The University of Nottingham

Faculty of Engineering

Department of Electrical and Electronic Engineering

Driver Prediction Model for Autonomous Vehicle within a Virtual Testing Platform Tailored to Malaysia Driving Scenarios

CHEOK JUN HONG

Supervised by

Ir. Ts. Dr. Vimal Rau Aparow
Assoc. Prof. Dr. Tomas Maul
Dr. Ahamed Miflah Hussian Ismail

Thesis submitted to the University of Nottingham

for the degree of Doctor of Philosophy

Abstract

Virtual simulation is a vital tool for testing autonomous vehicle (AV) systems in hazardous scenarios due to its cost-effectiveness, reproducibility, and safety. The reliability of such simulations depends on the accuracy of vehicle dynamics, environmental models, and, critically, driver models, which must replicate human driving behaviour to ensure valid testing results. This study develops an artificial intelligence-based driver model tailored to the Malaysian driving environment, addressing significant differences in traffic behaviour between developing and developed countries. To achieve this, a non-linear 14 Degrees of Freedom (DOF) vehicle model was developed and validated through comparative analysis with experimental data to ensure accurate replication of vehicle handling characteristics. Real-world driving data were collected over 245 hours using an instrumented vehicle equipped with cost-effective off-the-shelf sensors, covering diverse road networks, including urban, rural, and highway scenarios. Additionally, a mixed-reality driving simulator, integrating IPG CarMaker with a 6-degree-of-freedom motion platform and virtual reality, was employed to capture realistic human driving behaviours. Thirty participants were invited, and their driving styles were classified into aggressive, normal, and slow categories. The model was trained using normal driver data to develop a baseline for human-like driving behaviour. A hybrid Convolutional Neural Network-Long Short-Term Memory (CNN-LSTM) model, incorporating attention mechanisms to capture spatial and temporal dependencies in driving behaviour, was implemented. The model achieved 84.63% accuracy in predicting steering, throttle, and braking inputs under simulated conditions. However, when tested with real-world data, accuracy declined to 67.23%, highlighting a generalization gap due to underrepresented road types, varying time-of-day conditions, and environmental factors such as weather variations. To mitigate this issue, further training was conducted using a combination of real-world and simulation data, improving the model's adaptability. The proposed driver model was benchmarked against existing deep learning-based driver models, demonstrating superior performance in replicating human-like driving behaviour within the Malaysian driving context. Despite its contributions, the study acknowledges limitations in data collection, including the limited number of participants, relatively short driving durations per driver, and insufficient representation of extreme driving behaviours. These constraints impact the generalizability of the model to all traffic scenarios. Future work should focus on expanding the dataset with more diverse driving conditions and optimizing the model to enhance its robustness in real-world applications. This research advances driver modelling by leveraging deep learning to create a more contextually relevant model for Malaysia, bridging the gap between virtual simulation and real-world driving behaviour. The developed model has significant implications for AV testing, driver training systems, and intelligent transportation applications in developing countries with complex driving environments.

Acknowledgement

First and foremost, I would like to express my deepest gratitude to my supervisor, Ir. Ts. Dr Vimal Rau Aparow, for his unwavering support, guidance, and encouragement throughout my PhD journey. His insightful feedback and invaluable advice have been instrumental in shaping this research and bringing it to fruition.

I extend my sincere thanks to my co-supervisors, Professor Dr Tomas Maul and Dr Ahamed Miflah Hussain Ismail, for their time, effort, and constructive criticism. Their expertise and perspectives have enriched this work.

I am profoundly grateful to my colleagues and friends in the Autonomous Vehicle Engineering System (AVES) research group for their camaraderie, support, and stimulating discussions. Special thanks to Mr Lee Kah Onn, Mr Alex Au Yong Lup Wai, and Dr Manoharan Aaruththiran Thambippillai for their collaborative spirit and for sharing their knowledge and experience.

I am also thankful to the administrative and technical staff at University of Nottingham Malaysia for their assistance and support. Their dedication and professionalism have made my research process smoother and more efficient. I acknowledge the financial support from Yayasan Tun Ismail Mohamed Ali Berdaftar (YTI) under Permodalan Nasional Berhad, which made this research possible. I am grateful for the opportunity to pursue my studies and conduct this research with their support.

I would also like to my family, especially my parents, for their unconditional love, patience, and encouragement throughout this journey. Their belief in me has been a source of strength and motivation. Lastly, to my friends outside the academic sphere, thank you for providing balance, joy, and perspective in my life. Their support and understanding have been invaluable. This thesis is dedicated to all those who have supported me in this journey, each in their own unique way. Thank you.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

2D	Two Dimensional
3D	Three Dimensional
5G	Fifth-Generation technology standard for cellular networks
A3C	Asynchronous Advantage Actor-Critic
ABS	Anti-Lock Braking System
ACC	Adaptive Cruise Control
ACCD	Accident Scenarios
AD	Autonomous Driving
ADAS	Advanced Driver Assistance System
ADC	Analog-to-Digital Converter
ADS	Autonomous Driving Systems
AI	Artificial Intelligence
AMoDS	AVES Motion Driving Simulator
ANN	Artificial Neural Network
API	Application Programming Interface
APO	Applications Online
APU	Asia Pacific University of Technology and Innovation
ASEAN	Association of Southeast Asian Nations
ASIL	Automotive Safety Integrity Level
AV	Autonomous Vehicle
AVES	Autonomous Vehicle Engineering System
BSI	British Standards Institution
BUS	Binary Unit System
CaaS	Car-as-a-Service
CAN	Controller Area Network
CARLA	Car Learning to Act
CAV	Connected and Autonomous Vehicle
CAVE	Cave Automatic Virtual Environment
CCT	Compact Convolutional Transformer
CETRAN	Centre of Excellence for Testing and Research of Autonomous Vehicles
C-NCAP	China New Car Assessment Program
CNN	Convolutional Neural Network
COM	Communication port
CPU	Central Processing Unit
CV	Computer Vision
LiDAR	Light Detection and Ranging
DDPG	Deep Deterministic Policy Gradient
DDQN	Double Deep Q-Network
DDT	Dynamic Driving Task
DIL	Driver-in-the-Loop
DLC	Double Lane Change

DMM	Decision Making Model
DNN	Deep Neural Network
DOF	Degree of Freedom
DRL	Deep Reinforcement Learning
DS	Driving Simulator
DT	Digital Twin
DTT	Digital Twin Technology
DVA	Direct Variable Access
DVI	Driver-Vehicle Interface
EAM	Execute Action Model
EEPROM	Electrically Erasable Programmable Read-Only Memory
EKF	Extended Kalman filter
EU / EURO	European Union
EV	Electric Vehicle
FB	Fixed-Base
FC	Fully Connected layer
FCN	Fully Convolutional Network
FFB	Force Feedback
FL	Front Left
FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit
FN	Fully connected Network
FOV	Field Of View
FR	Front Right
GB	Giga Bytes
GMM	Gaussian Mixture Model
GPS	Global Positioning System
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
GUI	Graphical User Interface
HIL	Hardware-in-the-Loop
HMD	Head-Mounted Display
HMI	Human-Machine Interface
HMM	Hidden Markov Model
I2C	Inter-Integrated Circuit
ID	Identifier
IMU	Inertial Motion Unit
IoT	Internet of Things
IRL	Inverse Reinforcement Learning
ISO	International Organization for Standardization
ISR	Interrupt Service Routine
ITC-SOPI	ITC-Sense of Presence Inventory
JNCAP	Japan New Car Assessment Program

KNN	K-Nearest Neighbour
KNCAP	Korea New Car Assessment Program
LKAS	Lane Keeping Assisted System
LRT	Light Rapid Transit
LSTM	Long Short-Term Memory
MAC	Multiply-Accumulate Operation
MaRSeD	Malaysian Road Scenario Database
MaGIC	Malaysian Global Innovation and Creativity Centre
MB	Mega Bytes
MCU	MicroController Unit
MIPI CSI	Mobile Industry Processor Interface Alliance Camera Serial Interface
MIROS	Malaysian Institute of Road Safety Research
MLP	Multilayer Perceptron
MPC	Model Predictive Control
MSE	Mean Square Error
NADS	National Advanced Driving Simulator
NASA	National Aeronautics and Space Administration
NAVSAT	Navigational Satellite
NCAP	New Car Assessment Program
NDS	Naturalistic Driving Studies
NHTSA	National Highway Traffic Safety Administration
NI	National Instruments
NM	Near Miss Scenario
NORM	Normal Driving Scenario
NTP	Network Time Protocol
OBS	Open Broadcaster Software
ODD	Operational Design Domain
OS	Operating System
OV	Optimal Velocity
PC	Personal Computer
PESC	System Performance Score
PPR	Pulse Per Revolution
PT	PowerTrain
PX4	PixHawk 4
RADAR	RAdio Detection and Ranging
RAFT	Recurrent All-Pairs Field Transforms for Optical Flow
RAM	Random Access Memory
RC	Remote Controlled
ReLU	Rectified Linear Unit
RGB	Red, Green, Blue
RGB-D	Red, Green, Blue - Depth
RL	Reinforcement Learning
RMS	Root Mean Square

RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
ROI	Regions of Interest
ROS	Robot Operating System
RR	Rear Right
RTK	Real-Time Kinematic
RTX	Real-Time eXtended
SAE	Society of Automotive Engineers
SBC	Single Board Computer
SCK	Serial Clock
SD	Standard Deviations
SEE	Scene Encapturing and Evaluation
SIL	Software-in-the-Loop
SMPC	Stochastic Model Predictive Control
SOPI	Sense of Presence Inventory
SOTIF	Safety of the Intended Functionality
SRS	Sim Racing Studio
SSD	Single Shot Detector/Solid State Drive
SVM	Support Vector Machine
ViT	Vision in Transformer
TB	Tera Bytes
TCP/IP	Transmission Control Protocol/Internet Protocol
TDS	Tractor driving simulator
TORCS	Texas Off Road Championship
TQ	Test Questionnaire
TR	Technical Report
UKF	Unscented Kalman filter
UN	United Nations
US	United States
USA	United States of America
USB	Universal Serial BUS
UTC	Coordinated Universal Time
UTM	Universiti Teknologi Malaysia
V2C	Vehicle-to-Cloud
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-everything
VDS	Video Data Stream
VGG	Visual Geometry Group
VJOY	Virtual Joystick
VOOF	Video Odometry using Optical Flow
VPM	Visual Perception Model
VR	Virtual Reality
VTD	Virtual Test Drive

YOLO	You Only Look Once
YTI	Yayasan Tun Ismail Mohamed Ali Berdaftar

Chapter 1: Introduction

1.1 Overview

As autonomous vehicle (AV) reaches higher Society of Automotive Engineers (SAE) levels with increasingly complex self-driving system and Advanced Driver Assistance System (ADAS) [1], testing the vehicle's safety also become a challenging task. This is mainly because of the complexity to conduct the testing procedures [2] and requires hundreds of millions of road network for safety testing in order to demonstrate the system's reliability in critical scenarios [3]. Therefore, automotive developers and safety testing agencies have proposed for virtual testing in simulated environment and physical testing in a closed-loop environment to ensure the credibility of the AV before deploying the vehicle in real traffic environment. Conventionally, before testing a vehicle in a real environment, the vehicle's system will be tested in a virtual simulation platform with built-in simulated sensor models, vehicle dynamics model and virtual driver model. The advantages of using such platforms are cost effective, easy to re-generate the test scenarios for safety assessment and possible to test the vehicle's system in critical scenarios. These critical scenarios are hard to be tested in real world which could harm the road users if the safety system failed. However, as the reliability of virtual simulation depends on the accuracy of the simulated sensor model, vehicle dynamic model, and environment model, it is important to reproduce the testing scenarios as close to the real-world environment in the virtual simulation platform. This is to ensure a good representative of the vehicle's safety in these scenarios during simulation testing.

Among the simulated models in the virtual simulation platform, driver model plays as an important module in the simulation platform. It functions as a decision-making controller that output driving manoeuvre to control the simulated vehicle in the virtual simulation platform. Driver models must be able to predict and reproduce the driving behaviour of human driver in specific scenarios, so that the testing result can be used as benchmark, when incorporate them into the advanced driving assistance system. However, every driver has their unique habits in handling and controlling the vehicle during accelerating, braking, and cornering in various scenarios. Not to be mentioned that there is a large gap in traffic conditions between a developing country and a developed country. It can be seen that previous research works on the driver model are mostly emphasizing on the traffic scenario from developed countries, which is not similar to developing countries such as in Southeast Asian countries like Malaysia. Therefore, development of a driver model that can reproduce the driving profile of drivers from developing countries is essential. This driving profile will be used as inputs for development of virtual simulation platform to evaluate the performance of autonomous vehicle based on various traffic conditions from developing countries.

Therefore, this study investigates the development of a driver model based on artificial intelligence (AI) system by using driving profile from developing countries. The developed driver model was implemented in simulation-based testing platform for safety assessment of the driver model based on various test scenarios from Malaysian traffic scenarios. In the development process of the driver model, deep learning techniques has been adapted in this research to enhance the reliability of test outcomes in virtual simulations. Leveraging deep

learning, this research study aims to capture the driving behaviour of typical Malaysian drivers. The driving behaviours of Malaysian drivers were used as data inputs for generating steering wheel and speed controls for vehicles within a virtual environment. Considering that deep learning was predicated based on artificial neural networks (ANN) architecture to simulate the cognitive functions and learning mechanisms of the human brain, it is anticipated that the derived driver model will accurately emulate the steering and velocity representing Malaysian driver's behaviours. This effort was proposed to narrow down the gap between virtual testing outcomes and real-world road-testing scenarios.

The main works of this study include the modelling of a non-linear passenger car model, development of a virtual simulator with mixed reality motion platform based on traffic environment of Malaysia. The traffic environments were captured using an instrumented vehicle and the measured data is used for the development of driver's behaviour prediction model for safety testing of autonomous vehicle in virtual environment. In the first stage, the study was performed analytically using computer simulation to develop the non-linear vehicle model. Then, the possibility of developing an instrumented vehicle for data recording of road traffic with cost-effective off-the-shelf components was investigated. Cost-effective components were used to reduce the cost needed for an instrumented vehicle so that it is affordable for developing countries to deploy in mass to increase the size of dataset. The instrumented vehicle was used to validate the non-linear vehicle model and vehicle model of the simulator using handling test based on SAE testing procedures. Next, the instrumented vehicle was used for data recording by driving at the selected Malaysian road network. The data recorded was categorized as "scenarios" whereby the scenarios are analysed and classified using YOLOv8 to extract road users and traffic objects information. Based extracted information, several virtual test cases were created using vehicle driving simulator. On the other hand, the interface for simulator with motion platform and virtual reality were also developed to provide test drivers immersive driving experience in the virtual simulation. Finally, a virtual driver model that can produce realistic human driving style was developed and tested based on driving data of human driver recorded.

1.2 Problem Statement

The quest for safer and more efficient roadways is an ongoing challenge, particularly in developing countries where traffic conditions are notably unpredictable and diverse. One approach adopted by automotive developers and policymakers is the use of autonomous vehicle (AV) technology. In order to ensure a safe deployment of AV on road, a driver model is required which can adapt to the traffic environment of the deployment area. The development of driver models that can accurately predict and mimic human driving behaviour is crucial for the advancement of intelligent transportation systems, autonomous vehicles, and driver assistance technologies. However, most of the existing driver models are predicated on data from developed countries, which may not accurately reflect the driving environment and behaviour in developing nations. These environments are often marked by significant differences due to variations in road infrastructure, traffic regulations, and cultural attitudes towards driving [4]. This disparity poses a significant challenge for automotive engineers and urban planners who strive to develop universally applicable systems that are attuned to the varied driving dynamics across the globe. Meanwhile, driving behaviour in developing

countries is distinguished by unique patterns that emerge from a blend of economic constraints, less structured traffic environments, and distinct societal norms [5]. These elements lead to a driving style that may not be sufficiently represented by current traffic environment and road conditions of developing countries, resulting in a gap for the predictive accuracy of driver behaviour simulations. Therefore, this research aims to bridge this gap by devising a comprehensive driver model that captures the distinct driving behaviours prevalent in developing countries. The goal of this study is to develop an artificial intelligence-based driver model that accurately evaluates vehicle dynamic systems and performs safe decision to handle traffic congestions without any hazard level while driving on the road. Moreover, the driver model is designed as the benchmark model for various traffic conditions to support for the safety assessment of an AV using Malaysian driving scenarios.

Deep learning, with its ability to learn from large datasets and identify intricate patterns, presents a promising avenue for developing such models. This study seeks to utilize the deep learning capability to forge a driver model that authentically mirrors the driving behaviour in developing countries. The model will be trained on an extensive dataset encompassing a broad spectrum of driving scenarios, environmental conditions, and driver interactions. Utilizing advanced neural network architectures, the model aims to deliver a realistic simulation of human driving behaviour, which is crucial for the evaluation and enhancement of various automotive systems. The significance of this research lies in its potential to enhance road safety and efficiency in developing countries, where the rate of traffic-related accidents and fatalities remains disproportionately high. The deep learning-based driver model will also contribute to the global effort of creating more adaptive and intelligent transportation systems that can operate seamlessly across different regions. Therefore, it is important to enhance the existing driver model to produce driving inputs that more closely align with developing countries such as Malaysia and incorporate personalized driver characteristics based on the road environment and traffic conditions where the autonomous vehicle will operate. The issues raised herein underscore the necessity for studies like the proposed one.

1.3 Aim and Objectives

The aim of this research is to develop a novel driver estimation and prediction models optimized for environment of Malaysia using deep learning algorithm for a virtual estimation platform for safety testing and deployment of autonomous vehicle in Malaysia. The driver estimation model is proposed based on the behaviour of human drivers as well as dynamic objects such as pedestrians, cyclist, and motor bikers in urban cities. The expected outcome of the driver model using deep learning is to reproduce the driving behaviour of actual Malaysian drivers. The testing result of the driver model using virtual simulator platform can provide a good representation of the vehicle's safety system in the actual scenarios.

The main objectives of this work are the following:

1. To develop and validate a full vehicle dynamic model with integrated lateral and longitudinal dynamic model.
2. To design a suitable driver prediction algorithm using deep learning with validated vehicle model.

3. To integrate the trained driver model developed into the virtual simulation platform for scenario-based safety testing of autonomous vehicle safety testing based on Malaysian traffic conditions such as congested city traffic and rural roads.
4. To evaluate the performance of driver prediction algorithm using virtual environment model and real-world data based on Malaysian driving scenario.

1.4 Scope of the Research

The scope of this research covers the followings:

1. Derivation of a non-linear 14 degrees of freedom vehicle model and development of vehicle model using MATLAB Simulink platform.
2. System configuration of instrumented vehicle using cost-effective sensors such as camera, global positioning system (GPS), inertial measurement unit (IMU), steering wheel rotation encoder, and pedal sensor to capture driving inputs, vehicle responses and surrounding environment.
3. Validation of vehicle model in terms of longitudinal and lateral manoeuvre tests using the instrumented vehicle.
4. Data collection of traffic scenarios using the instrumented vehicle at several locations across the Malaysian road network, covering urban, rural, and highway environments. The data were collected under varying traffic conditions, including heavy congestion in urban areas, moderate traffic in suburban regions, and free-flow conditions on rural roads. Additionally, the study accounted for challenging road conditions, such as poorly maintained roads, potholes, inconsistent lane markings, and abrupt lane merges. To ensure comprehensive real-world representation, driving scenarios were recorded at different times of the day (morning, afternoon, evening) and under diverse weather conditions (clear, rainy).
5. Development of virtual critical scenarios, encompassing various speeds, road shapes, and traffic conditions within the IPG CarMaker simulator using data collected from the instrumented vehicle based on international standards and operational design domain.
6. Establishment of a virtual driving simulator, equipped with 6 degrees of freedom motion platform, IPG CarMaker simulation tool and virtual reality tool to incorporate driver-in-the-loop (DiL) system.
7. Development of several driver model structures based on deep learning techniques and training the models using driving dataset from the vehicle driving simulator.
8. Identification of most dominant driver model based on performance of the model's capability to generate lateral and longitudinal controls that can represent the optimistic driving style for Malaysian driving scenario.
9. Performance evaluation of the driver model using Software-in-the-loop (SIL) simulation and validation using Hardware-in-the-loop (HIL) simulation in terms of Mean Square Errors of longitudinal and lateral controls compared to average drivers.

1.5 Methodology

This study focuses on developing a data-driven driver model capable of emulating human driving behavior in a virtual environment. The methodology consists of three main phases: (1) Vehicle Model Development and Validation, (2) Data Collection and Driving Simulator Development, and (3) Driver Model Development and Performance Evaluation. These phases ensure a systematic approach to model validation, data acquisition, and driver model optimization.

Phase 1: Vehicle Model Development and Validation

To investigate vehicle dynamic behaviour in response to driving inputs, a nonlinear 14-degree-of-freedom (DOF) vehicle model was developed. The model was validated using IPG CarMaker, ensuring consistency between simulated vehicle responses and real-world dynamics. To further enhance model validation, an instrumented vehicle was constructed, equipped with sensors capable of capturing both road environment data and vehicle states. The vehicle was subjected to international standard tests including ISO 3888-1 (double lane change test), ISO 3888-2 (severe lane change test), ISO 4138 (steady-state circular test), and ISO 15037 (general vehicle dynamics test procedures), evaluating its ability to replicate real-world driving behaviour. If discrepancies were detected, refinements were made to improve accuracy. Figure 1-1 illustrated the methodologies in a flowchart for phase 1.

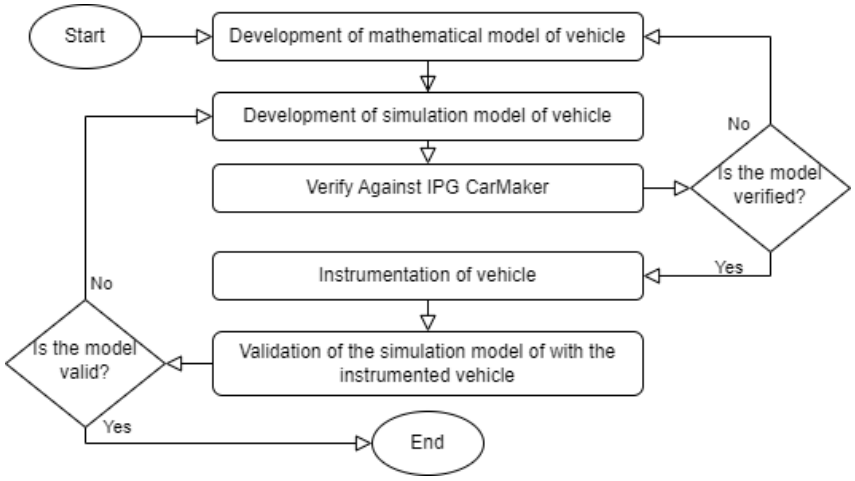


Figure 1-1 Phase 1 flow chart

Phase 2: Data Collection and Driving Simulator Development

Since the goal is to develop a driver model suited for Malaysian driving conditions, real-world driving data was essential. A data collection campaign was conducted using the instrumented vehicle across congested road networks, capturing road actor interactions and driving behaviours. The collected data was analysed and classified into distinct driving scenarios based on the presence of vehicles, pedestrians, and other road actors. If the dataset lacked scenario diversity, additional data was collected to ensure comprehensive coverage. To facilitate large-scale data collection under controlled conditions, a motion platform-based Driver-in-the-Loop (DIL) simulator was developed. The simulator incorporated a 6-DOF

motion platform for realistic vehicle dynamics feedback, steering and pedal interfaces for driver interaction, and a virtual reality (VR) headset for immersive road scene representation. Human drivers participated in simulator-based driving sessions, covering various test scenarios. The recorded driving inputs and vehicle states were pre-processed into structured training datasets for the driver model. Figure 1-2 shows the methodologies in a flowchart for phase 2.

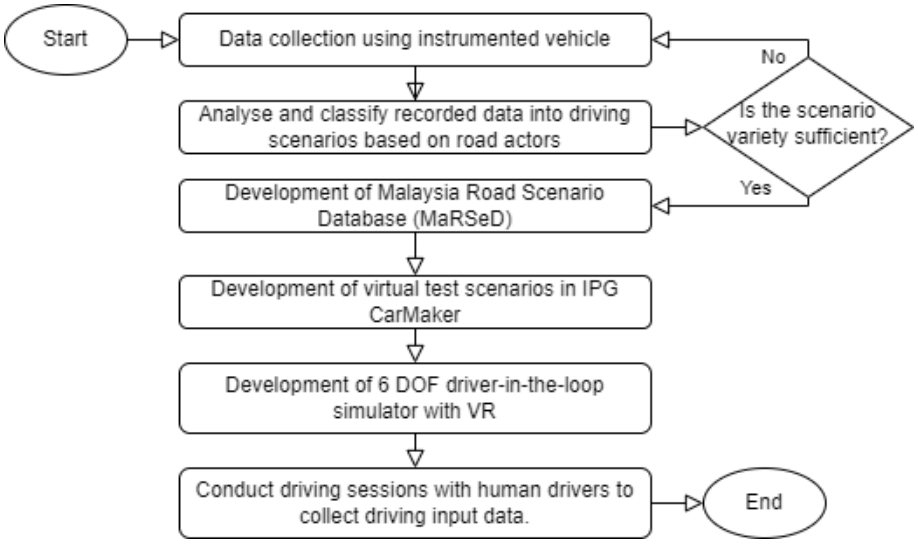


Figure 1-2 Phase 2 flow chart

Phase 3: Driver Model Development and Performance Evaluation

A deep learning-based driver model was developed using a custom CNN-LSTM architecture. The model was trained with input data consisting of sequential image frames from the driving simulator, and vehicle state parameters such as speed and yaw rate. The model outputs steering angles, throttle, and braking commands, effectively mimicking human driving behaviour. Multiple architectures were tested to identify the most accurate model. The trained model was then evaluated for its ability to generate human-like driving responses. If the model's accuracy was insufficient, it was refined through additional training and hyperparameter tuning. If the model met the required accuracy threshold, it was integrated into IPG CarMaker, replacing the default IPG Driver. To assess real-world applicability, the final model was further tested with real-world driving data from the instrumented vehicle. A comparative analysis was conducted between the developed driver model and the IPG Driver, ensuring the model's robustness and adaptability to varying input conditions. Figure 1-3 illustrated the methodologies in a flowchart for this research activity.

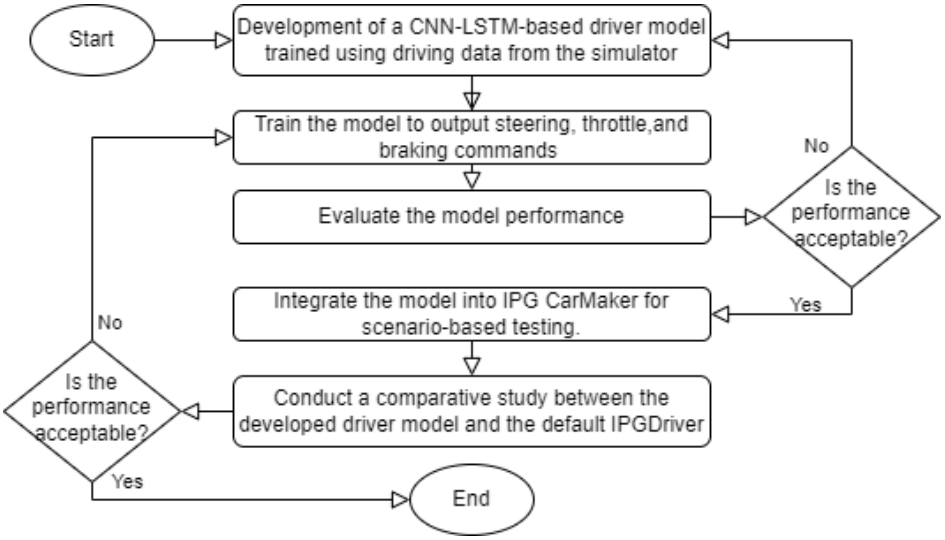


Figure 1-3 Phase 3 flow chart

1.6 Summary of Research Contribution

The This research presents a novel approach to developing a data-driven driver model tailored specifically for Malaysian driving environments, addressing a critical gap in existing driver models which are primarily optimized for western road conditions. The main contributions of this work can be summarized as follows:

1. **Development of an Integrated Virtual Simulation Platform**
   An end-to-end virtual simulation platform was developed, enabling human driving data collection and safety assessment of Level 3 autonomous vehicles. This system integrates IPG CarMaker with Simulink, a 6-DOF motion platform, and a virtual reality headset, providing an immersive driver-in-the-loop (DIL) simulation environment. The incorporation of realistic force feedback and a 360-degree virtual cockpit view enhances the realism of driver behavior data collection.
2. **Affordable, Real-World Data Collection using a Low-Cost Instrumented Vehicle**
   A vehicle was instrumented with a low-cost sensor suite comprising IMUs, encoders, pedal sensors, and cameras to record real driving behaviours on Malaysian roads. The dataset captures critical driving events, including abrupt braking, quick acceleration, lane changes, and vehicle following, creating a comprehensive driver behaviour database for safety assessments in developing countries.
3. **Rigorous Validation of the Vehicle Model**
   A non-linear 14-DOF vehicle dynamics model was developed and validated using real-world data from an instrumented vehicle. The validation process included SAE standard vehicle tests such as double lane change and step steer tests, ensuring the accuracy of vehicle response simulation within the virtual environment.
4. **Novel Deep Learning-Based Driver Model for Malaysian Driving Environment**
   The most significant contribution of this study is the development of a deep learning-based driver model that accurately predicts human driving inputs (steering, throttle, and brake) in Malaysian road conditions. This model incorporates a CNN-LSTM architecture, leveraging spatiotemporal dependencies between visual perception and

vehicle dynamics. Unlike conventional driver models that are either rule-based or trained on Western datasets, this model:

- Captures unique Malaysian driving behaviours, including interactions with diverse road users (motorcyclists, pedestrians, three-wheeled riders).
- Trains on a mixed dataset of simulation and real-world driving data, ensuring generalizability across virtual and physical environments.
- Outperforms the default IPGDriver model and existing driver model in replicating human-like driving behaviour, demonstrating an 84.63% accuracy in simulating Malaysian driving patterns.

5. **Integration and Benchmarking Against Industry Standards**
   The driver model was seamlessly integrated into IPG CarMaker, replacing the default IPGDriver for scenario-based autonomous vehicle safety testing. A comparative analysis was conducted, benchmarking the performance of the proposed model against the built-in IPGDriver in various test scenarios, highlighting the limitations of generic driver models in accurately representing real-world driving conditions in Malaysia.

1.7 Potential Impact of the Research

Based on the outcome of this work, the potential impact of the research can be summarized into two potential impacts. First, the virtual autonomous vehicle safety assessment platform is applicable for safety assessment of autonomous vehicle in Malaysia as well as other developing countries. This work is also helpful for Malaysian government transportation agencies to conduct safety performance of automated vehicle. In future, it can also be used in the Malaysian autonomous vehicle testbed centre as well as cooperate with automotive industries agencies, university research centres and other companies.

Secondly, the driver model that was tailored and trained to fit the culture a traffic of developing Asian countries can ease the development of driver model that can represent the majority drivers in Malaysia as well as other developing countries. Various types of human behaviours such as pedestrians, cyclist, motorbikes, three wheeled riders as well as vehicle drivers will be analysed and used to supervise the machine learned driver models. This driver model will be well trained based on Malaysian cultures and human behaviours where it will ease the implementation of the autonomous vehicles in Malaysian environments. Moreover, any developing countries with similar nature of traffic behaviours can adopt this driver model trained by deep learning to implement in the autonomous vehicle. Besides, this driver model could be used as a prototype for those automotive industries that aim to deploy their newly developed autonomous vehicles without necessarily being concerning about the nature of driving different environments. As for future prospect, supervised driver model in autonomous vehicle could lead for proper time management of deploying vehicles on the road based on traffic congestion. Besides, this could lead to the deployment of autonomous vehicles as a public transportation mode where it can be a solution to further improve the traffic congestion in most of the urban cities in developing countries.

1.8 Structure and layout of the thesis.

This thesis comprises of seven main chapters and the chapters were organized as follows: The first chapter is the introduction chapter of this study. This chapter presents the research background, problem statement, aim and objectives, scopes of the study, methodology used for research work, summary of contribution and overall outline of the thesis.

The chapter 2 presents the literature review on related subjects relating to this research study. In this chapter, various type of simulation platform for autonomous vehicle is investigated. Next, the investigation of instrumentation of vehicle and sensors used for traffic environment and driving data collection is presented in this chapter as well. Then, the published research related to the modelling of driver model using conventional mathematical modelling and using deep learning is presented in this chapter. Furthermore, the deep learning techniques commonly used in the autonomous vehicle field were discussed. Additionally, the adaption of international standards and digital twin technology were introduced in this chapter as well. Finally, the significant contribution based on the limitation and gap identified from the literatures is presented in chapter 2.

Chapter 3 delves into the mathematical modelling of a vehicle and the validation of an instrumented vehicle. Initially, a 14-degree-of-freedom vehicle model is developed using mathematical modelling techniques, with several assumptions made to capture the vehicle's characteristics. Subsequently, a vehicle model is developed using Simulink based on the derived mathematical equations. The mathematical modelling process begins with the derivation of the seven-degree-of-freedom vehicle ride model, followed by the derivation of the Pacejka tire model. Additionally, the chapter covers the derivation of the seven-degree-of-freedom vehicle handling model and the vehicle kinematics model. Finally, the vehicle model is fine-tuned and verified by comparing the vehicle state output of the derived model with that of the IPG CarMaker, using double lane change and bump tests.

Chapter 4 focuses on the instrumentation of the vehicle, covering sensor selection, synchronization, and mounting for the collection of driving data and vehicle response. The chapter also presents the validation results of the 14-degree-of-freedom mathematical model and the instrumented vehicle. These results were derived from field tests conducted with the instrumented vehicle and simulations using IPG CarMaker and Simulink vehicle models, following SAE tests for autonomous vehicles, such as double lane change, step steer, and emergency braking tests at different speeds. Additionally, the chapter discusses the selection of several road networks for the collection of driving style and road traffic environment data.

Chapter 5 details the integration of the 6-degree-of-freedom autonomous vehicle testing simulator with a motion platform and virtual reality, along with the configuration of the virtual platform to develop a simulation environment based on the actual traffic conditions in Malaysia. First, the chapter presents the system configuration of the 3D virtual simulation platform, which can be used as a driver-in-the-loop simulator to collect driving datasets necessary for the development of the driver model. Following this, a human-machine interface for controlling the virtual vehicle in the simulator using a steering wheel and pedals kit hardware is discussed. Additionally, the development of an interface for the motion platform and IPG CarMaker

virtual simulation software is outlined. Next, the chapter describes the development of an interface for the simulator and virtual reality headset. A dataset classification tool was also developed to extract and classify critical scenarios from raw data. Based on these critical scenarios, virtual scenarios were created for the development and testing of the driver model. Finally, the performance of the built-in driver model in IPG CarMaker was compared to the driving style of a human driver, highlighting the limitations of the current model and emphasizing the importance of developing an improved driver model.

Chapter 6 presents the development of driver model and integration of the driver model into the simulation platform developed. In this chapter, the development of several types of driver models using deep learning algorithms was explored using human driving data inputs. The human driving data was recorded by inviting participants to drive on the virtual reality motion simulation platform with different virtual scenarios. The best driver model that able to provide the highest accuracy in estimating the driving input of average Malaysian drivers was selected as the final model. The driver model selected then tested by integrating with the Simulink vehicle model developed to test ability of the driver model driving a virtual vehicle based on the recorded input data. Once the ability of driving a virtual vehicle was verified, the driver model was finally integrated into the IPG CarMaker to replace the default IPGDriver so that Malaysian driving style can be simulated in simulation platform for testing of autonomous vehicle in Malaysia. Finally, the performance and accuracy of the driver model to provide Malaysian driving inputs was evaluated and presented in this chapter.

Finally, in chapter 7, the conclusion chapter for the overall thesis is presented. This chapter summarizes the work done in this study. This chapter also provides recommendation for future research work.

Chapter 2: Literature Review

2.1 Overview

This chapter provides a comprehensive literature review pertinent to the safety assessment of autonomous vehicles and driver models. Structurally, it is organized as follows: The first section introduces the scope of the chapter. The second section reviews the autonomy levels of autonomous vehicles. The third section discusses the current trends in virtual autonomous vehicle simulation platforms, focusing on their limitations and applicability for implementation in Malaysia. The fourth section highlights the development of an autonomous vehicle simulation motion platform, designed as a driver-in-the-loop simulator. The fifth section examines the existing data collection platforms and technologies utilized within this study. The sixth section emphasizes the significance of sensor fusion for the instrumented vehicle and explores various relevant technologies. The seventh section discusses the advantages and drawbacks of integrating 5G technology in autonomous driving. The eighth section analyses the prevailing technological trends in driver model development and their limitations. The ninth section focuses on the background of international standards for safety testing of autonomous vehicles and introduces digital twin technology. Additionally, vehicle modelling is discussed in the eleventh section. The twelfth section addresses the potential research gaps identified through the review of existing literature. Finally, the chapter concludes with a summary.

2.2 Autonomy Levels of Autonomous Vehicles

The development of autonomous vehicles (AVs) represents one of the most transformative advancements in modern engineering, with the potential to revolutionize transportation systems, enhance road safety, and reduce environmental impacts. The Society of Automotive Engineers (SAE) International has established a widely accepted framework, SAE J3016, which classifies vehicle automation into six levels (Level 0 to Level 5) based on the degree of human intervention required [6]. This classification system provides a critical foundation for understanding the technological, regulatory, and societal implications of AVs. This review explores the historical context, technical distinctions, and real-world examples of each autonomy level, while addressing the challenges and future directions of this rapidly evolving field.

The concept of autonomous driving dates back to the early 20th century, with visionary ideas such as General Motors' 1939 Futurama exhibit, which envisioned automated highways [7]. However, practical research began in earnest in the 1980s with projects like Carnegie Mellon University's Navlab, which developed some of the first semi-autonomous vehicles [8]. The 2004–2007 DARPA Grand Challenges marked a turning point, showcasing the feasibility of autonomous navigation in complex environments and spurring significant investment in AV technology [9]. By the 2010s, companies like Google (Waymo), Tesla, and traditional automakers began deploying prototypes, leading to the need for a standardized framework to define and communicate the capabilities of AVs. The SAE J3016 standard, first published in

2014 and revised in 2018 and 2021, emerged as the definitive guide for classifying autonomy levels [6].

The SAE levels provide a clear hierarchy of automation, ranging from no automation to full autonomy, each with distinct technical requirements and operational implications.

- Level 0 (No Automation): The human driver is responsible for all aspects of driving, including steering, acceleration, and braking. Conventional vehicles without advanced driver assistance systems (ADAS) fall into this category. Examples include most vehicles produced before the mid-2000s [6].
- Level 1 (Driver Assistance): The vehicle can assist with either steering or acceleration/deceleration, but not both simultaneously. Adaptive cruise control (ACC) and lane-keeping assist (LKA) are common Level 1 features. For instance, Toyota's 2017 Corolla introduced radar-based ACC, allowing the vehicle to maintain a safe distance from the car ahead while requiring constant driver supervision [10].
- Level 2 (Partial Automation): The vehicle can control both steering and acceleration/deceleration simultaneously under specific conditions, but the driver must remain engaged and monitor the environment at all times. Tesla's Autopilot (2015) and General Motors' Super Cruise (2017) are prominent examples of Level 2 systems. These systems rely on a combination of cameras, radar, and ultrasonic sensors to enable features like lane-centering and traffic-aware cruise control [11].
- Level 3 (Conditional Automation): The vehicle can handle all aspects of driving in certain environments, such as highways, but may request human intervention when system limits are reached. Honda's 2021 Legend, available in Japan, and Mercedes-Benz's Drive Pilot (2023), available in Germany, are among the first commercially available Level 3 vehicles. These systems use advanced sensor suites, including lidar, to enable hands-free driving in geofenced area [12].
- Level 4 (High Automation): The vehicle can operate autonomously within a defined operational design domain (ODD), such as urban areas or specific weather conditions, without requiring human intervention. Waymo's robotaxi service in Phoenix, Arizona (launched in 2020), and Cruise's autonomous ride-hailing fleet in San Francisco (2022) are examples of Level 4 systems. These vehicles rely on high-definition maps, lidar, and AI algorithms to navigate complex environments [13].
- Level 5 (Full Automation): The vehicle can operate autonomously in all conditions and environments without any human input. No commercially available Level 5 vehicles exist as of 2023, but companies like Zoox and Aurora are actively developing prototypes with the goal of achieving Level 5 capabilities by the end of the decade [14].

The progression through SAE autonomy levels reflects significant advancements in sensor technology, machine learning, and computational power. Each level introduces new capabilities and challenges. Levels 0–2 focus on enhancing driver safety and convenience, while Levels 3–5 aim to eliminate human error, which is responsible for over 90% of road accidents [15]. However, higher autonomy levels also raise critical technical, ethical, and regulatory challenges. For instance, Level 3 systems must address the "handover problem," where the human driver must regain control in emergencies, while Level 4 and 5 systems

require robust fail-safes and ethical decision-making algorithms for edge cases [16]. Regulatory frameworks vary globally, with regions like Europe adopting UN-R157 for Level 3 certification, while the U.S. relies on state-level regulations, creating a fragmented landscape for AV deployment [17]. Additionally, public trust and acceptance remain significant barriers, as high-profile accidents involving AVs have raised concerns about safety and reliability [18].

