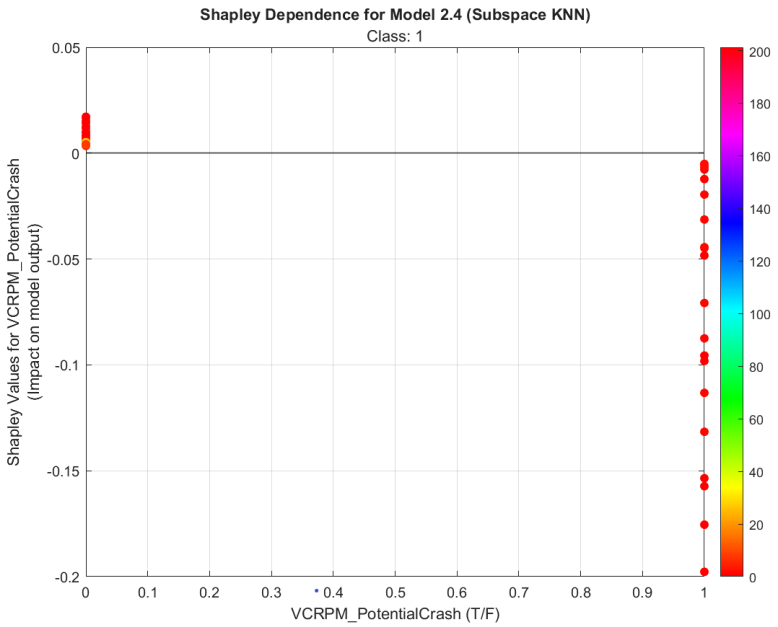


Figure D.2: Shapley dependence of the crash occurrence (class 1) for subspace KNN model using potential crash events between the ego vehicle and other vehicle.

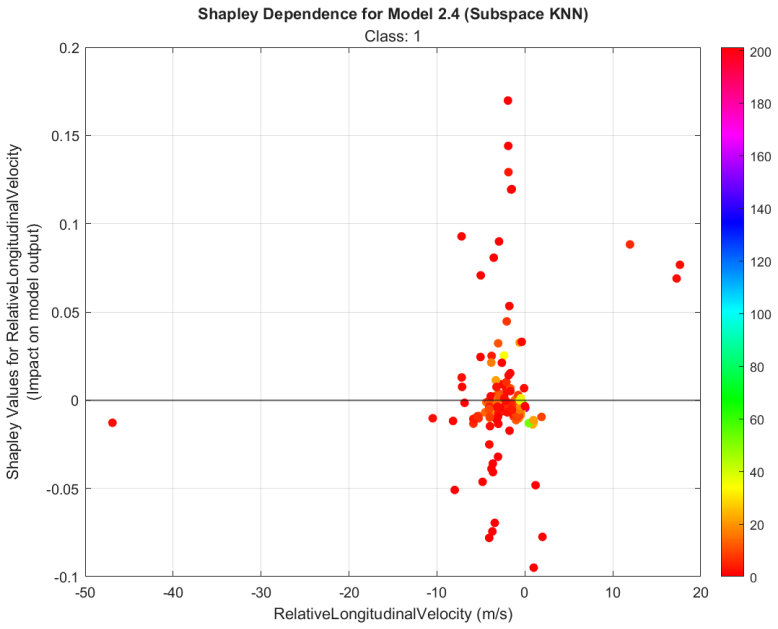


Figure D.3: Shapley dependence of the crash occurrence (class 1) for subspace KNN model using relative longitudinal velocity between the ego vehicle and other vehicle.

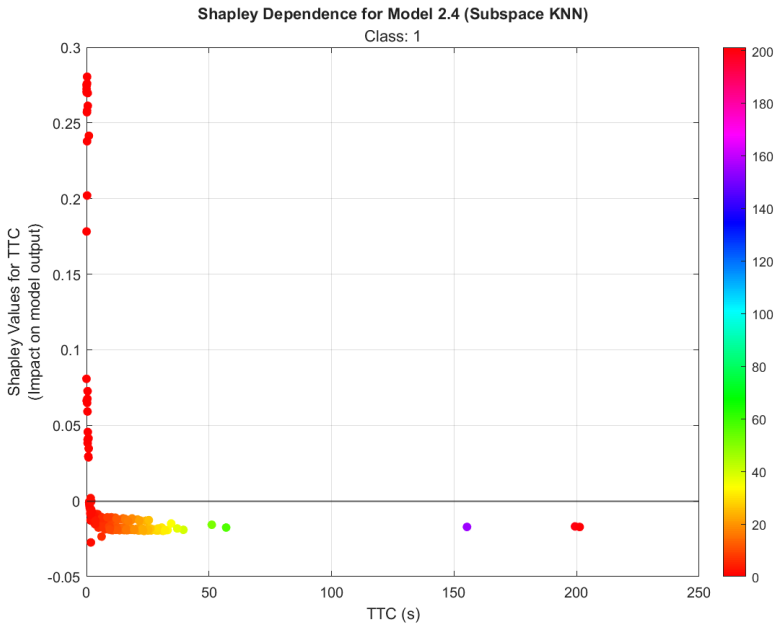


Figure D.4: Shapley dependence of the crash occurrence (class 1) for subspace KNN model using TTC values.

References

Abdel-Aty, M., & Ding, S. (2024). A matched case-control analysis of autonomous vs human-driven vehicle accidents. *Nature Communications*, *15*(1), 4931. https://doi.org/10.1038/s41467-024-48526-4

Abdel-Aty, M., Hasan, T., & Anik, B. M. T. H. (2024). An advanced real-time crash prediction framework for combined hard shoulder running and variable speed limits system using transformer. *Scientific Reports*, *14*(1), 26403. https://doi.org/10.1038/s41598-024-75350-z

Ahmad, N. S. (2020). Robust $\mathscr{H}\infty$-Fuzzy Logic Control for Enhanced Tracking Performance of a Wheeled Mobile Robot in the Presence of Uncertain Nonlinear Perturbations. *Sensors*, *20*(13).

Ahmed, M. I. B., Zaghdoud, R., Ahmed, M. S., Sendi, R., Alsharif, S., Alabdulkarim, J., Saad, B. A. A., Alsabt, R., Rahman, A., & Krishnasamy, G. (2023). A Real-Time Computer Vision Based Approach to Detection and Classification of Traffic Incidents. *Big Data and Cognitive Computing*, *7*(1).