As a conclusion, the SAE autonomy levels provide a comprehensive framework for understanding the evolution of autonomous vehicle technology. While significant progress has been made, achieving full autonomy (Level 5) remains a complex, multidisciplinary challenge. Continued collaboration among engineers, policymakers, and ethicists will be essential to realize the transformative potential of AVs and ensure their safe and equitable integration into society.

2.3 Virtual Autonomous Vehicle Simulation Platform

Looking at the automotive industry, there are several types of testing methodologies for safety system assessment of autonomous vehicle. One of the methods is to use real-world testing or replay as shown in Figure 2-1. The vehicle is tested on selected real-world road environment directly or the data are collected by driving the real vehicle on-road, then the data is replayed back to test the vehicle safety system. The advantage of this method is that the data collected is high fidelity due to the physical sensors were used. However, this also meant that the only the scenarios observed by the vehicle can be tested. Hence, this method is not scalable as rare critical scenarios cannot be tested frequently in a go and time consuming. The second method is structured testing as shown in Figure 2-2, where a test track is specially designed to provide specific test scenarios depending on the system or sensor required to test. This method is also high fidelity because the data were also obtained from the real-world. However, this method is very expensive because one must build a test track. Furthermore, this method is also not scalable as the vehicle cannot be tested hundreds of times in the exact same conditions.

The third method is using a virtual simulator for the simulation-based testing of autonomous vehicles as shown in Figure 2-3. Compared to the previous two methods, virtual simulators are cost effective and would not be any safety issue as the testing is not run in the real-world. By using realistic sensor models, virtual simulators can also provide data with a high level of fidelity. Virtual simulator is becoming more important in facilitating development of new autonomous system in industry. This is because from mechanical subsystems to structural integrity, electronics systems, and software development, virtual simulator is able to give deeper insight into all aspects of the development lifecycle. Aberdeen [19] shows that the abilities of a virtual simulator have been demonstrated to speed up the product development across the product lifecycle as well as increased business value for stakeholders. With the help of simulation toolkit, the development cost, time to market, and product quality improved by 22%, 21%, and 17%, respectively [19]. In general, simulation platform for automated vehicle can be categorized into three types:

- Low-level simulator: fixed-base (FB) simulator with fixed user display,

- Mid-level simulator: the simulator consists of advanced imaging technologies such as large projection screen, real vehicle, and a motion base,
- High-level simulator: the simulator is accelerated using motion platform with more than 6 degrees of freedoms.

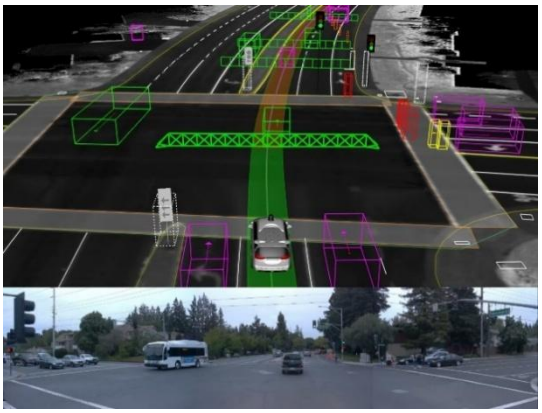


Figure 2-1 Waymo's real-world testing [20]

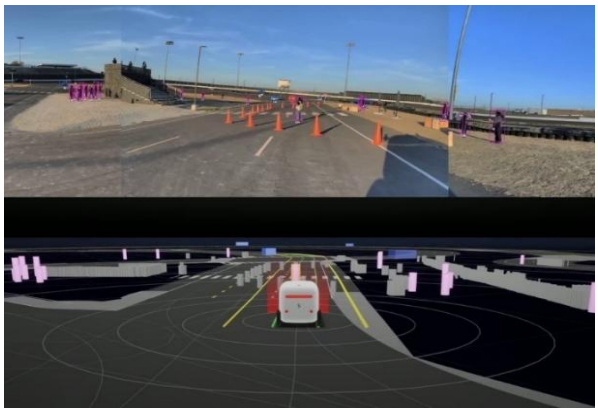


Figure 2-2 Zoox's structured testing on test track [21]

Figure 2-3: Types of vehicle simulation platform [22]*. (a) Delft University of Technology's Fixed-base simulator; (b) Toyota's vehicle simulator; (c) University of Iowa's National Advanced Driving Simulator (NADS).*

General Motors was one of the early companies that start using simulator for vehicle testing, beginning in the 1960s [23]. Figure 2-4 shows one of the two 360-degree simulators of the automaker equipped with image generator that can produce 5-terabyte-per-second images and dynamically respond to the steering and pedal inputs within 50ms [24]. These 360-degree simulators allow pitch, row, and yaw of a vehicle to simulate movement of the vehicle as well as using as a driving simulator to study driver's facial expressions and biometrics [25]. This allows the simulator to be used for testing of design concept, advanced driver assistance, and safety system.

Figure 2-4: General Motors' 360-degree simulator in Technical Centre in Warren, Michigan [24]

Numerous automotive developers have been developing automated vehicle platform such as RTLAB ORCHESTRA, NVIDIA DRIVE AV simulator, Real-time Technologies, MSC Software Simulations, rFpro, AVL, aVDS, ANSYS and LGSVL as shown in Table 2-1.

Table 2-1: Comparison between different Autonomous Vehicle Simulators

AV Simulators	Required Criteria for testing AV									
	Testing standard				European countries			ASEAN countries		
	SAE	ISO 26262	EURO NCAP	ASEAN NCAP	Road users' behaviour	Road and traffic environments	Driver model	Road users' behaviour	Road and traffic environments	Driver model
AVL Autonomous Driving and ADAS [26]	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	No
LGSVL Simulator [27]	No	No	No	No	Yes	Yes	Yes	No	No	No
aVDS [28]	No	No	No	No	Yes	Yes	Yes	No	No	No
ANSYS ADAS [29]	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	No
RT-LAB Orchestra [30]	No	No	No	No	Yes	Yes	Yes	No	No	No
NVIDIA DRIVE Constellation AV Simulator [31]	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	No
Realtime Technologies [32]	No	No	No	No	Yes	Yes	Yes	No	No	No
Automated Driving [33]	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	No
rFpro virtual testing [34]	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	No

These platforms have been used as the virtual testing platform for autonomous vehicle before deployment in real world. However, it can be noted that most of the autonomous vehicle simulators focused on road and traffic environment in developed countries since the research and testing of autonomous vehicles were carried out in European countries. Besides, most of their testing have neglected the interaction between human driving and pedestrian behaviours. Moreover, the autonomous vehicle simulators were used to evaluate the highway driving condition using advanced driving assisted system such as Adaptive Cruise Control (ACC) and Lane Keeping Assisted System (LKAS). These types of active safety systems were used in highway driving scenarios with vehicle speed of 50 km/h – 80 km/h. Moreover, the scenarios considered were mostly involved with passenger and commercial vehicles and did not consider the vulnerable road users' behaviours in these types of scenarios. It is noted that some automotive industries and research centres were actively focusing on virtual simulation testing platform for autonomous vehicle. Based on the report from Autonomous Vehicle Test & Development Symposium 2019 held at Stuttgart, Germany, the research and development on virtual testing platform has been initiated by most of the European countries [35].

First of all, Siemens Company showed a positive approach in the development of virtual testing platform for autonomous vehicle using LMS.Imagine Lab Amesim in developed countries. Meanwhile, TASS International has developed the autonomous vehicle simulation

platform using the PreScan software. This platform focuses on the environmental and human effects for the autonomous vehicle in developed countries. Another company called PTV Group used VISSIM simulation software started to explore on the autonomous vehicle virtual testing. This company main focus was on the traffic flow simulation and recently, they have showed positive interest to explore autonomous vehicle technology using their simulation platform. Then, MIRA Company explored on the simulation in verification and validation of autonomous vehicle with the support of PEGASUS project in Germany [36]. Similarly, CETRAN, Singapore also focused on the simulation testing and validation from physical testing of autonomous vehicle in Singapore [37]. Both projects were designed to evaluate the performance of autonomous vehicles before actual deployment in open environments. However, these projects considering Singapore and European countries' road and traffic environment for the testing. On the other hand, driving simulator also has been used as part of virtual testing platform for autonomous vehicles such as OKTAL and ANSIBLE MOTION. Both companies explored this research because they would like to explore the human driving behaviours and replicate as one of the autonomous driving behaviours for European countries.

Based on the current research trend, it can be observed that most of the safety testing for autonomous vehicles were actively developed and explored in European countries. In Asian countries, CETRAN Simulation Platform in Singapore [37], NVIDIA Virtual Based Simulation Platform, Japan [31] and NAVER Labs [38] and Morai in Korea [39] were actively focusing on safety testing autonomous vehicle in virtual testing by considering their own countries road users' behaviours and environment. Therefore, implementing their virtual testing platform in Malaysian road and traffic environment required a huge modification in the testing platform. Moreover, this modification will be very expensive to be implemented since it required a lot of information on road and traffic environment, road users and driving behaviours in Malaysia. Besides, most of the virtual testing platforms were designed based on the New Car Assessment Program (NCAP) from each country such as EURO NCAP for European countries and Singapore, J-NCAP for Japan and K-NCAP for Korea. Therefore, it is imperative to consider safety testing for autonomous vehicles based on human drivers and road users' interaction from developing countries. This requires collecting information about real-world traffic environments in developing countries, in our case, in Malaysia. Moreover, it is essential to consider the ASEAN NCAP testing standards as part of the virtual testing platform for safety assessment of autonomous vehicle in order the platform will suit the current automotive testing standards in Malaysia.

Furthermore, by studying the system architecture of the virtual testing platforms available in the automotive industry in the past few years [40], [41], it is observed that most of them were built on top of physics/dynamic engines such as Unity game engine and Unreal Engine to create realistic graphics, sensors, vehicle dynamics and beautiful environments. Waymo (Google) has the most mature simulation environment development when compared to other companies with over 5 billion self-driving miles simulated. However, the perception problem was not addressed in the simulation environment because of the lack of realistic graphics. Uber is another self-driving vehicle developing company that has developed a powerful simulation environment for self-driving vehicles. Uber also provides open-source visualisation toolkits for evaluation of mobility and geography [42]. However, their simulation

platform, like Waymo's, does not offer photo-realistic visuals. Besides, there are also open-source simulators such as CARLA (Car Learning to Act), Baidu Apollo and Microsoft AirSim. CARLA is an open-source simulator for autonomous vehicles created by Intel Labs, Toyota Research Institute, and Barcelona's Computer Vision Centre.

CARLA is built on top of Unreal Engine to provide realistic graphics environment and enables for testing from perception to vehicle control. Microsoft AirSim is an open-source simulator not only for drones but also can be used for cars and more. The simulator is designed for artificial intelligence and computer vision experiments. Similarly, Microsoft AirSim uses the Unreal Engine and Unity game engines to help create a very detailed 3D urban environment and virtual vehicles. Moreover, Baidu Apollo [43] is another open-source simulator based on Unity game engine intended for training and validation of the perception algorithms of the self-driving vehicle. Training and validation of perception algorithms requires ground truth data such as three-dimensional (3D) position and boundaries of obstacles, lane markings and other road users. Traditionally, these data are labelled manually by human which is cost and time consuming and subject to error. Whereas Apollo simulator can simulate realistic sensor to generate precise ground truth data at low cost and accelerate the development of perception algorithm for self-driving vehicles with high quality training and validation data. However, the simulator focused on graphics and lack the detail of vehicle dynamics. Gazebo [44] is a widely used simulator in robotics related research fields. This is because its modular architecture enables the simulator to be used with variety of sensor types and physic engines. However, creating a complicated environment in Gazebo is difficult and time consuming. Furthermore, Gazebo also lacks the technology to render realistic graphics that are available in many game engines such as Unreal and Unity game engines.

Simulation tools like CarSim [45] are renowned for their accuracy and detailed methods in simulating the performance of passenger vehicles and light-duty trucks. Validated by automotive engineers for over two decades, CarSim is a preferred tool for vehicle dynamics analysis, active controller development, and safety system engineering. Its standalone operation, along with interfaces to MATLAB/Simulink, NI LabVIEW, and FMI/FMU, allows for versatile integration into various development processes. CarSim's built-in controllers and support for a wide range of model options make it a comprehensive solution for vehicle simulation. Its extensive real-world validation and standalone capabilities make it a reliable and efficient choice for many engineers. However, its focus on vehicle dynamics might limit its scope in broader simulation scenarios that require interaction with more complex environments. Virtual Test Drive (VTD) [46] offers a complete toolkit for driving simulation applications. Designed for the development of ADAS and automated driving systems, as well as training simulators, VTD stands out for its ability to create, configure, and evaluate virtual environments and scenarios for road and rail-based simulations. Its modular design and open platform allow for easy interfacing and integration with third-party or custom packages, making it highly adaptable to specific project needs. VTD's comprehensive environment creation and scenario definition tools enable detailed and varied simulations, but the complexity of the tool may require a steeper learning curve for new users. PanoSim [47], developed in China, provides an integrated testing and simulation platform for autonomous driving. It includes high-fidelity vehicle dynamics models, realistic driving scenarios, and

physics-based sensor models. PanoSim supports offline simulation, real-time simulation with various loops, and digital-twin simulation with virtual reality fusion. Its flexibility and openness facilitate third-party integration and application development, which is crucial for companies looking to customize their simulation experiences. PanoSim's emphasis on integration and customization offers great flexibility, but this comes at the cost of requiring more resources to tailor the platform to specific needs.

It can be found due to the game engines are designed and optimized for games in mind, these platforms lack the capabilities of configurable vehicle dynamics model, personalized driver model, and vulnerable road user's behaviours that are essential for automotive testing. Game engines also unable to provide physically accurate vehicle and sensor models which is one of the critical requirements for AV simulations. For AV simulations, the sensor computation loads are required to run across different nodes on a server to improve the performance of the simulations. However, game engines are designed for gamers which mostly only run on single GPU only. Table 2-2 shows comparison between the requirements of an AV simulator and the quality that can be provide by game engine to fulfil the task of an AV simulator.

On the other hand, a well-known software in the automotive industry that satisfy these requirements is under IPG Automotive products, known as IPG CarMaker [48]. Unlike most other virtual platform that utilise physics engine to create vehicle dynamics model and environment model, IPG CarMaker provides high accuracy vehicle dynamics model that can reproduce the output of a real vehicle. In [49], a vehicle model of the Ford Fiesta car was generated and used to perform lane change test. The outputs obtained from the IPG CarMaker were compared with the sensor data collected with an instrumented Ford Fiesta experimental car performing the same lane change manoeuvres to validate the simulation model and they observed that the outputs obtained from the simulation model correlate well to the outputs of the experimental vehicle. Although the focus realm of the IPG CarMaker is on vehicle dynamics, unlike CarSim, IPG CarMaker provides toolkits for developers to create complex environment, planning of traffic for non-ego vehicles and pedestrians, and realistic sensor with different arrangement and configurations. Although it is a closed-source commercial software, it is open for integration of Simulink models for the developers to develop custom models and plugins such as sensor model and ADAS controller in Simulink which can then be integrated into the IPG CarMaker software. Besides, the state of the virtual environment in the IPG CarMaker such as the vehicle state and environment state can be accessed and controlled from external using Robot Operating System (ROS) [50], C code and Python through the CarMaker's Applications Online (APO) and TCP/IP socket in real-time. This enable the software to be used not only as a virtual simulator for AV testing but also as a driving simulator for human's driving data collection for development of driver model and controller for the AV's system.

Table 2-2 Unsuitability of game engines for autonomous vehicle simulation

	AV Simulator Requirements	Game Engine
Vehicle Dynamics Model	Accurate and configurable	Not accurate, limited parameters configurable
Sensors	Physically accurate, Repeatability	Not accurate
Road User	Realistic and configurable behaviour	Non-deterministic behaviour
Scalability	Test case automation, scalability across multiple computation nodes	Not designed to scale across multiple nodes
Driver Model	Accurate and realistic driving behaviour	Simple driver model
Simulation World	Ability to develop and modify world with ease, ability to replay simulation	Requires expertise to build 3D models of simulation world, No history replay
Modularity	Easy to develop extension using third-party software	Proprietary format

2.4 Driver-in-the-loop simulator

In order to develop an autonomous vehicle safety testing platform simulator, human driver's interaction with the vehicle is an important aspect to be considered. This is because lower-level autonomous vehicle system requires drivers to take control of the vehicle during emergency or critical situations. Study from Koopman and Wagner [16] shows that validating such real-time system could take around one billion driven hours, which is impractical to be done on test tracks and in real-world environments. Therefore, to study the driver's behaviour during critical situations, a driver-in-the-loop simulator that can simulate the real-world testing is required. However, due to limited intrinsic fidelity of the virtual models, it is still impossible to replace real-world assessment with simulator testing as a reliable tool for validating the system under test completely [51]. To fill the gap between real-world and simulated world, an immersive motion simulator is essential. This simulator will provide on-board impression for the driver as the driver is driving using an actual vehicle in simulated environment.

To control the vehicle in the virtual environment, several input devices are needed for the driver such as the steering wheel and pedals. The steering wheel for simulator makes use of electronic position sensor to determine the displacement of the steering wheel. Therefore, the precision and range of this sensor that determine the smoothness of the steering input of a driver may be the one of the important differences between brands. Another consideration for choosing a steering wheel is the range of motion. Cheaper steering wheel normally do not allow for more than ± 135 degrees of steering rotation, whereas more advanced and expensive wheels allow for up to $\pm$ 540 degrees of steering rotation. The last crucial requirement for a steering wheel is the Force Feedback (FFB) feature that can simulate forces on a steering wheel. From the FFB, driver can determine whether the wheel slides or the car hit a bump. Whereas for the pedals, similar to steering wheels, the pedal positions are measured using electronic sensors. Cheaper pedals make use of a spring for rebound, whereas advanced pedals use hydraulic systems which allow for customizable press force and actuation point to provide more realistic feel. Normally, the pedals are included in package with the steering wheel.

For realistic simulation, it is necessary to use advanced steering wheel that can provide accurate position readings, large motion range and force feedback functionality. Some of the popular brands that can offers such advanced steering wheel and pedals are such as Logitech, Thrustmaster, Simucube and Fanatec. In order to choose the suitable steering wheel and pedals for this work, the offering from different brands is compared as shown in Table 2-3. Considering the price of the product, the Logitech G29 has the best value while providing advanced class specification at cheaper price. Due to the more affordable price of Logitech G29, there are more tutorials and videos about Logitech G29 setup. Besides, Simulink also provides a built-in plugin block to obtain raw input data from the Logitech G29 which can ease the development process of the driving simulator using Simulink. Therefore, it would benefit the research most to choose the Logitech G29 steering wheel with pedals kit.

Table 2-3: Comparison between steering wheels from Logitech, Thrustmaster, Simucube and Fanatec.

	Logitech G29	Thrustmaster TMX	Simucube 2 Pro	Fanatec CSL Elite
Motion range	±900 degrees	±900 degrees	±1080 degrees	±1080 degrees
Sensor	10-bit resolution	12-bit resolution	22-bit resolution	12-bit resolution
Force feedback	Dual-Motor Force Feedback	Force Feedback	Force Feedback	Force Feedback
Pedal	Three-pedal board, nonlinear brake pedal	Three-pedal board, customizable press force	Two-pedal, customizable press force	Three-pedal board, customizable press force
Compatibility	Compatible with all mounts	Compatible with all mounts	Required custom simulator skeleton	Compatible with all mounts
Price	$299.99	$459.98	$2144	$599.90

Since driving simulators aim to replicate real-world driving to deceive perception, understanding how the human body perceives the driving experience is crucial. In driving simulators, self-motion perception is formed by combining multiple human sensory systems, as described in [52]. This is mainly because driving heavily relies on the human visual system, one of the most critical sensors for driving simulation. Most of the motion perception in a three-dimensional environment is accounted by the visual system [53]. The optical flow, combined with visual direction and extra-retinal direction, forms the visual information that the brain interprets to define heading [54]. While all the motion perception is formed by the visual and auditory systems in static simulators. In contrast, dynamic motion simulators engage other body sensors, such as the somatosensory system, to enhance the driving experience. The somatosensory system contains neurons and receptors that help human's brain to recognize body position and motion [54]. Hence, a motion simulator needs to trick the somatosensory system of the driver to get the perception of position and acceleration. As demonstrated in [55], by vibrating postural muscles, an illusion of motion can be created. Thus, the higher the fidelity of the cabin view and the motion produced by the motion platform, the more realistic the driving experience becomes.

Several design configurations of parallel manipulators are appropriate for motion platform applications, including the Stewart platform [56]. Numerous driving simulators employ these types of manipulators for their motion platforms. Examples include the Daimler Benz driving simulator [57] and the National Advanced Driving Simulator (NADS) at the

University of Iowa [58] and 6 degree of freedom driving simulator developed by Khusro et al. [59]. However, these motion platforms are high cost and not suitable for non-commercial testing centres due to use of space constraints. Therefore, most research institutes used mid-level motion platform that can provide required motion when developing their driver-in-the-loop simulator. For example, Universiti Teknologi Malaysia (UTM), Delft University of Technology and Ain Shams University developed 6 degree of freedom vehicle driving simulators using steward platform [60], [61], [62]. Different design configurations of motion platform also exist, such as the parallel cable drive configuration [63] and the 5-axis motion platform [64].

The field of view of visual is the horizontal angle of the generated image seen by the subject. Generally, a better sense of velocity can be provided by a wider field of view [65], whereas a narrow field of view can make certain manoeuvres difficult due to insufficient visual information, such as when turning at intersections. There are various types of design configuration for providing visual feedback to a driver.

- **Single Screen**: A single screen generates the field of view, with the rear-view mirror typically rendered as part of the screen. Usually, the field of view is less than 60° horizontally.
- **Single Projector**: Instead of a screen, a single projector is used.
- **Three Screens**: Three screens are attached to a frame, allowing for a field of view of more than 150°. However, there are screen edges separating the image.
- **Three Projectors**: Three projectors are used to project onto a (usually curved) screen. Edges are typically not visible due to soft blending techniques, allowing for a field of view of up to 190°.
- **Multi-Projectors**: 6 to 9 projectors can be setup to render a full 360◦ view.
- **Virtual Reality Headset**: A special helmet with two near-eye displays, which the user views through an optical system that compensates for the short focus.

From literatures, there were many types of driver-in-the-loop simulator that had been developed. Most studies developed their driver-in-the-loop simulator using environment based on Unity3D, Unreal Engine due to high fidelity graphics [66], [67]. As mentioned in the previous section, the vehicle dynamics models, sensor models from these simulators are based on game physics engine which are not as accurate and reliable as industrial engineering simulation software such as CarSim and IPG CarMaker. However, most of studies that used IPG CarMaker focused on static driving platform [68], [69], but there are limited literatures that developed a driver-in-the-loop motion simulator using IPG CarMaker [70]. Existing implementations with IPG CarMaker typically use monitors or projection screens for the cabin view [68], [70], which can limit the driver's field of view and, consequently, limit the perception of velocity [65]. This narrow field of view can be problematic when executing certain manoeuvres, such as turning at crossroads, due to the lack of visually accessible information. To overcome this limitation, recent studies have focused on the implementing virtual reality (VR) headset to provide a fully immersive experience by virtually placing the user inside the virtual environment [66]. However, while there has been research on

implementing virtual reality in IPG CarMaker for pedestrian-in-the-loop simulators [51], [71], studies utilizing VR headsets for DIL simulator development remains underexplored.

2.5 Data Collection Platform

In order to design virtual scenarios for scenario-based testing, driving data in real-world is needed. There are many AV datasets publicly available for research. The most popular is the KITTI dataset that has been widely used as benchmark for development of AV. Besides, there are many automotive datasets recently developed such as the ApolloScape, Waymo, nuScenes, ONCE, RADIATE and Boreas due to the increase gain of research interest in autonomous vehicle field as shown in Table 2-4. Most of the dataset provides front camera view data and have sensor configuration of Global Positioning System (GPS), Inertial Measurement Unit (IMU) and Light Detection and Ranging (LiDAR). Some dataset however provides rear camera view, Radar sensor and driving input data. Waymo is a new dataset which is larger than KITTI and has more data diversity such as night and rain condition. However, Waymo dataset only provides front camera view and lack of rear or surrounding camera. ApolloScape, nuScenes and ONCE are some of the datasets that provide Asian road environment. However, the ApolloScape dataset does not consist of driving input data and lack of raining weather data which is important for developing countries. Similar to ApolloScape, ONCE dataset is captured in Asian road environment and provides both front and rear camera view data. ONCE has more data diversity as it consists of data captured in raining weather but stills lack driving input data. Meanwhile, nuScenes dataset is the most comprehensive dataset including every important element for automotive research from low level to high level autonomous vehicle. RADIATE focussed on Radar sensing solution instead of optical sensors so the dataset only consists of front camera data and the dataset also lack driving input data. Similar to RADIATE, Boreas dataset is one of the latest automotive datasets available and has large diversity of data including seasonal changes and weather such as snow falling, rain and sun. However, it lacks the rear camera view and driving input data.

From Table 2.4, it can be noted that there is also no existing dataset that focusing on developing countries. Besides, these datasets used high end sensors on their instrumented vehicle such as Velodyne HDL-64e Lidar used in KITTI dataset, Navtech Radar used in RADIATE, Velodyne Alpha-Prime (128-beam) Lidar used in Boreas dataset. These sensors can provide high accuracy and high-quality data, but the cost of these sensors is quite prohibitive for a developing country's economy. Hence, implementing such data collection frameworks are difficult in developing countries such as ASEAN countries. In order to develop a dataset that is suitable for these developing countries, there is a need of cost-effective data collection framework that can meet reliability and performance requirements. Using a scenario-based virtual testing simulator approach, any real-world driving condition can be recreated, and thus endless testing can be carried out which can reduce the hazardous level. This data collection framework does not need to consider for autonomous driving but solely to collect data on human driving inputs and the surrounding traffic environment.

Table 2-4 Comparison between existing autonomous vehicle dataset

Dataset	Year	Camera View		RGB resolution	Sensor Configuration	Driving input		Diversity		Location
		Front	Rear			Steer	Gas & Brake	Light intensity	Rain	
KITTI [72]	2013	Yes	No	1392x512	GPS+IMU+Lidar	No	No	No	No	Germany
ApolloScape Lidar [73]	2018	Yes	Yes	3384x2710	GPS+IMU+Lidar	No	No	Yes	No	China
Waymo Open [20]	2020	Yes	No	1920x1080	GPS+IMU+Lidar	Yes	Yes	Yes	Yes	USA
nuScenes [74]	2020	Yes	Yes	1600x900	GPS+IMU+Lidar+ Radar	Yes	Yes	Yes	Yes	Boston & Singapore
ONCE [75]	2021	Yes	Yes	1920x1080	GPS+IMU+Lidar	No	No	Yes	Yes	China
RADIATE [76]	2021	Yes	No	672x376	GPS+IMU+Lidar+ Radar	No	No	Yes	Yes	UK
Boreas [77]	2022	Yes	No	2448x2048	GPS+IMU+Lidar+ Radar	No	No	Yes	Yes	Canada
Argoverse [78]	2019	Yes	Yes	1920x1200	Lidar+Radar	Yes	Yes	Yes	Yes	USA
Audi Autonomous Driving Dataset (A2D2) [79]	2020	Yes	Yes	1920x1080	Lidar+IMU+Lidar+ Radar+Microphone	Yes	Yes	Yes	Yes	Germany
Ad-datasets [80]	2022	Yes	Yes	1920x1080	GPS+IMU+Lidar+ Radar	Yes	Yes	Yes	No	USA
A*3D [81]	2020	Yes	No	1920x1080	Lidar	No	No	Yes	Yes	Singapore
BDD100K [82]	2020	Yes	No	1280x720	GPS+IMU	No	No	Yes	Yes	USA
DriveSeg [83]	2020	Yes	No	1920x1080	GPS+IMU	No	No	No	No	USA
Comma.ai [84]	2018	Yes	No	874x1164	GPS+IMU	Yes	Yes	No	No	USA
Ford [85]	2020	Yes	Yes	1288x964	GPS+IMU+Lidar	No	No	Yes	No	USA
Lyft Level 5 [86]	2020	Yes	Yes	1280x720	Lidar+Radar	No	No	Yes	No	USA
Oxford RobotCar [87]	2017	Yes	Yes	1280x960	GPS+IMU+Lidar	No	No	Yes	Yes	UK

As the real-world dataset gathered from this vehicle is subsequently employed to construct the virtual environment and scenarios for the scenario-based virtual simulator. As a result, the emphasis on sensor quality is not as critical as it would be for sensors utilized in autonomous driving vehicles, since the scenarios are recreated in a virtual environment that can yield highly quality results. Lower-cost sensors can indeed facilitate the widespread deployment of data-capturing vehicles on the roads, enabling the collection of a larger volume of traffic environment data. By reducing the cost of implementation, it becomes more feasible to equip a greater number of vehicles with data-capturing capabilities, thereby expanding the

coverage and diversity of the collected data. This increased availability of data can contribute to more comprehensive analyses and insights for development and safety improvements of autonomous driving systems in developing countries. From literature search, there is no existing automotive data collection system for autonomous vehicle which cost less than 1000$.

Developing a cost-effective instrumented vehicle for data collection requires understanding the types of data necessary for an autonomous vehicle dataset. Cameras are commonly used as the primary sensor for automotive data collection. Mcity data collection platform [88] equipped a total of five cameras on an instrumented Lincoln MKZ. Two of the cameras are front facing cameras where one of the cameras used lens with wider field of view for object detection and the other camera used lens with narrower field of view for traffic signals and signs. A wide field of view camera is placed facing backward of the instrumented vehicle to capture traffic behind the vehicle. The other two cameras are placed on the dashboard to capture relevant driver pose during driving such as head and eye movements which is important to study driver's driving behaviour. From literature, it can be seen that there are many data collection platforms that rely solely on camera sensors for vehicle perception such as Cityscapes and ApolloScape [89], [90].

However, it had been demonstrated by previous researchers that to perform object localization using only camera images are very challenging task [91], [92]. Therefore, most of the recent data collection platforms used combination of camera to provide visual information and ranging sensor such as Radar and LiDAR to provide distance information of the environment [76], [93]. These ranging sensors is robust in all light conditions [94] unlike camera sensor where the visibility of the object is influenced by the light conditions (For example, rain, fog, night, position of the sun and headlight from other vehicles) which had been demonstrated in [95]. The Radar sensor can be used to detect objects by analysing the Doppler spectrum as the spectrums are different for different objects [96]. However, radar sensors have low field of view angle and resolution when compared to camera sensor. Whereas automotive 3D Lidars such as the Velodyne's HDL-64E 3D LiDAR [97] are capable to provide 360 degrees field of view and higher resolution with 64 or more sparse rotating laser beams. The laser beams are used to generate dense point cloud for omnidirectional environment modelling [98]. Conversely, LiDAR's performance is influenced by weather condition such as fog and rain [99]. The sensor is also high cost and required higher computation power which is not ideal for a cost-effective data collection platform.

Over the next decade, many autonomous vehicles are expected to be developed and introduced to the market. To gain user acceptance, these vehicles must be both safe and reliable, while also providing a comfortable driving experience. Although comfort is subjective and influenced by driving style, studies have shown that users prefer autonomous vehicles that drive similarly to their own style [100]. Recent studies such as [101] and [102] focused on developing human driver-like manoeuvres to offer a personalized driving experience, thus enhancing user acceptance of autonomous vehicles. Driving inputs such as steering angle and pedal inputs are critical for creating an autonomous vehicle dataset tailored to personalized driving style research. Most existing frameworks capture these inputs from the Control Area Network (CAN) bus [102]. However, accessing CAN bus data requires direct access to the

vehicle's CAN bus, which is not feasible for many vehicles. Alternatives include snap-on systems that can be added to existing steering wheels to measure the driver's input [88]. Other studies have demonstrated the estimation of steering wheel angles using Inertial Motion Units (IMU) on consumer devices like tablets and smartphones [103], as well as using cameras to capture steering wheel rotation and obtain angles through optical flow-based estimation [104]. For gas and brake pedal inputs, research has utilized specially designed load cells attached to the pedals to measure the force applied by the driver [105].

Besides, Global Positioning System (GPS) is also important data for an AV for navigation and position information of the vehicle. GPS also works in all weather conditions and had been demonstrated to be high accuracy in providing vehicle position, speed, time. For instance, study from [106], [107] used speed and acceleration data obtained from GPS to detect and categorize driving behaviour into abnormal, normal and aggressive driving styles. Furthermore, the precision reliability and availability of the GPS signal in Bandar Baru Bangi, Malaysia had been tested and shown to be more than 99.7% [108]. Therefore, GPS sensor will be an adequate sensor for development of data collection platform in Malaysia. However, GPS sensor normally has low data rates and tall buildings in the urban area, tunnel and underground will obstruct the signals of the GPS and result in positioning failures [109]. Therefore, GPS are normally used with other sensors such as IMU [110] to increase accuracy and reliability.

Finally, IMU is important to study the vehicle’s attitude, lateral and longitudinal accelerations, and jerks of the vehicle to improve user comfort. Study from Vaitkus et al. [111] had used IMU sensor to determine and classify driving styles. IMU sensors can provide signals at high data rates, but its accuracy degrades over time due to gyroscope drift [112]. Positioning fixes can be applied from other data sources such as LiDAR and GPS to compensate for standalone deficiencies using sensor fusion [113].

2.6 Sensor Fusion

Sensor fusion is a fundamental task for the perception of an autonomous driving system, which involves combining data from multiple sensors such as cameras, LiDARs, radars, and ultrasonic sensors to achieve a more accurate and robust understanding of the vehicle's surroundings [114]. Popular sensor fusion techniques like the Kalman filters [115] and its variation such as the Extended Kalman filters (EKF) and Unscented Kalman filters (UKF) can enhance the performance and safety of autonomous vehicles by overcoming the limitations and uncertainties of individual sensors, such as occlusions, noise, blind spots, and varying weather and lighting conditions [116]. Sensor fusion methods for autonomous driving can be broadly classified into three categories based on the level of abstraction at which the fusion takes place: low-level fusion, intermediate-level fusion, and high-level fusion. In this section, the use of sensor fusion in autonomous driving, the advantages and disadvantages of various sensor fusion methods, and their suitability for various weather conditions are discussed.

Low-level fusion, also known as raw data fusion or early fusion, refers to the process of fusing sensor data before any feature extraction or object detection is performed. This approach can exploit the complementary nature of different sensor modalities and preserve the

maximum amount of information available from the sensors. However, low-level fusion also faces several challenges, such as high computational complexity, data synchronization, spatial and temporal alignment, and sensor calibration [117]. Some examples of low-level fusion methods for autonomous driving are:

- Image-LiDAR fusion: This method fuses images and point clouds to generate dense depth maps or semantic maps of the scene [118]. This can improve the performance of tasks such as object detection, segmentation, or classification. Image-LiDAR fusion can be achieved by using deep neural networks that learn to fuse features or representations from both modalities [119], or by using geometric methods that project point clouds onto images or vice versa [120].
- Image-radar fusion: This method fuses images and radar detections to generate enhanced images or radar maps of the scene [121], [122]. This can improve the visibility and robustness of objects in adverse weather conditions or low-light scenarios. Image-radar fusion can be achieved by using deep neural networks that learn to fuse features or representations from both modalities [123], or by using geometric methods that project radar detections onto images or vice versa [124].
- LiDAR-radar fusion: This method fuses point clouds and radar detections to generate fused point clouds or radar maps of the scene [125]. This can improve the range and resolution of objects in the scene and reduce the false positives and false negatives of radar detections. Radar-LiDAR fusion can be achieved by using probabilistic methods that model the uncertainty and noise of both sensors [126], or by using deep neural networks that learn to fuse features or representations from both modalities [127].

Intermediate-level fusion, also known as feature-level fusion or mid-level fusion, refers to the process of fusing sensor data after some feature extraction or object detection is performed on each sensor modality. This approach can reduce the computational complexity and data dimensionality compared to low-level fusion, while still retaining some information from each sensor. However, intermediate-level fusion also requires a common representation for different sensor modalities and a consistent confidence measure for each detection or feature [126]. Some examples of intermediate-level fusion methods for autonomous driving are:

- Bounding box-based fusion: This method fuses the bounding boxes of detected objects from different sensors, such as cameras and lidars, using geometric or probabilistic models. The fused bounding boxes can provide more reliable and consistent object localization and classification results [128]. This can improve the safety and efficiency of autonomous driving by avoiding collisions and optimizing trajectories.
- Region-based fusion: This method fuses the regions of interest (ROIs) from different sensors, such as cameras and radars, using pixel-level or feature-level alignment. The fused ROIs can provide more detailed and richer object appearance and motion information [129]. This can enhance the perception and understanding of complex scenes by capturing fine-grained details and dynamic changes.

- Deep feature-based fusion: This method fuses the deep features extracted from different sensors, such as cameras and lidars, using neural network architectures. The fused deep features can capture high-level semantic and contextual information from different modalities and enhance the performance of downstream tasks, such as object detection or segmentation [130]. This can enable more robust and flexible autonomous driving systems by adapting to various environments and scenarios.

High-level fusion, also known as decision-level fusion or late fusion, refers to the process of fusing sensor data after a high-level abstraction or interpretation is performed on each sensor modality, such as object labels, semantic maps, traffic rules, and driving intentions, to support high-level tasks such as navigation, planning, and control. This approach can handle heterogeneous sensor modalities and different levels of uncertainty in each sensor. However, high-level fusion also loses some information from each sensor and may suffer from inconsistency or ambiguity in the decision making [131]. Examples of high-level fusion methods are:

- Semantic mapping fusion is the process of combining semantic information from different sensors to create a rich and consistent representation of the environment. Semantic information can include object categories, attributes, locations, shapes, sizes, and relations. Semantic mapping fusion can be achieved by using probabilistic graphical models [132], deep neural networks [133], or hybrid methods [134]. Semantic mapping fusion can enhance the accuracy and robustness of localization and mapping, as well as provide useful information for situation awareness and decision making.
- Situation awareness fusion is the process of fusing semantic information from different sensors to infer the current situation and context of the driving scenario [135]. Situation awareness fusion can involve reasoning about the traffic rules, road conditions, weather conditions, driving behaviours, and intentions of other agents. Situation awareness fusion can be achieved by using rule-based methods [136], Bayesian networks [137], or reinforcement learning [138]. Situation awareness fusion can improve the safety and efficiency of autonomous driving, as well as enable adaptive and cooperative behaviours.
- Decision making fusion is the process of fusing semantic information from different sensors to generate optimal decisions for autonomous driving. Decision making fusion can involve planning trajectories, selecting manoeuvres, negotiating with other agents, and executing actions. Decision making fusion can be achieved by using optimization methods [139], game theory [140], or imitation learning [141]. Decision making fusion can enhance the performance and intelligence of autonomous driving, as well as ensure compliance with ethical and legal principles.

The suitability of different sensor fusion methods for various weather conditions depends on several factors, such as the type and quality of sensors used, the robustness and adaptability of the algorithms employed, and the availability and reliability of prior knowledge or contextual information. In general, low-level fusion methods tend to perform better in clear weather conditions where the sensors can provide high-quality data with minimal noise and

distortion. However, low-level fusion methods may degrade significantly in adverse weather conditions such as rain, fog, snow, or dust, where the sensors may suffer from reduced visibility, increased noise, or false detections [142]. Intermediate-level fusion methods can cope better with moderate weather conditions where some features or objects can still be detected reliably by some sensors. However, intermediate-level fusion methods may also fail in extreme weather conditions where the features or objects are too occluded or distorted by the environmental factors [143]. High-level fusion methods can handle more complex and uncertain situations where different sensors may provide conflicting or incomplete information. Nevertheless, high-level fusion methods may also introduce errors or biases in the decision making if the prior knowledge or contextual information is inaccurate or outdated [144].

In conclusion, sensor fusion is a key technology for autonomous driving that can improve the perception capabilities and safety performance of autonomous vehicles by integrating data from multiple sensors. Each level has its own advantages and disadvantages depending on the application scenario and weather condition. Therefore, choosing an appropriate sensor fusion method for autonomous driving requires a careful trade-off between computational efficiency, information preservation, robustness, and accuracy.

2.7 5G in Autonomous Driving

Autonomous driving hinges on three essential components: sensors, data integration, and safety decision-making. Historically, the limitations of network latency and reliability necessitated making most of the decisions locally within the vehicle. This placed significant demands on the vehicle, thereby delaying the widespread adoption of autonomous driving technologies. With the emergence of 5G, which is the fifth generation of mobile wireless technology that offers high speed, low latency, and high reliability for data transmission, these situations will be improved. The 5G can enable or streamline intelligent automation in different aspects of autonomous driving, such as tele-operated driving, high-definition maps, and cooperative collision avoidance.

Tele-operated driving is a scenario where a human driver remotely controls a vehicle through a wireless network. This can be useful for situations where autonomous driving is not feasible or safe, such as complex urban environments, emergency situations, or extreme weather conditions. Tele-operated driving requires a high level of situational awareness and real-time feedback from the vehicle to the driver. Therefore, 5G is essential for providing low-latency and high-bandwidth communication between the vehicle and the remote-control centre. A recent paper by Jin et al. [145] presents an experiment of tele-operated driving using 5G technology in a test track. The paper shows that 5G can achieve an end-to-end latency of less than 10ms and a throughput of more than 100 Mbps, which are sufficient for tele-operated driving with high-quality video streaming and haptic feedback.

High-definition maps are another key component of autonomous driving that provide accurate and detailed information about the road environment, such as traffic signs, lane markings, road geometry, and dynamic objects. High-definition maps can enhance the perception and planning capabilities of autonomous vehicles and improve their safety and

efficiency. However, high-definition maps also pose significant challenges for data acquisition, processing, storage, and transmission. 5G can help to overcome these challenges by enabling fast and reliable data transfer between the vehicles and the cloud servers. A recent paper by Attaran [146] discusses how 5G can facilitate the creation and update of high-definition maps for autonomous driving by using edge computing and artificial intelligence. The paper also highlights the benefits of 5G for enabling vehicle-to-everything (V2X) communication, which can allow vehicles to share map data with each other and with the infrastructure.

Cooperative collision avoidance is a use case where multiple vehicles coordinate their actions to avoid potential collisions. This can improve the safety and efficiency of autonomous driving, especially in scenarios where there are uncertainties or conflicts in the traffic situation. Cooperative collision avoidance requires a high level of communication and cooperation between the vehicles, which can be enabled by 5G. A recent paper by Hetzer et al. [147] proposes a framework for cooperative collision avoidance using 5G-V2X communication. The paper introduces a novel algorithm that can optimize the trajectories of multiple vehicles based on their positions, velocities, and intentions. The paper also presents a simulation study that shows that the proposed framework can achieve better performance than existing methods in terms of collision avoidance rate, travel time, and fuel consumption.

In conclusion, 5G is a key enabler for intelligent automation in autonomous driving. 5G can provide low-latency, high-bandwidth, and high-reliability communication for various use cases, such as tele-operated driving, high-definition maps, and cooperative collision avoidance. These use cases can improve the safety, efficiency, and user experience of autonomous driving. However, there are also many challenges and open issues that need to be addressed for the successful deployment of 5G for autonomous driving, such as network coverage, security, standardization, regulation, and interoperability.

2.8 Driver Model

The driver model plays a crucial role in virtual simulators for ADAS by simulating or predicting a human driver's actions and intentions, enabling ADAS to assist drivers effectively. Existing driver models typically represent the average driver, with fixed parameters that cannot adapt to different individuals. Accurate modelling of driving behaviour is a challenging task because human behaviour is inherently non-deterministic and influenced by factors such as individual variability, driving conditions, and interactions with other road users. Numerous studies have been conducted on modelling of driver behaviour models for ADAS [148], [149], [150].

Current approaches for driver behaviour modelling can be categorized into two types of implementations, 1. Mathematical formulation, 2. Machine learning from data. Typically, both implementations follow a common process flow, which can be outlined as below:

1. Collection of driving behaviour – Driving data are collected from a group of drivers using an instrumented vehicle or a virtual driving simulator. These data are analysed to study the different driver’s behaviour.

2. Modelling Driver behaviour – A driver model can be formulated mathematically by studying the driver's behaviour from the data collected or learned from the data collected using machine learning techniques.
3. Validation of the driver model – The driver model is validated and compared to the driving behaviour of the human drivers. Typically, the driver model will first be evaluated offline, with recorded driving data to verify that the model reproduces the driving behaviour accurately. Next, the driver model will be evaluated online by integrating the driver model into the virtual simulation platform and compared to the driving behaviour of human driver in the virtual simulated vehicle to verify the model's performance in the virtual simulation platform. Finally, the virtual simulation platform with the driver model is compared to the human driver driving in real traffic to verify the accuracy of the virtual simulation platform representing the actual d riving behaviour.

2.8.1 Mathematical Formulation of Driver Model

Modelling of driver has been studied since the early 1950s [151]. Since then, there is an increase of research in the modelling of driver, especially car following model, with the first non-linear car following model developed by General Motors in year 1961, also known as the General Motors model [152]. Most of the driver behaviour modelled using mathematical formulation focuses on one or few aspects only such as decision making based on the driving behaviour, lane changing, car following, trajectory following, acceleration, and braking models. In this section, the driver models based on mathematical formulation will be reviewed.

In order to develop a driver model for virtual vehicle that can be used for testing of active control units, Preusse [153] defined the driver model as a sequence of numerical optimal control problems that required to be solved to predict the future tracking controls. The driver behaviour model was defined by an objective function and solved using Direct transcription and Sequential Quadratic Programming. However, this approach does not guarantee a global optimum and a single-track model for vehicle dynamics model that might not be able to reproduce responses of a real vehicle. Meanwhile, Kristoffer and Tony [154] constructed a car-following driver model using state space modelling method. Their vehicle model was modelled using a state vector which consist of vehicle kinematics and driver's intent. To express the car-following behaviour, they used the well-established Gipps model [155] to estimate the velocity for the vehicle so that the vehicle can stop safely. This enables the driver model to predict the future traffic situation and adjust the vehicle to stay safe. However, the mathematical model was only simulated in a two-dimensional road network and unable to run at real-time according to the authors.

Besides, Miyajima et al. [156] developed driver models for car following and pedal braking using a statistical method of a Gaussian mixture model (GMM) [157]and an optimal velocity (OV) model [158] estimated by a non-linear function. The driver models developed using the two methods were evaluated with driving data collected from eight human drivers in driver identification test. Their result shows that the GMM driver model was able to perform better than the OV model by up to 14.3%. They also demonstrated the driver models were able

to capture and predict driving behaviour during car following and lane changing manoeuvres, using multi-modal driving signals such as gas/brake pedal force, steering-wheel angle, distance between vehicles, vehicle speed, vehicle acceleration [159]. However, the research lacks the consideration of interaction with traffic signs and other road users such as motorcyclists and pedestrians.

Utilizing Model Predictive Control (MPC) [160] method for driver modelling was regarded as main research challenges [161]. Driver model based on MPC was required to predict the future response of a vehicle using a dynamics model of the vehicle and produce control inputs such as steering-wheel angle, gas/brake pedal force to control the vehicle so that the difference between the desired trajectory and the actual trajectory of the vehicle can be minimized. Using MPC method, MacAdam [162] developed a linear optimal trajectory tracking controller that can be updated to simulate the adaptation and learning abilities of an actual human driver. Meanwhile, Qu et al. [163] proposed a driver steering model using Stochastic MPC (SMPC) method where the uncertainties or variations in the model that cannot be deal by the classic MPC were modelled to have statistical properties. For example, uncertainty in road coefficient was modelled as driver's knowledge regarding the road condition's variations. Then, the SMPC can be used to minimize a cost function which is a weighted combination of trajectory tracking and ease of driver control. They also proposed a driver model which used switching based SMPC method [164]. This method enables the SMPC to switch between different driving strategies to adapt to varied road conditions. The result of the switching based SMPC is promising and able to track the desired trajectory accurately. However, the current driver model is limited to lateral control with fixed vehicle speed and would require future work to incorporate the longitudinal control into the driver model.

Many of the research focussed on either lateral or longitudinal driver model development only. For lateral driver model that is responsible to the steering-wheel control, there are research that studies the modelling of eye movement and sensory delays of a human driver [165], [166], [167]. Whereas for longitudinal driver models that control the speed of the vehicle, these models are based on the reaction time, following distances and acceleration limits of a human driver [168], [169]. However, studies have shown that driver that provide lateral input (steer) and longitudinal input (brake) has a higher chance of avoiding collisions. Therefore, Zheng et al. [170] proposed a motion planning model that can provide lateral and longitudinal lane changing. Furthermore, Schnelle et al. [171], [172], [173] proposed a driver model that combines the lateral and longitudinal driver models to provide vehicle velocity and steering-wheel angle like a human driver in real-world collision avoidance and lane-change scenarios. The lateral driving model used is a first-order compensatory transfer function and an anticipatory component that consist of the desired trajectory of the driver. Whereas the longitudinal driving model consists of a feedforward component for selecting driver speed based on the curvature of the desired trajectory, and a feedback component for selecting following distance and speed relative to the preceding vehicle. Figure 2-5 shows the architecture of the combined longitudinal and lateral driver model. The driver model developed can reproduce most of the behaviour of the driver during lane-change and collision avoidance manoeuvres.

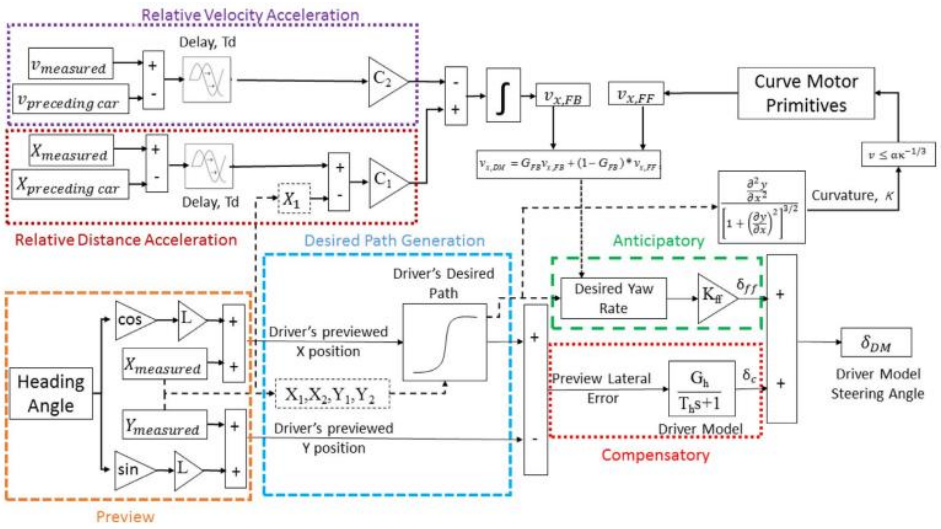


Figure 2-5: Architecture of the combined lateral and longitudinal driver model [171]

2.8.2 Machine Learning Based Driver Model

Research of modelling of driver model for virtual simulation platform have been long established as discussed in a survey [174]. Most existing work on driver estimation and prediction model focus on feature extractor engineering, integrated with basic temporal sequence model. These models require prior or expert knowledge to select features that are correlated to each action of interest of the driver. Many state-of-the-art machine learning algorithms such as Fuzzy-Logic [175], K-Nearest Neighbour (KNN) [176], Gaussian Mixture Model (GMM) [177], Hidden Markov Model (HMM) [178], Support Vector Machine (SVM) [179] and Multilayer Perceptron (MLP) [180] were also studied to model the driving behaviours and achieved remarkable result in various applications. However, most of the machine learning approaches had various shortcomings as well. For example, KNN model will increase the time complexity when computing the distance metrics that define similarities and dissimilarities between the unknown input and all known samples, if the training data is unbalance. HMMs such as first-order Markov model are also limited to contextual information representation, where the output is independent, and the present state depends solely on the prior state. Furthermore, since driver behaviour model construction can be formulated as a time series anomaly prediction problem, human-designed features that normally cannot guarantee optimal features could lead to restrictions on model precision and robustness. This is due to the driver prediction model should be capable of extracting features that are only important to predict future driver behaviours based on the scenarios. Therefore, a feature learning framework is required to take advantage of the sensor-rich input. In addition, most of the driver model focussed on either longitudinal control or lateral control only [181], [182], with a few carries out both longitudinal and lateral control synchronously [180].

Acosta and Kanarachos [180] proposed a hybrid driver model that combines a MPC and feedforward neural network, for autonomous drifting. The hybrid design used MPC for path following, whereas the feedforward neural network is used to estimate the unknown drifting operating point and tire friction properties from data of a real driver drifting in a driver-in-the-loop (DIL) simulator. For MPC, the vehicle dynamics model used to predict the vehicle

response is a two-track vehicle planar dynamics model. The resulting driver model was robust enough to perform autonomous drifting even in road terrains that are not included in the training data. The advantages of using such a hybrid model instead of end-to-end machine approach is because it is very hard to fully verify and validate an end-to-end "black box" system [183]. However, the proposed idea was only implemented in two-dimensional (2D) planar, limited to drifting manoeuvre, and lack of consideration of other road users and traffic objects which will present in a real-world road environment.

Deep learning is a great feature learning technique that make use of Artificial Neural Network (ANN) composed of layers of connected artificial neurons made of non-linear computational units to learn features directly from the input to eliminate the needs of feature engineering. Wu et al. [184] demonstrated their driver model based on deep learning outperform other machine learning methods, such as random forest. There are two types of implementations of deep learning for driver model which are the sequential perception-planning-action pipeline system and the End-to-end learning system [185]. Figure 2-6 shows the differences in system architecture design between sequential pipeline system and end-to-end system. For sequential pipeline system, each component in the system can be designed individually using artificial intelligence method or non-learning method. The components will then combine to form a complete driver model. Whereas for end-to-end system, the system is a direct mapping of a vehicle's on-board sensor data to control commands for the vehicle such as steering-wheel angle, gas/brake pedal pressure.

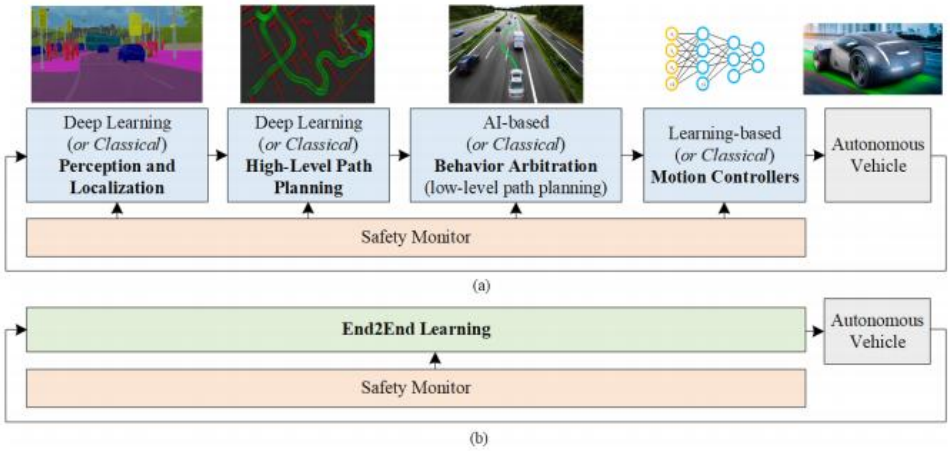


Figure 2-6: Deep learning techniques for driving model [185]*: (a) sequential perception-planning-action pipeline system; (b) End-to-end system.*

Sequential pipeline system is the traditional approach that divides driving task of a driver model into several parts such as perception and localization (object detection and lane detection), path planning, and motion controller. Normally, these components are researched separately. For perception and localisation, the lane detection task normally used computer vision techniques such as edge detection [186], and Hough transform [187]. Whereas for object detection task, normally object detection deep neural network (DNN) such as Single Shot Detector (SSD) [188] and You Only Look Once (YOLO) [189], [190] were used. Then, the path planning and motion controller will plan the trajectory and vehicle's motion using the

information extracted during the perception and localisation stage, such as lane marking and traffic objects' information.

An example of driver model developed using sequential pipeline approach is in [191]. The intelligent virtual driver developed consists of 3 main subsystems, Visual Perception Model (VPM), Decision Making Model (DMM) and Execute Action Model (EAM). VPM is based on Scene Encapturing and Evaluation (SEE) model that provides scene capture and visual information processing. DMM is developed as a rule-based approach that execute decision based on visual information received from VPM and rule weightings. Finally, EAM will control the vehicle in terms of vehicle acceleration, braking and steering based on the decision made by DMM. However, one of the main disadvantages of the sequential pipeline approach is that minor error in the system will accumulate from stages to stages and finally result to an inaccurate prediction.

Unlike sequential pipeline system, the end-to-end system as demonstrated by Bokarski et al. [192] used a single CNN only that takes raw camera frames as input and generate vehicle control signals directly. End-to-end system is able to provide better performance because the hyper-parameters of the model were self-optimised using driving data collected, hence the unnecessary or irrelevant features and steps were removed and will also resulting in a smaller system [193]. End-to-end driver model can either be DNN that trained offline using real-world driving data [194] and/or synthetic data generated from simulator [195], [196], or based on Deep Reinforcement Learning (DRL) that trained and validated in the simulator [197].

(a) CNN

The most used deep learning techniques is the convolutional neural network (CNN) that is utilized for image and speech processing tasks. CNNs made up of several layer of convolutional and subsampling layers with non-linear neural activations to extract feature maps from the input. Then, the feature maps are feed into a fully connected neural network for retrieval, clustering or classification [1]. In CNN, the input Red-Green-Blue (RGB) image is treated as a 3D matrix, while computing the dot product between the filter and the input entries to produce a smaller 2D output of that filter. A mathematical equation of 2D convolution is denoted as Equation 2.1 below.

$$y[i,j] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} h[m,n] \cdot x[i-m, j-n] \tag{2.1}$$

For convolutional process, a filter (kernel) size and the corresponding strides is defined. Stride is the number of shifting pixels over the input matrix. With an input matrix of $h \times w \times d$, applying a convolution filter of $f_h \times f_w \times d$, the resultant output matrix will be $(h - f_h + 1) \times (w - f_w + 1) \times 1$. If the filter does not convolve perfectly towards the input matrix or image, either zero padding towards the edge or drop out the remainder part of the input matrix can be performed.

The convolutional layers served as feature extractors. It can detect the similarity between current and previous layers of the same region and produce the respective feature maps

for the given input image. The neurons which represent local similarity feature maps will therefore form the local receptive fields. In the same convolutional layer, feature maps contain non-identical weights such to enable it to obtain multiple features at each location. Convolution is done between input image or feature maps with trained weights to generate a new feature map, and the output is delivered through a nonlinear activation function to acquire the final classification output. Activation functions are crucial in a neural network model as it determines the accuracy and respective output of a model. It also has major influence in the computational efficiency and converging time for a deep neural network. A general activation function notation is given as Equation 2.2 below:

$$Y = Activation\left(\sum weight \times input\right) + bias \tag{2.2}$$

The activation function which is used widely in state-of-the-art CNNs such as ResNet [198] is ReLU, as it is the simplest form of activation function and is computationally efficient. Defined mathematically as Equation 2.3 below:

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{2.3}$$

The ReLU outputs the input directly if it is positive; otherwise, it returns zero. This simple yet effective function introduces non-linearity into the model while maintaining computational efficiency. ResNet introduced an "identity shortcut connection," which offers several advantages, including:

- Accelerating the speed of training deep networks
- Increasing the network's depth while maintaining its width, thereby reducing the number of parameters
- Mitigating the Vanishing Gradient Problem effect in Conventional CNNs
- Achieving higher network accuracy, particularly in Image Classification

This is because Conventional CNNs are hard to train because backpropagating the gradient to earlier layers can result in repeated multiplication that causes the gradient to become extremely small over time. The issue is known as the vanishing gradient problem, which can be addressed by Batch Normalization, a technique utilized in ResNet. Another issue that arises when deep networks begin to converge is degradation. To tackle these challenges, the residual network (ResNet) was introduced. Residual neural networks explicitly fit the mapping F(x)+x, where x represents the input and F(x) denotes the mapping of the stack of non-linear layers (residual mapping). This enable the residual block to carry important information in the previous layer to the next layers which is similar to a principle introduced with LSTM cells. With this residual block, the ResNet can be made by stacking these residual blocks layers by layers. Even though this short connection looks like an addition to a conventional neural network architecture, the ResNet was able to go deep to 152 layers with much lower parameter than conventional CNN which is smaller in depth at the same time providing faster training performance. This is because the arithmetic addition of F(x)+x did not increase the network

parameters. Besides, it effectively solves the problem of degradation of conventional CNN as the network become deeper.

In conventional neural network, it is assumed that two successive inputs are independent of each other. However, it is not true for couples of applications. For instance, it can be noted that the driver prediction model is a driving behaviour prediction based on time series data captured from sensors such as accelerometer, GPS information and video recording. Therefore, researchers such as Ha and Choi [199] proposed a state-of-the-art activity recognition model that applies convolution and pooling process along the time-series sensor data. Moreover, Recurrent Neural Networks (RNNs) such as Long Short-Term Memory (LSTM) [200] that are commonly used to process natural language, have feedback loops or gates (input and forget gates) to maintain information that had been calculated previously in memory over time.

(b) LSTM

Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memories (LSTMs) have demonstrated effectiveness in various ADAS applications, as evidenced by their successful application in driver activity prediction [201], [202]. Jain et al. [203] utilized an RNN with LSTM units to capture long-term dependencies over time, applying their model to predict driver manoeuvres using real-world datasets. Olabiyi et al. [204] proposed a method employing deep bidirectional RNNs to anticipate driver actions, leveraging both past and future contextual information from sensor data.

LSTMs, featuring a memory cell and three gates (input gate i, forget gate f, and output gate o), are pivotal in handling sequential data due to their ability to selectively retain or discard information over time. Figure 2-7 illustrates the architecture of an LSTM unit, which updates its internal state c_k based on current inputs x_k, previous hidden state h_{k-1}, and previous cell state c_{k-1}. This is governed by recursive equations (2.4).

$$\begin{aligned} i_k &= \sigma(W_i x_k + W_i h_{k-1} + b_i) \\ f_k &= \sigma(W_f x_k + W_f h_{k-1} + b_f) \\ g_k &= tanh(W_{Ig} x_k + W_c h_{k-1} + b_g) \\ c_k &= f_k \odot c_{k-1} + i_k \odot g_k \\ o_k &= \sigma(W_o x_k + W_o h_{k-1} + b_o) \\ h_k &= o_k \tanh \odot (c_k) \end{aligned} \tag{2.4}$$

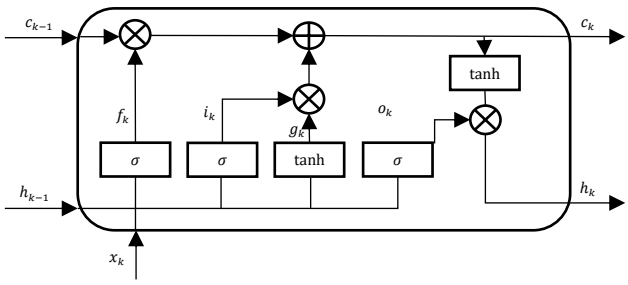


Figure 2-7 Architecture of LSTM [200]

Where, σ represents the sigmoid non-linearity, $tanh$ denotes the hyperbolic tangent non-linearity, ⊙ is the element-wise product, and W and b terms denote weight matrices and bias vectors, i_k,o_k, g_k, f_k, and c_k represents input gate, output gate, input modulation gate, forget gate, and memory cell and at time k. In spite of LSTMs' capability to model long-term dependencies, training them effectively can be challenging. In this context, a series of linear convolution layers are employed to extract the dynamics of the model, enhancing its ability to learn complex temporal patterns.

(c) CNN-LSTM

CNNs are widely recognized for their capability to efficiently extract features from images by converting them into flat vectors. It is conventional to integrate a CNN at the beginning of the "encoder" module. When trained in conjunction with the "encoder-decoder" LSTM module, this CNN module contributes to the development of an advanced end-to-end model that exceeds the complexity of basic models. Furthermore, Ordóñez and Roggen [205] also proposed a state-of-the-art model with the combination of CNN and LSTM for wearable activity recognition. These designs enable neural networks to predict future output, which is depended on the previous computations. In fact, research from Sonja [206] shows that the past behaviour of a driver is the main contribution factor required to predict the driver's intention due to the motive behind an action can be reasoned. Ever since NVIDIA developed the End-to-end driving model known as PilotNet [207] in 2017, there have been a lot of research that utilized the similar architecture in driver model in the past few years. This approach trains CNN by mapping raw sensory data from the vehicle directly to the vehicle control commands. Meanwhile, Xu et al. [208] transform the autonomous driving problem into a vehicle future motion prediction problem by introducing temporal features into the end-to-end visual model. The method proposed is a combination of Fully Convolutional Network and Long-Short Term Memory (FCN-LSTM), where the FCN act as visual encoder to learn visual representation in each video frame, while the LSTM will act as temporal encoder to provide motion history information.

This CNN-LSTM architecture has been applied to different aspect of automated driving such as localisation, prediction, decision making and control. For example, in localisation, Selvaraj et al. [209] combined CNN and convolutional LSTM (ConvLSTM) for detecting lanes and localisation of the car relative to the map from road images. In prediction, CNN-LSTM can be used to forecast the future states and behaviours of road users, such as their trajectories, speeds, and intentions [210]. In decision making, Cheng et al. [211] proposed a lateral manoeuvre decision making network based on CNN-LSTM architecture to make either lane change, or lane keep decision based on distance and speed of preceding and neighbour vehicles information obtained from motion image sequence. In control, CNN-LSTM model can be used to adjust either the steering angle or the speed of the self-driving car according to the planned path and the surrounding conditions [212][213][214][215]. Valiente et al. [216] employed a CNN-LSTM based network for autonomous steering control in cooperative self-driving vehicles. To enhance the steering angle control accuracy, they proposed a system that enables information sharing between these cooperative vehicles.

Similar idea has also been applied in [217], where the network used to develop driver steering model is a C-LSTM (Convolutional-Long Short-Term Memory). Whereas in [218], the proposed driver model requires multiple CNNs act as feature encoder and four LSTMs network that act as temporal encoder to process information coming from surround-view cameras. Then two more fully connected networks (FNs) are needed to provide control commands in terms of future steering angle and future speed. Whereas Qin et al. [219] proposed a car-following model using CNN-LSTM architecture to control speed of the vehicle so that a safety distance from the leading vehicle can be maintained. Vijitkunsawat et al. [220] successfully deployed CNN-LSTM algorithm on a scaled RC-Car platform powered by Nvidia Jetson Nano for self-driving car navigation at three different speeds with high accuracy. However, these studies only considered either controlling the steering angle while keeping constant speed or vice versa. Yang et al. [221] proposed a multi-modal system for self-driving to address previous end-to-end model that can only perform single task learning. By using multi-modal network, suitable model can be used for different task which in this case, the system predicts steering angle using CNN and speed value using LSTM. Instead of dividing the network into two, Gu et al. [222] proposed an end-to-end self-driving model based on CNN-LSTM to predict both steering angle and speed control within a single model. They used LSTM to form the "encoder-decoder" structure to learn information hidden in the input features and managed to achieve high accuracy using Waymo Open dataset. However, the scenarios tested are just moving in straight line without large lateral motion such as car-following and deceleration.

Zhang et al. [223] then further improve the CNN + LSTM architecture by introducing the attention mechanism [224] to the network. This is because the default LSTM or other Recurrent neural network (RNN) based network like the Gated Recurrent Unit (GRU) [225] have to encode the entire input sequence into a single vector. However, every feature maps have different contribution to the prediction of driver behaviour and the LSTM only has fixed-length memory cell. This will cause loss of important information if the time series sensor data is large and complex. By introducing the attention mechanism into the network, the attention unit which is a neural network will learn the weightage for each feature map. Hence, only the feature maps that are relevant will remains and used as input for the LSTM or RNN. Zhao et al. [226]proposed an end-to-end self-driving model that can predict both steering angle and speed control but joined with attention mechanism to improve accuracy of scene perception based on vision. However, similar to the previous model, due to lack of input data such as trajectory waypoint, GPS data, both models do not have the ability to perform task such as path following and self-driving navigation tasks.

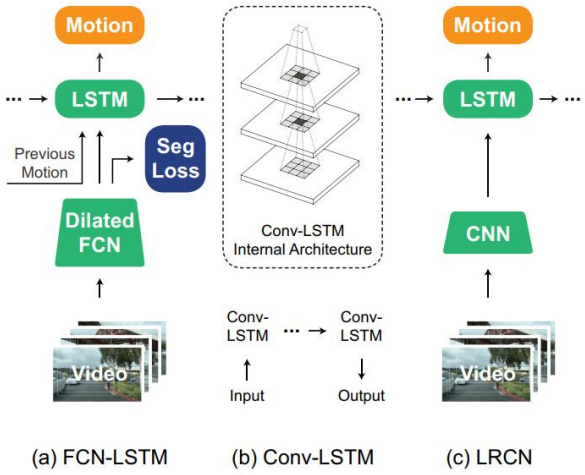


Figure 2-8: Architecture of different types of visual-temporal end-to-end deep neural network [208]

Instead of using traditional method where CNN model is trained with offline training data, Perot et al. [227] proposed a driving model based on CNN + LSTM architecture but was trained using Reinforcement Learning (RL) in the Texas Off Road Championship (TORCS) game simulator. In RL, the DNN learn by trial and error in an interactive environment and uses reward and punishment as positive and negative feedback to the DNN. To maximize the accumulated reward and optimise the driving policy using only RGB image from a front facing camera as input, asynchronous advantage Actor-Critic (A3C) method is used to learn the vehicle controls in the simulator because it does not require experience replay for decorrelation during training. The system is then enhanced in [228] with full lateral and longitudinal control including hand brake control for drifting. The enhanced system also proved to converge faster and have better generalization of the car control.

Other driver models based on Reinforcement Learning used different training strategies. For example, in [229], the Double Deep Q-Network (DDQN) [230] was used to implement the driving policy of the driver model based on classical Reinforcement Learning. Whereas in [231], the driving policy was implemented using Deep Deterministic Policy Gradient (DDPG) [232] algorithm that was proposed by DeepMind. This algorithm has good convergence, good stability, does not require a model, and can be used in a continuous action domain. It can be said that the most challenging part of developing driver model based on RL is to find the right reward function for the agent, so that the driver model will act like humans. In contrast, faulty reward function will make the trained model fails [233]. To obtain the optimum reward function, Huang et al. [234] used Inverse Reinforcement Learning (IRL) method to learn the reward function directly from human naturalistic driving data. This is done by adjusting the weights of the reward function so that the driving policy matches the decision-making scheme of humans. By using IRL, the proposed driver model could reduce error in human likeness by up to 24%. However, IRL relies on the assumption that the human's decision-making scheme is optimal with respect to an unknown reward function [235].

2.9 Safety Assessment of Autonomous Vehicle Systems

The process of verification and validation is crucial in the development and deployment of Autonomous Vehicles (AVs), ensuring that these complex systems operate safely and reliably. Standardization of testing procedures has emerged as a critical step towards achieving this goal, but the road to a unified approach has been winding and complex. ISO standards play a pivotal role in establishing the necessary frameworks for safety testing. The initial efforts in standardizing AV testing focused on adapting existing frameworks. With the groundwork laid in foundational research [16], researchers explored applying the well-established ISO 26262 standard for functional safety in road vehicles [236], [237]This standard, designed for traditional cars, provided a strong foundation for safety considerations. However, as AV technology advanced, it became clear that adaptations were needed. The unique functionalities and reliance on complex algorithms in AVs necessitated a more nuanced approach. The International Organization for Standardization Technical Report (ISO/TR) 4804:2020, for instance, provides comprehensive guidelines for the design, verification, and validation of cybersecurity and safety for automated driving systems [238]. It emphasizes a safety- and cybersecurity-by-design approach, focusing on vehicles with level 3 and level 4 features as defined by SAE J3016:2018 [239]. This standard outlines methods for developing dependable systems by integrating the three dependability domains: safety of the intended functionality, functional safety, and cybersecurity. These domains work in tandem to create a robust system capable of withstanding various operational challenges

As research progressed, the complexities involved in standardizing AV testing became apparent [240]. The sheer diversity of real-world scenarios that AVs need to navigate effectively posed a significant challenge. Unlike traditional vehicle testing on controlled tracks, AVs need to be prepared for everything from sunny highways to unpredictable weather conditions on winding mountain roads. Additionally, the vast array of sensor technologies employed by different developers (cameras, LIDAR, Radio Detection and Raging (RADAR)) further complicated the issue. Different technologies necessitate different testing methods to ensure proper functioning. Responding to these challenges, a collaborative approach emerged with the proposal for a standardized, open-source repository of testing scenarios [241]. This collaborative approach allows researchers and developers to share a vast array of standardized test cases categorized by complexity and driving conditions. This not only accelerates development but also ensures a wider range of situations are considered, fostering the creation of more robust AV systems.

(a) Adoption of International Standards for Safety Testing of Autonomous Vehicle

The work of Koopman et al. [242] underscores a significant challenge in the development and testing of autonomous vehicles (AVs), particularly in relation to the Operational Design Domain (ODD). The ODD is a foundational concept in AV development, defining the specific conditions under which an AV is designed to operate safely. It encompasses a range of variables, including geographic, climatic, and traffic-related factors. The precise delineation of an AV's ODD is crucial for the development of relevant test scenarios and for ensuring that the vehicle can handle real-world driving situations within its

designated domain. However, the limitations of current simulation-based testing methods, as highlighted by Koopman et al., stem from their inability to fully capture the complexity and variability inherent in real-world environments that an AV's ODD must account for. Simulations are essential for AV development because they allow for the safe and controlled testing of vehicles under a wide range of scenarios. However, if these simulations are not sufficiently representative of the real world, they may fail to prepare the AV for the full spectrum of challenges it will face within its ODD.

To address this, Koopman et al. emphasized the standardization of simulation environments and the integration of real-world sensor data into these simulations. By doing so, the simulations become more aligned with the ODD, enhancing their ability to replicate the diverse scenarios an AV will encounter. This integration ensures that AVs are tested against more realistic and challenging conditions, thereby improving their readiness for deployment in real-world traffic situations. In essence, the authors call for standardization in simulation environments and the integration of real-world sensor data to ensure that the ODD is thoroughly and accurately represented in testing protocols. This approach helps bridge the gap between theoretical testing and practical application, ensuring that AVs are truly prepared for the complexities of the environments they are designed to navigate.

While earlier research works [243], [244], [245], [246], [247], [248], [249] have focused on adapting existing standards like ISO 21448 Safety of the Intended Functionality (SOTIF) [250] and ISO 26262 Road Vehicles — Functional Safety. These standards were pivotal in earlier studies focusing on the development and deployment of autonomous vehicles in physical environments. a new player has emerged which is the ISO 3450X series standards. The ISO 34501 Road vehicles — Test scenarios for automated driving systems — Vocabulary [251] is the first document in the ISO 3450X series standard to specify the vocabulary and terminology used for the test scenario creations. ISO 34501 standardizes the vocabulary and terminology for creating test scenarios. It aligns with existing standards such as ISO 21448 and ISO 26262-1, ensuring consistency across technical documents. This standard is crucial for maintaining a unified language and understanding among researchers and practitioners working on automated driving systems. ISO 34502 Road vehicles — Test scenarios for automated driving systems — Scenario based safety evaluation framework [252] outlines the safety tasks, scenario-based safety evaluation processes, and the relationship between the developed framework and other standards and legislation. It focuses on inputs such as Operational Design Domain (ODD), regulations, and expert knowledge, identifying critical scenarios based on risk factors. The document also describes the derivation of concrete test scenarios, test executions, and safety evaluations. This comprehensive approach ensures that all relevant factors are considered in the safety assessment process.

ISO 34503 Road Vehicles — Test scenarios for automated driving systems — Specification for operational design domain [253] specifies the ODD, which identifies the capabilities of Autonomous Driving Systems (ADS). The ODD represents the operating conditions within the test environment in which an ADS can safely perform the dynamic driving task (DDT). This document, applicable to level 3 and level 4 ADS, leverages the BSI Flex 1889 framework to detail ODD and scenario levels. The specification of the ODD is

crucial for understanding the limits and capabilities of ADS in various conditions. The BSI Flex 1889 [254] provides a framework for classifying scenario abstraction levels, essential for designing scenario-based testing. Four types of scenario levels are defined: functional scenarios, abstract scenarios, logical scenarios, and concrete scenarios. Functional scenarios are non-formal, human-readable explanations, often accompanied by visualizations. Abstract scenarios, on the other hand, are formalized and machine-readable, described based on cause-effect relationships. Logical scenarios involve parameterized sets of scenarios, incorporating ranges of variable scenario parameters. Concrete scenarios are specific scenarios with fixed parameters, typically written in OpenDrive and OpenScenario format.

ISO 34504 Road vehicles — Test scenarios for automated driving systems — Scenario categorization [255] focuses on scenario categorization based on the ODD defined in previous standards. It introduces a tagging system to classify scenarios for the safety assessment of ADS. Tags categorize scenarios based on characteristics such as

1. dynamic objects
2. scenery aspects
3. environmental situations
4. desired test applications.

This categorization facilitates a structured approach to scenario analysis and ensures that all relevant factors are considered in the safety assessment. The top-level taxonomy in Figure 2-9 categorizes Operational Design Domain (ODD) attributes related to scenario categorization, classifying basic ODD based on scenery aspects, environmental situations, and dynamic objects. Each scenario category represents driving conditions at specific locations. Figure 2-10 illustrates the relationship between scenarios and scenario categories. For example, scenario category A might be "daytime" and category B might be "jaywalking pedestrian." The overlapping elements of these tags represent scenarios occurring during daytime with a jaywalking pedestrian. Real-world data from instrumented vehicles is used to gather relevant scenarios based on deployment locations for autonomous vehicles. ODD is dependent on the application of autonomous vehicles, classifying operating conditions based on scenery, environment, and dynamic object detection.

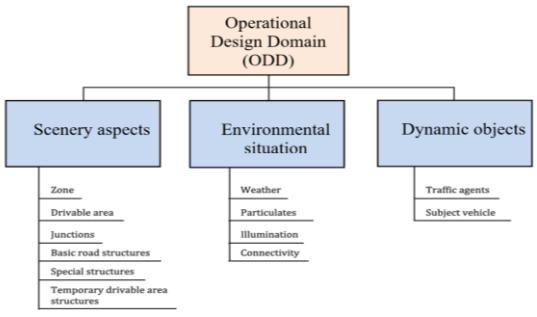


Figure 2-9 Top level taxonomy with ODD attributes [256]

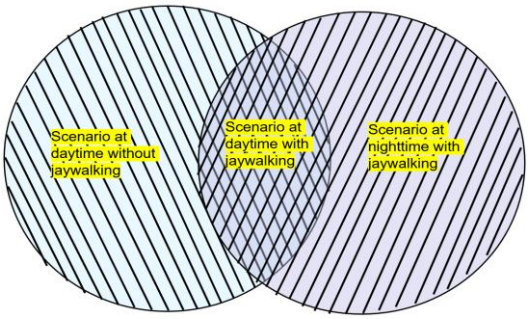


Figure 2-10 Relationship between scenario and scenario category

The difference between a scenario and a test case lies in the ODD, which defines the required test cases for safety assessment. ODD varies among vehicle types (e.g., passenger vehicle, buses, shuttles like NAVYA or WeRide), and test cases are generated based on

identified ODD and potential scenarios. Test cases are a subset of scenarios, defined based on real-world scenarios, and are not generated outside of the ODD. Figure 2-11 depicts the relationship between real-world data, scenarios, ODD, and test cases. Not all test scenarios are suitable for simulation and physical assessment as test cases. Test cases are designed based on specific characteristics necessary for assessment, such as test objectives, input data, procedures, identifiers, platform capabilities, and expected results. Scenario evaluation, based on ISO 21448 (SOTIF), uses "Known/Unknown" and "Safe/Unsafe" scenario categories as shown in Figure 2-12 to categorize scenarios for critical level evaluation. Test cases for autonomous vehicles are generated based on scenario evaluation results and testing capabilities.

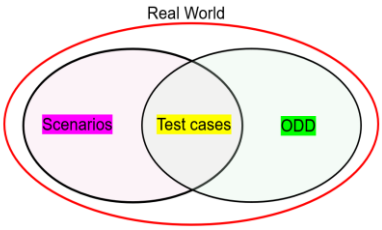


Figure 2-11 Relationship between Real World, scenarios, ODD and test cases

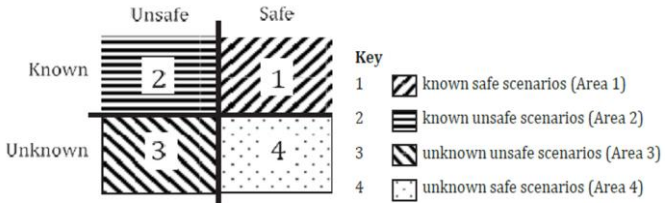


Figure 2-12 Illustration of Known/Unknown and Safe/Unsafe Scenario categories

The upcoming ISO 34505 Road Vehicles – Test scenarios for automated driving systems – Scenario Evaluation and Test Case Generation" [257], marks a significant step forward in standardizing AV testing, addressing the gap identified in earlier approaches. ISO 34505 adopts a two-pronged approach, focusing on both scenario evaluation and test case generation:

1. Scenario Evaluation: This stage assesses the effectiveness of various test scenarios designed to challenge an AV's capabilities. The standard outlines factors to consider during evaluation, such as:
    a. Relevance: Does the scenario represent a realistic situation an AV might encounter on the road?
    b. Comprehensiveness: Does the scenario adequately test various aspects of the AV's perception, decision-making, and control systems?
    c. Difficulty: Does the scenario pose a significant challenge to the AV, pushing its boundaries and identifying potential weaknesses?
    d. Measurability: Can the outcome of the test scenario be objectively evaluated to determine the AV's performance?
2. Test Case Generation: Once a scenario is deemed effective, ISO 34505 provides a framework for converting it into a concrete test case. This involves defining:
    a. Initial Conditions: The starting state of the AV and its surrounding environment, including weather, traffic, and road infrastructure.
    b. Stimuli: The specific events or actions that trigger the AV's response within the scenario, such as a car suddenly changing lanes or a pedestrian crossing the road.
    c. Expected Results: The desired outcome of the AV's behaviour in response to the stimuli within the scenario. This could involve successful avoidance of an obstacle, maintaining safe headway with other vehicles, or correctly interpreting traffic signals.

d. Pass/Fail Criteria: Clearly defined metrics for determining whether the AV's performance meets expectations within the test case.