Ahmed, S., Hossain, M. A., Ray, S. K., Bhuiyan, M. M. I., & Sabuj, S. R. (2023). A study on road accident prediction and contributing factors using explainable machine learning models: analysis and performance. *Transportation Research Interdisciplinary Perspectives*, *19*, 100814. https://doi.org/https://doi.org/10.1016/j.trip.2023.100814

Alai, H., & Rajamani, R. (2024). Low-cost camera and 2-D LIDAR fusion for target vehicle corner detection and tracking: Applications to micromobility devices. *Mechanical Systems and Signal Processing*, *206*, 110891. https://doi.org/https://doi.org/10.1016/j.ymssp.2023.110891

Alexakos, C., Katsini, C., Votis, K., Lalas, A., Tzovaras, D., & Serpanos, D. (2021). Enabling Digital Forensics Readiness for Internet of Vehicles. *Transportation Research Procedia*, *52*, 339-346. https://doi.org/https://doi.org/10.1016/j.trpro.2021.01.040

Alghodhaifi, H., & Lakshmanan, S. (2021). Autonomous Vehicle Evaluation: A Comprehensive Survey on Modeling and Simulation Approaches. *IEEE Access*, *9*, 151531-151566. https://doi.org/10.1109/ACCESS.2021.3125620

Ali, Y., Hussain, F., & Haque, M. M. (2024). Advances, challenges, and future research needs in machine learning-based crash prediction models: A systematic review. *Accident Analysis & Prevention*, *194*, 107378. https://doi.org/https://doi.org/10.1016/j.aap.2023.107378

Alim, M., Das, T., Yadav, V. K., Dev, K., Srivastava, A., & Nigam, K. (2023). Road Accident Investigation. In P. Shrivastava, J. A. Lorente, A. Srivastava, A. Badiye, & N. Kapoor (Eds.), *Textbook of Forensic Science* (pp. 469-507). Springer Nature Singapore. https://doi.org/10.1007/978-981-99-1377-0_16

Alrejjal, A., Farid, A., & Ksaibati, K. (2022). Investigating factors influencing rollover crash risk on mountainous interstates. *Journal of Safety Research*, *80*, 391-398. https://doi.org/https://doi.org/10.1016/j.jsr.2021.12.020

Anis, M., Li, S., Geedipally, S. R., Zhou, Y., & Lord, D. (2024). Real-time risk estimation for active road safety: Leveraging Waymo AV sensor data with hierarchical Bayesian extreme value models. *arXiv e-prints*, arXiv:2407.16832. https://doi.org/10.48550/arXiv.2407.16832

Arcaini, P., Zhang, X. Y., & Ishikawa, F. (2022, 4-14 April 2022). Less is More: Simplification of Test Scenarios for Autonomous Driving System Testing. 2022 IEEE Conference on Software Testing, Verification and Validation (ICST),

Arnold, E., Al-Jarrah, O. Y., Dianati, M., Fallah, S., Oxtoby, D., & Mouzakitis, A. (2019). A Survey on 3D Object Detection Methods for Autonomous Driving Applications. *IEEE Transactions on Intelligent Transportation Systems*, *20*(10), 3782-3795. https://doi.org/10.1109/TITS.2019.2892405

Ashraf, M. T., Dey, K., Mishra, S., & Rahman, M. T. (2021). Extracting Rules from Autonomous-Vehicle-Involved Crashes by Applying Decision Tree and Association Rule Methods. *Transportation Research Record*, *2675*(11), 522-533. https://doi.org/10.1177/03611981211018461

Basso, F., Pezoa, R., Varas, M., & Villalobos, M. (2021). A deep learning approach for real-time crash prediction using vehicle-by-vehicle data. *Accident Analysis & Prevention*, *162*, 106409. https://doi.org/https://doi.org/10.1016/j.aap.2021.106409

Beck, J., Arvin, R., Lee, S., Khattak, A., & Chakraborty, S. (2023). Automated vehicle data pipeline for accident reconstruction: New insights from LiDAR, camera, and radar data. *Accident Analysis & Prevention*, *180*, 106923. https://doi.org/https://doi.org/10.1016/j.aap.2022.106923

Behboudi, N., Moosavi, S., & Ramnath, R. (2024). Recent Advances in Traffic Accident Analysis and Prediction: A Comprehensive Review of Machine Learning Techniques. *arXiv e-prints*, arXiv:2406.13968. https://doi.org/10.48550/arXiv.2406.13968

Berhanu, Y., Schröder, D., Wodajo, B. T., & Alemayehu, E. (2024). Machine learning for predictions of road traffic accidents and spatial network analysis for safe routing on accident and congestion-prone road networks. *Results in*

Engineering, *23*, 102737. https://doi.org/https://doi.org/10.1016/j.rineng.2024.102737

Brach, M., Mason, J., & Brach, R. M. (2022). *Vehicle accident analysis and reconstruction methods*. Sae international.

Brummelen, J. V., O'Brien, M., Gruyer, D., & Najjaran, H. (2018). Autonomous vehicle perception: The technology of today and tomorrow. *Transportation Research Part C: Emerging Technologies*, *89*, 384-406. https://doi.org/https://doi.org/10.1016/j.trc.2018.02.012

Butt, F. A., Chattha, J. N., Ahmad, J., Zia, M. U., Rizwan, M., & Naqvi, I. H. (2022). On the Integration of Enabling Wireless Technologies and Sensor Fusion for Next-Generation Connected and Autonomous Vehicles. *IEEE Access*, *10*, 14643-14668. https://doi.org/10.1109/ACCESS.2022.3145972

Caesar, H., Kabzan, J., Tan, K. S., Fong, W. K., Wolff, E., Lang, A., Fletcher, L., Beijbom, O., & Omari, S. (2021). NuPlan: A closed-loop ML-based planning benchmark for autonomous vehicles. *arXiv e-prints*, arXiv:2106.11810. https://doi.org/10.48550/arXiv.2106.11810

Calvi, A., D'Amico, F., Ferrante, C., & Ciampoli, L. B. (2020). A driving simulator study to assess driver performance during a car-following maneuver after switching from automated control to manual control. *Transportation Research Part F: Traffic Psychology and Behaviour*, *70*, 58-67. https://doi.org/https://doi.org/10.1016/j.trf.2020.02.014

Calvi, A., D'Amico, F., Ciampoli, L. B., & Ferrante, C. (2020). Evaluation of driving performance after a transition from automated to manual control: a driving simulator study. *Transportation Research Procedia*, *45*, 755-762. https://doi.org/https://doi.org/10.1016/j.trpro.2020.02.101

Campbell, S., O'Mahony, N., Krpalcova, L., Riordan, D., Walsh, J., Murphy, A., & Ryan, C. (2018, 21-22 June 2018). Sensor Technology in Autonomous Vehicles : A review. 2018 29th Irish Signals and Systems Conference (ISSC),

CDV. (2024). *Autonomous Vehicle Crashes*. https://www.avcrashes.net/

Čehovin, L., Leonardis, A., & Kristan, M. (2016). Visual Object Tracking Performance Measures Revisited. *IEEE Transactions on Image Processing*, *25*(3), 1261-1274. https://doi.org/10.1109/TIP.2016.2520370

Chavez-Garcia, R. O., & Aycard, O. (2016). Multiple Sensor Fusion and Classification for Moving Object Detection and Tracking. *IEEE Transactions on Intelligent Transportation Systems*, *17*(2), 525-534. https://doi.org/10.1109/TITS.2015.2479925

Chen, Y., Zou, Y., Kong, X., & Wu, L. (2024). Investigating the impact of influential factors on crash types for autonomous vehicles at intersections. *Journal of Transportation Safety & Security*, *16*(10), 1089-1116. https://doi.org/10.1080/19439962.2023.2289403

Cheng, H. (2011). *Autonomous intelligent vehicles: theory, algorithms, and implementation*. Springer Science & Business Media.