Unlike adapting generic safety standards, ISO 34505 focuses specifically on AV testing needs by providing targeted testing. This framework ensures test scenarios and cases are directly relevant to the challenges that might encounter by autonomous vehicles in the real world. This also enable developers to ensure the AV is tested against scenarios that is likely encounter within its ODD. By establishing a structured approach for scenario evaluation and test case generation, ISO 34505 streamlines the testing process. This reduces redundancy and ensures developers focus on the most impactful test cases. Besides, the ISO 34505 also enhanced comparability. This is because standardized test cases allow consistent evaluation of AV performance across different developers and platforms. This facilitates benchmarking and promotes progress in the field. The link between test scenarios and derived test cases also made explicit in ISO 34505. This traceability helps developers understand the rationale behind each test and identify the aspects of the AV system being evaluated. The connectivity between ISO standards, as illustrated in Figure 2-13, guides the safety assessment methodology for the validation and verification framework.

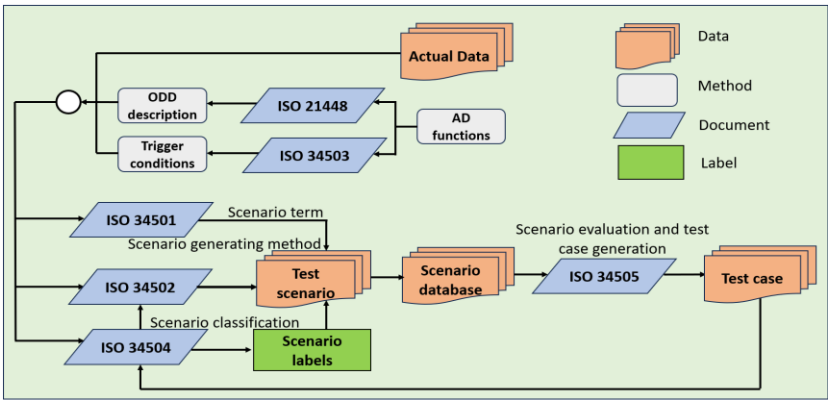


Figure 2-13 Relationship of ISO 3450X series standards

(b) New Car Assessment Programme (NCAP)

In addition to these standards, several New Car Assessment Programs (NCAPs) worldwide play crucial roles in assessing vehicle safety, including Euro NCAP [258], Global NCAP, Association of Southeast Asian Nations (ASEAN) NCAP, United States (US) NCAP, and China C-NCAP. The New Car Assessment Programme (NCAP) is a critical element in ensuring vehicle safety, offering a standardized approach to evaluating the safety features of new vehicles. Among the various NCAPs worldwide, Euro NCAP stands as a prominent entity, established with the support of several European countries and organizations. It provides consumers with safety assessments through rigorous testing procedures, thus guiding informed purchasing decisions. Global NCAP, another significant body, operates under the Towards Zero Foundation and aims to improve vehicle safety globally, often focusing on democratizing vehicle safety in support of UN Global Goals. ASEAN NCAP, specifically tailored for Southeast Asian countries, was established by the Malaysian Institute of Road Safety Research (MIROS) and Global NCAP. It emphasizes raising consumer awareness and enhancing vehicle

safety standards within the region. The US counterpart, the National Highway Traffic Safety Administration (NHTSA)'s NCAP, known for its 5-Star Safety Ratings, helps consumers compare safety features more effectively and encourages manufacturers to incorporate advanced safety technologies. Japan's JNCAP, initiated by the Ministry of Land, Infrastructure, Transport and Tourism, along with the National Agency for Automotive Safety & Victims' Aid, informs consumers about safer vehicles and promotes the development and popularization of these vehicles. These programs collectively contribute to a global effort to improve vehicle safety, with each NCAP addressing specific regional needs and regulations, yet all aiming towards the common goal of reducing vehicular accidents and enhancing passenger safety.

In the context of autonomous vehicles (AVs), the role of NCAPs is increasingly significant. AVs introduce a new paradigm in driving, where advanced driver-assistance systems (ADAS) take on tasks traditionally performed by human drivers. NCAPs assess the effectiveness of these systems in preventing and mitigating accidents, which is crucial as the industry moves towards higher levels of automation. The evaluation criteria include the performance of automatic emergency braking, lane-keeping assistance, and other proactive safety features that can dramatically reduce the likelihood of collisions. As AV technology continues to evolve, the NCAPs' testing methodologies are also adapting. The inclusion of ADAS in safety assessments reflects the changing landscape of vehicle safety, where the focus is shifting from passive safety features like airbags and crumple zones to active systems that can prevent accidents from occurring in the first place. This evolution in safety standards underscores the importance of NCAPs in guiding both consumers and manufacturers towards safer automotive technologies and practices. The comprehensive approach of NCAPs to vehicle safety, encompassing both traditional and autonomous vehicles, demonstrates their indispensable role in the automotive industry. By providing transparent and rigorous assessments, NCAPs empower consumers with the knowledge to make safety a priority when purchasing vehicles. Moreover, they drive innovation among manufacturers, pushing the industry towards continuous improvement in vehicle safety features. As vehicles become more connected and autonomous, the NCAPs' assessments will remain a key factor in shaping the future of safe mobility on a global scale.

(c) International Standards for Vehicle Dynamics and Safety Testing of Vehicle

The validation of instrumentation for data collection and autonomous vehicle (AV) testing is also a critical aspect of ensuring the safety, reliability, and performance of modern vehicles. This section explores the international standards that govern these processes to provide a framework for evaluating vehicle dynamics, safety, and performance under various conditions, which is essential for the development and deployment of autonomous vehicles. International standards for vehicle dynamics and safety testing like the ISO 3888-1:2018 - Double Lane Change Test specifies the dimensions and procedures for a double lane-change test, which evaluates a vehicle's ability to perform severe lane-change manoeuvres. This test is crucial for assessing the vehicle's stability, handling, and safety under extreme conditions. The standard applies to passenger cars and light commercial vehicles with a gross vehicle mass of up to 3.5 tons. It defines the test track layout, including the dimensions of the lanes and the required vehicle speeds, ensuring consistent and repeatable testing conditions [259].

Besides, SAE provided guidelines for testing a vehicle's braking and acceleration performance. These guidelines are critical for assessing the vehicle's ability to stop safely and accelerate efficiently under various conditions. It includes detailed procedures for measuring braking distance, deceleration rates, and acceleration times, ensuring that vehicles meet safety and performance benchmarks [260]. Next, ISO 4138:2021 - Step Steer Test outlines the methodology for conducting step steer tests, which measure a vehicle's lateral transient response. This test is essential for evaluating the vehicle's handling characteristics, particularly its response to sudden steering inputs. The standard specifies the test conditions, including vehicle speed, steering angle, and data collection requirements, to ensure accurate and reliable results [261]. The ISO 7401:2011 - Lateral Transient Response Test then complements ISO 4138 by providing additional methods for evaluating a vehicle's lateral transient response. This standard focuses on the vehicle's behaviour during sudden lane changes or obstacle avoidance manoeuvres, which are critical for autonomous vehicle testing. It specifies the test conditions and data collection methods to ensure consistency across different testing environments [262].

In addition, the ISO 15037-1:2019 addresses the general conditions necessary for testing of suspension systems and vehicle components. This standard is particularly relevant for autonomous vehicles, as it ensures that the suspension system can handle dynamic loads and maintain stability under various driving conditions. It provides guidelines for measuring the dynamic behaviour of suspension systems, including ride comfort and handling performance [263]. Moreover, ISO 22737:2021 - Low-Speed Automated Driving Systems provides specific guidelines for testing low-speed automated driving systems, such as those used in urban environments. This standard is particularly relevant for validating the performance of AVs in complex, low-speed scenarios, such as pedestrian crossings and traffic congestion [264].

Research in AV safety testing adopts a comprehensive approach, integrating simulations, empirical testing in real-world conditions, and scenario-based evaluations [265]. These tests yield critical insights into the AV’s interaction dynamics with various road users, diverse weather conditions, and unforeseen events. A thorough safety assurance framework is essential for AV deployment, encompassing meticulous system design, rigorous requirements analysis, detailed documentation, and robust verification traceability. Such a framework ensures that AVs are deployed successfully and safely. The process of verification and validation is fundamental to the development of AVs, ensuring their reliability and security. Adhering to established standards such as ISO 3888, ISO 4138, ISO/TR 4804:2020, ISO 26262, and ISO 34505, and integrating NCAP evaluations, constitutes a solid foundation for a comprehensive safety assurance strategy. As AV technology evolves, it is imperative to continually revise and enhance these standards and testing methodologies. This ongoing process will address the evolving challenges associated with autonomous driving systems, ensuring that safety remains at the forefront of AV development and deployment.

2.10 Application of Digital Twin in Autonomous Vehicle Safety Testing

The concept of Digital Twin (DT) was first proposed by Grieves in 2003 but initially did not gain attention [266]. However, advancements in information technologies and hardware such as sensors, computers, artificial intelligence, and big data have made digital twins feasible. NASA reintroduced the concept in 2012 [267], leading to increased interest and investment in digital twin research by universities and enterprises. From 2017 to 2019, DTs were listed among Gartner's top ten strategic technology trends, highlighting their potential to integrate physical and virtual spaces effectively for intelligent manufacturing [268]. Modelling methods are crucial in implementing digital twins, and various scholars have made significant contributions in this area. A digital twin framework consisting of virtual space, real space, and the connections to allow the flow of information and data between these spaces was introduced by Grieves et al. [269]. A five-dimension DT model was proposed Tao et al., encompassing virtual entity, real-world entity, service system, connection network, and digital twin data. Physics, Geometry, behaviour, and rule models are included in the virtual entity [270]. The digital twin concept and its implementation across different stages of lifecycle of a product is investigated by Zhuang et al. [271]. A reference DT model comprising cyber things, real-world things, and hybrid things is presented by Alam et al., providing foundational structures for DT systems [272].

Recently, DTs have found applications across diverse fields such as medical care [273], machinery manufacturing [274], aerospace [275], shipbuilding [276], and urban construction [277]. The automotive industry, in particular, has emerged as a significant application area for digital twins, attracting considerable scholarly attention and research efforts. In the context of autonomous vehicles, simulation-based DTs play an important role in speeding up AV verification and reducing development costs [266]. These virtual replicas can simulate a wide range of traffic and environmental scenarios, eliminating the needs for massive real-world testing. For instance, waiting months for major snowstorm for on-road AVs testing can be bypassed by creating a digital twin that simulates a major snowstorm and generates high-quality testing data [278]. Vehicle dynamic simulators, such as those for testing cruise control systems, have been extensively utilized in the automotive industry for testing purposes [279]. Similarly, the aerospace industry has long employed simulation for various applications. Testing AVs in controllable virtual environments can significantly reduce time and costs of development. Simulators are widely utilised to evaluate the path-planning and decision-making modules across different scenarios. These simulators can provide perception data including the states and positions of the ego vehicle and other traffic actors[280]. This method is scalable and straightforward but lacks the fidelity to accurately represent real-world conditions, leading to several issues.

Simulation tests cannot fully substitute for physical tests of the pipeline of the autonomous vehicle software, which includes sensing, localization, path planning, vehicle control, and decision-making. Physical tests are also constrained by environmental factors such as weather and lighting conditions. Simulators often utilize virtual town maps rather than real-world road tests, and these do not accurately simulate road conditions [281]. To evaluate

Autonomous Driving (AD) functions such as highway ramps, which rely on traffic regulations and road geometry, a digital twin map is essential. Furthermore, pre-programmed car and pedestrian animations in simulations cannot mimic the complex interactions of real traffic, making it challenging to judge scenarios like aggressive driving and junctions. AV software testing in simulations often faces challenges with low-fidelity sensor data. Techniques such as ray casting and depth mapping, which simulate lidar sensors, struggle to accurately replicate real-world phenomena such as reflection and diffusion. These discrepancies underscore the disparities between simulations and real-world conditions. In contrast to AV software development, automotive hardware development benefits from "digital twin" physical simulation tools like Modelica and MATLAB, which expedite the development process [282].

Besides, in the automotive industry, DTs are employed to generate digital replicas of vehicles, facilitating personalized service and maintenance based on data on vehicle use and performance. These digital replicas can be either model copies or networked systems [283]. Engineers utilize simulations to explore AI and predict breakdowns and wear before the car reaches the assembly line. This approach can significantly reduce the need for extensive road testing and maintenance, thereby saving unforeseen costs. DT technology holds the potential to replicate and enhance various aspects of the system of a smart electric vehicle (EV), which could have significant implications [284]. Effective DT vehicle modelling necessitates a comprehensive understanding of the DT environment. Automakers use DTs to generate accurate digital replicas of automobiles, allowing for more targeted service and maintenance based on car performance data. By studying AI through simulations, engineers can predict breakdowns and wear, thereby reducing the need for maintenance and physical road testing for autonomous vehicles. DT technology has the capability to effectively replicate and enhance the systems of smart EVs. Consequently, successful DT vehicle modelling hinges on a thorough understanding of the DT environment.

In Yu et al.’s paper [285], they shared real-life experiences of DTs as a practical approach for developing autonomous driving systems. They highlighted how digital twins create detailed, precise, and reliable models of the real-world environments, thereby reducing the need for physical testing. The primary contributions include the identification of limitations inherent in conventional AD simulation approaches and the demonstration of how digital twins can address these challenges. The authors summarized their experience in designing the autonomous driving DT system into three fundamental principles. They elaborated on the system’s architecture and components, which include gathering real-world mapping data, replicating sensor data, and synthesizing traffic participants. The contributions of the paper by Dong et al. [286] include detailing the AD DT system's structure and components, the creation of a working prototype under the digital twin paradigm, and the tracking of critical metrics such as time lag and localization precision. The system integrates Human-Machine Interface (HMI) devices, utilizing the HoloLens for controlling vehicles within a 3D environment, while the driving simulators with first-person perspective offer complete driver control.

Moreover, Niaz et al. [287] examines the origins and deployment phases of digital twin technology, highlighting applications such as autonomous motion control, predictive mobility,

ADAS, battery management systems, vehicle health management systems, intelligent charging, electric power drive systems, and vehicle power electronic converters. Steinmetz et al. [288] contribute to the literature by proposing the use of digital twins (DT) to enable Car-as-a-Service (CaaS) in densely populated locations with significant traffic due to commercial expansions. The proposed design consists of several components: the city infrastructure, middleware for entity connectivity, DT models operating on the middleware, and applications such as car-sharing. They demonstrated the implementation of this concept and highlighted key areas for future development. Meanwhile, Tsinarakis et al. [289] demonstrate that the entire electric car development process can be simulated using a Petri-net-based digital twin. This digital twin enables real-time data sharing between the physical system and its digital counterpart, leading to improved and quicker decision-making. The two systems collaborate to calculate and implement actions, generate updated schedules, and provide users with various scenarios (optimistic, most likely, pessimistic) regarding potential task delays.

In a recent study, a Vehicle-to-Cloud (V2C) communicating linked car digital twin structure was proposed [290]. This structure integrates the Driver-Vehicle Interface (DVI) of the vehicle with cloud server data, enabling the display of advisory speed information to the driver for vehicle control. This digital twin framework is designed to enhance Advanced Driver Assistance Systems (ADAS) by leveraging the digital twin concept. The proposed structure underwent testing in real-world scenarios focused on cooperative ramp merging which involved three passenger vehicles. Meanwhile, Tsinghua University is working on a digital twin system focused on Connected and Autonomous Vehicles (CAVs), allowing for multi-vehicle tests even without physical cars [291]. This approach combines physical and virtual automobiles to achieve desired outcomes. A sand table testbed facilitates the smooth operation of small cars, while a game engine creates a virtual environment through full-element modelling. This virtual space can display the real-time state of the sand table. The research suggests using a cloud vehicle to replace smaller autos.

However, there are limitations to simulation-based testing. Five major challenges highlighted in implementing DT technology across various fields are:

1. **Data-related concerns**: This includes concerns regarding privacy, trust, cybersecurity, acquisition, governance, convergence, and large-scale analysis. Some behaviours, like socioeconomic inequality, ecological stability, and political instability, are difficult to quantify, making it challenging for designers to replicate them accurately [292]. Addressing these concerns are crucial, especially in the early stages of DT development, to understand their potential impact on stakeholders and the environment.
2. **Lack in implementation of guidelines, standards, and regulations**: The lack of universally accepted interoperability and standards in industries, such as manufacturing, limits the widespread adoption of DTs [293]. Establishing clear guidelines and standards through articles and research can help define the benefits and structures of DTs, facilitating their adoption.
3. **High implementation costs**: Overcoming the challenge of high implementation costs, which arise from the requirement for additional sensors and computing power, is crucial for advancing DTs to higher maturity levels. This advancement is essential for

integrating DTs into commercial designs and achieving widespread deployment, especially in less affluent regions where resources are scarce [294].

4. **Utilizing big data and AI for large-scale and long-term analysis**: Big data algorithms and Internet of Things (IoT) technology play a crucial role in successful DT implementations, as they help generate and analyse vast amounts of data [295]. Effectively leveraging these technologies can enhance data connectivity and processing, enabling bidirectional flow of information and autonomous operations.
5. **High speed connectivity**: The adoption of advanced communication standards, such as 5G, plays a crucial role in facilitating real-time data connectivity and enhancing the operational efficiency of DTs [296]. The benefits of 5G technology, including high-speed connectivity, ultra-low power consumption, and improved reliability, are pivotal for ensuring the smooth operation of DTs, particularly in applications like smart cities.

Addressing these challenges is vital for the successful implementation and widespread adoption of DTs across various industries.

In summary, despite facing several challenges, digital twins offer significant benefits. In the automotive industry, they facilitate the creation of digital vehicle copies, enabling personalized service and maintenance. Engineers can predict breakdowns and wear using simulation models, potentially saving unforeseen costs. Moreover, digital twin technology can enhance smart electric vehicle systems, impacting sustainability and efficiency. The integration of Digital Twin Technology (DTT) with real-world data involves a systematic approach encompassing data collection, processing, and simulation. By leveraging real-time sensor data, external data sources, AI algorithms, and robust connectivity, DTT creates highly accurate and dynamic scenarios for autonomous vehicle systems. These scenarios are invaluable for testing, validating, and optimizing autonomous vehicle performance, ensuring these vehicles can operate safely and efficiently in the real world. As DTT continues to advance, its ability to integrate with real-world data will further enhance, improving the fidelity and utility of the scenarios it can produce for autonomous vehicle systems. Therefore, to fully leverage digital twin capabilities, understanding the digital twin environment is crucial for accurate vehicle modelling and optimization.

2.11 Vehicle Modelling

The dynamic modelling of vehicles is a cornerstone in the design and control of autonomous driving systems. A robust understanding of vehicle dynamics is essential not only for evaluating ride comfort and handling performance but also for developing control strategies that ensure safety and efficiency in autonomous vehicles. In the literature, a wide range of vehicle dynamic models have been developed, each characterized by its DOF and tailored to different research objectives. The DOF in a dynamic model represent the independent motions that the vehicle or its subsystems can undergo. For example, a simple two-DOF model captures the vertical translation motions of both the sprung and unsprung masses, whereas more sophisticated models allow for rotational movements such as pitch and roll.

A quarter car model is one of the most elementary representations of vehicle dynamics and is typically formulated as a two-DOF system. In this model, one DOF represents the vertical motion (or bounce) of the sprung mass—a quarter of the vehicle's body mass—while the second DOF captures the vertical motion of the unsprung mass (the wheel and tire assembly). Owing to its simplicity and low computational burden, the quarter car model has been widely adopted in early-stage suspension design studies and ride comfort analyses. For example, studies comparing the vibration attenuation of quarter, half, and full car models demonstrate that the quarter car formulation can effectively capture the essential vertical dynamics while sacrificing the ability to represent lateral or rotational effects [297]. However, the quarter car model does not capture the effects of vehicle pitching or rolling, and hence its scope is limited when predicting full-vehicle behaviour in more complex scenarios such as those encountered in autonomous driving.

To overcome these limitations, researchers developed the half car model, which typically incorporates four DOF. This model considers two unsprung masses—one at the front and one at the rear—and introduces an additional DOF to represent the pitch motion of the chassis. By capturing the rotation about the lateral axis, the half car model enables the simulation of front–rear load transfer, which is crucial for assessing both ride comfort and road-holding performance under various road excitations [297]. A further increase in modeling fidelity is achieved with full car models that incorporate seven DOF. In such models, the vehicle body is generally allowed three independent motions: vertical translation (bounce), pitch (rotation about the lateral axis), and roll (rotation about the longitudinal axis). In addition, each of the four wheels is modelled with one DOF, usually capturing its vertical displacement. The seven-DOF full car model offers a significantly enhanced representation of the vehicle's dynamic response because it captures the interactions between chassis rotational dynamics and the individual suspension responses. Research using these models has provided valuable insights into active suspension design and control strategies that improve both ride comfort and stability [298], [299].

Even more detailed are the 14 DOF models, which include additional DOF to capture the rotational dynamics of the wheels and other localized motions such as lateral deformations in suspension components. In these models, besides the three DOF for the chassis and the four for the vertical motions of the unsprung masses, extra states are added to represent, for example, wheel rotational inertia and tire deformation effects. Although such detailed formulations promise to simulate subtle interactions—like the coupling between tire deformation and vehicle stability—the literature indicates that practical applications of 14 DOF models in autonomous vehicle control are sparse. The challenges here include extensive parameter identification, increased computational complexity, and difficulties in real-time implementation [300].

Beyond these vertical dynamics models, other vehicle models have been developed to address specific research needs. Kinematic models, such as the single-track or "bicycle" model, are commonly used for path planning and trajectory tracking tasks because they efficiently represent lateral and longitudinal motions without delving into the complexities of vertical dynamics. These models, while simpler, are particularly useful when the primary interest is in

planar motion [299]. At the opposite end of the spectrum, highly detailed multibody dynamic models may include 38 DOF or even exceed 100 DOF, capturing interactions among the chassis, suspension, powertrain, steering system, and even human body dynamics. Such models, often implemented in advanced simulation environments, offer unparalleled fidelity and are typically used for offline analyses or for validating new control strategies under extreme conditions [301].

In addition, there are specialized models that integrate active control elements. For instance, recent work on active suspension systems collaborating with an active aerodynamic surface in a quarter car model illustrates how additional forces can be integrated into the state-space formulation to improve ride comfort and road-holding performance [302]. Collectively, these models form a hierarchy in which the selection of a vehicle dynamic model depends on the intended application. Simpler models (quarter and half car) are advantageous for preliminary control design and real-time applications, while higher-fidelity models (7 DOF and 14 DOF) are necessary for a comprehensive understanding of vehicle behaviour under a wide range of driving conditions. However, a notable gap in the literature remains: the integration of 14 DOF models into real-time autonomous vehicle control frameworks is still underdeveloped due to the challenges of computational load and parameter calibration. Future research is expected to address these issues through advanced model order reduction techniques and robust parameter estimation methods, leading to improvements in both safety and performance for autonomous vehicles.

Tire modelling is another important element in vehicle dynamics modelling, which has been a topic of intensive research for decades, as it provides the crucial link between vehicle dynamics and road behaviour. Early models sought to capture the basic physical phenomena governing tire's road interaction, while more modern formulations focus on achieving a balance between computational efficiency and accuracy. The evolution of tire models reflects this balance, ranging from the simple analytical models based on physical principles to complex empirical formulations that are fitted to experimental data.

One of the most influential models is the Magic Formula tire model, developed by Hans B. Pacejka in the 1980s. Despite its lack of a strict physical foundation, the Magic Formula gained widespread acceptance due to its ability to accurately replicate measured tire behavior over a wide range of operating conditions. The model employs a sine-arctan formulation to represent tire forces as functions of slip parameters, incorporating numerous coefficients that account for effects such as vertical load, camber, and combined slip [303]. Its flexibility and relatively low computational cost have made it a standard in vehicle dynamics simulations and control applications, even though its empirical nature means that its parameters must be carefully determined from experimental data. Researchers have noted that while the Magic Formula is highly accurate within the range of data used for calibration, its extrapolation beyond these conditions can be problematic.

In contrast to the Magic Formula, semi-empirical models like the Fiala tire model and the TMeasy model incorporate a more direct physical interpretation of the tire's behaviour. The Fiala model, dating back to the 1950s, conceptualizes the tire as a deformable structure where

a “brush” of elements represents the contact patch. This model divides the behaviour into regions of pure adhesion and partial sliding, offering insights into the physical mechanisms behind force generation. Although it uses a limited number of parameters and is relatively simple to implement, the Fiala model’s validity is often restricted to small slip conditions, which limits its usefulness in aggressive manoeuvres or when significant nonlinearities are present [304].

The TMeasy model represents another effort to bridge the gap between purely empirical and physically based approaches. It uses analytical approximations of tire force curves while keeping the number of parameters manageable. These parameters are typically identified through laboratory experiments, allowing the model to capture both pure slip and combined slip characteristics with reasonable accuracy. However, the TMeasy model often struggles with predicting aligning torque and exhibits limitations when extrapolating to conditions outside those for which it was calibrated. Despite these drawbacks, its computational efficiency and physical interpretability make it attractive for applications such as vehicle control design, where a balance between simplicity and fidelity is essential [305].

Physically based models, particularly the brush tire models, take a more fundamental approach by representing the tire-road interface as an array of bristles that deform under load. These models are grounded in mechanics, with parameters such as the tire’s material properties and geometric characteristics entering directly into the equations. Brush models can often provide better extrapolation under varying operating conditions because their parameters have clear physical meanings. However, they are not without limitations; many brush models assume simplified contact patch shapes and constant friction coefficients, which can lead to inaccuracies during transient or high-slip events. Recent extensions to brush models, including those that incorporate aspects of the LuGre friction model, have aimed to capture stick-slip phenomena and frictional hysteresis, thereby improving the predictive capability of these models. Yet, the increased complexity associated with these extensions often results in higher computational costs and more challenging parameter identification processes [306].

Comparative studies in the literature have shown that while the Magic Formula tends to offer the best fit to steady-state experimental data, its “black box” nature and the large number of parameters can be drawbacks, especially when extending the model to account for environmental variations such as temperature, tire wear, or inflation pressure. Semi-empirical models like TMeasy, by using fewer parameters, offer a more physically intuitive approach, but they may not capture all the nuances of tire behaviour under extreme conditions. Physically based models, including both traditional brush models and their modern extensions, excel in providing robustness and better extrapolation capabilities, though often at the cost of increased computational effort and complexity.

In recent years, advances in computational power have enabled the use of more complex physical models in real-time simulations. For example, models such as FTire have been adapted for real-time applications, demonstrating that even highly detailed physics-based models can be used in simulations if computational efficiency is properly managed. This trend suggests that future tire models may increasingly blend the high accuracy of empirical

approaches with the robustness and physical interpretability of analytical models, offering improved predictions across a broader range of operating conditions [307].

In summary, the literature on vehicle dynamics and tire modelling reveals a rich history of development, marked by a continual trade-off between accuracy, computational efficiency, and physical insight. For vehicle dynamics modelling, a clear understanding of the DOF involved is essential to capture the nuances of vehicle behaviour, particularly as autonomous driving systems demand increasingly sophisticated control solutions. Although simplified models continue to be valuable for initial studies and control design, the development and integration of more detailed models, such as the 14 DOF full car model, represent critical research directions for future advancements in autonomous vehicle technology. Whereas for tire modelling, the Magic Formula model remains a benchmark for many applications, while semi-empirical and physically based models offer valuable alternatives that may be better suited to applications requiring extrapolation to extreme conditions. As measurement techniques improve and computational resources continue to grow, future research is likely to focus on hybrid models that combine the best features of these approaches, leading to more reliable and versatile tools for vehicle dynamics analysis and control.

2.12 Research Gap

The literature review highlights a significant gap in the development of autonomous vehicle simulation platforms and driver models that cater to the road environments and driving behaviours specific to developing countries. Existing simulators and datasets mostly focus on urban or rural environments in developed nations, necessitating extensive modifications for suitability in training autonomous vehicles for developing country road environments. In Malaysia, there's a need of available simulator platforms and datasets tailored to testing autonomous vehicle safety systems in the local context. Moreover, virtual simulation platforms often overlook the interaction between human drivers and other road users, a critical aspect in developing countries where traffic is less organized, pedestrians might crossroads unpredictably, and road markings may be poorly maintained. These factors significantly influence driving behaviour and must be considered in developing autonomous vehicle simulation platforms.

Meanwhile, earlier standards like ISO 21488 and ISO 26262 provided foundational frameworks for assessing the reliability and safety of advanced systems, focusing on functional safety and managing potential risks associated with electronic and electrical systems within vehicles. These standards have been instrumental in guiding the industry through its nascent stages. However, as technology progresses, the need for updated and more comprehensive standards becomes apparent. The BSI Flex 1889 and the ISO 3450X series represent a new wave of standards aimed at addressing the complexities of Operational Design Domains (ODD) and specific scenarios that autonomous vehicles may encounter. The absence of literature developing scenario-based testing for ODDs in line with these newer standards indicates a significant research gap. This gap presents an opportunity for scholars and industry experts to pioneer methods that will ensure the safe integration of autonomous vehicles into transportation systems, considering the diverse and dynamic environments they will operate in. Bridging this

gap is crucial for advancing the field and maintaining public trust in these emerging technologies. Developing literature and practical testing methodologies based on BSI Flex 1889 and the ISO 3450X series would mark a significant step forward. Such efforts would potentially set new benchmarks for the safety and reliability of autonomous vehicle technology, ensuring that these vehicles can perform safely and effectively in all anticipated operational conditions.

Furthermore, existing driver models based on mathematical formulations or traditional AI algorithms like GMM and HMM typically focus on limited driving scenarios and lack the ability to consider previous decisions, actions, or interactions of other road users. For instance, end-to-end driver models using CNN, such as PilotNet, may identify a pedestrian but struggle to predict their movement speed or direction, hindering accurate collision avoidance manoeuvres during emergency braking. To address these limitations, employing recurrent neural networks (RNNs) with feedback loops enables driver models to retain information from previous states, allowing them to estimate the intentions of other road users more effectively. Combining CNNs with recurrent networks in hybrid architectures offers promise by processing time series data while maintaining an end-to-end design.

Another avenue involves training driver models through reinforcement learning in interactive simulator environments. However, while suitable for optimizing driving policies, this method may not effectively replicate human driver actions. Inverse reinforcement learning, although valuable in learning from human expert demonstrations, relies on the assumption that the learned reward function is optimal. Additionally, the use of simplified vehicle models in MPC may limit the accuracy of vehicle motion representation in simulators. Developing driver models for safety testing simulators differs from those for self-driving vehicles. While the former aims to reproduce human driver anomalies or errors, most models focus on perfect driving skills. There's a need of research on driver models combining both lateral and longitudinal vehicle control through end-to-end deep learning methods.

This endeavour seeks to address critical gaps by creating a novel end-to-end CNN+LSTM-based driver model capable of emulating human driving behaviour in developed countries with the ability of path following. It involves simulating diverse and realistic Malaysian road scenarios within a 3D environment to capture human driver actions. By incorporating an intricate vehicle model into imitation learning process, this approach aims to improve the precision of predicting vehicle responses.

Table -2-5 Shortcomings of the existing CNN-LSTM network

Paper	Application	Shortcomings
End-to-end Learning of Driving Models from Large-scale Video Datasets [208]	Driver Model	• The model primarily relies on visual data, which might not be sufficient for complex driving tasks. The lack of additional inputs such as trajectory waypoints, GPS data, and high-definition maps can limit the model's ability to perform precise path following and navigation tasks. • The model is trained on a corpus of demonstrated behaviour, which may not cover all driving scenarios. This limitation means the model might struggle with rare or unexpected events not represented in the training data.

		• The learned policy is evaluated based on its ability to predict future actions in a held-out dataset but is never executed in a real-world or simulated environment. This lack of real-world validation raises questions about the model's robustness and reliability in practical applications • The approach combines a fully convolutional network (FCN) with a long short-term memory (LSTM) network for sequential image data. While effective, this method can suffer from information loss if the time-series data is large and complex. The integration of attention mechanisms could address this issue by highlighting relevant features, but this is not fully explored in the paper
Lane detection and localization using hybrid deep neural network model [209]	Perception	• Used CNN and ConvLSTM for lane detection and car localization, which is effective for perception but lacks integration with real-time control and decision-making.
Spatio-Temporal Image Representation and Deep-Learning-Based Decision Framework for Automated Vehicles [211]	Decision Making	• The model relies heavily on simulated data and lacks real-world testing to validate the model's effectiveness and safety • The model is limited to 2D planar and specific manoeuvres, lacking consideration for real-world road users and traffic objects.
Controlling steering angle for cooperative self-driving vehicles utilizing CNN and LSTM-based deep networks [216]	Driver Model	• The model only controls steering angle • The model relies heavily on image data shared via V2V communication, which may not always be reliable or available. • The model's performance may vary with changes in vehicle speed or distance, affecting the relevance of shared images. • The model's ability to generalize to different environments or vehicle types is not discussed.
End-to-end Deep Learning for Steering Autonomous Vehicles Considering Temporal Dependencies [217]	Driver Model	• The model only controls steering angle • The model relies on the front camera images only • Traditional CNN-based methods do not account for the temporal relationship between frames, missing motion features.
A CNN-LSTM car-following model considering generalization ability [219]	Driver Model	• The model only controls vehicle speed • The model is limited to 2D planar and specific manoeuvres, lacking consideration for real-world road users and traffic objects
Comparison of machine learning algorithms on self-driving car navigation using Nvidia Jetson Nano [220]	Driver Model	• The accuracy of model decreases with the addition of more obstacles and higher speed levels. • The model control steering only • Limited to specific scenarios and speed levels.
End-to-end multi-modal multi-task vehicle control for self-driving cars with visual perceptions [221]	Driver Model	• Multi-modal multi-task network to handle both steering and speed control. • The model predicts discrete speed commands (accelerating, decelerating, maintaining speed) with pre-fixed levels, which can limit the smoothness of vehicle control. • Using only visual inputs restricts the accuracy of command predictions, especially in scenarios where the vehicle's current speed or road conditions should alter the command decision.
An LSTM-based autonomous driving model using a waymo open dataset [222]	Driver Model	• End-to-end single model network to handle both steering and speed control. • The framework faces challenges in manual feature combination due to feature explosion, difficulty in design, and recognition of combined features.

		• The model is trained on front-camera images from Waymo cars and may not generalize well to other car models with different camera positions • The model requires a specific input length (10 previous frames), which may not capture sufficient history for accurate predictions or could be unnecessarily complex. • The current training and testing approach don't use predictions from previous frames as input, which differs from real-world scenarios where predictions would build on each other.
A Deep Learning Framework for Driving Behaviour Identification on In-Vehicle CAN-BUS Sensor Data [223]	Driver model	• Modelling the temporal dynamics of feature activations explicitly is complex. • The high-dimensional nature of CAN-BUS data leads to a large feature space, increasing the complexity of model training. • The framework needs further validation on large-scale Naturalistic Driving Studies (NDS) datasets to ensure its effectiveness in real-world scenarios.
End-to-end autonomous driving decision model joined by attention mechanism and spatiotemporal features [226]	Driver model	• The model has limited interpretability compared to rule-based methods, making it challenging to understand the decision-making process. • The model's performance in emergencies and its response ability are not fully proven in real-world scenarios due to the lack of depth information and reliance on virtual datasets.

In summary, the research gap in driver models using deep learning lies in the need for models that can generalize to diverse driving scenarios, adapt in real-time, incorporate contextual information effectively, ensure interpretability and safety, and transfer effectively from simulation to real-world environments. Therefore, the focus of this study will be on addressing these gaps by developing a driver model that leverages deep learning techniques while considering these critical aspects.

Another avenue involves training driver models through reinforcement learning in interactive simulator environments. However, while suitable for optimizing driving policies, this method may not effectively replicate human driver actions. Inverse reinforcement learning, although valuable in learning from human expert demonstrations, relies on the assumption that the learned reward function is optimal. Additionally, the use of simplified vehicle models in MPC may limit the accuracy of vehicle motion representation in simulators. Developing driver models for safety testing simulators differs from those for self-driving vehicles. While the former aims to reproduce human driver anomalies or errors, most models focus on perfect driving skills. There's a need of research on driver models combining both lateral and longitudinal vehicle control through end-to-end deep learning methods.

It involves simulating diverse and realistic Malaysian road scenarios within a 3D environment to capture human driver actions. By incorporating an intricate vehicle model into imitation learning process, this approach aims to improve the precision of predicting vehicle responses. The goal is to generate commands for both lateral and longitudinal vehicle motions, effectively closing the disparity in replicating average human driving behaviour in developing countries. Such an approach is pivotal for rigorously testing autonomous vehicle safety systems.

2.13 Summary

The literature review conducted in this chapter identifies key gaps in current driver modelling approaches and safety assessment frameworks for autonomous vehicles. While existing simulation-based validation methods provide a controlled environment for testing, they often fail to represent real-world driving conditions, especially in the context of developing countries like Malaysia. Most driver models have been developed based on western driving datasets, such as KITTI and nuScenes, which do not adequately capture heterogeneous traffic conditions, mixed road users, and unstructured driving patterns in developing regions.

Furthermore, the integration of Operational Design Domain (ODD) frameworks in autonomous vehicle (AV) testing is still in its early stages. ISO 34502 provides a scenario-based safety evaluation framework that outlines the process of identifying, designing, and validating test scenarios based on ODD attributes, such as road environment, weather, and dynamic objects. However, limited research has systematically integrated these standards into driver modelling frameworks. This study seeks to bridge this gap by incorporating ISO 34502 and ISO 34503 to define a structured methodology for driver behaviour prediction under realistic ODD conditions.

Existing driver models are generally categorized into rule-based approaches (e.g., Gipps' car-following model, optimal control-based models) and data-driven deep learning models (e.g., CNN-LSTM architectures). Rule-based approaches perform well in structured environments, but they lack adaptability to complex traffic scenarios. On the other hand, deep learning-based models, while achieving high predictive accuracy, often suffer from limited interpretability and poor domain generalization. This study addresses these limitations by developing a hybrid CNN-LSTM driver model trained on real-world Malaysian driving data, ensuring greater generalizability across different traffic conditions.

In addition, safety validation of driver models and AVs requires adherence to standardized ISO vehicle dynamics tests. ISO 3888-1 and ISO 3888-2 define double lane change manoeuvres, which are crucial for assessing a driver model's response to sudden obstacle avoidance scenarios. Similarly, ISO 4138 evaluates steady-state circular driving to assess a vehicle's understeer and oversteer characteristics. However, few studies have integrated these tests into deep learning-based driver models. This research incorporates these ISO test standards into the driver prediction framework, ensuring comprehensive safety validation in both virtual and real-world environments.

The scenario categorization methodology outlined in ISO 34504 introduces a systematic way to classify testing scenarios based on dynamic objects, environmental factors, and test objectives. This classification aligns with the risk-based safety assessment approach in ISO 26262 and ISO 21448, which focus on functional safety and safety of the intended functionality. By leveraging these standards, this study develops a structured test scenario framework, ensuring traceability and reproducibility of driver behavior evaluation under diverse driving conditions.

In conclusion, the present study addresses key gaps in driver model development and validation by:

- Developing a driver-in-the-loop simulator that integrates VR and 6-DOF motion feedback for improved realism.
- Proposing a hybrid CNN-LSTM driver model trained on real-world Malaysian traffic data, enhancing adaptability to local driving conditions.
- Incorporating international standards such as ISO 34502, ISO 26262, and ISO 21448 into the driver modelling framework, ensuring compliance with scenario-based safety evaluation methodologies.

By integrating deep learning-based driver models with ODD-based scenario classification and ISO safety standards, this research establishes a novel framework for driver prediction and safety validation, contributing to the advancement of autonomous vehicle testing methodologies in developing countries. Based on the following literature review and research gaps, the methodology of this study has been designed and addressed in the following sections of this thesis.

Chapter 3: Development of Vehicle Dynamics Model

3.1 Overview

The chapter introduces the mathematical formulation of the 14 Degrees of Freedom (DOF) vehicle dynamics model, which comprises two primary modules: the 7 DOF ride model and the 7 DOF handling model. This model enables comprehensive examination of lateral, longitudinal, vertical, and rotational vehicle motions driven by specific driving inputs. The inclusion of the vehicle dynamics model is critical in this study as it plays an essential role in integrating with the driver model. This integration is necessary to produce accurate vehicle responses based on the driving inputs generated by the driver model. Therefore, ensuring the vehicle model accurately replicates real vehicle responses is crucial for benchmarking the driver model against ground truth data.

This chapter is organized as follow: The first section provides an overview of the importance of development of vehicle dynamics model in this study. The next section presents the assumptions made during the process of modelling. In the third section, the modelling of 7 DOF vehicle ride model followed by the modelling of Pacejka tire model in the fourth section. The fifth section presents the development of the 7 DOF vehicle handling model. Sixth section discussed about the longitudinal and lateral slip model. The seventh section shows the vehicle kinematics model. The model verification is included in the eighth section using IPG CarMaker. The last section discusses the summary and conclusion of this chapter.

3.2 Modelling Assumptions

Several assumptions were made while developing the vehicle model to balance the model's fidelity with computational efficiency. Each assumption has been carefully chosen to capture the essential dynamics of the vehicle while ensuring that the simulation remains tractable for real-time analysis and integration with driver modelling frameworks. First, the vehicle chassis (sprung mass) is assumed to behave as a rigid body with 4 wheels (unsprung mass) attached to it at the corner. This assumption simplifies the dynamic equations by neglecting high-frequency flexible modes, which are not significant for the range of frequencies encountered in normal driving manoeuvres. The rigid body assumption is common in vehicle dynamics modelling because it permits the focus to be placed on the gross motion such as longitudinal, lateral, and yaw motions without the additional complexity of structural deformations [303], [308]. This level of abstraction is particularly justified when the primary objective is to assess driver behaviour and vehicle response under typical operating conditions rather than detailed structural analysis.

Second, the model assumes a flat, planar road surface. Although real-world road profiles can vary significantly, this assumption is made to reduce the complexity of the model while still capturing the principal effects of tire-road interaction and vehicle dynamics. In controlled simulation environments, particularly when validating vehicle dynamics against standard test manoeuvres (e.g., double lane changes, step steer tests), a planar road is a reasonable approximation. Moreover, this assumption allows the focus to be placed on the

vehicle's intrinsic dynamics without the confounding effects of road irregularities. This approach is supported by numerous studies in vehicle dynamics where the flat-road assumption has been used successfully to validate handling and braking performance under controlled conditions [309], [310].

Another critical assumption is that the mass and inertial properties of the vehicle are considered constant during the simulation. In practice, variations due to fuel consumption or load transfer during transient manoeuvres exist; however, these changes are generally small relative to the overall mass distribution and have a negligible effect on the dynamic response in the operating conditions of interest. The constancy of these properties is essential for ensuring that the derived equations remain linear (or quasi-linear) in the parameters of interest, which in turn simplifies both the formulation and the numerical solution of the dynamic equations. This assumption is standard in many full-vehicle models because it simplifies the computation and parameter identification process while still capturing the essential dynamics [311], [312].

The model further decouples the vehicle dynamics into two main subsystems—one for ride (vertical, pitch, and roll motions) and one for handling (longitudinal, lateral, and yaw motions). This decoupling is based on the observation that, under normal driving conditions, the interactions between ride and handling dynamics are relatively weak. By treating these subsystems separately, the model can be made more computationally efficient without a significant loss in accuracy. This method has been validated in several previous studies and is recognized as a pragmatic approach in comprehensive vehicle modelling [312], [313]. A further assumption involves the use of linear or quasi-linear approximations for the suspension system. While actual suspension behaviour can be highly nonlinear, assuming linear stiffness and damping characteristics over the operating range is often sufficient to capture the primary response of the system. This simplification reduces the number of parameters that need to be identified and calibrated, thereby streamlining the simulation process without a significant loss in accuracy for the targeted testing scenarios as corroborated by established texts [314].

Finally, the model assumes external disturbances, such as aerodynamic forces, are either negligible or can be incorporated into the formulation through simplified additive terms. For the speed range and operating conditions considered in this study, the aerodynamic effects are not the dominant factors influencing vehicle dynamics compared to tire forces, suspension responses, and load transfer phenomena. This assumption further simplifies the modelling process, allowing the focus to remain on the key dynamics that govern vehicle behaviour during the test scenarios relevant to autonomous vehicle safety assessment [310], [312].

3.3 Seven DOF Vehicle Ride Model

In this section, the detail mathematical modelling of a 7 DOF vehicle ride model is presented. The 7 DOF model used considered the pitch, roll and vertical motion of the vehicle body, as well as the vertical displacement of the four unsprung masses as shown in Figure 3-1.

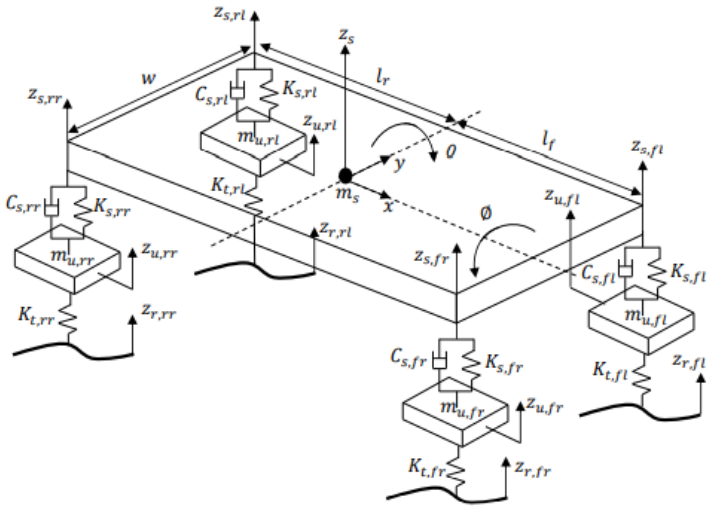


Figure 3-1: 7 degree of freedom vehicle riding model [315]

Looking at Figure 3-1, from Newton second law, the equilibrium vertical force acting on the sprung mass is the summation of all the suspension forces (spring and damper forces) in the vertical direction and the vertical acceleration of the sprung mass can be obtained as shown in Equation (3.1)

$$\sum F_b = m_s \ddot{z}_s$$

$$F_{sfl} + F_{dfl} + F_{sfr} + F_{dfr} + F_{srl} + F_{drl} + F_{srr} + F_{drr} = m_s \ddot{z}_s$$

$$\ddot{z}_s = \frac{F_{sfl} + F_{dfl} + F_{sfr} + F_{dfr} + F_{srl} + F_{drl} + F_{srr} + F_{drr}}{m_s} \tag{3.1}$$

Meanwhile, the pitch, θ and roll, $\emptyset$ motion of the sprung mass can be obtained as shown in equation 3.2 and 3.3 respectively.

$$\circlearrowleft \sum M_p = I_p \ddot{\theta}$$

$$-(F_{sfl} + F_{dfl} + F_{sfr} + F_{dfr})l_f + (F_{srl} + F_{drl} + F_{srr} + F_{drr})l_r = I_p \ddot{\theta}$$

$$\ddot{\theta} = \frac{-(F_{sfl} + F_{dfl} + F_{sfr} + F_{dfr})l_f + (F_{srl} + F_{drl} + F_{srr} + F_{drr})l_r}{I_p} \tag{3.2}$$

$$\circlearrowleft \sum M_p = I_r \ddot{\emptyset}$$

$$0.5w[(F_{sfl} + F_{dfl} + F_{srl} + F_{drl}) - (F_{sfr} + F_{dfr} + F_{srr} + F_{drr})] = I_r \ddot{\emptyset}$$

$$\ddot{\emptyset} = \frac{0.5w[(F_{sfl} + F_{dfl} + F_{srl} + F_{drl}) - (F_{sfr} + F_{dfr} + F_{srr} + F_{drr})]}{I_r} \tag{3.3}$$

where, F_{sfl} is the suspension spring force at front left corner,

F_{dfl} is the suspension damper force at front left corner,

F_{sfr} is the suspension spring force at front right corner,

F_{sdr} is the suspension damper force at front right corner,

F_{srl} is the suspension spring force at rear left corner,

F_{drl} is the suspension damper force at rear left corner,

F_{srr} is the suspension spring force at rear right corner,

F_{drr} is the suspension damper force at rear right corner,

m_s is the mass of the sprung mass,

$\ddot{z}_s$ is the acceleration of sprung mass at centre of gravity,

I_p is the moment of inertia at pitch axis,

I_r is the moment of inertia at roll axis,

$\ddot{\theta}$ is the pitch acceleration at centre of gravity of sprung mass,

$\ddot{\emptyset}$is the roll acceleration at centre of gravity of sprung mass,

l_f is the distance between the centre of gravity of sprung mass and the front wheels,

$l_r$is the distance between the centre of gravity of sprung mass and the rear wheels,

w is the wheelbase of the vehicle.

The 3 DOF of the vehicle ride model already derived in previous session, what is left is the 4 DOF for suspension system at the four corners. The suspension system of the vehicle can be illustrated as a 2 DOF mass-spring damper system as shown in Figure 3-2.

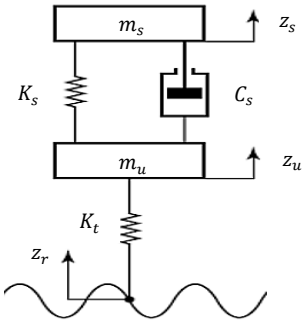


Figure 3-2: 2 degree of freedom suspension system of the vehicle model.

The suspension spring and damper forces at the four corners can be obtained as shown in Equation 3.4.

$$\begin{aligned} F_{sfl} &= K_{s,fl}(z_{u,fl} - z_{s,fl}) \\ F_{dfl} &= C_{s,fl}(\dot{z}_{u,fl} - \dot{z}_{s,fl}) \\ F_{sfr} &= K_{s,fr}(z_{u,fr} - z_{s,fr}) \\ F_{dfr} &= C_{s,fr}\left(\dot{z}_{u,fr} - \dot{z}_{s,fr}\right) \\ F_{srl} &= K_{s,rl}(z_{u,rl} - z_{s,rl}) \end{aligned} \tag{3.4}$$

$$F_{drl} = C_{s,rl}(\dot{z}_{u,rl} - \dot{z}_{s,rl})$$

$$F_{srr} = K_{s,rr}(z_{u,rr} - z_{s,rr})$$

$$F_{drr} = C_{s,rr}(\dot{z}_{u,rr} - \dot{z}_{s,rr})$$

where, $K_{s,fl}$ is the stiffness of the front left suspension spring,

$K_{s,fr}$ is the stiffness of the front right suspension spring,

$K_{s,rl}$ is the stiffness of the rear left suspension spring,

$K_{s,rr}$ is the stiffness of the rear right suspension spring,

$C_{s,fl}$ is the damping of the front left suspension,

$C_{s,fr}$ is the damping of the front right suspension,

$C_{s,rl}$ is the damping of the rear left suspension,

$C_{s,rr}$ is the damping of the rear right suspension,

$z_{u,fl}$ is the vertical displacement of the front left unsprung masses,

$z_{u,fr}$ is the vertical displacement of the front right unsprung masses,

$z_{u,rl}$ is the vertical displacement of the rear left unsprung masses,

$z_{u,rr}$ is the vertical displacement of the rear right unsprung masses,

$\dot{z}_{u,fl}$ is the vertical velocity of the front left unsprung masses,

$\dot{z}_{u,fr}$ is the vertical velocity of the front right unsprung masses,

$\dot{z}_{u,rl}$ is the vertical velocity of the rear left unsprung masses,

$\dot{z}_{u,rr}$ is the vertical velocity of the rear right unsprung masses.

Next, the vertical displacement of the sprung mass at each corner can be described in the vertical displacement of sprung mass at centre of gravity and the angles of pitch and roll as shown in Figure 3-3. This motion can be formulated as shown in Equation (3.5).

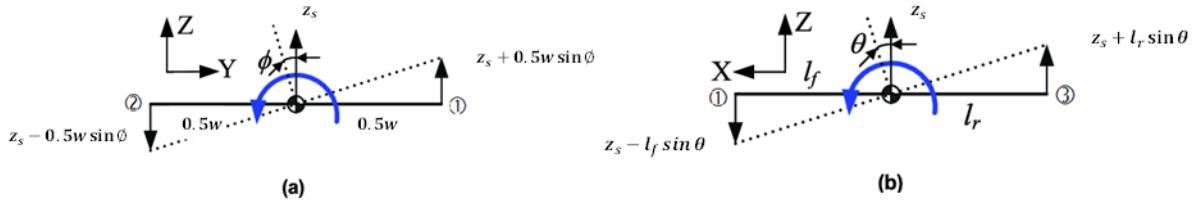


Figure 3-3 Displacement of the sprung mass during roll(a) and pitch(b) motion

$$\begin{aligned} z_{s,fl} &= z_s - l_f \sin\theta + 0.5w \sin\emptyset \\ z_{s,fr} &= z_s - l_f \sin\theta - 0.5w \sin\emptyset \\ z_{s,rl} &= z_s + l_r \sin\theta + 0.5w \sin\emptyset \\ z_{s,rr} &= z_s + l_r \sin\theta - 0.5w \sin\emptyset \end{aligned} \tag{3.5}$$

Since the angles of pitch, θ and roll, $\varnothing$ are assumed to be very small as these motions were minimized by the vehicle's suspension system, thus the Equation (3.5) can be simplified to Equation (3.6).

$$\begin{aligned} z_{s,fl} &= z_s - l_f\theta + 0.5w\emptyset \\ z_{s,fr} &= z_s - l_f\theta - 0.5w\emptyset \\ z_{s,rl} &= z_s + l_r\theta + 0.5w\emptyset \\ z_{s,rr} &= z_s + l_r\theta - 0.5w\emptyset \end{aligned} \tag{3.6}$$

where, $z_{s,fl}$ is the vertical displacement of front left sprung mass,

$z_{s,fr}$ is the vertical displacement of front right sprung mass,

$z_{s,rl}$ is the vertical displacement of rear left sprung mass,

$z_{s,rr}$ is the vertical displacement of rear right sprung mass,

$\dot{\theta}$ is the pitch rate at centre of gravity of the sprung mass,

$\dot{\varnothing}$ is the roll rate at centre of gravity of the sprung mass,

z_S is the vertical displacement of the sprung mass at centre of gravity,

$\dot{z_S}$ is the vertical velocity of the sprung mass at centre of gravity.

The suspension spring forces, and damper forces can then be obtained by substituting Equation (3.4) into Equation (3.2). From the resulting equation, Equation (3.1) can be expressed as Equation (3.7) below:

$$\begin{aligned} m_s\ddot{z}_S = {} & K_{s,fl}(z_{u,fl} - z_s + l_f\theta - 0.5w\emptyset) + C_{sfl}(\dot{z}_{u,fl} - \dot{z}_s + l_f\dot{\theta} - 0.5w\dot{\emptyset}) \\ & + K_{s,fr}(z_{u,fr} - z_s + l_f\theta + 0.5w\emptyset) + C_{sfr}(\dot{z}_{u,fr} - \dot{z}_s + l_f\dot{\theta} + 0.5w\dot{\emptyset}) \\ & + K_{s,rl}(z_{u,rl} - z_s - l_r\theta - 0.5w\emptyset) + C_{srl}(\dot{z}_{u,rl} - \dot{z}_s - l_r\dot{\theta} - 0.5w\dot{\emptyset}) \\ & + K_{s,rr}(z_{u,rr} - z_s - l_r\theta + 0.5w\emptyset) + C_{srr}(\dot{z}_{u,rr} - \dot{z}_s - l_r\dot{\theta} + 0.5w\dot{\emptyset}) \end{aligned} \tag{3.7}$$

Similarly, the pitch angle, θ and roll angle, $\varnothing$ from Equation (3.2) and Equation (3.3) of the sprung mass can be re-described as Equation (3.8) and Equation (3.8) respectively as shown below:

$$\begin{aligned} I_p\ddot{\theta} = {} & -[K_{s,fl}(z_{u,fl} - z_s + l_f\theta - 0.5w\dot{\varnothing}) + C_{s,fl}(\dot{z_{u,fl}} - \dot{z_s} + l_f\dot{\theta} - 0.5w\dot{\varnothing}) + \\ & K_{s,fr}(z_{u,fr} - z_s + l_f\theta + 0.5w\dot{\varnothing}) + C_{s,fr}(\dot{z_{u,fr}} - \dot{z_s} + l_f\dot{\theta} + 0.5w\dot{\varnothing}]l_f + \\ & [K_{s,rl}(z_{u,rl} - z_s - l_r\theta - 0.5w\dot{\varnothing}) + C_{s,rl}(\dot{z_{u,rl}} - \dot{z_s} - l_r\dot{\theta} - 0.5w\dot{\varnothing}) + \end{aligned}$$

$$K_{s,rr}(z_{u,rr}-z_s-l_r\theta+0.5w^{\varnothing})+C_{s,rr}(\dot{z}_{u,rr}-\dot{z}_s-l_r\dot{\theta}+0.5\dot{w^{\varnothing}})]l_r \quad (3.8)$$

$$\begin{aligned} I_r\ddot{\varnothing} = 0.5\text{w}\,[&(K_{s,fl}(z_{u,fl}-z_s+l_f\theta-0.5w^{\varnothing})+C_{s,fl}(\dot{z}_{u,fl}-\dot{z}_s+l_f\dot{\theta}-0.5\dot{w^{\varnothing}})+K_{s,rl}\\ &(z_{u,rl}-z_s-l_r\theta-0.5w^{\varnothing})+C_{s,rl}(\dot{z}_{u,rl}-\dot{z}_s-l_r\dot{\theta}-0.5\dot{w^{\varnothing}}))-(K_{s,fr}(z_{u,fr}-z_s+\\ &l_f\theta+0.5w^{\varnothing})+C_{s,fr}(\dot{z}_{u,fr}-\dot{z}_s+l_f\dot{\theta}+0.5\dot{w^{\varnothing}})+K_{s,rr}(z_{u,rr}-z_s-l_r\theta+0.5w\\ &^{\varnothing})+C_{s,rr}(\dot{z}_{u,rr}-\dot{z}_s-l_r\dot{\theta}+0.5\dot{w^{\varnothing}})] \end{aligned} \quad (3.9)$$

The equation of motion of the unsprung masses at each corner can be described in the corresponding suspension spring force and damper force as shown in Equation (3.10) below:

$$\begin{aligned} F_{tfl}-F_{sfl}-F_{dfl}&=m_{u,fl}\ddot{z}_{u,fl}\\ F_{tfr}-F_{sfr}-F_{dfr}&=m_{u,fr}\ddot{z}_{u,fr}\\ F_{trl}-F_{srl}-F_{drl}&=m_{u,rl}\ddot{z}_{u,rl}\\ F_{trr}-F_{srr}-F_{drr}&=m_{u,rr}\ddot{z}_{u,rr} \end{aligned} \quad (3.10)$$

where, F_{tfl} is the tire forces acting at front left of sprung mass,
F_{tfr} is the tire forces acting at front right of sprung mass,
F_{trl} is the tire forces acting at rear left of sprung mass,
F_{trr} is the tire forces acting at rear right of sprung mass,
$\ddot{z}_{u,fl}$ is the acceleration of the Front left unsprung masses,
$\ddot{z}_{u,fr}$ is the acceleration of the Front right unsprung masses,
$\ddot{z}_{u,rl}$ is the acceleration of the Rear right unsprung masses,
$\ddot{z}_{u,rr}$ is the acceleration of the Rear right unsprung masses.

By substituting Equation (3.8) and Equation (3.9) into Equation (3.10), the summation of vertical forces at each corner of the sprung mass can be obtained as Equation (3.11) below:

$$\begin{aligned} m_{u,fl}\ddot{z}_{ufl} &= K_{t,fl}(z_{r,fl}-z_{u,fl})-K_{s,fl}(z_{u,fl}-z_s+l_f\theta-0.5w^{\varnothing})-\\ &\qquad C_{s,fl}(\dot{z}_{u,fl}-\dot{z}_s+l_f\dot{\theta}-0.5\dot{w^{\varnothing}})\\ m_{u,fr}\ddot{z}_{ufr} &= K_{t,fr}(z_{r,fr}-z_{u,fr})-K_{s,fr}(z_{u,fr}-z_s+l_f\theta+0.5w^{\varnothing})-\\ &\qquad C_{s,fr}(\dot{z}_{u,fr}-\dot{z}_s+l_f\dot{\theta}+0.5\dot{w^{\varnothing}}\\ m_{u,rl}\ddot{z}_{url} &= K_{t,rl}(z_{r,rl}-z_{u,rl})-K_{s,rl}(z_{u,rl}-z_s-l_r\theta-0.5w^{\varnothing})-\\ &\qquad C_{s,rl}(\dot{z}_{u,rl}-\dot{z}_s-l_r\dot{\theta}-0.5\dot{w^{\varnothing}}\\ m_{u,rr}\ddot{z}_{urr} &= K_{t,rr}(z_{r,rr}-z_{u,rr})-K_{s,rr}(z_{u,rr}-z_s-l_r\theta+0.5w^{\varnothing})-\\ &\qquad C_{s,rr}(\dot{z}_{u,rr}-\dot{z}_s-l_r\dot{\theta}+0.5\dot{w^{\varnothing}} \end{aligned} \quad (3.11)$$

Finally, the equation of the normal force, F_Z acting on the tires can be derived from Equation (3.4) as Equation (3.12) which can then be used for the development of tire model.

$$
\begin{aligned}
F_{z,fl} &= {m_s g l_r}/{2(l_r + l_f) + m_{u,fl} g + F_{s,fl} + F_{d,fl}} \\
F_{z,fr} &= {m_s g l_r}/{2(l_r + l_f) + m_{u,fr} g + F_{s,fr} + F_{d,fr}} \\
F_{z,rl} &= {m_s g l_f}/{2(l_r + l_f) + m_{u,rl} g + F_{s,rl} + F_{d,rl}} \\
F_{z,rr} &= {m_s g l_f}/{2(l_r + l_f) + m_{u,rr} g + F_{s,rr} + F_{d,rr}}
\end{aligned} \tag{3.12}
$$

where, $F_{z,fl}$ is the normal force acting on the Front left tire,
$F_{z,fr}$ is the normal force acting on the Front right tire,
$F_{z,rl}$ is the normal force acting on the Rear left tire,
$F_{z,rr}$ is the normal force acting on the Rear right tire,
$m_{u,fl}$ is the mass of the Front left unsprung mass,
$m_{u,fr}$ is the mass of the Front right unsprung mass,
$m_{u,rl}$ is the mass of the Rear left unsprung mass,
$m_{u,rr}$ is the mass of the Rear right unsprung mass,
g is the Gravitational force, 9.81 ms^{-2}.

3.4 Pacejka Tire Model

An accurate tire model is an essential part of a vehicle model to simulate the longitudinal force, lateral force and aligning moment acting on the tires. A good choice of tire model that can provide a good accuracy in estimating the longitudinal and lateral dynamics is the Pacejka Tire Model or also known as the "Magic" formula [303]. This choice is underpinned by the extensive empirical validation and industry acceptance of the Magic Formula, which effectively captures the nonlinear relationship between tire slip and generated forces. Alternative models, such as the Brush Model or the Fiala Model, provide simplified representations of tire behaviour that tend to be inadequate for capturing the full nonlinear response, especially under extreme conditions such as emergency manoeuvres. While more advanced methods—such as neural network-based tire models or finite element-based approaches—could potentially offer higher fidelity, these methods require considerably more computational resources and complex parameter identification processes. Therefore, the Pacejka Magic Formula presents the best compromise between accuracy and computational efficiency, ensuring that tire forces are modelled with sufficient realism for the purposes of safety testing and driver behaviour prediction. The general force or moment equations for the tire model are given in Equation (3.13), Equation (3.14) and Equation (3.15) below.

$$y(x) = D \sin\left[C \tan^{-1}\left(Bx - E(Bx - \tan^{-1}(Bx))\right)\right] \tag{3.13}$$

$$Y(X) = y(x) + S_v \tag{3.14}$$

$$x = X + S_h \tag{3.15}$$

where, B is the stiffness factor,
C is the shape factor,

D is the peak value,
E is the curvature factor,
S_v is the vertical shift,
S_h is the horizontal shift,
X is either slip angle or longitudinal slip depending on the usage of the equations.

X will denote the slip angle, α when the equations are used for lateral force and aligning moment. Meanwhile, X will also denote the longitudinal slip, κ when the equations are used for longitudinal force and brake force. B, C, D and E are parameter dependent coefficients which varies according to the types of surfaces and the normal force of the tires, F_Z. The equations for the coefficients are given in Equation (3.16), Equation (3.17) and Equation (3.18) below:

$$D = a_1 F_z^2 + a_2 F_Z \tag{3.16}$$

$$B \cdot C \cdot D = \frac{a_3 F_z^2 + a_4 F_Z}{e^{a_5 F_Z}} \tag{3.17}$$

$$E = a_6 F_z^2 + a_7 F_Z + a_8 \tag{3.18}$$

The vertical shift and horizontal shift are affected by the camber angle, γ_c, with equations as shown in Equation (3.19) and Equation (3.20) below:

$$S_h = a_9 \gamma_c \tag{3.19}$$

$$S_v = (a_{10} F_z^2 + a_{11} F_z)\gamma_c \tag{3.20}$$

Meanwhile, the value of the shape factor C is given as below:
$C = 1.30$ for lateral force, F_y,
$C = 1.65$ for longitudinal force, F_x,
$C = 2.40$ for aligning moment, M_Z.

Finally, the coefficients used for a_1 to a_{11} are given in [303] as shown in Table 3-1 below:

Table 3-1 Values of Coefficients $a_1$to a_{11} for Pacejka Tire Model

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}
F_x	-21.3	1144	49.6	226	0.069	-0.006	0.056	0.486	-	-	-
F_y	-22.1	1011	1078	1.82	0.208	0.000	-0.354	0.707	0.028	0.000	14.8
M_z	-2.72	-2.28	-1.86	-2.73	0.110	-0.070	0.643	-4.04	0.015	-0.066	0.945

3.5 Seven DOF Vehicle Handling Model

The 7 DOF vehicle handling model composed of 3 DOF representing the lateral, longitudinal and yaw motion of the vehicle sprung mass, and 1 DOF representing the rolling motion for each wheel which make up to the remaining 4 DOF. The 7 DOF vehicle handling model is illustrated as shown in Figure 3-4.

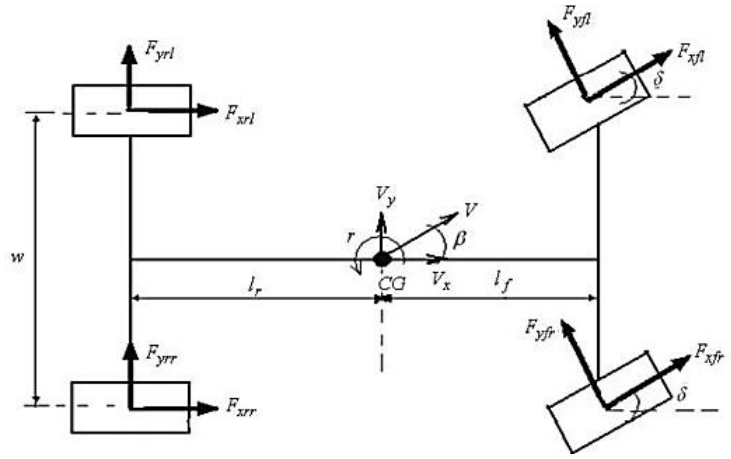


Figure 3-4: 7 degree of freedom handling model [316]

Based on Figure 3-4, an equation that defines the total longitudinal forces of the vehicle model can be obtained as in Equation (3.21) and Equation (3.22):

$$\sum F_x = m_s \dot{v}_x = F_{x,rr} + F_{x,rl} - \left(F_{y,fr} + F_{y,fl}\right)\sin\delta_f + \left(F_{x,fr} + F_{x,fl}\right)\cos\delta_f + \left(m_s g \sin\theta_r\right) - \sum F_d(v_x) \tag{3.21}$$

$$\sum F_d(v_x) = F_a + F_r = 0.5\rho A C_d v_x^2 + m_s g C_r v_x \tag{3.22}$$

Whereas the lateral forces of the vehicle model can be obtained as Equation (3.23) below:

$$\sum F_y = m_s \dot{v}_y = F_{y,rr} + F_{y,rl} + \left(F_{y,fr} + F_{y,fl}\right)\cos\delta_f + \left(F_{x,fr} + F_{x,fl}\right)\sin\delta_f \tag{3.23}$$

The self-aligning moment, M_Z about the z-axis can be obtained as Equation (3.24) below. The moment, M_Z is assumed to have the same direction with yaw motion, φ.

$$\sum M_y = I_s \ddot{\varphi} = \left[F_{x,rr} - F_{x,rl} - \left(F_{y,fr} - F_{y,fl}\right)\sin\delta_f + \left(F_{x,fr} - F_{x,fl}\right)\cos\delta_f\right]w/2 - \left[F_{y,rr} + F_{y,rl}\right]l_r + \left[\left(F_{y,fr} + F_{y,fl}\right)\cos\delta_f + \left(F_{x,fr} + F_{x,fl}\right)\sin\delta_f\right]l_f + \left[M_{z,fl} + M_{z,fr} + M_{Z,rl} + M_{z,rr}\right] \tag{3.24}$$

where, F_a is aerodynamic force,

F_r is rolling resistance force,

θ_r is the gradient of the road,

C_d is the coefficient of the aerodynamic,

C_r is the coefficient of rolling resistance of tire,

$F_{x,fl}$ is the front left tire's longitudinal forces,

$F_{x,fr}$ is the front right tire's longitudinal forces,

$F_{x,rl}$ is the rear left tire's longitudinal forces,

$F_{x,rr}$ is the rear right tire's longitudinal forces,

$F_{y,fl}$ is the front left tire's lateral forces,
$F_{y,fr}$ is the front right tire's lateral forces,
$F_{y,rl}$ is the rear left tire's lateral forces,
$F_{y,rr}$ is the rear right tire's lateral forces,
$M_{z,fl}$ is the front left tire's moment,
$M_{z,fr}$ is the front right tire's moment,
$M_{z,rl}$ is the rear left tire's moment,
$M_{z,rr}$ is the rear right tire's moment,
δ_f is the wheel steer angle in degree.

The inertial acceleration in longitudinal, a_x and lateral, a_y direction is defined by the acceleration in lateral, v_y and longitudinal, v_x directions and centripetal acceleration in lateral $v_y\dot{\varphi}$ and longitudinal $v_x\dot{\varphi}$ directions. Equation (3.25) represent the acceleration in longitudinal and lateral directions.

$$\begin{aligned} \dot{v_x} &= a_x + v_y\dot{\varphi} \\ \dot{v_y} &= a_y + v_x\dot{\varphi} \end{aligned} \tag{3.25}$$

With the acceleration equations, the longitudinal and lateral velocities v_x and v_y can be obtained by simply performing integration to Equation (3.25). From Figure 3-5, it can be observed that the roll motion of the vehicle body is affected by the lateral acceleration, a_y obtained from Equation (3.25). To calculate the roll motion of the vehicle body, the centre of roll motion with a roll angle, ϕ is assumed at a position defined by a height of h_{rc} from the ground and is below the center of gravity of the vehicle body. With this assumption, the roll motion of the vehicle body during lateral acceleration can be calculated with the summation of moment acting at the x-axis of the centre of roll motion as shown in equation below.

$$(I_r + m_s(h - h_{rc})^2)\ddot{\phi} = m_s a_y(h - h_{rc})\cos\phi + m_s g(h - h_{rc})\sin\phi - k_\phi\phi - \beta_\phi\dot{\phi} \tag{3.26}$$

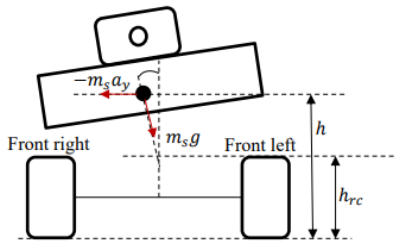


Figure 3-5: Roll motion due to lateral acceleration of the vehicle [2]

Similarly, the pitch motion of the vehicle body is affected by the longitudinal acceleration during acceleration or braking of the vehicle. The centre of the pitch motion is assumed below the centre of gravity of the vehicle body as shown in Figure 3-6.

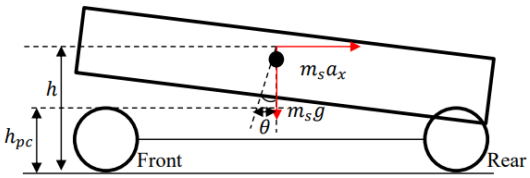


Figure 3-6: Pitch motion due to longitudinal acceleration of the vehicle [317]

The pitch motion can be formulated as the summation of moment acting at the y-axis of the centre of pitch motion as shown in equation below.

$$\left(I_p + m_s(h - h_{pc})^2\right)\ddot{\theta} = m_s a_x(h - h_{pc}) + m_s g(h - h_{pc})\sin\theta - k_\theta\theta - \beta_\theta\dot{\theta} \tag{3.28}$$

The remaining four degrees of freedom (DOF) in the 7 DOF handling model refer to the rotational motion of the vehicle's four wheels. This motion is described by the wheel's angular velocity, ω. A wheel's dynamic motion is illustrated as shown in Figure 3-7 below:

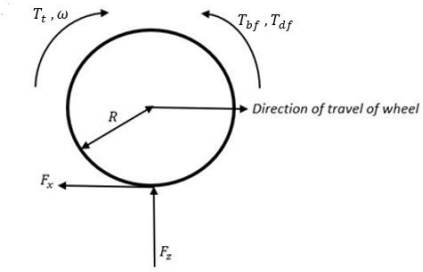


Figure 3-7: Motion acting on the wheel [2]

The equation of motion of the wheel is a summation of torque including the throttle torque, T_t, brake torque, T_b, viscous friction torque, T_d and rotational motion of the wheel. Since the vehicle is front wheel drive, the throttle torque, T_t is assumed to be zero. The equation of motion of a wheel can be obtained as shown in Equation (3.28).

$$\begin{aligned}
I_w\dot{w_{fr}} &= T_{t,fl} = T_{t,fl} + T_{d,fl} - T_{b,fl} + \left[F_{x,fl} \times R_w\right] \\
I_w\dot{w_{fr}} &= T_{t,fr} = T_{t,fr} + T_{d,fr} - T_{b,fr} + \left[F_{x,fr} \times R_w\right] \\
I_w\dot{w_{rl}} &= T_{t,rl} = T_{t,rl} + T_{d,rl} - T_{brl} + \left[F_{x,rl} \times R_w\right] \\
I_w\dot{w_{fl}} &= T_{t,rr} = T_{t,rr} + T_{d,rr} - T_{b,rr} + [F_x, rr \times R_w]
\end{aligned} \tag{3.28}$$

Where, the viscous friction torque, T_d can be written as Equation (3.29) below:

$$\begin{aligned}
T_{d,fl} &= \omega_{fl} \times C_{f,f} \\
T_{d,fr} &= \omega_{fr} \times C_{f,f} \\
T_{d,rl} &= \omega_{rl} \times C_{f,r} \\
T_{d,rr} &= \omega_{rr} \times C_{f,r}
\end{aligned} \tag{3.29}$$

Where, $C_{f,f}$ is the viscous friction coefficient of front wheels,

$C_{f,r}$ is the viscous friction coefficient of rear wheels.

3.6 Longitudinal and Lateral Slip Model

In previous section, the accelerations and velocities in longitudinal and lateral directions are obtained. So, in this section, the equations that define the longitudinal and lateral slip of the tires are obtained. First, the vehicle body side slip angle, β can be defined as Equation (3.30) below:

$$\beta = \tan^{-1} \frac{\int_0^\infty a_y}{\int_0^\infty a_x} \tag{3.30}$$

The lateral slip angle at the front, α_f and rear tires, α_r can be obtained as Equation (3.31).

$$\begin{aligned} \alpha_f &= \tan^{-1}\left(\frac{v_y + l_f\dot{\varphi}}{v_x}\right) - \delta_f \\ \alpha_r &= \tan^{-1}\left(\frac{v_y + l_r\dot{\varphi}}{v_x}\right) \end{aligned} \tag{3.31}$$

The difference between the equation of the front and rear tires is because the steering angle for the rear wheels are always zero, where $\delta_r = 0$. Meanwhile, the longitudinal slip at the front, λ_f and rear, λ_r can be calculated using the equations below.

$$\begin{aligned} \lambda_f &= v_{wxf} - \omega_f R_w / v_{wxf} \\ \lambda_r &= v_{wxr} - \omega_r R_w / v_{wxr} \end{aligned} \tag{3.32}$$

The longitudinal slip under braking condition is used in this model due to the assumption of positive pitch motion during braking condition. The longitudinal velocities of the front and rear tires, v_{wxf} and v_{wxr} that are required in Equation (3.32) are given by the following equations.

$$\begin{aligned} v_{wxf} &= \left[\sqrt{\left(v_y + l_f r\right)^2 + v_x^2}\right] \cos\alpha_f \\ v_{wxr} &= \left[\sqrt{\left(v_y + l_r r\right)^2 + v_x^2}\right] \cos\alpha_r \end{aligned} \tag{3.33}$$

3.7 Vehicle Kinematics Model

The velocities obtained in the vehicle handling model are in local coordinate. In order to control the trajectory of the simulated vehicle, the local coordinate need to be resolved into the global coordinate. The global coordinate, X and Y of the vehicle in global frame can be obtained by using Equation (3.34) below.

$$\begin{aligned} X &= \int_0^{X_0} \left(V_x \cos\varphi - V_y \sin\varphi\right) dt \\ Y &= \int_0^{X_0} \left(V_x \sin\varphi + V_y \cos\varphi\right) dt \end{aligned} \tag{3.34}$$

3.8 Verification of the Vehicle Model

The mathematical model derived in the previous sections is developed in MATLAB's Simulink platform. Figure 3-8 shows the block diagram of the vehicle model developed.

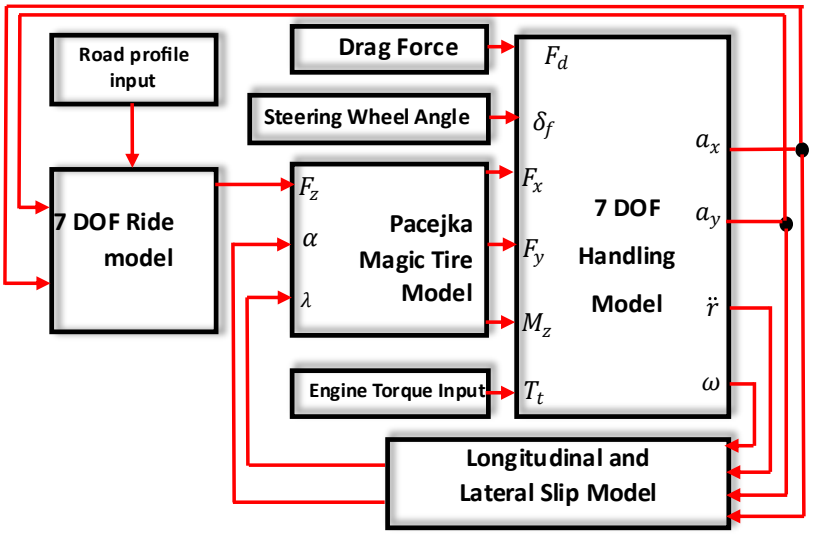


Figure 3-8: 14 degree of freedom vehicle dynamics Simulink model

In order to validate the vehicle model developed, IPG CarMaker is used to develop simulation test cases by providing inputs for the vehicles and measure the outputs in terms of vehicle lateral and longitudinal behaviours. Then, the same control inputs from IPG CarMaker are provided to the vehicle model in Simulink and the outputs of the vehicle model are verified with IPG CarMaker responses. The simulation used the ode4 Runge-Kutta solver with a fixed step time of 0.001s to balance the numerical stability and computational cost. Besides, vehicle dynamics models that incorporate high-frequency phenomena—such as tire-road interactions modelled by Pacejka's Magic Formula and rapid transient responses from braking or steering inputs—often require a sufficiently small time step to capture these dynamics without introducing numerical instability or excessive discretization error [303]. The parameters of the vehicle model are provided in Table 3-2 below:

Table 3-2: Simulated vehicle's parameters

Parameters	Abbreviation	Value	Unit
Moment of inertia of front and rear wheel	I_w	15	kgm^2
Sprung mass	m_s	1463	kg
Unsprung mass	$m_{u,fl}, m_{u,fr}, m_{u,rl}, m_{u,rr}$	25	kg
Height of centre of gravity	h	0.567	m
Tire radius	R_w	0.318	kg
Front length from centre of gravity	l_f	1.240	m
Rear length from centre of gravity	l_r	1.712	m
Frontal area	A	1.5	m^2
Wheelbase width	w	1.092	m
Moment of inertia at pitch axis	I_x	1.8745×10^3	kgm^2
Moment of inertia at roll axis	I_y	584.097	kgm^2
Moment of inertia at yaw axis	I_z	2058	kgm^2
Spring stiffness of tire	$K_{t,fl}, K_{t,fr}, K_{t,rl}, K_{t,rr}$	350000	N/m
Spring stiffness of suspension	$K_{s,fl}, K_{s,fr}, K_{s,rl}, K_{s,rr}$	35000	N/m
damper stiffness of suspension	$C_{s,fl}, C_{s,fr}, C_{s,rl}, C_{s,rr}$	3000	Nms^{-1}

3.8.1 Verification of Vehicle Ride Behaviour

First, the ride model developed is validated using a test track developed in IPG CarMaker. This test is used to evaluate the suspension performance and effects of the road profile on the vehicle dynamics. The test track is a straight track with total driving distance of 70m and consists of 3 bumps along the road separated by 15m. The three bumps have height of 0.02 m, 0.05 m and 0.1 m respectively. During the test, the vehicle is set to maintain a constant speed of 10 km/h, 40 km/h and 60 km/h to evaluate vehicle dynamics across a representative range of operating conditions. These specific speeds are chosen to encompass low-speed, moderate-speed, and higher-speed scenarios, each of which presents distinct dynamic behaviours and challenges.