Choi, Y., Lee, W., Kim, J., & Yoo, J. (2021). A Variable-Sampling Time Model Predictive Control Algorithm for Improving Path-Tracking Performance of a Vehicle. *Sensors*, *21*(20).

CRASH, V. (2024). *Virtual CRASH Accident Reconstruction Software*. Virtual CRASH, LLC. Retrieved 15 March 2023 from https://www.vcrashusa.com/home

Čulík, K., Kalašová, A., & Štefancová, V. (2022). Evaluation of Driver's Reaction Time Measured in Driving Simulator. *Sensors*, *22*(9).

DMV. (2024). *Autonomous Vehicle Collision Reports* Department of Motor Vehicles (DMV), CA, USA Retrieved from www.dmv.ca.gov/portal/vehicle-industry-services/autonomous-vehicles/autonomous-vehicle-collision-reports

Domova, V., Currano, R. M., & Sirkin, D. (2024). Comfort in Automated Driving: A Literature Survey and a High-Level Integrative Framework. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, *8*(3), Article 98. https://doi.org/10.1145/3678583

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., & Koltun, V. (2017). *CARLA: An Open Urban Driving Simulator* Proceedings of the 1st Annual Conference on Robot Learning, Proceedings of Machine Learning Research. https://proceedings.mlr.press/v78/dosovitskiy17a.html

Elassad, Z. E. A., Ameksa, M., Elassad, D. E. A., & Mousannif, H. (2024). Efficient Fusion Decision System for Predicting Road Crash Events: A Comparative Simulator Study for Imbalance Class Handling. *Transportation Research Record*, *2678*(5), 789-811. https://doi.org/10.1177/03611981231192985

Elassad, Z. E. A., Mousannif, H., & Al Moatassime, H. (2020). Class-imbalanced crash prediction based on real-time traffic and weather data: A driving simulator study. *Traffic Injury Prevention*, *21*(3), 201-208. https://doi.org/10.1080/15389588.2020.1723794

FARO. (2024). *Crash Reconstruction*. FARO. Retrieved 20 September from https://www.faro.com/en/Application/Forensic-Analysis-and-Pre-incident-Planning/Crash-Reconstruction

Feng, X., Dawam, E. S., & Li, D. (2019, 19-23 Aug. 2019). Autonomous Vehicles' Forensics in Smart Cities. 2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI),

Filion, A. (2019). *Hyperparameter Optimization | Applied Machine Learning, Part 3*. MathWorks. Retrieved 12 November 2024 from https://www.mathworks.com/videos/applied-machine-learning-part-3-hyperparameter-optimization-1547849445386.html

García-Fernández, Á. F., Rahmathullah, A. S., & Svensson, L. (2020). A Metric on the Space of Finite Sets of Trajectories for Evaluation of Multi-Target Tracking Algorithms. *IEEE Transactions on Signal Processing*, *68*, 3917-3928. https://doi.org/10.1109/TSP.2020.3005309

Ghafarian, M., Watson, M., Mohajer, N., Nahavandi, D., Kebria, P. M., & Mohamed, S. (2023). A Review of Dynamic Vehicular Motion Simulators: Systems and Algorithms. *IEEE Access*, *11*, 36331-36348. https://doi.org/10.1109/ACCESS.2023.3265999

Ghazali, M., Gupta, I., Abdallah, M. B., Clarke, J., Indragandhi, V., & Hartavi, A. E. (2023). Performance Comparison of Three Rival AI-Powered Intelligent Trajectory Tracking Controllers for an Autonomous Delivery Van. *Transportation Research Procedia*, *72*, 3039-3045. https://doi.org/https://doi.org/10.1016/j.trpro.2023.11.852

Guo, F., & Li, D. (2019). Adaptive Sliding Mode Control of Vehicular Platoons With Prescribed Tracking Performance. *IEEE Transactions on Vehicular Technology*, *68*(8), 7511-7520. https://doi.org/10.1109/TVT.2019.2921816

Haque, M. M. (2024). *VCRPM Crash Dataset* Version V1). https://github.com/MahfuzRnD/Crash_Dataset_MMH_103Scn_v1

Haque, M. M., Ghobakhlou, A., & Narayanan, A. (2024). Multi-Tracking Sensor Architectures for Reconstructing Autonomous Vehicle Crashes: An Exploratory Study. *Sensors*, *24*(13).

Hasanujjaman, M., Chowdhury, M. Z., & Jang, Y. M. (2023). Sensor Fusion in Autonomous Vehicle with Traffic Surveillance Camera System: Detection, Localization, and AI Networking. *Sensors*, *23*(6).

Hossain, M., Abdel-Aty, M., Quddus, M. A., Muromachi, Y., & Sadeek, S. N. (2019). Real-time crash prediction models: State-of-the-art, design pathways and ubiquitous requirements. *Accident Analysis & Prevention*, *124*, 66-84. https://doi.org/https://doi.org/10.1016/j.aap.2018.12.022

Hou, H., Shen, C., Zhang, X., & Gao, W. (2023). CSMOT: Make One-Shot Multi-Object Tracking in Crowded Scenes Great Again. *Sensors*, *23*(7).

Houston, J., Zuidhof, G., Bergamini, L., Ye, Y., Chen, L., Jain, A., Omari, S., Iglovikov, V., & Ondruska, P. (2020). One Thousand and One Hours: Self-driving Motion Prediction Dataset. *arXiv e-prints*, arXiv:2006.14480. https://doi.org/10.48550/arXiv.2006.14480

Huang, M., Zhang, X. S., Bhatti, U. A., Wu, Y., Zhang, Y., & Yasin Ghadi, Y. (2024). An interpretable approach using hybrid graph networks and explainable AI for intelligent diagnosis recommendations in chronic disease care. *Biomedical Signal Processing and Control*, *91*, 105913. https://doi.org/https://doi.org/10.1016/j.bspc.2023.105913

Huang, T., Wang, S., & Sharma, A. (2020). Highway crash detection and risk estimation using deep learning. *Accident Analysis & Prevention*, *135*, 105392. https://doi.org/https://doi.org/10.1016/j.aap.2019.105392

Huang, Y., Zhang, S., Li, X., Lu, R., & Sun, Q. (2022, 17-19 June 2022). Object Tracking Performance Evaluation Method Based on Adaptive Threshold and Background Suppression. 2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC),

Hussain, F., Ali, Y., Li, Y., & Haque, M. M. (2023). Real-time crash risk forecasting using Artificial-Intelligence based video analytics: A unified framework of generalised extreme value theory and autoregressive integrated moving average model. *Analytic Methods in Accident Research*, *40*, 100302. https://doi.org/https://doi.org/10.1016/j.amar.2023.100302

Ignatious, H. A., Hesham, E.-S., & Khan, M. A. (2023). Sensor Technology for Autonomous Vehicles. In R. Narayan (Ed.), *Encyclopedia of Sensors and Biosensors (First Edition)* (pp. 35-51). Elsevier. https://doi.org/https://doi.org/10.1016/B978-0-12-822548-6.00122-9

Ignatious, H. A., Sayed, H.-E., & Khan, M. (2022). An overview of sensors in Autonomous Vehicles. *Procedia Computer Science*, *198*, 736-741. https://doi.org/https://doi.org/10.1016/j.procs.2021.12.315

Imprialou, M., & Quddus, M. (2019). Crash data quality for road safety research: Current state and future directions. *Accident Analysis & Prevention*, *130*, 84-90. https://doi.org/https://doi.org/10.1016/j.aap.2017.02.022

Islam, M. R., Abdel-Aty, M., Islam, Z., & Abdelraouf, A. (2024). Real-Time Framework to Predict Crash Likelihood and Cluster Crash Severity. *Transportation Research Record*, *2678*(1), 202-217. https://doi.org/10.1177/03611981231170623

Islam, M. R., Abdel-Aty, M., Wang, D., & Islam, Z. (2024). Spatial Ensemble Distillation Learning for Large-Scale Real-Time Crash Prediction. *IEEE Transactions on Intelligent Transportation Systems*, 1-16. https://doi.org/10.1109/TITS.2024.3432854

Jacob, B., & Violette, E. (2012). Vehicle Trajectory Analysis: An Advanced Tool for Road Safety. *Procedia - Social and Behavioral Sciences*, *48*, 1805-1814. https://doi.org/https://doi.org/10.1016/j.sbspro.2012.06.1155

Johnson, J. (2024). *Tesla's FSD Displays Superhuman Reaction Time, Able To Avoid A Car Moving Into Its Lane*. Torque News. Retrieved 30 September 2024 from https://www.torquenews.com/14335/teslas-fsd-displays-superhuman-reaction-time-able-avoid-car-moving-its-lane#:~:text=The%20system's%20ability%20to%20react,and%20react%20to%20potential%20hazards.

Kamnik, R., Perc, M. N., & Topolšek, D. (2020). Using the scanners and drone for comparison of point cloud accuracy at traffic accident analysis. *Accident Analysis & Prevention*, *135*, 105391. https://doi.org/https://doi.org/10.1016/j.aap.2019.105391

Kang, D., & Kum, D. (2020). Camera and Radar Sensor Fusion for Robust Vehicle Localization via Vehicle Part Localization. *IEEE Access*, *8*, 75223-75236. https://doi.org/10.1109/ACCESS.2020.2985075

Katarína, M., Gustáv, K., & Ján, P. (2021). Usage of Digital Evidence in the Technical Analysis of Traffic Collisions. *Transportation Research Procedia*, *55*, 1737-1744. https://doi.org/https://doi.org/10.1016/j.trpro.2021.07.166

Kaur, P., Taghavi, S., Tian, Z., & Shi, W. (2021, 28-29 April 2021). A Survey on Simulators for Testing Self-Driving Cars. 2021 Fourth International Conference on Connected and Autonomous Driving (MetroCAD),

Ke, R., Cui, Z., Chen, J., Zhu, M., Yang, C., Zhuang, Y., & Wang, B. (2023). Lightweight Edge Intelligence Empowered Near-Crash Detection Towards Real-Time Vehicle Event Logging. *IEEE Transactions on Intelligent Vehicles*, *8*(4), 2737-2747. https://doi.org/10.1109/TIV.2023.3241934

Khader, M., & Cherian, S. (2023). *An Introduction to Automotive Lidar*. Texas Instruments. Retrieved 11 December 2024 from https://www.ti.com/lit/pdf/slyy150

Khan, M. A., Sayed, H. E., Malik, S., Zia, T., Khan, J., Alkaabi, N., & Ignatious, H. (2022). Level-5 Autonomous Driving—Are We There Yet? A Review of Research Literature. *ACM Comput. Surv.*, *55*(2), Article 27. https://doi.org/10.1145/3485767

Kolla, E., Adamová, V., & Vertaľ, P. (2022). Simulation-based reconstruction of traffic incidents from moving vehicle mono-camera. *Science & Justice*, *62*(1), 94-109. https://doi.org/https://doi.org/10.1016/j.scijus.2021.11.001

Kothari, C. R. (2004). *Research Methodology: Methods and Techniques*. New Age International (P) Limited. https://books.google.co.nz/books?id=hZ9wSHysQDYC

Krishnendhu, S. P., & Mohandas, P. (2023). SAD: Sensor-Based Anomaly Detection System for Smart Junctions. *IEEE Sensors Journal*, *23*(17), 20368-20378. https://doi.org/10.1109/JSEN.2023.3297205

Kumar, N., Acharya, D., & Lohani, D. (2021). An IoT-Based Vehicle Accident Detection and Classification System Using Sensor Fusion. *IEEE Internet of Things Journal*, *8*(2), 869-880. https://doi.org/10.1109/JIOT.2020.3008896

Kuo, P.-F., Hsu, W.-T., Lord, D., & Putra, I. G. B. (2024). Classification of autonomous vehicle crash severity: Solving the problems of imbalanced datasets and small sample size. *Accident Analysis & Prevention*, *205*, 107666. https://doi.org/https://doi.org/10.1016/j.aap.2024.107666

Law, P. D. (2024). *What Is An Accident Reconstructionist?* Retrieved 15 October 2024 from https://www.patrickdaniellaw.com/blog/accident-reconstructionist/

Lee, Y.-G., Oh, J.-Y., Kim, D., & Kim, G. (2023). SHAP Value-Based Feature Importance Analysis for Short-Term Load Forecasting. *Journal of Electrical Engineering & Technology*, *18*(1), 579-588. https://doi.org/10.1007/s42835-022-01161-9

Lei, T., Peng, J., Liu, X., & Luo, Q. (2021). Crash prediction on expressway incorporating traffic flow continuity parameters based on machine learning approach. *Journal of Advanced Transportation*, *2021*(1), 8820402.

Lengyel, H., Maral, S., Kerebekov, S., Szalay, Z., & Török, Á. (2023). Modelling and Simulating Automated Vehicular Functions in Critical Situations—Application of a Novel Accident Reconstruction Concept. *Vehicles*, *5*(1), 266-285.