- Low-Speed (10 km/h): At this speed, the vehicle operates under conditions where inertial forces are minimal, and the dynamics are influenced by factors such as steering input and tire compliance. Simulating at 10 km/h allows for the assessment of vehicle control and stability during manoeuvres such as parking or navigating tight spaces.
- Moderate-Speed (40 km/h): This speed represents typical urban driving conditions. Evaluating vehicle dynamics at 40 km/h provides insights into handling characteristics during common city driving scenarios, including lane changes and gradual turns. Studies have utilized similar speeds to assess driving behaviour and vehicle response in urban settings [318].
- High-Speed (60 km/h): Operating at 60 km/h introduces more obvious inertial effects, making it suitable for assessing vehicle stability and control during manoeuvres. Research indicates that vehicle dynamics can change significantly at speeds above 60 km/h, affecting factors such as ride comfort and steering stability [319].

By selecting these speeds, the simulation aims to comprehensively evaluate vehicle performance across a spectrum of realistic driving conditions, ensuring that the model accurately captures the nuances of vehicle dynamics pertinent to each speed range. Figure below shows the test track developed in the IPG CarMaker.

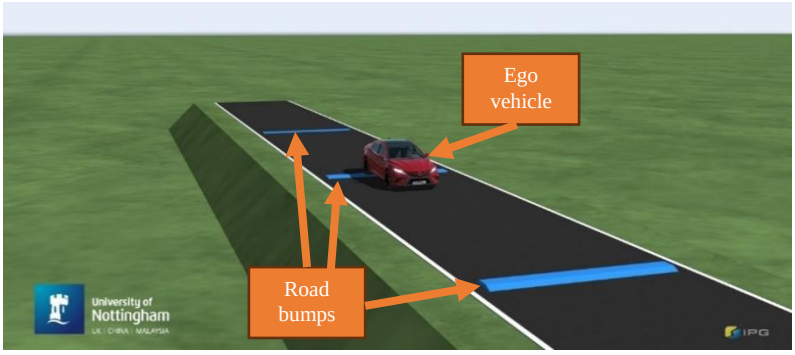


Figure 3-9: Ride model test track developed in IPG CarMaker

Two road sensors were attached to the wheels of the vehicle, one of the sensors at the front wheel and another sensor at the rear wheel to record the road profile data experience by the vehicle's front and rear wheels. The configuration of the road sensors is as shown in Figure 3-10 below. The two green dots on the vehicle are the position of the road sensors located. The

preview distance is set to 0.0m and the “Consider bumps” option is checked so that the sensor will capture the height of the bumps at the position of the front and rear wheel.

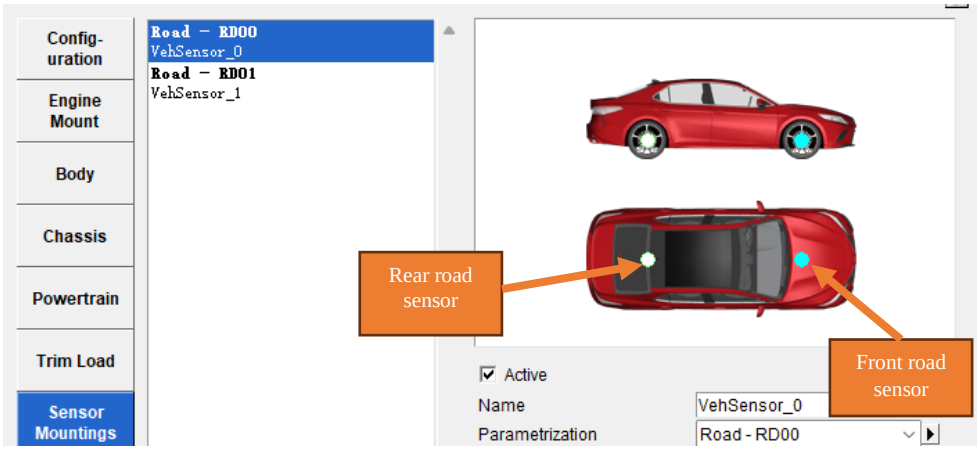


Figure 3-10: Configuration of road sensors in IPG CarMaker

The road profile data captured using road sensors attached to the front and rear wheels is plotted using MATLAB as shown in Figure 3-11 below.

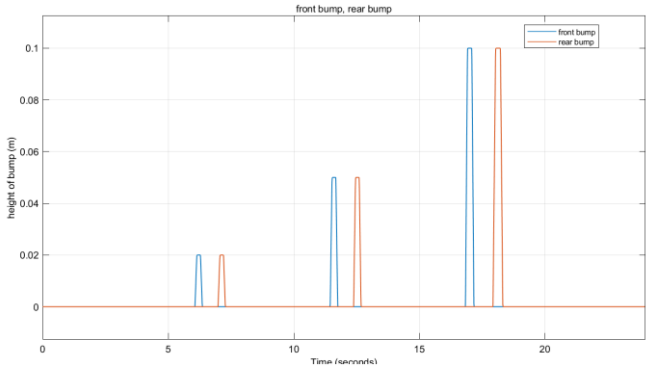
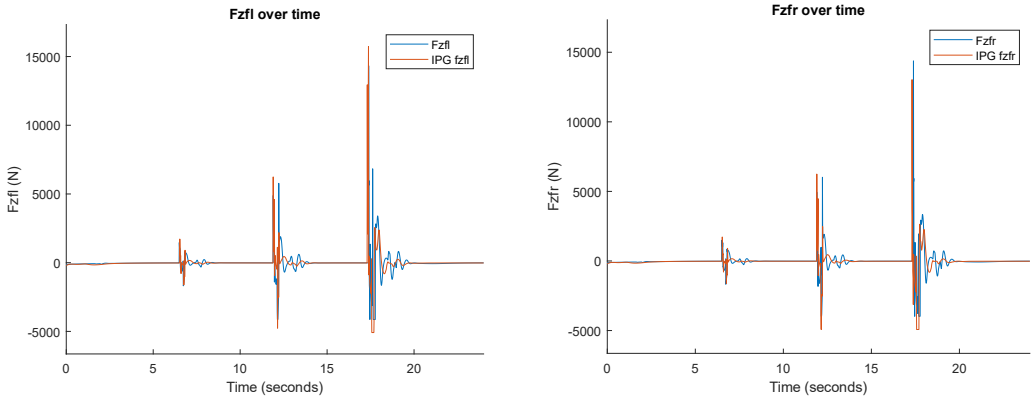


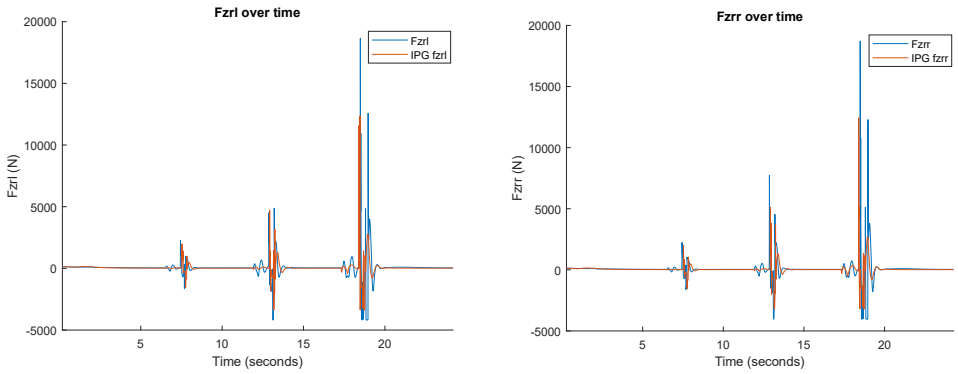
Figure 3-11: Road profile captured from IPG CarMaker

By providing the above road profile data to the vehicle model in Simulink, the normal forces acting on the sprung mass for vehicle test speed of 10 km/h can be obtained as show in Figure 3-12 below.



(a) F_{zfl} against time

(b) F_{zfr} against time



(c) F_{zrl} against time *(d) F_{zrr} against time*

Figure 3-12: Riding model test result at 10 km/h

The output obtained from the vehicle ride model in the Simulink is compared with the simulated normal force, F_z data from the IPG Carmaker. As observed from Figure above, the result from the Simulink ride model developed closely follow the trend of the simulated data from the IPG CarMaker with minor deviation. By using data analysis method, Root Mean Square (RMS) as shown in Equation (3.35), the percentage of error between the Simulink model developed and the simulation data from IPG CarMaker can be calculated as shown in Table 3-3.

$$Root\ Mean\ Square\ Error, RMSE = \sqrt{\sum_{t=1}^{N} \frac{\left(Output_{actual}(t) - Output_{desired}(t)\right)^2}{N}} \tag{3.35}$$

Table 3-3: Percentage of RMS error for riding model test

Observation data	Root Mean Square (RMS) at 10 km/h		Percentage of error (%)
	IPG CarMaker	Simulink	
F_{zfl}	4702 N	4416 N	6.089
F_{zfr}	4545 N	4416 N	2.845
F_{zrl}	3288 N	3297 N	0.2551
F_{zrr}	3138 N	3297 N	5.051

Based on the table above, the Root Mean Square Error (RMSE) values show that the data obtained is within acceptable range which is less than 15% with the highest only up to 6.089% [320]. This indicates that the Simulink vehicle ride model developed is able to reproduce the desired output from the IPG CarMaker with minor error only. This minor error is occurred due to the difference between the vehicle parameter used in the Simulink model and the IPG CarMaker where in the Simulink model, constant values are used for the parameters such as suspension spring constants,K_s and damping constants, C_s; while in the IPG CarMaker, the suspension's constants are defined using a look-up table as shown in Figure 3-13 below. Therefore, due to the relationship between the damping force and the velocity is non-linear in the IPG CarMaker, this causes the slight error in the values obtained.

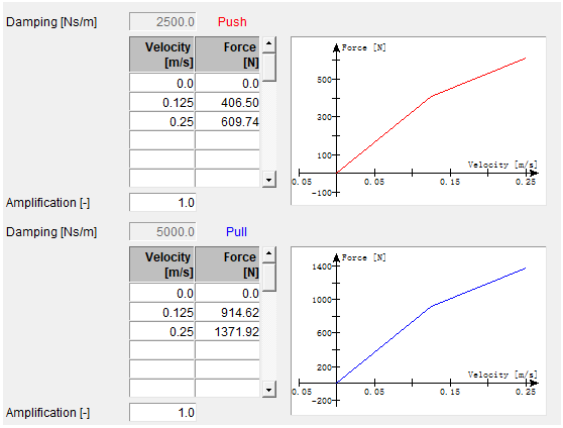
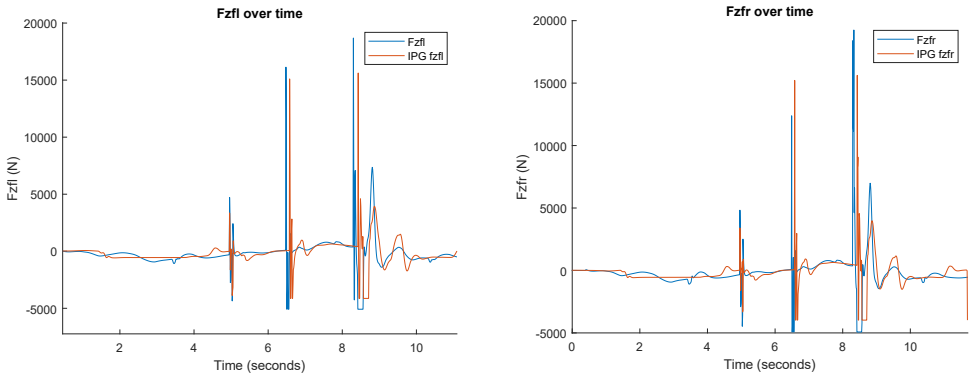
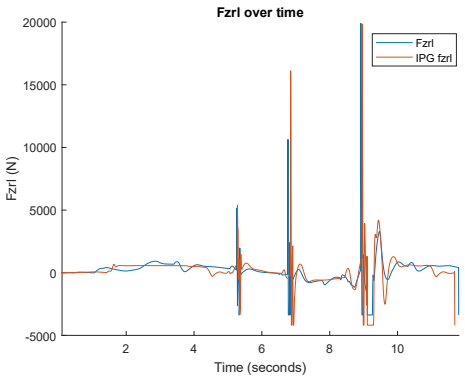


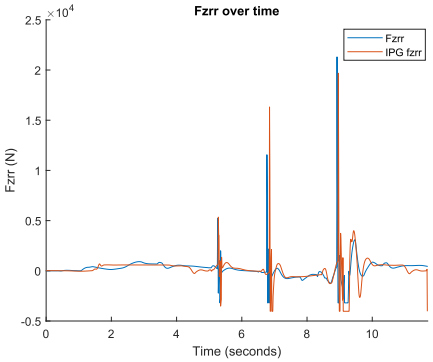
Figure 3-13: Damping parameters used in IPG CarMaker

The 7 DoF riding model is further tested at target speed of 40 km/h and 60 km/h according to SAE recommended speed for testing[321]. Figure 3-14 and Figure 3-15 show the response obtained from the Simulink model. It can be observed that the model is able to achieve response similar to the response from the IPG CarMaker even at these higher vehicle speeds. The verification tests show the percentage of RMS error for normal force acting at the front-left, front-right, rear-left and rear-right tires are less than 15%. For vehicle speed of 40 km/h, the RMS value for front-left, front-right, rear-left and rear-right normal force responses obtained from Simulink are 4478 N, 4478 N, 3427 N and 3427 N, respectively. Whereas the RMS value of the responses obtained from the IPG CarMaker are 4448 N, 4289 N, 3549 N and 3381 N respectively. Based on the result, the percentage of RMSE can be calculated as 0.6369%, 4.388%, 3.423% and 1.369%, respectively. Besides, during the vehicle speed of 60 km/h, the RMS values for response obtained from Simulink model are 4479 N, 4479 N, 3429 N and 3429 N, respectively. On the other hand, the RMS value calculated from response from IPG CarMaker vehicle model are 4460 N, 4299 N, 3583 N and 3395 N, respectively. Hence, the percentage of RMSE can be identified as 0.42%, 4.184%, 3.75% and 0.9989%, respectively. It can also be observed from Table 3-4 that the percentage of RMS error for normal force acting at each wheel is less at higher speed is less than that at lower speed in the previous verification test. This is because at higher speed the value of spring and damping coefficients of the IPG CarMaker model are closer to the coefficients in the Simulink model, hence the model is able to obtain a better result. Therefore, from these verification test, it can be observed that the vehicle riding model developed can be used as a plant model for analysis of driver model, which will be discussed in the next chapter.



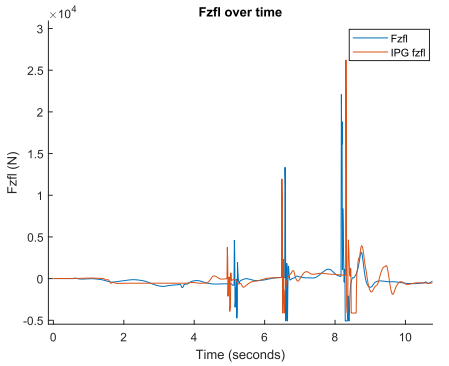
(a) F_{zfl} against time *(b) F_{zfr} against time*

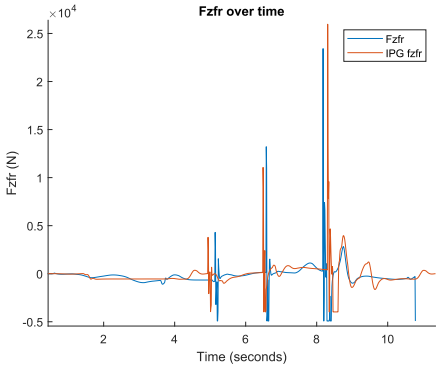




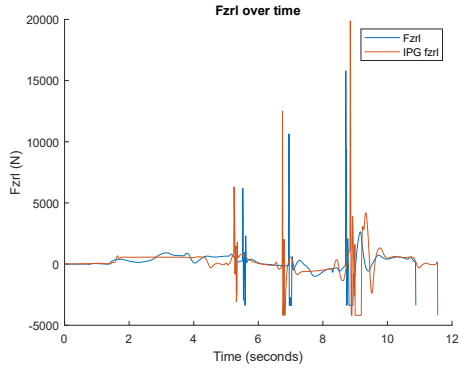
(c) F_{zrl} against time *(d) F_{zrr} against time*

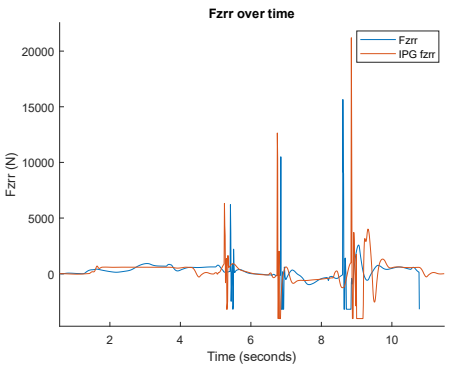
Figure 3-14: Riding model test result at 40 km/h





(a) F_{zfl} against time *(b) F_{zfr} against time*





(c) F_{zrl} against time *(d) F_{zrr} against time*

Figure 3-15: Riding model test result at 60 km/h

Table 3-4: Percentage of RMS error for riding model test at 40 km/h and 60 km/h

Observation data	RMS value				Percentage of RMSE (%)	
	40 km/h		60 km/h			
	IPG CarMaker	Simulink	IPG CarMaker	Simulink	40 km/h	60 km/h
F_{zfl}	4448 N	4478 N	4460 N	4479 N	0.6369	0.42
F_{zfr}	4289 N	4478 N	4299 N	4479 N	4.388	4.184
F_{zrl}	3549 N	3427 N	3583 N	3429 N	3.423	3.75
F_{zrr}	3381 N	3427 N	3395 N	3429 N	1.369	0.9989

3.8.2 Verification of Vehicle Lateral Behaviours

In this subsection, the verification of vehicle handling model using lane change test is used. The testing methodology used is the Double Lane Change (DLC) based on the test track specification of the International Organization of Standardisation's ISO 3888-1 which can be illustrated in Figure 3-16. Based on this specification, a test track developed for the DLC test in IPG CarMaker is as shown in Figure 3-17. In this simulation, the vehicle drives forward with a constant speed of 90km/h, and steering input was applied to the vehicle so that the vehicle passes through the route defined by the red pylons. The steering input and wheel steer angle for this manoeuvre are shown in Figure 3-18.

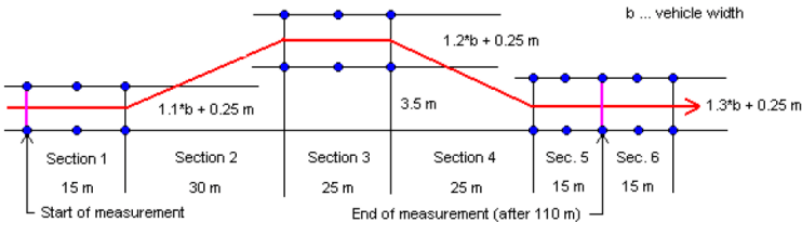


Figure 3-16: The schematic drawing of double lane change test based on ISO 3888-1 [322]

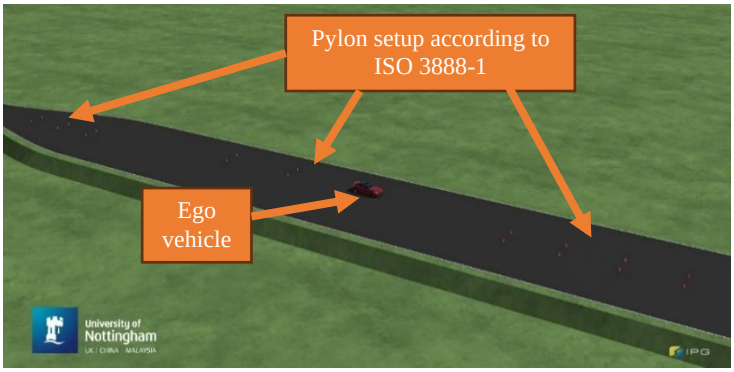


Figure 3-17: Double Lane Change test track developed in IPG CarMaker

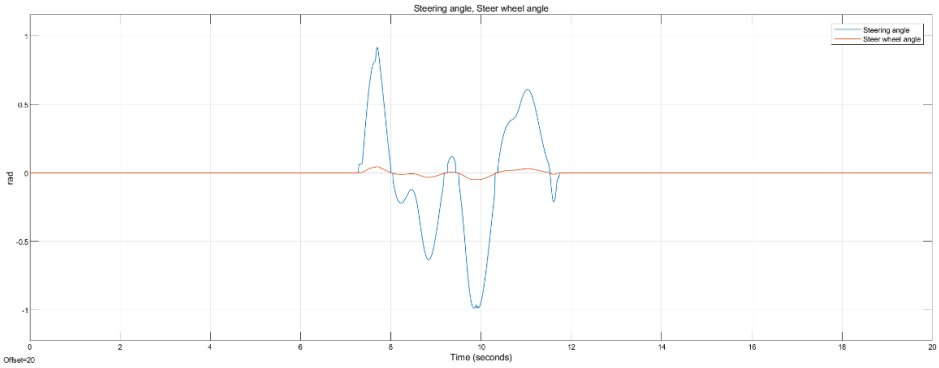
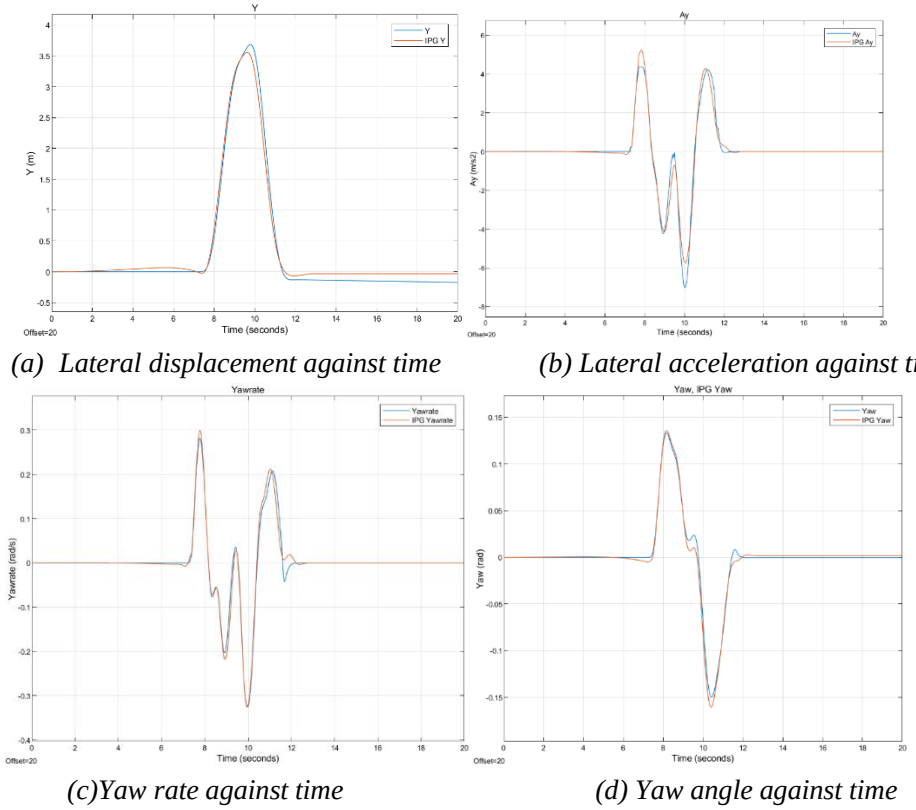


Figure 3-18: Steering-wheel angle and steer wheel angle simulated using IPG CarMaker

By providing the same steering input to the vehicle model in Simulink, the results of the handling model verification for double lane change test are shown in Figure 3-19(a) to Figure 3-19(d). The responses of the vehicle model that are investigated are the yaw rate, yaw angle, lateral acceleration, and lateral displacement.



(a) Lateral displacement against time *(b) Lateral acceleration against time*

(c)Yaw rate against time *(d) Yaw angle against time*

Figure 3-19: Double Lane Change test result

The comparative analysis of the vehicle response profiles from the Simulink model and the IPG CarMaker model reveals a high degree of correlation. Notably, the Simulink model's outputs align closely with those from the IPG CarMaker, exhibiting only marginal discrepancies. Quantitatively, the root mean square (RMS) errors for lateral displacement and lateral acceleration are calculated to be 3.617% and 2.587%, respectively. Similarly, the RMS errors for yaw rate and yaw angle are 2.245% and 4.163%, respectively. Given that these errors fall well within the permissible threshold of 15%, it substantiates the validity of the Simulink vehicle handling model in replicating the dynamic responses observed in the IPG CarMaker. The assumption of the chassis to behave as a rigid body could be one of the reasons caused these errors. This assumption inherently neglects certain high-frequency flexible dynamics and localized deformations that may occur in an actual vehicle, especially during aggressive manoeuvres. As a result, discrepancies arise between the simulated yaw motion and those from the IPG CarMaker.

In addition to the rigid body assumption, the suspension system is modelled using linear or quasi-linear approximations for its spring and damping characteristics. In reality, the suspension exhibits nonlinear behaviour over a broad range of operating conditions. Such simplifications can lead to deviations in the predicted load transfer during yaw motion, thereby contributing to the percentage error observed in Figure 3-19. Moreover, the tire forces—calculated using Pacejka's Magic Formula—are subject to uncertainties due to the empirical nature of the model. Although the Magic Formula has been widely adopted for its balance between accuracy and computational efficiency, it remains an approximation of the complex tire–road interactions and does not fully account for variations caused by factors such as tire

wear, temperature fluctuations, or road surface irregularities.

Furthermore, parameter uncertainties in key inputs—such as chassis moment of inertia, suspension stiffness, and damping coefficients—are known to affect the accuracy of the dynamic response. Sensitivity studies in vehicle dynamics literature have demonstrated that even minor variations in these parameters can result in significant differences in the roll moment balance [308]. Thus, the percentage error trend shown in Figure 3.19 reflects not only the cumulative effect of modelling assumptions and simplifications but also the inevitable uncertainties in parameter estimation.

In summary, it can be clarified that the percentage error trend represents the normalized difference between the simulation and experimental yaw dynamic variables. This error is primarily attributable to the rigid body assumption for the chassis, the linearization of the suspension behaviour, and the empirical approximations in the tire model, compounded by uncertainties in key model parameters. By analysing these sources of error, insight into the limitations of the current modelling approach is obtained and the potential area for improvement is identified, thereby enhancing the fidelity of future vehicle dynamics simulations.

Table 3-5: Percentage of RMS error for handling model double lane change test

Observation data	Root Mean Square (RMS)		Percentage of error (%)
	IPG CarMaker	Simulink	
Lateral displacement, Y	0.7233	0.7494	3.617
Lateral acceleration, a_y	1.142	1.113	2.587
Yaw rate	0.05458	0.05336	2.245
Yaw angle	0.02936	0.02814	4.163

3.9 Summary

In this chapter, a fourteen-degree-of-freedom vehicle model is derived. This model integrates several subsystems, including a seven-degree-of-freedom ride model, a seven-degree-of-freedom handling model, a lateral and longitudinal slip model, a powertrain model, and a steering model. Using this mathematical model, a Simulink-based vehicle model is developed. The Simulink model is then verified for vehicle ride and handling behaviours. The vehicle ride behaviour is validated by comparing the Simulink model's responses with those from IPG CarMaker's virtual vehicle, tested on a straight road with several bumps. The results show RMS errors of 6.089%, 2.845%, 0.2551%, and 5.051% for the normal forces acting on the front left, front right, rear left, and rear right wheels, respectively. For the handling behaviour, a double lane change test is conducted. The Simulink model's responses are compared with those from IPG CarMaker, yielding RMS errors of 3.617%, 2.587%, 2.245%, and 4.163% for lateral displacement, lateral acceleration, yaw rate, and yaw angle, respectively. As the RMS errors are all less than 15%, this demonstrates that the developed Simulink vehicle model can accurately represent the virtual vehicle model in IPG CarMaker under similar testing conditions.

Chapter 4: Validation of Vehicle Models using an Instrumented Vehicle

4.1 Overview

In the previous chapter, the detailed derivation of the vehicle model was explored to understand the lateral, vertical, and longitudinal dynamics. Based on the verification analysis of the developed vehicle model, the required system configuration to develop an instrumented vehicle was identified. In this chapter, an instrumented vehicle for data collection is developed using cost-effective sensors. The instrumented vehicle is developed using off-the-shelf sensors capable of synchronizing the recorded video data with other vehicle data as drivers operate the vehicle on public roadways.

A personally owned vehicle, Volkswagen Polo Sedan with a 1600 cc displacement engine and automatic transmission was selected as the instrumented vehicle due to budgetary constraints that precluded the acquisition of a dedicated research vehicle to ensure the feasibility of the study within the available financial resources. The vehicle has a weight of 1182 kg, a wheelbase of 2552 mm, a width of 1466 mm, hydraulic power rack and pinion steering, MacPherson Strut front suspension, and torsion beam rear suspension. The vehicle's braking system uses a hydraulic system with ventilated discs on the front axle and drums on the rear axle and features an Anti-Lock Braking System (ABS). In the next section, the setup for the instrumented vehicle will be discussed.

The following sections describe the testing procedure using the Society of Automotive Engineers (SAE) method and the validation of the 14DOF vehicle model based on the outcomes using the results from the instrumented vehicle during testing procedures. After the validation process, the vehicle is utilized to capture real-world driving data, which is essential for the validation of the vehicle dynamics model and the development of the driver model. Additionally, this chapter discusses the importance of adhering to international standards for scenario-based testing and classification, which are crucial for the safety assessment of autonomous vehicles. Finally, the chapter concludes with the preparation of the Malaysian Road Scenario Database (MaRSeD) and the classification of driving scenarios based on severity, contributing to the creation of a comprehensive testing environment for autonomous vehicle safety.

4.2 System configuration of cost-effective instrumented vehicle

For the development of instrumented vehicle, various types of sensors were mounted on the vehicle, as shown in Figure 4-1. The data acquisition system consists of two cameras (front and rear camera), a steering angle sensor, two pedal sensors (gas and brake pedal), an Inertial Measurement Unit (IMU) and a Global Positioning System (GPS). In order to gather and synchronize the sensor data, a laptop has been used as a Host Personal Computer (PC) for in the instrumented vehicle. Besides, the Host PC also used for the data logging process.

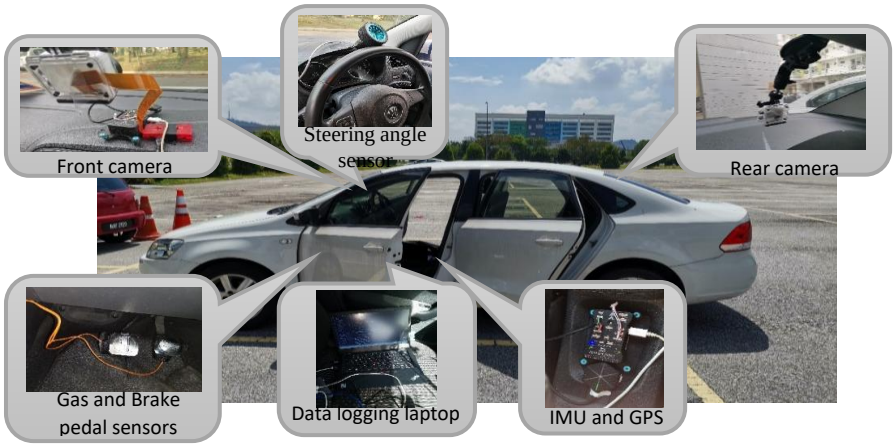


Figure 4-1 Sensor configuration of the instrumented vehicle

The camera sensors were installed on the front and rear of the vehicle to capture the surrounding traffic and road environment. The camera sensors were mounted on the windshield of the vehicle using suction cup windshield car mount. The camera sensors used are Raspberry Pi 8 Megapixels IMX219 camera from Sony with up to 30 fps when recording at full binned Field of View (FOV) resolution of 1640x1232 pixels. By default, the camera comes with 62.2 degrees of horizontal FOV lens which is too narrow for the usage of capturing surrounding environment. Therefore, the Pi camera's lens was replaced with a wide-angle lens with horizontal FOV of 160 degrees so that more information can be captured by the camera. The cameras were connected to Raspberry Pi single board computer (SBC) through MIPI CSI connection. However, the Raspberry Pi SBC only has a single MIPI CSI port on-board. So, in this case, two Raspberry Pi SBC are required for two cameras. Raspberry Pi Zero SBC is selected in this study for cost effective, low-power (can be powered from laptop's Universal Serial Bus (USB) port) and has sufficient computation power for video recording task. The SBC encoded the captured raw data from the camera into motion jpeg video format and stream the video to the laptop through USB communication. Figure 4-2 shows a sample frame of the environment captured using the front and rear camera on the instrumented vehicle.

Figure 4-2 Front and rear camera footage captured by the instrumented vehicle.

Meanwhile, the steering wheel angle was measured using an optical encoder. The optical encoder used is RE30E-500-213 from Nidec which provides up to 500 Pulse per Revolution (PPR). The 500 pulse per revolution can be referred as one pulse is equivalent to 0.72 degrees. To configure the optical encoder for angular position measurement of the steering wheel, an Arduino Uno microcontroller is used to obtain pulse signal from the sensor and convert the pulse signal into angular position. The optical encoder has four wires, "Power +"

and “Power –” for powering the sensor and two signals namely output “A” and output “B” for encoder signals. The hardware connection of the optical encoder to the Arduino is shown in Table 4-1.

Table 4-1 Physical connection of Steering wheel optical encoder sensor to Arduino

Wire Colour	RE30E-500-213	Arduino Pin
Red	Power +	5V
Black	Power -	Ground
White	Output “A”	2 (hardware interrupt)
Green	Output “B”	3 (hardware interrupt)

According to the datasheet of the RE30E, the pulse signal is depicted as a square wave. There are a total of four phase changes in the optical encoder output: two for output "A" and two for output "B," as illustrated in Figure 4-3. Hardware interrupt pins on the Arduino are used to track any changes in the pulse signal from outputs "A" and "B". A hardware interrupt is a signal sent to the processor that immediately stops the current code execution to handle an urgent task. This mechanism allows the processor to respond quickly to external events, such as changes in the pulse signal from the optical encoder, without wasting time constantly checking the signal state. The interrupt service routine (ISR) is triggered at the rising edge of the pulse, and it increments or decrements a count value based on the direction of rotation. Since the interrupt is triggered twice for a single pulse, the optical encoder's resolution is 0.36 degrees per count. To convert the encoder's angle of rotation to the steering wheel's angle of rotation, the diameters of both the encoder and the steering wheel are measured to calculate the transmission ratio, much like calculating a gear ratio, as shown in Figure 4-5. Therefore, the steering wheel angle is determined by multiplying the number of encoder’s counts by the encoder's angular resolution and the transmission ratio as shown in Equation 4.1. The measured angular position is then transmitted to a laptop for data logging via USB serial communication.

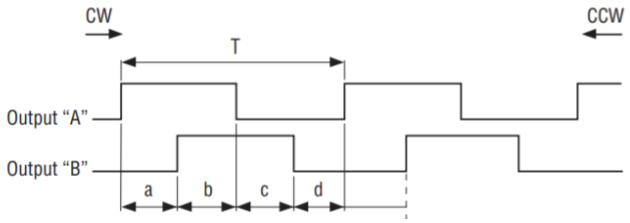


Figure 4-3 Phase relationship between Output "A" and Output "B" to determine the direction of rotation

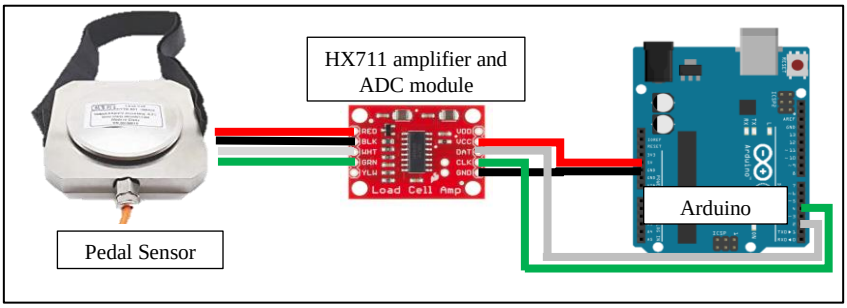


Figure 4-4 Physical connection of Pedal sensor-HX711-Arduino

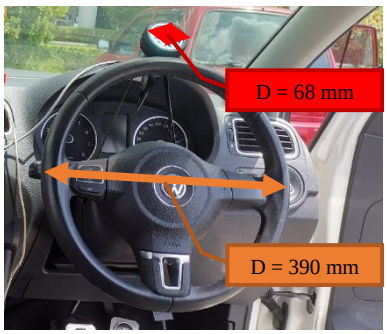


Figure 4-5 Transmission ratio from the steering wheel to the optical encoder

$$Steering\ Wheel\ Angle = number\ of\ counts \times 0.36°\ per\ count \times \frac{68\ mm}{390\ mm} \tag{4.1}$$

The gas and braking pedal forces applied by the driver was measured by using load cells that were fixed on the gas and braking pedal. In this research, the DYZ-105 sensor which is specially designed for pedal force measurement was used to obtain the pedal force applied by the driver. The operating range of the pedal sensor is from 0 N to 2000 N. To read the output from the sensor, a load cell amplifier, HX711, is used. The HX711 reads changes in the resistance of the load cell and amplify the analogue signal. It then uses its built-in 24-bit Analog-to-Digital Converter (ADC) to convert the analogue signal into digital data. An Arduino Uno is used as a bridge to connect the laptop to the HX711 through I2C interface. The hardware connection of the pedal sensors is shown in Figure 4-4 and the definition of each connection is defined in Table 4-2. Interrupt service routine is used to obtain the latest available data converted from analogue signal from the HX711. The data obtained is then transmit to the laptop through USB serial communication for data recording.

Table 4-2 Definition of connection between Pedal sensor, HX711 and Arduino

Wire Colour	DYZ-105	HX711 Amplifier	HX711 ADC	Arduino Pin
Red	Power +	Excitation +	Vcc	5V
Black	Power -	Excitation -	Ground	Ground
Green	Signal +	Amplifier +	SCK	4
White	Signal -	Amplifier -	DT	2 (hardware interrupt)

The load cell in the pedal sensor is highly sensitive and this causes the outcome will vary based on external inputs. Therefore, pedal sensors need to be calibrated before it can be used for data collection process. Since strain gauges linearly relate strain to force applied, a linear relationship can be used to calibrate the load cell. The equation representing this relationship is defined in Equation 4.2.

$$y = mx + b \tag{4.2}$$

Where y is the value of the load applied on the sensor,

x is the HX711's ADC value,

m is a constant value which represent the slope of the calibration line,

b is the tare value when the load applied on the sensor, y is zero.

To calibrate the pedal sensor, two calibration points are required: the zero value and a known mass value. First, the pedal sensor is powered on, and the calibration is initiated using Arduino without any force applied to the sensor to obtain the tare value from the HX711's ADC. Then, an object with a known mass is placed on the sensor, and the sensor reading is measured based on the ADC value from the HX711. This data is used to calculate the slope of the calibration line using the Arduino. The tare and calibration values are saved to the Arduino's EEPROM to avoid re-calibration for future use with similar sensors under the same configuration. Figure 4-6 shows the serial monitor printout during the calibration process of the pedal sensor.

To obtain the attitude of the vehicle, a Pixhawk (PX4) controller which consists of ST Micro L3GD20 3 axis gyroscope, ST Micro LSM303D 3-axis accelerometer/magnetometer and Invensense MPU-6000 accelerometer/gyroscope is used in this study. Besides, a U-blox m8n GPS is connected to the PX4 controller to access GPS data of the vehicle. The m8n GPS is a cost effective low powered GPS when compared to centimetre-level GPS such as Real-Time Kinematic (RTK) or Trimble Real-Time eXtended (RTX) GPS which are very expensive. However, the m8n still able to deliver high precision and sub-metre accuracy up to 0.6 meters.

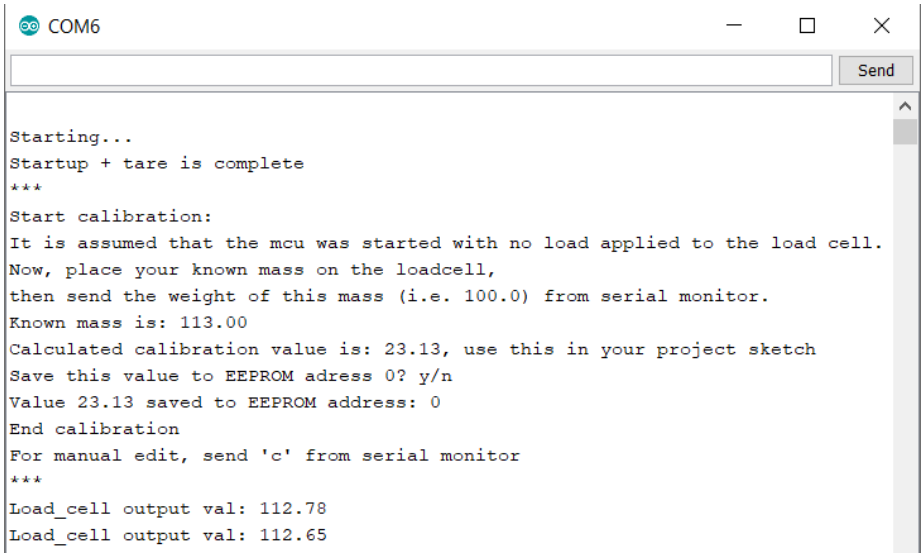


Figure 4-6 Calibration process of pedal sensor via Arduino serial monitor

The laptop namely, Lenovo ThinkPad is used as the HOST PC for data recording process. The laptop is designed using an Intel Core i5 10th gen processor with 8 GB of Random-access Memory (RAM). Since video data recording task requires high disk write throughput, a one Tera Bytes (TB) Solid State Drive (SSD) with write speed of 500 MB/s was used for data storage. For connectivity with the sensors, the laptop however only supports up to two USB 3.0 Type A ports and one USB 3.1 Type C port while the sensor setup required up to five USB ports. Therefore, an external USB hub is used to provide additional USB ports required. The laptop also has a built-in 2.995 Ah / 46 Wh battery that can be used to power the laptop, as well as sensors through USB port's 5V power line without any other external power supply. Despite having small sized battery, by using low powered laptop, SBCs and MCUs in the setup, the built-in battery was able to provide runtime of up to 3 hours in a single full charge which is sufficient for most data recording applications. Table 4-3 shows the power

consumption measured during data recording. The connection of the sensors to the laptop for data recording is shown in Figure 4-7.