Li, G., Yang, Y., Zhang, T., Qu, X., Cao, D., Cheng, B., & Li, K. (2021). Risk assessment based collision avoidance decision-making for autonomous vehicles in multi-scenarios. *Transportation Research Part C: Emerging Technologies*, *122*, 102820. https://doi.org/https://doi.org/10.1016/j.trc.2020.102820

Li, Q., Queralta, J. P., Gia, T. N., Zou, Z., & Westerlund, T. (2020). Multi-Sensor Fusion for Navigation and Mapping in Autonomous Vehicles: Accurate Localization in Urban Environments. *Unmanned Systems*, *08*(03), 229-237. https://doi.org/10.1142/S2301385020500168

Li, Y., Yuan, W., Zhang, S., Yan, W., Shen, Q., & Wang, C. (2024). Choose Your Simulator Wisely: A Review on Open-Source Simulators for Autonomous Driving. *IEEE Transactions on Intelligent Vehicles*, *9*(5), 4861-4876. https://doi.org/10.1109/TIV.2024.3374044

Liu, P., Guo, Y., Liu, P., Ding, H., Cao, J., Zhou, J., & Feng, Z. (2024). What can we learn from the AV crashes? – An association rule analysis for identifying the contributing risky factors. *Accident Analysis & Prevention*, *199*, 107492. https://doi.org/https://doi.org/10.1016/j.aap.2024.107492

Liu, Q., Wang, X., Liu, S., Yu, C., & Glaser, Y. (2024). Analysis of pre-crash scenarios and contributing factors for autonomous vehicle crashes at intersections. *Accident Analysis & Prevention*, *195*, 107383. https://doi.org/https://doi.org/10.1016/j.aap.2023.107383

Liu, Q., Wang, X., Wu, X., Glaser, Y., & He, L. (2021). Crash comparison of autonomous and conventional vehicles using pre-crash scenario typology. *Accident Analysis & Prevention*, *159*, 106281. https://doi.org/https://doi.org/10.1016/j.aap.2021.106281

Liu, Q., Zhou, W., Zhang, Y., & Fei, X. (2021, 14-16 May 2021). Multi-target Detection based on Multi-sensor Redundancy and Dynamic Weight Distribution for Driverless Cars. 2021 International Conference on Communications, Information System and Computer Engineering (CISCE),

Liu, Y., Liu, Z., Luo, X., & Zhao, H. (2022). Diagnosis of Parkinson's disease based on SHAP value feature selection. *Biocybernetics and Biomedical Engineering*, *42*(3), 856-869. https://doi.org/https://doi.org/10.1016/j.bbe.2022.06.007

Liu, Y., Wan, X., Xu, W., Shi, L., Deng, G., & Bai, Z. (2022). An intelligent method for accident reconstruction involving car and e-bike coupling automatic simulation and multi-objective optimizations. *Accident Analysis & Prevention*, *164*, 106476. https://doi.org/https://doi.org/10.1016/j.aap.2021.106476

Lundberg, S. M., & Lee, S.-I. (2017). Consistent feature attribution for tree ensembles. *arXiv e-prints*, arXiv:1706.06060. https://doi.org/10.48550/arXiv.1706.06060

Ma, X., Huo, Z., Lu, J., & Wong, Y. D. (2025). Deep Forest with SHapley additive explanations on detailed risky driving behavior data for freeway crash risk prediction. *Engineering Applications of Artificial Intelligence*, *141*, 109787. https://doi.org/https://doi.org/10.1016/j.engappai.2024.109787

Ma, Y., Wang, Z., Yang, H., & Yang, L. (2020). Artificial intelligence applications in the development of autonomous vehicles: a survey. *IEEE/CAA Journal of Automatica Sinica*, *7*(2), 315-329. https://doi.org/10.1109/JAS.2020.1003021

Marti, E., Miguel, M. A. d., Garcia, F., & Perez, J. (2019). A Review of Sensor Technologies for Perception in Automated Driving. *IEEE Intelligent Transportation Systems Magazine*, *11*(4), 94-108. https://doi.org/10.1109/MITS.2019.2907630

Mateen, A., Hanif, M. Z., Khatri, N., Lee, S., & Nam, S. Y. (2022). Smart Roads for Autonomous Accident Detection and Warnings. *Sensors*, *22*(6).

MathWorks. (2017). *Extended Kalman Filters*. MathWorks. Retrieved 01 February 2024 from https://www.mathworks.com/help/fusion/ug/extended-kalman-filters.html

MathWorks. (2018). *Constant velocity state update*. MathWorks. Retrieved 02 February 2024 from https://www.mathworks.com/help/fusion/ref/constvel.html

MathWorks. (2019a). *Introduction to Tracking Metrics*. MathWorks. Retrieved 16 February 2024 from https://www.mathworks.com/help/fusion/ug/introduction-to-tracking-metrics.html

MathWorks. (2019b). *Multi-Object Trackers*. MathWorks. Retrieved 18 January 2024 from https://au.mathworks.com/help/fusion/multi-object-trackers.html

MathWorks. (2020). *What Is Track-Level Fusion? | Understanding Sensor Fusion and Tracking, Part 6*. https://www.mathworks.com/support/search.html/videos/sensor-fusion-part-6-what-is-track-level-fusion-1598607201282.html

MathWorks. (2021a). *Calculate Generalized Optimal Subpattern Assignment Metric*. Retrieved Feb. 2 from https://au.mathworks.com/help/fusion/ref/generalizedoptimalsubpatternassignmentmetric.html

MathWorks. (2021b). *Introduction to Track-To-Track Fusion*. MathWorks. Retrieved 12-01-2024 from https://www.mathworks.com/help/fusion/ug/introduction-to-track-to-track-fusion.html

MathWorks. (2024a). *Automated Driving Toolbox*. MathWorks. Retrieved 02 February from https://au.mathworks.com/products/automated-driving.html

MathWorks. (2024b). *Forward Collision Warning Using Sensor Fusion*. Retrieved 12 July 2024 from https://www.mathworks.com/help/driving/ug/forward-collision-warning-using-sensor-fusion.html?searchHighlight=forward%20collision%20distance&s_tid=srchtitle_forward%20collision%20distance_1

MathWorks. (2024c). *Navigation Toolbox*. MathWorks. Retrieved 05 January from https://au.mathworks.com/products/navigation.html

MathWorks. (2024d). *Optimize Cross-Validated Classifier Using bayesopt*. MathWorks. Retrieved 15 November 2024 from https://www.mathworks.com/help/stats/bayesian-optimization-case-study.html

MathWorks. (2024e). *Polynomial evaluation*. MathWorks. Retrieved Feb. 2 from https://www.mathworks.com/help/matlab/ref/polyval.html

MathWorks. (2024f). *What Is Cross-Validation?* MathWorks. Retrieved 15 November 2024 from https://www.mathworks.com/discovery/cross-validation.html

MathWorks. (2024g). *What Is Lidar?* MathWorks. Retrieved 10 November 2024 from https://www.mathworks.com/discovery/lidar.html?s_tid=srchtitle_support_results_2_Lidar

MathWorks. (2025). *Hyperparameter Optimization in Classification Learner App*. MathWorks. Retrieved 01 July 2025 from https://www.mathworks.com/help/stats/hyperparameter-optimization-in-classification-learner-app.html

Mo, W., Lee, J., Abdel-Aty, M., Mao, S., & Jiang, Q. (2024). Dynamic short-term crash analysis and prediction at toll plazas for proactive safety management. *Accident Analysis & Prevention*, *197*, 107456. https://doi.org/https://doi.org/10.1016/j.aap.2024.107456

Modas, A., Sanchez-Matilla, R., Frossard, P., & Cavallaro, A. (2020). Toward Robust Sensing for Autonomous Vehicles: An Adversarial Perspective. *IEEE Signal Processing Magazine*, *37*(4), 14-23. https://doi.org/10.1109/MSP.2020.2985363

Moradloo, N., Mahdinia, I., & Khattak, A. J. (2024). Safety in higher level automated vehicles: Investigating edge cases in crashes of vehicles equipped with automated driving systems. *Accident Analysis & Prevention*, *203*, 107607. https://doi.org/https://doi.org/10.1016/j.aap.2024.107607

Nadimi, N., Ragland, D. R., & Amiri, A. M. (2020). An evaluation of time-to-collision as a surrogate safety measure and a proposal of a new method for its application in safety analysis. *Transportation Letters*, *12*(7), 491-500. https://doi.org/10.1080/19427867.2019.1650430

Najm, W. G., Ranganathan, R., Srinivasan, G., Smith, J. D., Toma, S., Swanson, E. D., & Burgett, A. (2013). *Description of light-vehicle pre-crash scenarios for safety applications based on vehicle-to-vehicle communications*.

Natan, O., & Miura, J. (2022). Towards Compact Autonomous Driving Perception With Balanced Learning and Multi-Sensor Fusion. *IEEE Transactions on Intelligent Transportation Systems*, *23*(9), 16249-16266. https://doi.org/10.1109/TITS.2022.3149370

NHTSA. (2017-2023). *Crash Injury Research Engineering Network (Current)*. https://crashviewer.nhtsa.dot.gov/

NHTSA. (2022a). Event Data Recorders [A Proposed Rule by the National Highway Traffic Safety Administration]. *Federal Register*, 12. https://www.federalregister.gov/documents/2022/06/22/2022-12860/event-data-recorders

NHTSA. (2022b). Results of Event Data Recorders Pre-Crash Duration Study: A Report to Congress [Tech Report]. https://doi.org/https://doi.org/10.21949/1530243

NHTSA. (2022c). Summary report: standing general order on crash reporting for level 2 advanced driver assistance systems. *US Department of Transport*.

NHTSA. (2024). *ADS Incident Report Data*. https://www.nhtsa.gov/laws-regulations/standing-general-order-crash-reporting

Novat, N., Kidando, E., Kutela, B., & Kitali, A. E. (2023). A comparative study of collision types between automated and conventional vehicles using Bayesian probabilistic inferences. *Journal of Safety Research*, *84*, 251-260. https://doi.org/https://doi.org/10.1016/j.jsr.2022.11.001

Nurliyana, C., Lestari, Y. D., Prasetio, E. A., & Belgiawan, P. F. (2023). Exploring drivers' interest in different levels of autonomous vehicles: Insights from Java Island, Indonesia. *Transportation Research Interdisciplinary Perspectives*, *19*, 100820. https://doi.org/https://doi.org/10.1016/j.trip.2023.100820

Ortiz, M. F. P., & Dávila, C. E. R. (2023). Virtual reconstruction in traffic: a review of techniques. *Revista Tecnológica Ciencia y Educación Edwards Deming*, *7*(2).

Pamidimukkala, A., Kermanshachi, S., & Patel, R. (2023). An exploratory analysis of crashes involving autonomous vehicles. International Conference on Transportation and Development 2023,

Parseh, M., & Asplund, F. (2022). New needs to consider during accident analysis: Implications of autonomous vehicles with collision reconfiguration systems. *Accident Analysis & Prevention*, *173*, 106704. https://doi.org/https://doi.org/10.1016/j.aap.2022.106704

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, *24*(3), 45-77. https://doi.org/10.2753/MIS0742-1222240302

Pérez, J. A., Gonçalves, G. R., Morillo Barragan, J. R., Fuentes Ortega, P., & Caracol Palomo, A. A. M. (2024). Low-cost tools for virtual reconstruction of traffic accident scenarios. *Heliyon*, *10*(9), e29709. https://doi.org/https://doi.org/10.1016/j.heliyon.2024.e29709

Pour, H. H., Li, F., Wegmeth, L., Trense, C., Doniec, R., Grzegorzek, M., & Wismüller, R. (2022). A Machine Learning Framework for Automated Accident Detection Based on Multimodal Sensors in Cars. *Sensors*, *22*(10).

Qayyum, A., Usama, M., Qadir, J., & Al-Fuqaha, A. (2020). Securing Connected & Autonomous Vehicles: Challenges Posed by Adversarial Machine Learning and the Way Forward. *IEEE Communications Surveys & Tutorials*, *22*(2), 998-1026. https://doi.org/10.1109/COMST.2020.2975048

Qiu, H., Chen, J., Jain, A., Jiang, Q., McCartney, M., Kar, G., Bai, F., Grimm, D. K., Gruteser, M., & Govindan, R. (2018). Towards Robust Vehicular Context Sensing. *IEEE Transactions on Vehicular Technology*, *67*(3), 1909-1922. https://doi.org/10.1109/TVT.2017.2771623

Qu, A., & Wu, C. (2025). Revisiting the correlation between simulated and field-observed conflicts using large-scale traffic reconstruction. *Accident Analysis & Prevention*, *210*, 107808. https://doi.org/https://doi.org/10.1016/j.aap.2024.107808

Rahmathullah, A. S., García-Fernández, Á. F., & Svensson, L. (2017, 10-13 July 2017). Generalized optimal sub-pattern assignment metric. 2017 20th International Conference on Information Fusion (Fusion),

Ren, R., Li, H., Han, T., Tian, C., Zhang, C., Zhang, J., Proctor, R. W., Chen, Y., & Feng, Y. (2023). Vehicle crash simulations for safety: Introduction of connected and automated vehicles on the roadways. *Accident Analysis & Prevention*, *186*, 107021. https://doi.org/https://doi.org/10.1016/j.aap.2023.107021

Répás, J., Berek, L., & Schmidt, M. (2022, 12-13 Oct. 2022). Autonomous Vehicles Forensics-The next step of the Digital Vehicles Forensics. 2022 IEEE 1st International Conference on Cognitive Mobility (CogMob),

Rong, G., Shin, B. H., Tabatabaee, H., Lu, H., Lemke, S., Možeiko, M., Boise, E., Uhm, G., Gerow, M., Mehta, S., Agafonov, E., Kim, D., Sterner, E., Ushiroda, K., Reyes, M., Zelenkovsky, D., & Kim, S. (2020, 20-23 Sept. 2020). LGSVL

Simulator: A High Fidelity Simulator for Autonomous Driving. 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC),

Rosique, F., Navarro, P. J., Fernández, C., & Padilla, A. (2019). A Systematic Review of Perception System and Simulators for Autonomous Vehicles Research. *Sensors*, *19*(3).