Table 4-3 Power consumption of data acquisition system measured during operation.

Component	Voltage, V	Maximum Current, A	Power Consumption, W
Laptop	17.2	0.638	10.974
Raspberry Pi Zero + camera	5	0.385	1.925
Arduino Uno + 2x pedal sensor	5	0.058	0.290
Arduino Uno + steering sensor	5	0.062	0.310
Pixhawk + GPS	5	0.840	4.200

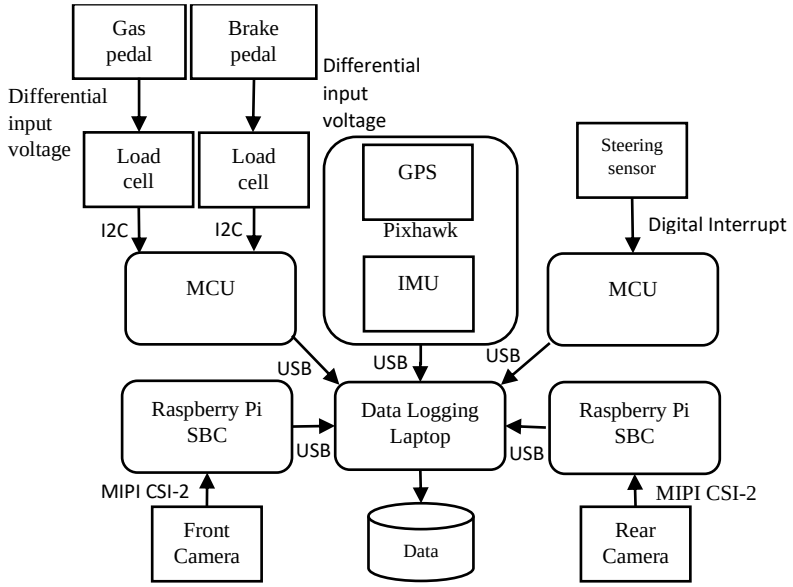


Figure 4-7 System architecture of data acquisition system

4.3 Sensor Synchronization and Sensor Fusion

Since every sensor mentioned in the previous section have different sampling data rate as shown in Table 4-4, there's a need to synchronize these sensors. All the sensors were synchronized to the Network Time Protocol (NTP) time which is synchronized to the Coordinated Universal Time (UTC) time reported by the laptop. For each sensor, a buffer was created to hold the latest readings reported by the sensor. To update all the sensors in simultaneously, threaded processes were created for each sensor so that each sensor can update the reading at their sampling rate. Another threaded process sampled at 100 Hz to match the sampling rate of the steering encoder. It captured all sensor data from the buffers, recorded the sensor data, and presented timestamps in the database. The file was named in the format: Sensordata-YYYY-MM-DD-HH-MM-SS.csv. Each data point recorded in the file has nine fields: *t, lat, lon, pitch, roll, yaw, gas, brake, steer*. The field t represents the timestamp; lat and lon represent the latitude and longitude data obtained from the GPS; $pitch$, $roll$ and yaw represent the attitude data obtained from the PX4 IMU; gas and $brake$ represent the gas and braking force measured by the pedal sensors; and $steer$ represents the steering angle obtained from the steering angle sensor.

Table 4-4 Output signal of sensors

Sensor	Specification	Sampling Rate	Channel	Data Captured
Camera	Raspberry Pi 8MP Sony IMX219 Camera	30 fps	2	Front and rear view of road
Steering sensor	RE30E-500-213-1 optical encoder	100 Hz	1	Steering angle
Pedal sensors	DYZ-105 force sensor	80 Hz	2	Gas and brake pedal force applied
IMU	Pixhawk L3GD20 3-axis 16-bit gyroscope, LSM303D 3-axis 14-bit accelerometer with magnetometer and MPU6000 3-axis accelerometer/gyroscope	25 Hz	1	Linear acceleration and angular velocity in x-y-z axis
GPS	U-BLOX NEO-M8N GPS with compass	10 Hz	1	Lateral and longitudinal position, and heading

For camera video recording, a free and open-source software, Open Broadcaster Software (OBS), was used. An OBS plugin was added to the software to insert a timestamp into each frame of the recorded footage. Since the raw camera data requires video encoding to be saved as a video file, there is a time delay between the real world and the recorded video. To measure the latency from the camera to the laptop, the camera was pointed at a laptop screen displaying a stopwatch application. The time difference between the stopwatch displays and the time captured by the camera was noted. For instance, if the stopwatch displayed 1:43.20 and the camera perspective showed 1:43.09, the latency from the camera to the laptop would be 0.11 seconds, as shown in Figure 4-8. Furthermore, the latency between the front and rear cameras can be measured using the same method. As shown in Figure 4-9, the time captured by both cameras is the same, indicating that the latency between the two cameras is less than 10 milliseconds and negligible. The recorded data is separated into different folders and identified by the location, date, and time at which they were recorded with the format of "*Location-YYYY-MM-DD-HH-MM"*. The data for each recording is organized as shown below:

- Location-YYYY-MM-DD-HH-MM
    - YYYY-MM-DD-HH-MM-SS.mkv
    - Sensordata-YYYY-MM-DD-HH-MM-SS.csv

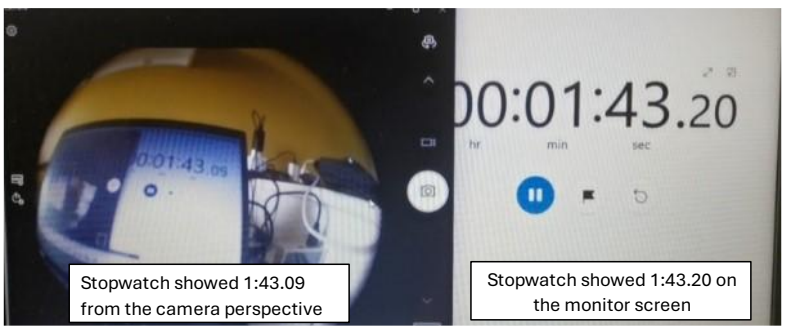
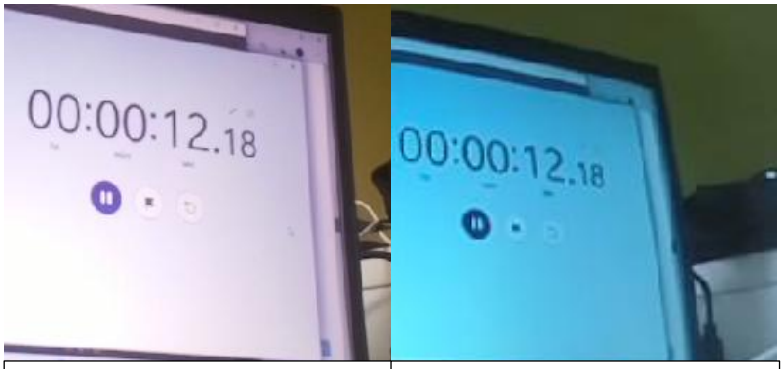


Figure 4-8 Camera facing monitor screen displaying a stopwatch application



Stopwatch showed 12.18 seconds from the front camera perspective	Stopwatch showed 12.18 seconds from the rear camera perspective

Figure 4-9 Latency between front and rear camera

To further optimize the latency when using a non-real-time operating system, the data capture pipeline was improved by employing a multithreaded process for each sensor node and a zero in-memory copy approach. This approach is similar to several methods based on the Robot Operating System (ROS) used in autonomous driving system to reduce latency and persist raw data [323], [324], [325]. By using multithreading, each sensor node can operate independently, reducing the processing time. The zero in-memory copy technique minimizes data duplication, further decreasing latency and improving overall system performance.

4.4 Sensor Fusion for Instrumented Vehicle

Sensor fusion is a process of combining data from multiple sensors, such as camera, IMU, and GPS, to obtain a more accurate and reliable estimate of the state of a system. In this study, the sensor fusion is done using ROS (Robot Operating System), which is a framework that provides tools and libraries for developing robotics applications. The reason for choosing ROS is because ROS has several packages that can be used for sensor fusion, such as *robot_localization*, *imu_tools*, and *navsat_transform_node*. One of the applications of sensor fusion using ROS is to assist with the different driving and weather conditions. For example, sensor fusion can help to improve the localization and navigation of autonomous vehicles by fusing data from camera, IMU, and GPS sensors. Camera sensor can provide visual information about the environment, such as lane markings, traffic signs, and obstacles. Meanwhile, IMU can measure the linear and angular acceleration of the vehicle, which can be used to estimate its orientation and velocity. On the other hand, GPS can provide the global position of the vehicle, which can be used to plan the route and avoid collisions. However, each sensor has its own limitations and uncertainties. Camera can be affected by lighting conditions, occlusions, and distortions while IMU can be affected due to drift and noise. Besides, GPS can have errors due to multipath effects, signal loss, and interference. Therefore, sensor fusion using ROS can help to overcome these challenges by combining the data from different sensors and applying filtering techniques, such as Kalman filter or particle filter. The sensor fusion helps to reduce the noise and errors from the sensors. Moreover, the sensor fusion able to improve the accuracy and consistency of the estimation for the state of the sensor fusions.

One of the most popular packages for sensor fusion in ROS is called as *robot_localization*, which provides nonlinear state estimation nodes that can fuse multiple sensor inputs using various flavours of Kalman filters. A Kalman filter is a recursive algorithm designed to estimate the state of a dynamic system based on noisy measurements by using a prediction-correction scheme. The Kalman filter can also handle uncertainties in the sensor data by using covariance matrices that represent the measurement errors and process noise. Meanwhile, *rospy*, a python client library for ROS is used to convert the raw sensor data into these sensor messages. In this study, *robot_localization* is used to fuse data from camera, IMU and GPS sensors for localization of the instrumented vehicle. In order to implement sensor fusion using *robot_localization*, several steps need to be taken, including data preparation, configuration of the Kalman filter nodes, and parameter tuning to achieve optimal performance. The first step is to ensure that the sensor data is published as ROS messages in the appropriate format and frame. To accomplish this, rospy, a python client library for ROS is used to convert the raw sensor data into these sensor messages:

- For camera data, the images are published as *sensor_msgs/Image* messages, with the camera frame designated as the header *frame_id*.
- For IMU data, the orientation, angular velocity, and linear acceleration are published as *sensor_msgs/Imu* messages, with the IMU frame specified as the header *frame_id.*
- For GPS data, the latitude, longitude, and altitude are published as *sensor_msgs/NavSatFix* messages, with the GPS frame assigned as the header *frame_id.*

Next, it is necessary to establish transforms between the camera frame and the IMU frame, as well as between the IMU frame and the GPS frame. This can be achieved using a tool provided by ROS which is the *tf2_ros::StaticTransformBroadcaster*. To use this tool, the transform between the camera frame and the IMU frame needs to account for any physical offset or rotation existing between the two sensors. Similarly, the frame transformation between the IMU frame and the GPS frame needs to account for any variations in magnetic declination or geodetic datum between the two sensors. To accomplish this, the mounting position of each sensor on the instrumented vehicle needs to be measured. Additionally, to link the camera frame with the odometry frame, *viso2_ros package* which provides a visual odometry algorithm is used in this configuration. The visual odometry algorithm estimates the motion of the camera from consecutive camera images and then publishes it as an odometry message.is used. The visual odometry algorithm estimates the motion of the camera from consecutive camera images and then publishes it as an odometry message.

Lastly, the Kalman filter nodes that will fuse the sensor data is configured. The *robot_localization* package provides two types of Kalman filter nodes, namely the *ekf_localization_node* and *ukf_localization_node*. The former uses an extended Kalman filter (EKF), which is suitable for linear or mildly nonlinear systems. The latter uses an Unscented Kalman Filter (UKF), which is suitable for highly nonlinear systems. In this study, the *ekf_localization_node* as EKF is utilized even though UKF has higher accuracy because EKF is more straight forward to implement and require less computation power [115].

In low-light or adverse weather conditions, the camera data may exhibit noise or unreliability. However, the IMU and GPS data can still provide accurate information. In such

cases, the weights of the IMU and GPS data in the Kalman filter can be increased to rely more on their information. Conversely, in urban or indoor environments, the GPS data may either be unavailable or imprecise, while the camera and IMU data remain valuable. In this scenario, it is recommended to assign higher weights to the camera and IMU data within our Kalman filter to rely more on their information. To adjust the weights of different sensor sources in the Kalman filter, their covariance matrices can be modified. A lower covariance value means a higher weight, and vice versa. For example, if the weight of IMU data need to be increased, its covariance values in *imu0_covariance* parameter can be decreased. If the weight of GPS data needs to be decreased, its covariance values in Navigational Satellite (NAVSAT) parameter can be increased.

4.5 Vehicle Model Validation using Instrumented Vehicle

In order to calibrate vehicle models using the sensors equipped in the instrumented vehicle, several field tests were performed according to SAE vehicle safety testing standard. This calibration is required to ensure the vehicle response and driving data recorded by the sensors are accurate and reliable. Three field test scenarios were used which are the double lane change test, step steer test, sudden braking test. All the tests were carried out at parking lot of Asia Pacific University of Technology and Innovation (APU). The open space has a dimension of 100 x 60 m of asphalt road. Road items such as traffic pylons were used to define the test tracks according to the ISO 3888-2 SAE standard for double lane change test, step steer lateral test according to ISO 4138, and braking longitudinal test according to ISO 3833 SAE standard.

4.5.1 Design of Field Test

The tests are carried out starting at a constant speed of 20 km/h and increasing the speed of 10 km/h between experiments until 30 km/h. The reason for limiting the vehicle speed for not greater than 30 km/h is due to the constrained length of the test track, which prevents safety testing at higher speeds. Additionally, urban areas generally experience heavy traffic, causing vehicles to travel at idle or low speeds. Moreover, in the Fourth United Nation Global Road Safety Week, World Health Organization has released a Managing speed document to call policymaker for reduction in urban maximum road travel speed to 30 km/h [326]. Figure 4-10 shows an outline of the SAE tests performed where there are three types of manoeuvre test for two test speeds. Altogether 6 test cases and each test cases were repeated for three times to ensure the result is consistent, reproducible, and more reliable.

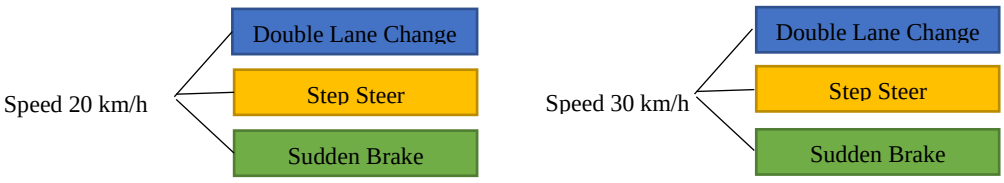


Figure 4-10 Test carried out according to SAE standards.

(a) Double Lane Change Test

In double lane change (DLC) test, the vehicle accelerates until the vehicle's speed reach a constant speed of 20 km/h and 30 km/h. Then, steering input is applied by the driver to the vehicle so that the vehicle passes through the route defined by the traffic pylons. The test track was setup according to the ISO 3888-2 standard for double lane changes as shown in Figure 4-11. Meanwhile, Figure 4-12 shows the instrumented vehicle carried out double lane change test. All sensor data that were collected throughout the test was recorded in storage to be analysed later.

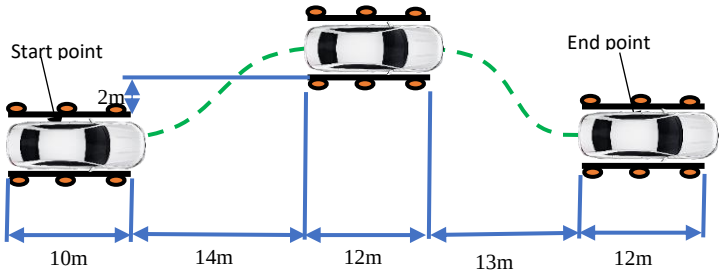


Figure 4-11 ISO 3888-2 test

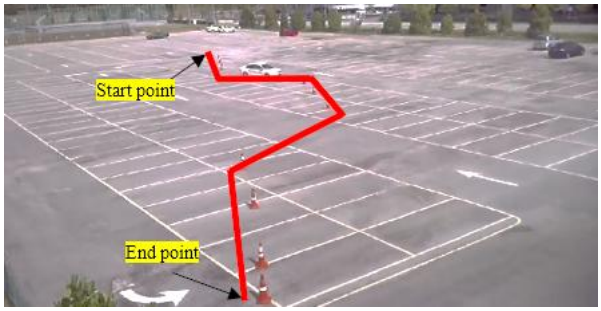


Figure 4-12 Double Lane Change field test

(b) Step Steer Test

In step steer test, the vehicle first moves forward with a constant speed then a constant steering angle is applied to the vehicle so that the vehicle turns left following the circular route defined by the traffic pylons. When the vehicle reached the exit, the constant steering angle is released so that the vehicle moves in a straight line. The testing is conducted at two different speeds such as 20 km/h and 30 km/h. The test track was setup according to the SAE ISO 4138 standard for lateral step steer test as shown in Figure 4-13. Meanwhile, Figure 4-14 shows the instrumented vehicle carried out step steer manoeuvre. All sensor data that were collected throughout the test was recorded in storage to be analysed later.

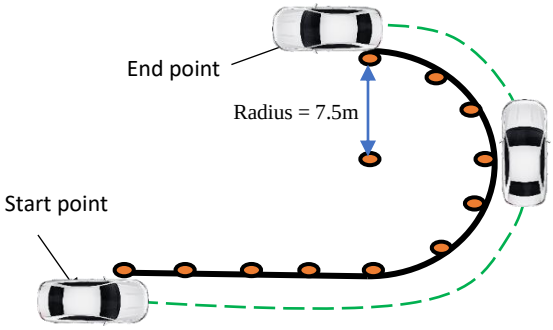


Figure 4-13 ISO 4138 test

(c) Braking Test

The test is begun with vehicle at stationary position at the starting point. Then, a full throttle is applied by the driver until the vehicle reaches the required speed. Then, a sudden braking with full pedal input is applied when the vehicle passes the vehicle braking point to stop the vehicle. This test is conducted according to the ISO 3833 longitudinal vehicle test as shown in Figure 4-15. Meanwhile, Figure 4-16 shows the instrumented vehicle performing accelerate and braking test. The gyroscope and the accelerometer inside the PX4 microcontroller are used in this test to record the acceleration and the pitching angle of the vehicle for the validation test.

Figure 4-14 Step steer field test

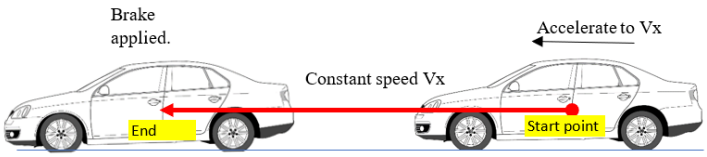


Figure 4-15 ISO 3883 test

Figure 4-16 Braking test.

4.5.2 Results validation process from Instrumented Vehicle

In order to validate the simulated vehicle with data recorded from instrumented vehicle, the same test tracks used in the field test were created in the vehicle simulation platform. IPG CarMaker, which is commonly used by automotive industry and validated in [327] using an instrumented Ford Fiesta car to perform lane change test is used. The test tracks created are as shown in Figure 4-17 for DLC test, Figure 4-18 for Step Steer test and Figure 4-19 for Braking test.

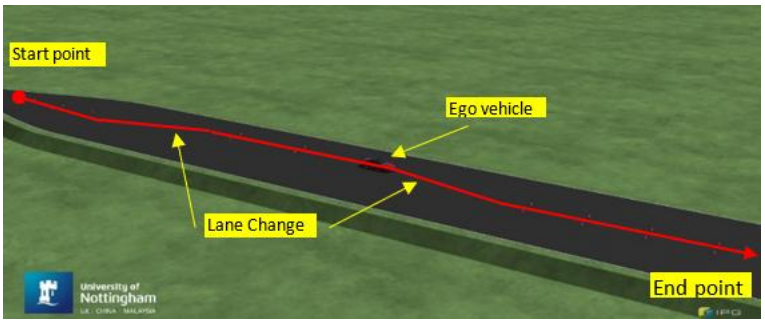


Figure 4-17 Double Lane Change test track developed using IPG CarMaker

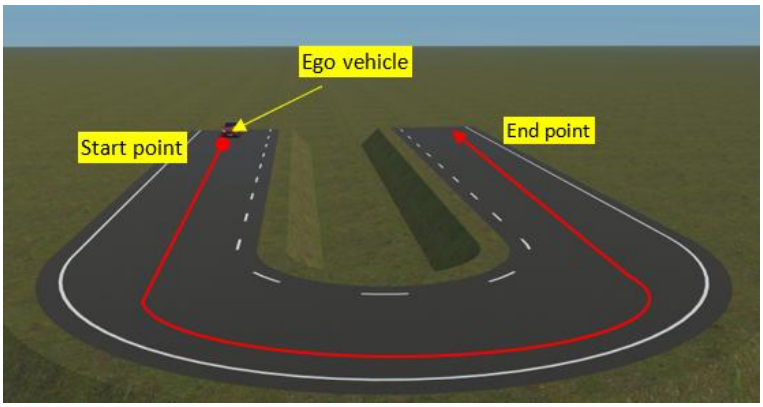


Figure 4-18 Step Steer test track developed using IPG CarMaker

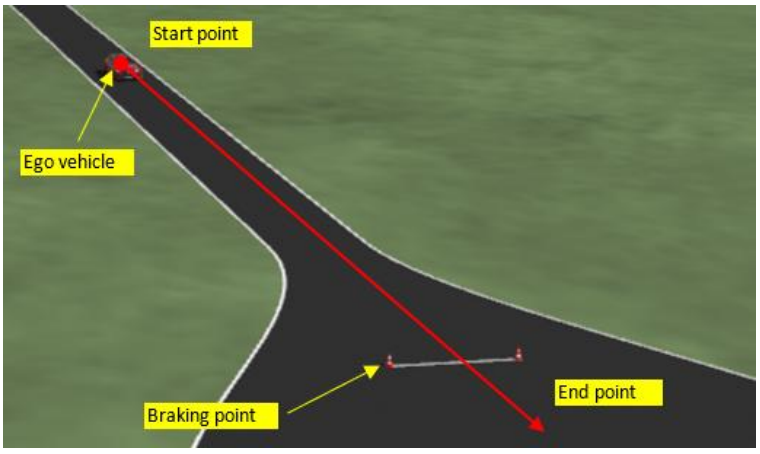


Figure 4-19 Braking test track developed using IPG CarMaker

The tests were carried out in the simulator with the same setting of the field test. The vehicle data from the IPG CarMaker simulation software were exported to compare with the

data recorded from the instrumented vehicle to validate the accuracy and reliability of the virtual vehicle model from IPG CarMaker and the 14 DOF vehicle model developed in the Chapter 3.

4.5.3 Validation of vehicle model using data from Instrumented Vehicle

In this section, the accuracy and validity of simulation are investigated by comparing the responses of the Simulink-based vehicle model and a virtual vehicle in IPG CarMaker to the data from the instrumented real vehicle. Three critical tests were performed: the Double Lane Change (DLC) Test, the Step Steer Test, and the Braking Test. These tests evaluate the lateral stability, yaw dynamics, and braking force distribution of the vehicle, ensuring that the models provide a reliable representation of real-world vehicle responses.

Identical human driving inputs, such as steering angle, throttle, and brake inputs, were provided to both the virtual vehicle in IPG CarMaker and the 14-degree-of-freedom vehicle model in Simulink validated in previous research [313]. The vehicle states obtained from these models were then compared with sensor data from the instrumented vehicle. This comparison was conducted for specific driving manoeuvres, including double lane change tests, step steer tests, and braking tests at constant speeds of 20 km/h and 30 km/h.

(a) Double Lane Change Test

The double lane change test is a critical manoeuvre used to evaluate the dynamic handling and stability characteristics of a vehicle. Comparing vehicle states in these tests provides comprehensive insights into the vehicle's performance under rapid lateral manoeuvres. The vehicle states that are investigated for the DLC test are the local position, lateral acceleration and yaw rate of the vehicle because these metrics are critical indicators of vehicle dynamics and handling performance. Comparing these specific results allows for a detailed assessment of how accurately the simulated models replicate the real vehicle's behaviour under different driving conditions.

For instance, the local position tracks the vehicle's movement in the lateral direction. It is crucial for understanding how accurately the vehicle can follow the intended path during a lane change. By comparing the local position, the precision of the simulated vehicle's path-following capabilities can be evaluated. Any deviations from the real vehicle's path can indicate areas where the simulation model needs improvement. While the yaw rate is the rate at which the vehicle rotates around its vertical axis. It is a critical parameter for evaluating turning performance and the vehicle's ability to maintain a stable trajectory during rapid directional changes. Comparing yaw rates helps determine how well the simulations replicate the real vehicle's rotational dynamics, which is essential for predicting the vehicle's behaviour during quick manoeuvres. Whereas the lateral acceleration measures the vehicle's ability to change direction laterally and is directly related to the grip and stability of the vehicle. It reflects how quickly and efficiently the vehicle can respond to steering inputs without losing traction. By comparing the lateral acceleration between the simulated models and the real vehicle, the

accuracy of the simulations in replicating real-world cornering forces and vehicle stability can be assessed.

The double lane change test required approximately one minute for each execution. Initially, the driver accelerated the instrumented vehicle to and maintained the required constant speed. Upon reaching the starting point of the test, the driver initiated the double lane change manoeuvre. Each test was repeated three times to ensure the consistency and reliability of the results, crucial for validating the simulated vehicles. To ensure clarity, only the vehicle states during the double lane change manoeuvre are presented in this section, with the acceleration segment excluded from the results. The result for double lane change at constant speed of 20 km/h is shown in Figure 4-22, Figure 4-23 and Figure 4-21. Whereas the result for double lane change at 30 km/h is shown in Figure 4-26, Figure 4-27 and Figure 4-25.

i. *Results for Double Lane Change test for vehicle constant speed of 20 km/h*

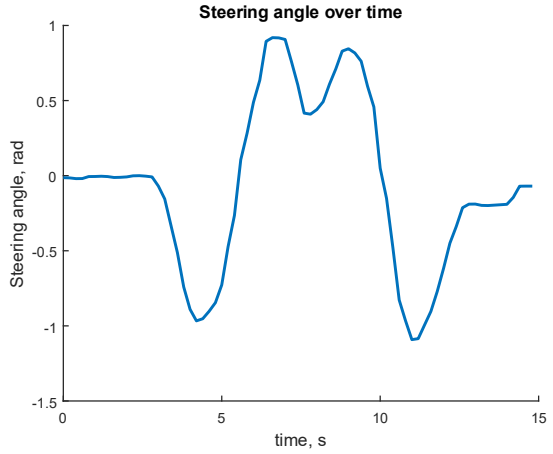


Figure 4-20 Steering angle for double lane change at 20 km/h

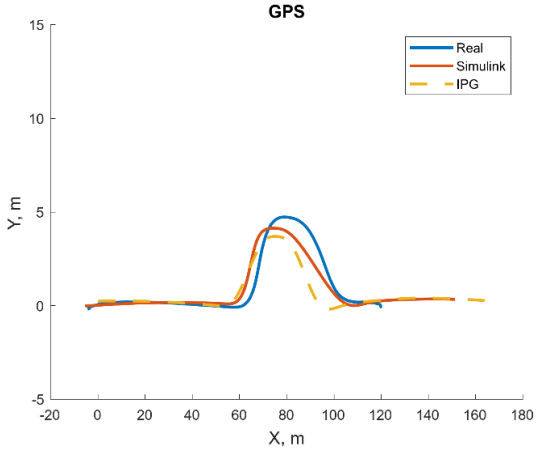


Figure 4-21 Local position plot for double lane change at 20 km/h

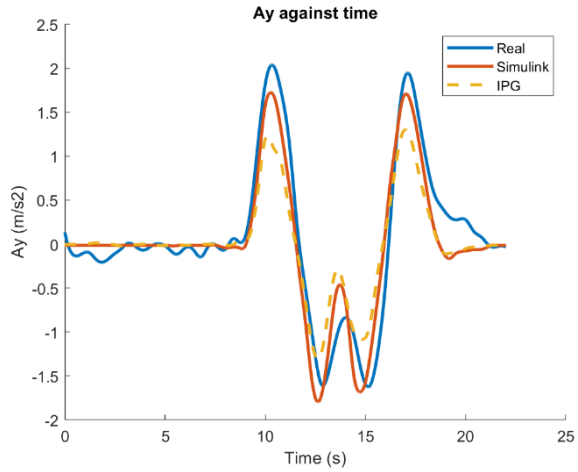


Figure 4-22 Lateral acceleration for double lane change at 20 km/h

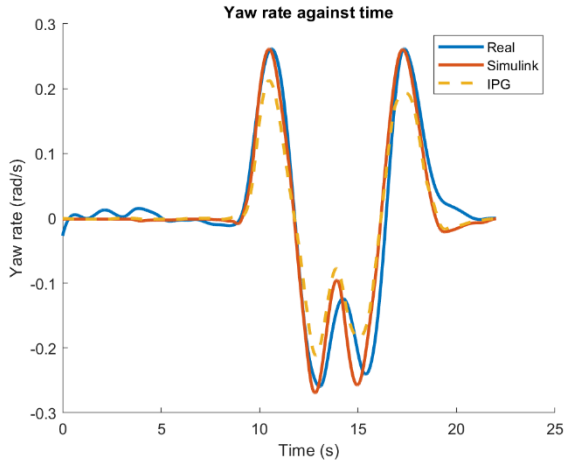


Figure 4-23 Yaw rate response for double lane change at 20 km/h

ii. Results for Double Lane Change test for vehicle constant speed of 30 km/h

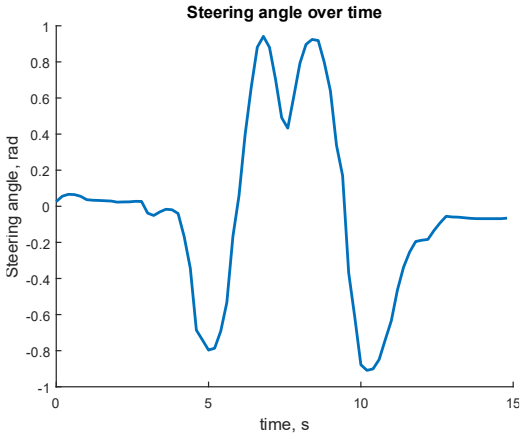


Figure 4-24 Steering angle for double lane change at 30 km/h

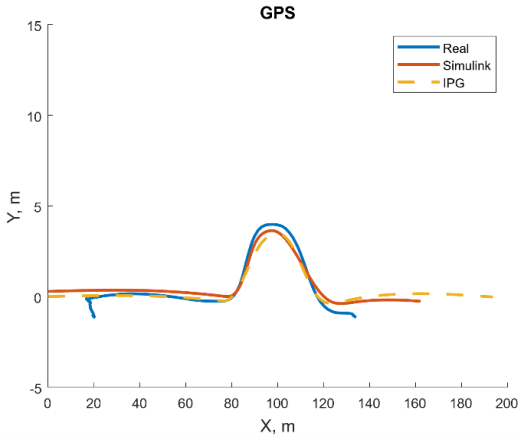


Figure 4-25 Local position plot for double lane change at 30 km/h

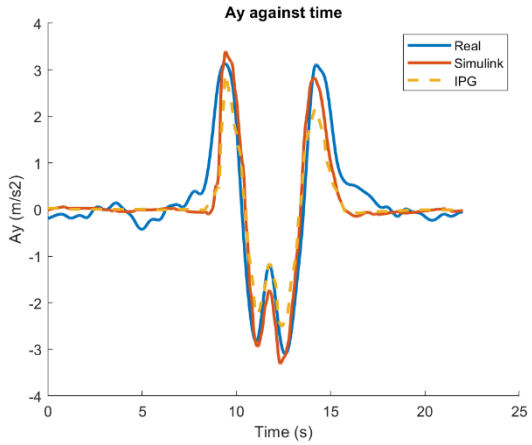


Figure 4-26 Lateral acceleration for double lane change at 30 km/h

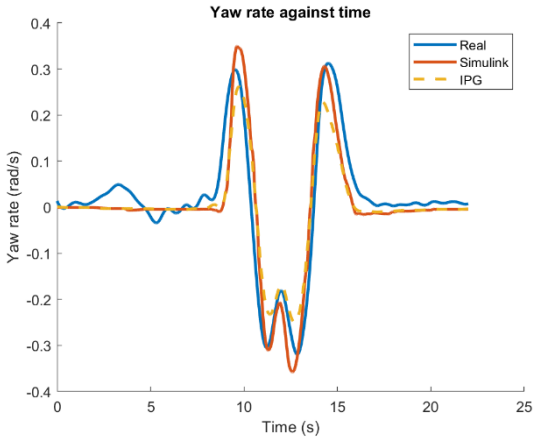


Figure 4-27 Yaw rate response for double lane change at 30 km/h

Table 4-5 Root Mean Square Error for30 km/h double lane change data.

Root means square error		
20 km/h	Lateral acceleration, m/s^2	5.216
	Yaw rate, rad/s	4.527
	Position, m	7.239
30 km/h	Lateral acceleration, m/s^2	5.179
	Yaw rate, rad/s	4.932
	Position, m	5.167

From the plot, it can be observed that the vehicle responses from both Simulink model and virtual vehicle from IPG CarMaker shared the similar characteristic with the vehicle state data obtained using the instrumented vehicle. Table 4-5 shows the percentage of RMS values for the overall responses from the sensor measurements when compared to the vehicle response from Simulink model developed. The lateral acceleration, yaw rate and displacement of the vehicle moving at 20 km/h show the percentage difference of RMS errors about 5.22%, 4.53% and 7.24%, respectively. Whereas the lateral acceleration, yaw rate and displacement of the

vehicle moving at 30 km/h show the percentage difference of RMS errors about 5.18%, 4.93% and 5.17%, respectively. It can also be observed that the real vehicle is much more responsive and have a much obvious peak and low value. Figures 4-20 and 4-24 illustrate the applied steering angles for the two speed conditions. It is observed that at 30 km/h, the magnitude of the steering input is larger compared to 20 km/h due to the increased lateral forces acting on the vehicle. The local position trajectories of the instrumented vehicle, shown in Figure 4-21 and Figure 4-25, align well with the simulated models, demonstrating that both models are capable of accurately replicating real-world lane change behaviour. However, minor deviations occur, particularly during the exit phase of the manoeuvre, where the real vehicle stabilizes slightly later than the simulations. This discrepancy can be attributed to variations in tire-road friction, unmodeled aerodynamic effects, and slight inconsistencies in the driver's input timing. The lateral acceleration and yaw rate responses during the DLC test, as presented in Figure 4-22 to Figure 4-27, indicate a strong correlation between the simulated and real-world results. The lateral acceleration peaks at the midpoint of the manoeuvre, where the highest lateral forces are encountered. The Simulink model slightly underestimates lateral acceleration, whereas the IPG CarMaker model overestimates the peak values. A similar trend is observed for yaw rate, where the peak values in the IPG CarMaker model are slightly higher than those in the real vehicle and Simulink model. These discrepancies arise due to differences in tire modeling, suspension compliance, and vehicle mass distribution. The Pacejka tire model, used in both simulation environments, assumes a continuous slip ratio, whereas real-world tires exhibit transient slip variations that influence lateral acceleration and yaw rate. Additionally, the instrumented vehicle experiences minor variations in weight distribution due to the presence of the driver and data logging equipment, which are not fully captured in the simulations. However, the root means square error (RMSE) calculated between the real model and IPG model for both 20km/h and 30km/h double lane change test are still under acceptable range which is less than 10%. Hence, it is shown that the simulated vehicle as developed in the Chapter 3 is valid and able to produce similar response to real vehicle during the transient state. Besides, it also shows that the instrumented vehicle is capable of capturing vehicle state when performing lateral manoeuvre.

(b) Results of Step Steer Test at 60 Degree

The step steer test evaluates the vehicle's transient response to sudden steering inputs, providing crucial insights into its dynamic behaviour. First, the vehicle is accelerated to the desired constant speed. After reaching this speed, the vehicle is stabilized to ensure consistent motion. Next, a sudden and decisive steering input is applied, such as turning the steering wheel to a specific angle. This manoeuvre is designed to test the vehicle's dynamic response to abrupt steering changes. To ensure the reliability of the results, the test is repeated three times, to ensure consistency.

Similar to the double lane change test, parameters such as lateral acceleration, yaw rate, and local position are compared to assess the simulation's accuracy in replicating real-world vehicle dynamics during these manoeuvres. The local position determines how well the simulated vehicle follows the intended path during sudden steering inputs, highlighting any

discrepancies with the real vehicle's behaviour. The yaw rate however helps to assess the accuracy of the simulation in predicting the vehicle's rotational behaviour under sudden steering manoeuvres. Finally, lateral acceleration evaluates how accurately the simulated vehicle replicates the real vehicle's ability to change direction laterally and reflects its dynamic stability during steering manoeuvres.