SAE. (2021). Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles In (Vol. J3016_202104): SAE International.

Santos, K., Dias, J. P., & Amado, C. (2022). A literature review of machine learning algorithms for crash injury severity prediction. *Journal of Safety Research*, *80*, 254-269. https://doi.org/https://doi.org/10.1016/j.jsr.2021.12.007

Saravanarajan, V. S., Chen, R.-C., Dewi, C., Chen, L.-S., & Ganesan, L. (2024). Car crash detection using ensemble deep learning. *Multimedia Tools and Applications*, *83*(12), 36719-36737. https://doi.org/10.1007/s11042-023-15906-9

Shah, G. A., Khan, S., Memon, S. A., Shahzad, M., Mahmood, Z., & Khan, U. (2022). Improvement in the Tracking Performance of a Maneuvering Target in the Presence of Clutter. *Sensors*, *22*(20).

Shah, S., Dey, D., Lovett, C., & Kapoor, A. (2018, 2018//). AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. Field and Service Robotics, Cham.

Shen, Z., Ji, W., Yu, S., Cheng, G., Yuan, Q., Han, Z., Liu, H., & Yang, T. (2023). Mapping the knowledge of traffic collision Reconstruction: A scientometric analysis in CiteSpace, VOSviewer, and SciMAT. *Science & Justice*, *63*(1), 19-37. https://doi.org/https://doi.org/10.1016/j.scijus.2022.10.005

Shi, X., Yang, F., Tong, F., & Lian, H. (2017, 21-23 April 2017). A comprehensive performance metric for evaluation of multi-target tracking algorithms. 2017 3rd International Conference on Information Management (ICIM),

Sinha, A., Vu, V., Chand, S., Wijayaratna, K., & Dixit, V. (2021). A Crash Injury Model Involving Autonomous Vehicle: Investigating of Crash and Disengagement Reports. *Sustainability*, *13*(14).

Society, R. (2022). *Forensic collision investigation: a primer for courts*. London SW1Y 5AG, UK: Royal Society Retrieved from https://royalsociety.org/-/media/about-us/programmes/science-and-law/DES7309Science-and-the-Law-PrimerCollision63101.pdf

Sohail, A., Cheema, M. A., Ali, M. E., Toosi, A. N., & Rakha, H. A. (2023). Data-driven approaches for road safety: A comprehensive systematic literature review. *Safety Science*, *158*, 105949. https://doi.org/https://doi.org/10.1016/j.ssci.2022.105949

Song, J., & Hyun, S.-H. (2024). Extended Kalman Filter-Based Vehicle Tracking Using Uniform Planar Array for Vehicle Platoon Systems. *Sensors*, *24*(7).

Song, Y., Chitturi, M. V., & Noyce, D. A. (2021). Automated vehicle crash sequences: Patterns and potential uses in safety testing. *Accident Analysis & Prevention*, *153*, 106017. https://doi.org/https://doi.org/10.1016/j.aap.2021.106017

Stark, C., Medrano-Berumen, C., & Akbaş, M. İ. (2020, 28-29 March 2020). Generation of Autonomous Vehicle Validation Scenarios Using Crash Data. 2020 SoutheastCon,

Struble, D. E., & Struble, J. D. (2020). *Automotive accident reconstruction: practices and principles*. CRC Press.

Su, Z., Ji, H., Tian, C., & Zhang, Y. (2024). Performance evaluation for multi-target tracking with temporal dimension specifics. *Chinese Journal of Aeronautics*, *37*(2), 446-458. https://doi.org/https://doi.org/10.1016/j.cja.2023.08.024

Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., Vasudevan, V., Han, W., Ngiam, J., Zhao, H., Timofeev, A., Ettinger, S., Krivokon, M., Gao, A., Joshi, A., . . . Anguelov, D. (2019). Scalability in Perception for Autonomous Driving: Waymo Open Dataset. *arXiv e-prints*, arXiv:1912.04838. https://doi.org/10.48550/arXiv.1912.04838

Ullah, I., Liu, K., Yamamoto, T., Zahid, M., & Jamal, A. (2023). Modeling of machine learning with SHAP approach for electric vehicle charging station choice behavior prediction. *Travel Behaviour and Society*, *31*, 78-92. https://doi.org/https://doi.org/10.1016/j.tbs.2022.11.006

Wang, J., Zhang, L., Huang, Y., & Zhao, J. (2020a). Safety of Autonomous Vehicles. *Journal of Advanced Transportation*, *2020*, 8867757. https://doi.org/10.1155/2020/8867757

Wang, J., Zhang, L., Huang, Y., & Zhao, J. (2020b). Safety of autonomous vehicles. *Journal of Advanced Transportation*, *2020*(1), 8867757.

Wang, S., Li, Y., Qi, G., & Sheng, A. (2023). Optimal Geometry and Motion Coordination for Multisensor Target Tracking with Bearings-Only Measurements. *Sensors*, *23*(14).

Wang, Y., Han, Z., Xing, Y., Xu, S., & Wang, J. (2024). A Survey on Datasets for the Decision Making of Autonomous Vehicles. *IEEE Intelligent Transportation Systems Magazine*, *16*(2), 23-40. https://doi.org/10.1109/MITS.2023.3341952

Wang, Y., Xu, C., Liu, P., Li, Z., & Chen, K. (2024). Assessing the predictability of surrogate safety measures as crash precursors based on vehicle trajectory data prior to crashes. *Accident Analysis & Prevention*, *201*, 107573. https://doi.org/https://doi.org/10.1016/j.aap.2024.107573

Warren, M. E. (2019, 9-14 June 2019). Automotive LIDAR Technology. 2019 Symposium on VLSI Circuits,

Wei, P., Zeng, X., Ouyang, W., & Zhou, H. (2024). Multi-Sensor Environmental Perception and Adaptive Cruise Control of Intelligent Vehicles Using Kalman Filter. *IEEE Transactions on Intelligent Transportation Systems*, *25*(3), 3098-3107. https://doi.org/10.1109/TITS.2023.3306341

Wendt, Z., & Cook, J. S. (2018). *Saved by the Sensor: Vehicle Awareness in the Self-Driving Age*. Machine Design. Retrieved 12 December 2024 from https://www.machinedesign.com/mechanical-motion-systems/article/21836344/saved-by-the-sensor-vehicle-awareness-in-the-self-driving-age

WHO. (2023). *Global Status Report on Road Safety 2023*. https://www.who.int/teams/social-determinants-of-health/safety-and-mobility/global-status-report-on-road-safety-2023

Wu, K.-W., Wu, W.-F., Liao, C.-C., & Lin, W.-A. (2023). Risk Assessment and Enhancement Suggestions for Automated Driving Systems through Examining Testing Collision and Disengagement Reports. *Journal of Advanced Transportation*, *2023*(1), 3215817.

Xiang, C., Feng, C., Xie, X., Shi, B., Lu, H., Lv, Y., Yang, M., & Niu, Z. (2023). Multi-Sensor Fusion and Cooperative Perception for Autonomous Driving: A Review. *IEEE Intelligent Transportation Systems Magazine*, *15*(5), 36-58. https://doi.org/10.1109/MITS.2023.3283864

Yahyaei, M., Seifert, G., Hempen, T., & Huber, W. (2022, 8-12 Oct. 2022). Review of exteroceptive sensors for autonomous driving. 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC),

Yao, S., Guan, R., Huang, X., Li, Z., Sha, X., Yue, Y., Lim, E. G., Seo, H., Man, K. L., Zhu, X., & Yue, Y. (2024). Radar-Camera Fusion for Object Detection and Semantic Segmentation in Autonomous Driving: A Comprehensive Review. *IEEE Transactions on Intelligent Vehicles*, *9*(1), 2094-2128. https://doi.org/10.1109/TIV.2023.3307157

Yao, Y., & Atkins, E. (2018, 4-7 Nov. 2018). The Smart Black Box: A Value-Driven Automotive Event Data Recorder. 2018 21st International Conference on Intelligent Transportation Systems (ITSC),

Yao, Y., & Atkins, E. (2021). The Smart Black Box: A Value-Driven High-Bandwidth Automotive Event Data Recorder. *IEEE Transactions on Intelligent Transportation Systems*, *22*(3), 1484-1496. https://doi.org/10.1109/TITS.2020.2971385

Yeong, D. J., Velasco-Hernandez, G., Barry, J., & Walsh, J. (2021). Sensor and Sensor Fusion Technology in Autonomous Vehicles: A Review. *Sensors*, *21*(6).

Yilmaz, A. C., Aci, C., & Aydin, K. (2016). Traffic accident reconstruction and an approach for prediction of fault rates using artificial neural networks: A case study in Turkey. *Traffic Injury Prevention*, *17*(6), 585-589. https://doi.org/10.1080/15389588.2015.1122760

Yuan, T., Da Rocha Neto, W., Rothenberg, C. E., Obraczka, K., Barakat, C., & Turletti, T. (2022). Machine learning for next-generation intelligent transportation systems: A survey. *Transactions on Emerging Telecommunications Technologies*, *33*(4), e4427. https://doi.org/https://doi.org/10.1002/ett.4427

Yusuf, S. A., Khan, A., & Souissi, R. (2024). Vehicle-to-everything (V2X) in the autonomous vehicles domain – A technical review of communication, sensor, and AI technologies for road user safety. *Transportation Research Interdisciplinary Perspectives*, *23*, 100980. https://doi.org/https://doi.org/10.1016/j.trip.2023.100980

Zazuli, M. I., Dikairono, R., Purwanto, D., Muhtadin, & Hakim, M. L. (2024, 17-19 Jan. 2024). Sensor Fusion System for Localization of Autonomous Car. 2024 International Conference on Green Energy, Computing and Sustainable Technology (GECOST),

Zeng, X., Wang, F., Wang, B., Wu, C., Liu, K. J. R., & Au, O. C. (2022). In-Vehicle Sensing for Smart Cars. *IEEE Open Journal of Vehicular Technology*, *3*, 221-242. https://doi.org/10.1109/OJVT.2022.3174546

Zhang, G., Jin, J., Chang, F., & Huang, H. (2024). Real-time traffic conflict prediction at signalized intersections using vehicle trajectory data and deep learning. *International Journal of Transportation Science and Technology*. https://doi.org/https://doi.org/10.1016/j.ijtst.2024.10.009

Zhang, Q., Liu, J., & Jiang, X. (2023). Lane Detection Algorithm in Curves Based on Multi-Sensor Fusion. *Sensors*, *23*(12).

Zhang, Y., Shao, Y., Shi, X., & Ye, Z. (2024). A Testing and Evaluation Method for the Car-Following Models of Automated Vehicles Based on Driving Simulator. *Systems*, *12*(8).

Zhang, Z., Nie, Q., Liu, J., Hainen, A., Islam, N., & Yang, C. (2024). Machine learning based real-time prediction of freeway crash risk using crowdsourced probe vehicle data. *Journal of Intelligent Transportation Systems*, *28*(1), 84-102. https://doi.org/10.1080/15472450.2022.2106564

Zhao, X., Sun, P., Xu, Z., Min, H., & Yu, H. (2020). Fusion of 3D LIDAR and Camera Data for Object Detection in Autonomous Vehicle Applications. *IEEE Sensors Journal*, *20*(9), 4901-4913. https://doi.org/10.1109/JSEN.2020.2966034

Zheng, L., & Sayed, T. (2020). A novel approach for real time crash prediction at signalized intersections. *Transportation Research Part C: Emerging Technologies*, *117*, 102683. https://doi.org/https://doi.org/10.1016/j.trc.2020.102683

Zheng, O., Abdel-Aty, M., Wang, Z., Ding, S., Wang, D., & Huang, Y. (2023). AVOID: Autonomous Vehicle Operation Incident Dataset Across the Globe. *arXiv e-prints*, arXiv:2303.12889. https://doi.org/10.48550/arXiv.2303.12889

Zhou, J., Zhang, Y., Guo, S., & Guo, Y. (2022, 31 Oct.-3 Nov. 2022). A Survey on Autonomous Driving System Simulators. 2022 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW),

Zhou, R., Huang, H., Lee, J., Huang, X., Chen, J., & Zhou, H. (2023). Identifying typical pre-crash scenarios based on in-depth crash data with deep embedded clustering for autonomous vehicle safety testing. *Accident Analysis & Prevention*, *191*, 107218. https://doi.org/https://doi.org/10.1016/j.aap.2023.107218

Zhou, R., Lin, Z., Zhang, G., Huang, H., Zhou, H., & Chen, J. (2024). Evaluating Autonomous Vehicle Safety Performance Through Analysis of Pre-Crash Trajectories of Powered Two-Wheelers. *IEEE Transactions on Intelligent Transportation Systems*, 1-13. https://doi.org/10.1109/TITS.2024.3392673

Zhu, M., Yang, H., Liu, C., Pu, Z., & Wang, Y. (2022). Real-time crash identification using connected electric vehicle operation data. *Accident Analysis & Prevention*, *173*, 106708. https://doi.org/https://doi.org/10.1016/j.aap.2022.106708