For step steer test, the results obtained at constant speed of 20 km/h is shown in Figure 4-30, Figure 4-31 and Figure 4-29. Whereas the result for step steer test at 30 km/h is shown in Figure 4-34, Figure 4-35 and Figure 4-33. The responses of the vehicle model that are investigated are the lateral acceleration, longitudinal acceleration, yaw rate, and local position of the vehicle.

i. *Step steer test results for vehicle constant speed of 20 km/h*

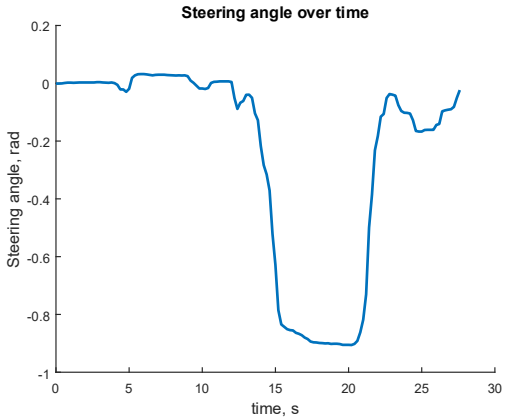


Figure 4-28 Steering angle for step steer test at 20 km/h

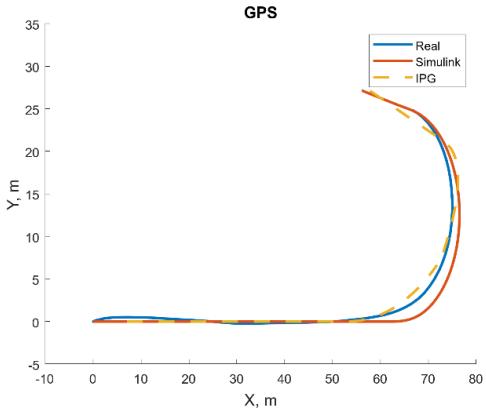


Figure 4-29 Local position plot for step steer test at 20 km/h

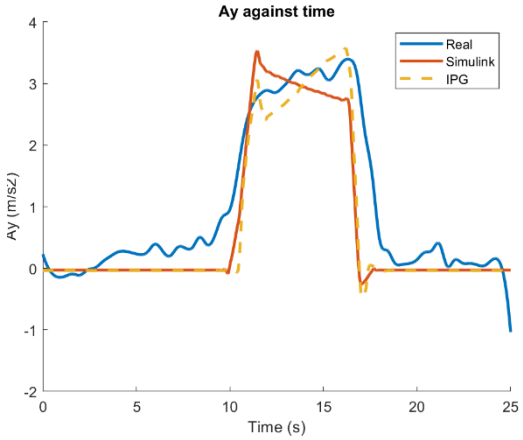


Figure 4-30 Lateral acceleration for step steer test at 20 km/h

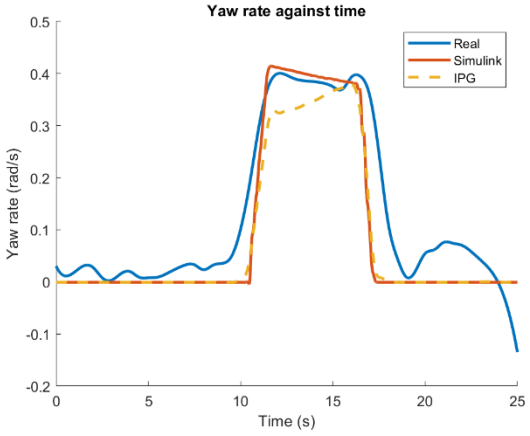


Figure 4-31 Yaw rate response for step steer test at 20 km/h

ii. *Step steer test results for vehicle constant speed of 30 km/h*

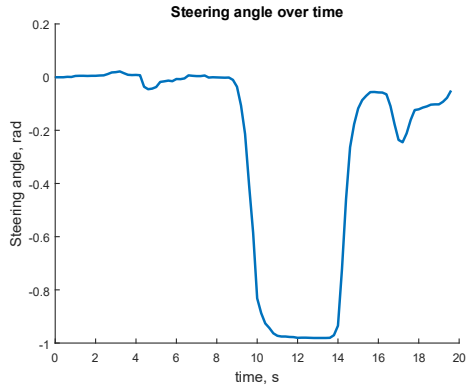


Figure 4-32 Steering angle for step steer test at 30 km/h

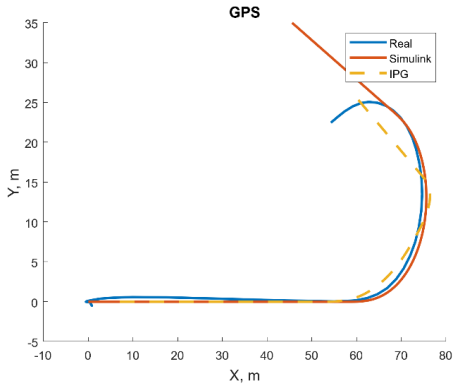


Figure 4-33 Local position plot for step steer test at 30 km/h

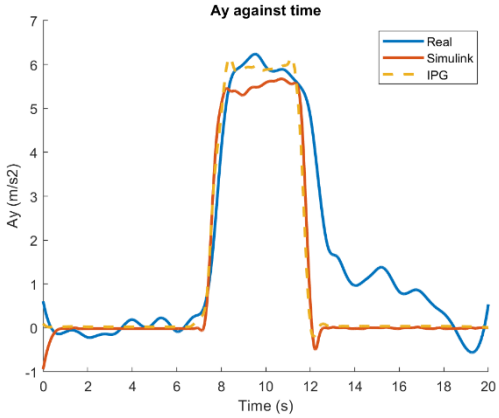


Figure 4-34 Lateral acceleration for step steer test at 30 km/h

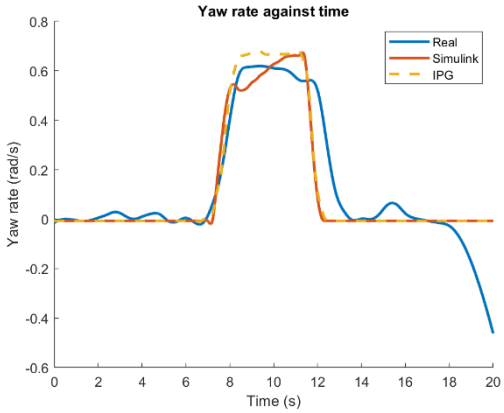


Figure 4-35 Yaw rate response for step steer test at 30 km/h

Table 4-6 Root Mean Square Error for step steer test data.

Root means square percentage error, %		
20 km/h	Lateral acceleration, m/s^2	7.156
	Yaw rate, rad/s	3.617
	Position, m	4.574
30 km/h	Lateral acceleration, m/s^2	7.394
	Yaw rate, rad/s	4.163
	Position, m	7.108

From the plot, it can be observed that the response of the vehicle obtained from real vehicle shared the similar characteristic with both the Simulink model and virtual vehicle from IPG CarMaker. Table 4-6 shows a summary of RMS percentage error for step steer at 20km/h and 30km/h when compared to the data obtained from the Simulink model developed. The lateral acceleration, yaw rate and displacement of the vehicle moving at 20 km/h show the percentage difference of RMS errors about 7.156%, 3.617% and 4.574%, respectively. Whereas the lateral acceleration, yaw rate and displacement of the vehicle moving at 30 km/h

show the percentage difference of RMS errors about 7.394%, 4.163%and 7.108%, respectively. Similar to the double lane change tests, higher peak and lower low values for the lateral acceleration measured from the real vehicle data when compared to the simulated vehicles. This can be explained by the displacement plot of the real vehicle and the simulated vehicle as shown in Figure 4-29 and Figure 4-33. From the displacement plot, it is observed that the radius of the circular motion of the real vehicle during step steer is small than the simulated vehicle. Since the steering angle input to the real vehicle and the simulated vehicles is the same, it can be deduced that the simulated vehicles do not encounter with unwanted disturbances in real time testing such as uneven road surfaces and mechanical friction between steering mechanism and wheel mechanism which is connected with rack and pinion system. Therefore, this caused some minor deviation in terms of vehicle lateral behaviour output between simulated vehicles and real vehicles even though the steering angle input is the same. Figure 4-31 and Figure 4-35 show the yaw rate response, where the real-world data reveals a slightly slower stabilization time compared to the simulated models. This occurs because real vehicle suspension components exhibit elasticity and compliance effects, causing a delayed stabilization of yaw rate. Similarly, Figure 4-30 and Figure 4-34 present the lateral acceleration response, which demonstrates that the simulated models predict higher peak lateral acceleration compared to real-world data. This suggests that the real vehicle undergoes additional damping effects, likely caused by the deformation of suspension components and variations in road surface texture. Furthermore, minor aerodynamic disturbances such as crosswinds may introduce slight variations in real-world yaw rate, whereas the simulations assume an idealized aerodynamic environment. The differences in response time and acceleration profiles highlight the need for a more sophisticated tire-suspension interaction model to capture real-world dynamics with greater fidelity. However, the root means square error (RMSE) calculated between the real model and IPG model for both 20km/h and 30km/h step steer test are still under acceptable range, which is less than 10%. Hence, it is noted that the vehicle model represents similar behaviour as the real vehicle which is capable of producing the similar lateral behaviours during the steady state response.

(c) Braking test

In this test, the vehicle will move at a constant speed and then applied with sudden braking input to come to halt the vehicle. This test is used to test the deceleration responses from the vehicle. The longitudinal speed, and the pitch angle will be used for comparison to show the similarity between the three model.

i. *Sudden Brake Test under 20km/h*

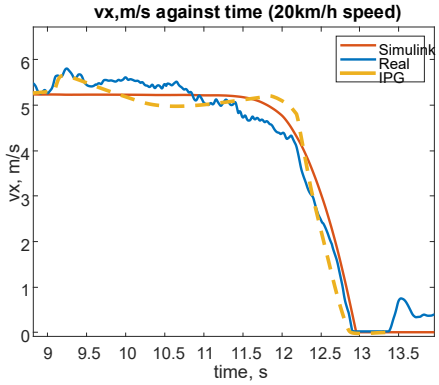


Figure 4-36 longitudinal velocity against time (20km/h)

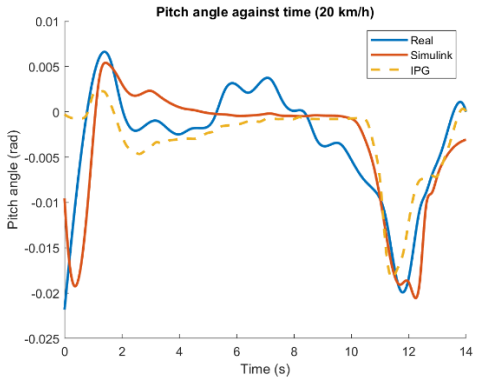


Figure 4-37 pitch angle against time (20km/h)

ii. *Sudden Brake Test under 30km/h*

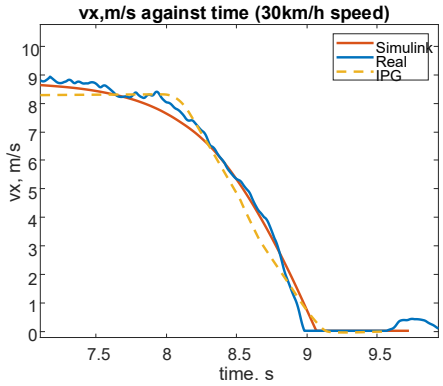


Figure 4-38 longitudinal velocity against time (30km/h)

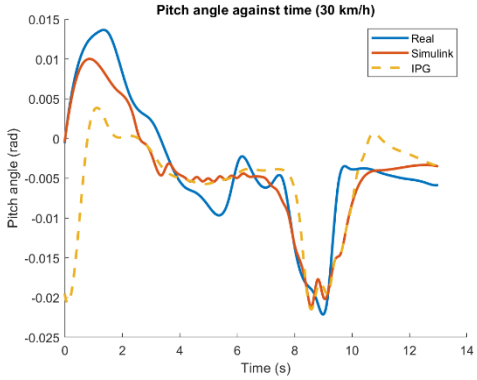


Figure 4-39 pitch angle against time (30km/h)

Table 4-7 Root Mean Square Error for braking test.

Root means square percentage error, %		
20km/h	Velocity, m/s	4.24
	Pitch angle, rad	6.52
30km/h	Velocity, m/s	3.90
	Pitch angle, rad	6.18

From the response from the plot, it shown that during the acceleration phase of the vehicle, both models shared the similar characteristic which can proved that the engine dynamic model and the engine mapping graph are reliable. Table 4-7 shows the RMS percentage error of the longitudinal velocity and pitch angle of the vehicle moving at 20 km/h and 30 km/h which are about 4.24%, 6.52%, 3.90% and 6.18%, respectively. Figure 4-36 and Figure 4-38 illustrate the velocity reduction profiles, which indicate that the real vehicle takes slightly longer to come to a complete stop compared to the simulated models. This suggests that the simulated braking force application is slightly overestimated. One reason for this discrepancy is the simplification of brake system dynamics in the simulation models, where factors such as thermal characteristic, brake piston, brake fluid pressure variations and mechanical response delays are not explicitly considered. Additionally, minor surface

irregularities and tire wear effects in the real-world braking test introduce variations in braking performance that are absent in the simulations. Figure 4-37 and Figure 4-39 show the pitch angle response during braking. It is observed that the real vehicle experiences a greater degree of forward pitch motion compared to the simulated models. This indicates that weight transfer effects are more significant in real-world conditions than what is predicted by the models. The simulations assume an idealized and symmetric braking force distribution, whereas in reality, braking force varies dynamically between the front and rear wheels, affecting pitch motion. Moreover, the condition of the braking system, such as brake pad friction characteristics and heat dissipation effect, further contributes to the observed differences. However, the root means square error calculated between the real model and Simulink model for both 20km/h and 30km/h test are still under acceptable range for the validation.

In summary, the comparative analysis demonstrates that the Simulink and IPG CarMaker vehicle models effectively replicate real-world vehicle behaviour under controlled test conditions. However, discrepancies arise due to tire-road interaction effects, suspension compliance, aerodynamic influences, and braking force distribution variations. The IPG CarMaker model tends to overestimate lateral acceleration and yaw rate due to its idealized tire model and reduced damping effects, whereas the Simulink model slightly underpredicts lateral acceleration due to simplifications in suspension compliance modelling. The braking test results suggest that further refinements are needed in the braking force distribution model and weight transfer dynamics to enhance the accuracy of pitch response predictions. These findings highlight key areas for future improvement, including incorporating advanced tire models, refining aerodynamic simulations, and integrating real-world road texture data into the vehicle model. Addressing these aspects will further enhance the reliability of simulation-based vehicle validation and improve the predictive accuracy of the developed models.

4.6 Data Collection on Dedicated Routes for Scenario-Based Testing

Following the system configuration of the instrumented vehicles for data recording capabilities of vehicle behaviours and surrounding environment, the vehicle is used to gather driving and traffic environment data on public roads. This collected data is analysed based on ISO standards such as ISO 34501, ISO 34502 and ISO 34503 for the Operational Design Domain (ODD) analysis. Based on the analysis, some of critical scenarios were extracted from the recorded datasets. The study involved repetitive driving condition using the instrumented vehicle across several selected road networks, namely:

a) Route 1: University of Nottingham Campus
   - Representing campus-like road settings, which facilitated the collection of data in controlled and unpredictable traffic scenarios.

b) Route 2: Jalan Pudu
   - Representing an urban road with moderate traffic and complexities.

c) Route 3: Cyberjaya MaGIC Route A

- Representing a modern cityscape with urban road features. This is also approved test track for autonomous vehicle by Ministry of Transport, according to the "Guidelines of Autonomous Vehicle Trials".

d) Route 3: Cyberjaya MaGIC Route A

- Representing a modern cityscape with highway lane driving features. This is also approved test track for autonomous vehicle by Ministry of Transport, according to the "Guidelines of Autonomous Vehicle Trials".

These locations were chosen based on the density of road users, including pedestrians, motorbikes, passenger vehicles, and commercial vehicles like buses and heavy vehicles. Furthermore, the emphasis was placed on areas with a high volume of pedestrians and motorbike users, aligning with the priorities of road safety agencies such as MIROS in investigating road accidents. Data collection was conducted under four distinct environmental conditions: morning, afternoon, evening, rainy day. During data collection, the instrumented vehicle was driven through the selected road networks while recording the data from the sensors mounted on the instrumented vehicle. The camera captured the visual information, the IMU captured the motion data, the GPS logged the vehicle's trajectory, and the steering wheel angle sensor and pedal sensor captured the driver's inputs evening, and rainy conditions.

4.6.1 Data Recording Route 1: University of Nottingham Malaysia Route

The route selected encompasses the University of Nottingham Malaysia, which experiences high traffic flow with various road users navigating in, out, and around the campus premises. The planned trajectory for data collection is detailed in Figure 4-40, incorporating roundabouts, parking lots, pedestrian crossings, speed bumps, and junctions to ensure a diverse dataset. The University of Nottingham Campus had a more controlled traffic environment with a mix of vehicles and pedestrians, leading to smoother driving patterns with less frequent acceleration and braking events. This is classified as the Operational Design Domain for campus environment. Data collection activities are conducted during peak hours in the morning, afternoon, and evening over a period of four weeks, encompassing weekends as well.

Figure 4-40 GPS trajectory of University of Nottingham Malaysia route (aerial image obtained from Google Earth)

4.6.2 Data Recording Route 2: Jalan Pudu Masjid Jamek

The route along Jalan Pudu and Masjid Jamek was selected due to its bustling city centre, characterized by heavy traffic and dense buildings and public transportation connectivity such as LRT and public bus services. Jalan Pudu, characterized by high traffic congestion and frequent stop-and-go conditions, exhibited more abrupt braking and acceleration patterns. The presence of pedestrians, commercial activities, and public transport services led to unpredictable driving behaviour. These has been classified as different ODD for scenario categorise compared to Route 1 as mentioned above. Figure 4-41 displays the planned data collection route for this area. Data collection occurred on this route for two weeks, encompassing morning, afternoon, and evening periods. During the morning, the vehicle moved slowly and often stopped due to heavy traffic from buses, cars, and motorcycles, especially along Jalan Pudu, causing traffic jams. After the morning peak hours, traffic eased as vehicle volume decreased. Weekends revealed tourists strolling and vehicles parked along Merdeka Square Road, with frequent police patrols.

Around noon, vehicular movement increased due to lunchtime traffic from workers frequenting the numerous restaurants lining Jalan Pudu. Contrary to the morning, afternoon traffic was faster paced, witnessing increased pedestrian activity and crossings, alongside numerous food delivery services' motorcycles. Merdeka Square Road experienced less congestion than Jalan Pudu, regardless of peak times, with only a few vehicles noted during each data recording lap. During the Muslim fasting month, from 4 PM onwards, numerous food trucks lined Jalan Pudu and Merdeka Square Road, attracting pedestrians buying food to break their fast. Consequently, traffic slowed during this period, exacerbated by police presence and roadblocks. These diverse traffic conditions and road actors at different times of the day offer valuable scenarios for analysis. Based on the above descriptions, it can be noted that the selected location contribution towards mixed traffic conditions which is heavily involved with pedestrians' motion, passenger and commercial vehicles which creates heavy traffic conditions. This has been classified as part of the ODD for the Jalan Pudu and Masjid Jamek routes.

Figure 4-41 GPS trajectory of Jalan Pudu and Masjid Jamek route (aerial image obtained from Google Earth)

4.6.3 Data Recording Route 3 and 4: Cyberjaya MaGIC Route A and Route B

Cyberjaya's Malaysian Global Innovation and Creativity Centre (MaGIC) Route A and MaGIC Route B constitute the initial two approved autonomous vehicle testing tracks by Futurise and the Ministry of Transport of Malaysia [328]. Figure 4-42 illustrates the layout for MaGIC Route A, while Figure 4-43 depicts MaGIC Route B. Comparatively, MaGIC Route A is a track located within the Futurise Centre which is less frequented by road users, limiting the number of valuable scenarios for data collection. In contrast, Route B connects to the main road, offering increased encounters with road users and diverse traffic situations, thus providing more valuable data for recording purposes. As both routes exhibit a lower frequency of road users and traffic scenarios, data collection was conducted over a span of nine days to ensure an adequate collection of scenarios. The lower traffic density resulted in more consistent driving behaviours, with less variability in speed and manoeuvring. However, the lower frequency of interactions with other road users provided fewer critical driving scenarios, limiting the diversity of recorded data. Despite this, these routes served as valuable testbeds for controlled autonomous vehicle trials. Moreover, these routes have been considered as one of the important locations for data collection as these routes have been designated as testing area for Autonomous Vehicle. Therefore, identifying the potential location to conduct the scenario-based testing is useful in this study.

Figure 4-42 GPS trajectory of MaGIC Route A (aerial image obtained from Google Earth)

Figure 4-43 GPS trajectory of MaGIC Route B (aerial image obtained from Google Earth)

Figure 4-44 displays the total dataset length recorded in hours for various conditions, totalling approximately 245 hours of road environment and driving data. It can be observed that the Jalan Pudu route has the highest recorded dataset length in hours. This is due to heavy traffic congestion, frequent stop-and-go conditions, and diverse road users. The presence of multiple intersections, pedestrian crossings, and commercial activities results in prolonged travel times and increased driving data. The complex urban environment requires frequent braking, acceleration, and steering adjustments, further extending the dataset length. In contrast, Pavilion Bukit Bintang despite being in a busy commercial district, has better-managed traffic flow, one-way streets, and designated pedestrian zones, reducing prolonged vehicle interactions and unnecessary delays. As a result, less time is spent in active driving, leading to fewer recorded data hours compared to Jalan Pudu.

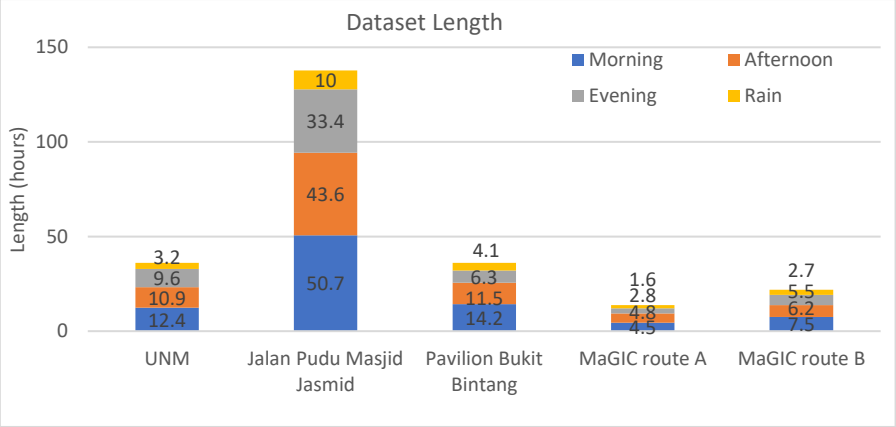


Figure 4-44 Dataset length for different conditions

4.7 Summary

This chapter presented the validation of the developed vehicle model using data collected from an instrumented vehicle. A low-cost sensor suite was integrated into the instrumented vehicle, enabling real-time acquisition of vehicle state parameters and surrounding traffic conditions. The validation process involved systematic testing using standardized driving manoeuvres, including the double lane change (DLC) test, step steer test, and braking test, to compare the Simulink-based 14-degree-of-freedom (DOF) vehicle model and IPG CarMaker virtual vehicle against real-world data.

The results of the double lane change (DLC) test demonstrated a high level of agreement between the simulated and real vehicle responses in terms of steering angle, lateral acceleration, yaw rate, and local position. However, minor discrepancies were observed, particularly during the exit phase of the manoeuvre, where the real vehicle exhibited a slightly delayed stabilization compared to the simulations. These deviations were primarily attributed to tire-road interaction variations, suspension compliance, and transient aerodynamic effects, which were not fully captured in the simulation models. The Simulink model tended to underpredict lateral acceleration, while the IPG CarMaker model exhibited slightly exaggerated yaw rate peaks, likely due to differences in tire model formulations and damping characteristics.

The step steer test evaluated the transient response of the vehicle to a sudden steering input. The yaw rate and lateral acceleration responses from the simulated models closely matched the real-world data, with slight differences in stabilization time and peak lateral acceleration values. The real vehicle exhibited greater damping effects, which were likely due to suspension compliance and tire deformation characteristics that were not fully modeled in the simulations. Additionally, external factors such as minor crosswinds and road surface irregularities may have contributed to variations in yaw rate stabilization in real-world conditions. The results suggested that further refinements in suspension dynamics and tire-road interaction modelling could improve the fidelity of the vehicle models.

The braking test examined vehicle deceleration characteristics, including velocity reduction and pitch angle response. The simulated models generally overestimated braking

force application, leading to slightly shorter stopping distances compared to the real vehicle. The real-world data indicated a more pronounced forward pitch motion, which was attributed to greater weight transfer effects in actual braking scenarios. The simplified braking force distribution models in the simulations assumed symmetric braking, whereas real-world braking dynamics involve non-uniform brake force allocation and transient variations in brake pad friction. These findings highlight the need for a more sophisticated brake system model to enhance the accuracy of longitudinal dynamics predictions.

Additionally, this chapter presented the data collection process on dedicated driving routes, including Jalan Pudu, Pavilion Bukit Bintang, University of Nottingham Malaysia (UNM) and MaGIC route A and B. This effort resulted in a total of 245 hours of driving data recorded across all selected road networks. The Jalan Pudu route recorded the longest dataset length due to its heavy urban congestion, frequent stop-and-go traffic, and diverse road users, which naturally extended the driving duration. In contrast, the Pavilion Bukit Bintang and UNM routes had shorter dataset lengths due to better-managed traffic flow and lower vehicle density, allowing vehicles to complete the routes more quickly. These variations in dataset length emphasize the influence of road network characteristics, traffic conditions, and driver interactions on data collection outcomes.

Overall, this chapter established the validity of the developed vehicle model by demonstrating strong correlations between simulated and real-world vehicle responses. The discrepancies observed highlight the limitations of tire modeling, suspension dynamics, aerodynamic influences, and braking force distribution assumptions. Future refinements should focus on incorporating higher-fidelity tire models, improving aerodynamic modeling, and enhancing real-world road texture integration to further improve simulation accuracy. These findings provide a critical foundation for the subsequent development of an AI-based driver model, ensuring that realistic vehicle dynamics are accurately captured within the simulation framework.

Chapter 5: Development of An Autonomous Vehicle Testing Platform

5.1 Overview

Safety assessment for autonomous vehicle has become one of the important elements before actual deployment on the road. This is to reduce the uncertainty that might occurs for the autonomous vehicle which lead road casualties such as traffic congestions or road accidents. One of the safety assessment methods is focusing on the simulation-based testing which can represents the actual environment and traffic scenarios. Besides, most of the autonomous vehicle developers are focusing on the safety issue for the vehicle, especially for Level 3 and Level 4 autonomous vehicle. A safety operator has been included as part of the vehicle operation to handle the vehicle during critical scenario. This has raised the additional concern on the capability of the safety operator to handle the autonomous vehicle during critical scenario. Therefore, driver-in-the loop simulation platform with human-machine-interface (HMI) is proposed as the safety testing platform for autonomous vehicle in this chapter. The simulation platform is developed using the simulation tool, IPG CarMaker. IPG CarMaker is used as the main component in the safety testing platform to develop the 3D virtual environment, traffic simulation, test cases, vehicle dynamic modelling and sensors simulation. Moreover, Simulink is used to design sub-models and algorithm such as ADAS controller, driver models and external steering and driving pedal mechanism which can then be integrated in the IPG CarMaker.

In the safety testing platform, other than quality and the fidelity of the simulation images or videos generated by the virtual simulation environment are important, but the degree of immersion also plays an important role for the safety assessment. This could replicate the similar driving feel from actual vehicle for the drivers or safety operators. Based on the degree of immersion, the behaviour of the drivers to handle the autonomous vehicle during critical scenario can be estimated. Moreover, it can be noted that the driver's experience such as driving in the real-world scenario is quite important for the development of the safety testing platform. This is mainly because the driver's experience using driving simulator can be used as an input to train the developed driver model, which will be discussed in Chapter 6. In order to develop the driver model, relevant driving data are required based on different drivers. The IPG CarMaker is used as the main simulation platform to generate high fidelity virtual simulation environment. However, it can be observed that as a software tool, it is unable to provide physical motion feedback to drivers without external hardware. Therefore, integration of simulation tool, IPG CarMaker with 6DOF motion platform and Virtual Reality headset are emphasized in this chapter. Based on the integration, several test cases are developed using IPG CarMaker for the driver-in-the-loop safety testing to gather driver's experience.

This chapter start with the development of an interface of Logitech G29 Driving Force Steering Wheel and Pedals kit to CarMaker using Simulink model. Then, the details of interfacing the Logitech G29 Steering Wheel and Pedal with the IPG CarMaker is discussed in this chapter. Next, this chapter presents the development of virtual safety testing for

autonomous vehicle that integrates with driver-in-the-loop simulator, 6DOF motion simulator and virtual reality. is also discussed. The integration of 6 degree of freedom (DoF) motion simulator with virtual reality (VR) is discussed in the following section. Then, this chapter also discussed on followed by the development of scenario-based testing using data recorded from the previous chapter. Furthermore, the performance of the built-in driver model of the IPG CarMaker compared to a human driver driving on the developed scenario-based testing platform is evaluated. Finally, the final section will summarise the chapter.

5.2 Configuration of Interface between CarMaker and Simulink

MATLAB Simulink is one of the most popular tools used for Model Based Software Development in the automotive industry. Simulink has been used in various application and able to interface with external tools such as CarSim and IPG CarMaker. In terms of integration of IPG CarMaker, this simulation tool also provides advanced software package for easier integration of custom Simulink models such as controller and driver models into IPG CarMaker software. Moreover, IPG CarMaker supports for co-simulation with Simulink and able to integrate external build-in Simulink model for the simulation-based testing. In this chapter, the main focus is the integration of external hardware into the IPG CarMaker simulation tool so that the virtual ego vehicle can receive driving inputs from the human driver.

In order to achieve this goal, a human interface hardware is required to allow human's interaction with the simulation during real-time testing. The hardware is required with physical interaction for steering inputs, brake input, throttle input and gear inputs (for manual driving only). Thus, Logitech G29 Driving Force Steering Wheels and Pedals are used in this study to enable human control of vehicle steering angle, gear, gas pedal position and brake pedal position as shown in Figure 5-1. The Steering wheel allows 900 degrees' ($-450°$ for full left and $+450°$ for full right) lock-to-lock rotation and force feedback to provide realistic driving experience. This configuration which is important to ensure that the human's responses are gathered for each test cases conducted using the IPG CarMaker. Moreover, the responses gathered from the hardware are compared with "DrivMan" Simulink block. This is to validate the responses collected from the simulation tool similar to the actions taken by same driver while driving in real life environment.

Figure 5-1: Logitech G29 Driving Force Steering Wheel and Pedals

In order to interface the Logitech G29 kit to IPG CarMaker, a Simulink model is developed as shown in Figure 5-2. The Simulink model is used to map the control signals from the Logitech G29 kit into steering angle, gear, gas pedal and brake pedal signals which can be accepted in the IPG CarMaker. By default, IPG CarMaker used built-in driver model, known

as IPGDriver to drive the virtual vehicle. In order to bypass the IPGDriver and use external driver's control input instead, the output of the default IPG's Simulink driver model is replaced with the corresponding outputs of the external driver model developed, as shown in Figure 5-3. The steering wheel and pedal interface were assessed by inviting participants to test it, with the goal of evaluating its realism and identifying any issues. Several scenarios developed to conduct this test. Figure 5-4 shows a participant driving the virtual vehicle model in IPG CarMaker via the virtual environment namely IPG Movie using the hardware interface. Through the human-machine-interface (HMI) using Logitech Steering and Pedal Input, the human inputs can be measured and able to control externally based on participant driving behaviour. However, it can be noted the limitations of the static driving simulator based on participant's feedback. The limitations are addressing lack of motion feedback to provide a sense of speed and vehicle motion, hindering the ability to handle the car naturally. Moreover, the participants also feedback that the visualization of the simulation setup does not provide a sufficient field of view to observe the side-view and rear-view of the vehicle. This drawback causes some difficulties for the participant to conduct critical test scenarios such as lane change conditions and overtaking scenarios, which requires surrounding view of the vehicle.

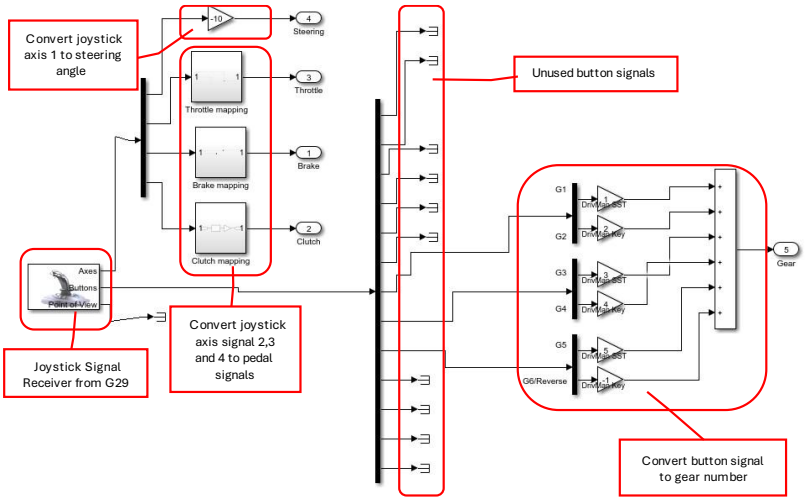


Figure 5-2: Logitech G29 Driving Force Simulink model.

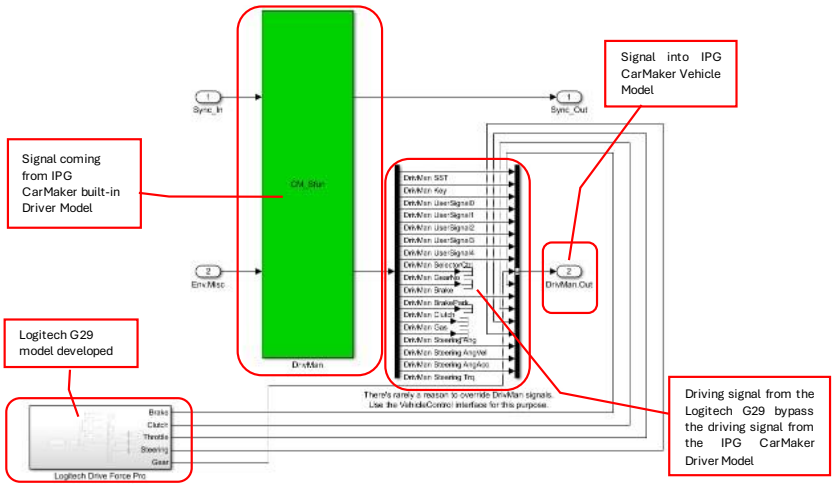


Figure 5-3: Logitech G29 interface with CarMaker for Simulink

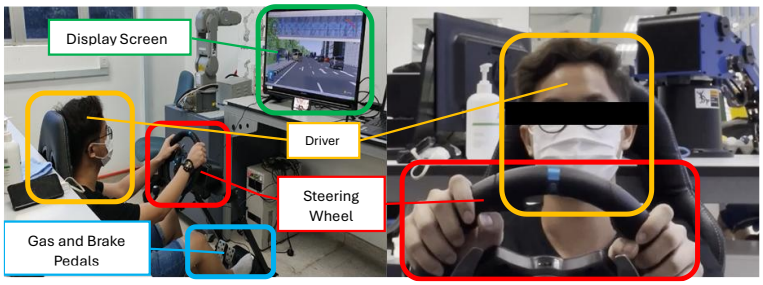


Figure 5-4: Human machine interface using IPG CarMaker and Logitech G29 steering wheel.

5.3 Integration of Virtual Simulator with 6 Degree of Freedom Motion Simulator

A motion simulator was developed to enhance the immersion of the driving simulator and provide the motion feedback that was lacking in previous tests, as indicated by participant feedback. This development aimed to ensure that the driving inputs applied by the drivers on the simulator were similar to those in the real world. The motion simulator has a floor footprint of $120 \times 250\ cm$, and features seven linear actuators designed to provide the 6-DOF motions: pitch, roll, yaw, surge, sway, and heave. The 6-DOF simulator is important for replicating vehicular movements, encompassing acceleration, braking, cornering, and impacts. This provides drivers an authentic driving experience, which is important in mitigating simulator sickness by aligning the simulated experience with the driver's anticipations and corporeal movements. Figure 5-5 shows the 3D model of the motion platform designed using SolidWorks and Figure 5-6 shows the actual 6 DOF motion platform developed.

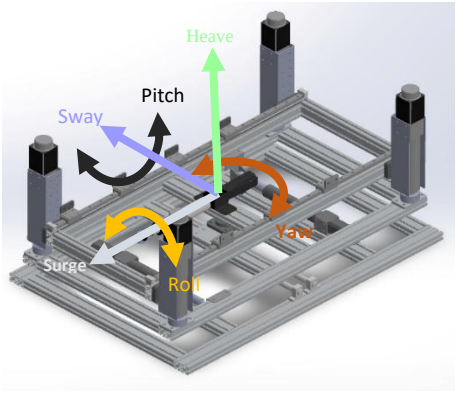


Figure 5-5 Motion simulator 3D model

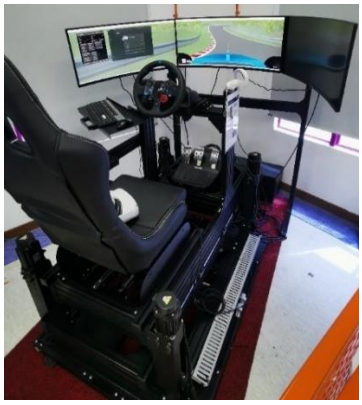


Figure 5-6 6 DOF motion platform

In this section, the detailed development of the interface between the IPG CarMaker and the 6 DOF motion simulator will be discussed. Controlled by the Thanos AMC-AASD15A servo motion controller, the 6 DOF motion platform's linear actuators are managed based on the received motion input. The default configuration does not equip with Application Programming Interface (API) to support the interaction between IPG CarMaker and the 6 DOF motion platform. On the other hand, the motion simulator was designed with the Sim Racing Studio (SRS) API. This interface was developed for the sole purpose of gaming activities. Thus, the existing API has been modified in order to enable the API to communicate between IPG CarMaker and the motion platform via a TCP/IP socket. However, creating a TCP/IP bridge with the motion simulator poses a significant challenge. This challenge stems from the basic

design of Simulink's TCP/IP function block, which is not inherently equipped to communicate via specific APIs on such networks.

5.3.1 Interfacing IPG CarMaker with Python

Generally, Simulink platform works great with IPG CarMaker for model-based development and co-simulation-based testing. However, to communicate with the motion platform using SRS API through TCP/IP network, an open-source platform is required to establish the interface between motion platform and Logitech G29 steering wheel and pedal kit. Therefore, Python is used in this study for the API configuration process. It can be observed that Simulink is not as convenient as Python due to more powerful and easy to use libraries such as TensorFlow and PyTorch are available through Python, which have vast community resources and supports. Therefore, developing an interface between IPG CarMaker and Python is very important not only for the communication with the motion platform but also for integration between IPG CarMaker with the external AI based driver model, which will be discussed in the next chapter. However, there is no established configuration to integrate between Python with IPG CarMaker, such as IPG CarMaker for Simulink. Nonetheless, IPG CarMaker allows TCP/IP sockets for communication with external open-source platforms. By establishing a TCP/IP bridge between the IPG CarMaker and the Python program, the Python program can access and manipulate the quantities of the IPG CarMaker via Direct Variable Access (DVA) during simulation. Once the TCP/IP bridge is established, the program can request the value of required quantity data by simply transmitting “DVAWrite” message encoded with the name of the quantity data, duration and the desired parameters’ values to IPG CarMaker. Whereas to read quantity data, it can be done by simply sending a “DVARead” message encoded with the name of the quantities. Then, IPG CarMaker will transmit back the specified value of the quantity as shown in Figure 5-7 instantaneously.

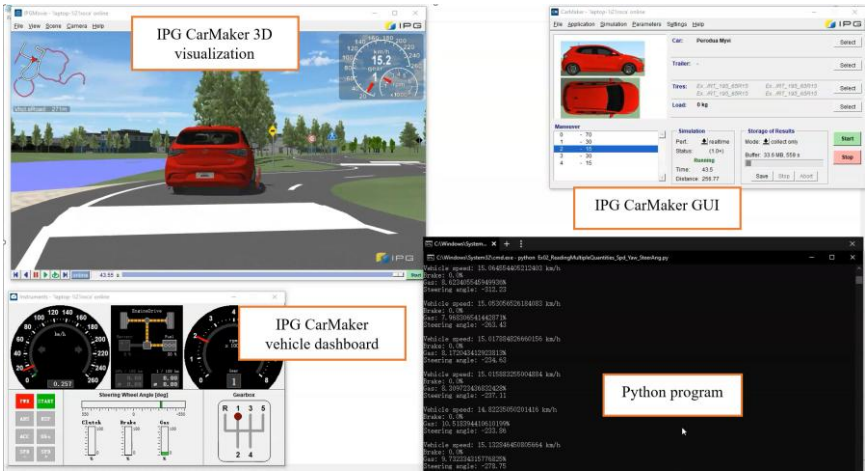


Figure 5-7 Reading vehicle speed, brake pedal, gas pedal and steering angle quantities from IPG CarMaker using Python.

Besides accessing and manipulating the quantities of IPG CarMaker, Video Data Stream (VDS) generated by virtual cameras are mounted on the ego vehicle. The data from virtual cameras can be exported over TCP/IP and accessed by Python program. This is one of the steps that has been implemented for the interface because it’s crucial for vision-based

autonomous vehicle's controllers to gather images as data input. Unlike accessing the quantities of the IPG CarMaker, a configuration file for IPGMovie is required before using the VDS. The configuration file is required to define the TCP/IP socket for streaming and camera's configuration such as resolution, framerate, relative distance, orientation of the camera and other parameters of the camera sensor. Additionally, more than one camera can be defined in the same configuration file as shown in Figure 5-8. However, increasing VDS will affect the latency of each frame cycle, which in result will slow down the simulation speed. Therefore, the number of cameras used for the streaming process is limited to 4 views only. After the configuration is completed and simulation initiated, each VDS camera frame will be rendered in each frame cycle sequentially before start of the next frame cycle.

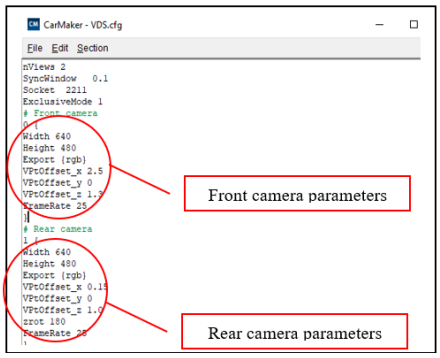


Figure 5-8 Example of VDS configuration with two cameras (front and rear) with TCP/IP socket of 2211

Each transmitted image is encoded with a header information string that has a format as follow:

"*VDS <label of camera> <image format> <time> <width>x<height> <length of the image in bytes>\n".

Python is used to decode and extract the image data from the VDS received. The image data is reshaped into RGB three color channels image based on the width and height information decoded from the header information string. Finally, the image can be used according to user's requirement such as feeding as inputs into a vision-based controller's algorithm and displaying on the screen using OpenCV library as shown in Figure 5-9.

Figure 5-9 Video stream data exported from four cameras mounted on ego vehicle, where frame 1 is from front camera, frame 2 is from rear camera, frame 3 is from right side mirror camera and frame 4 is from left side mirror camera.

5.3.2 Interfacing IPG CarMaker with Motion Simulator

After the interface between Python and IPG CarMaker is completed, the data exchange between both platforms can be achieved in real-time. By using the same API protocol, the data exchange from the motion platform and IPG CarMaker can be established. The parameters required by the motion simulator can be accessed and obtained by transmitting "DVARead" message from the IPG CarMaker via TCP/IP 16660 socket. The signals required from the IPG CarMaker, and the corresponding signals of the motion platform are shown in Table 5-1. By using the Sim Racing Studio API, the IPG CarMaker signals are encoded into a telemetry message which is sent to the motion simulator through TCP/IP socket 33001. Figure 5-10 shows the communication architecture between the IPG CarMaker and the motion simulator.

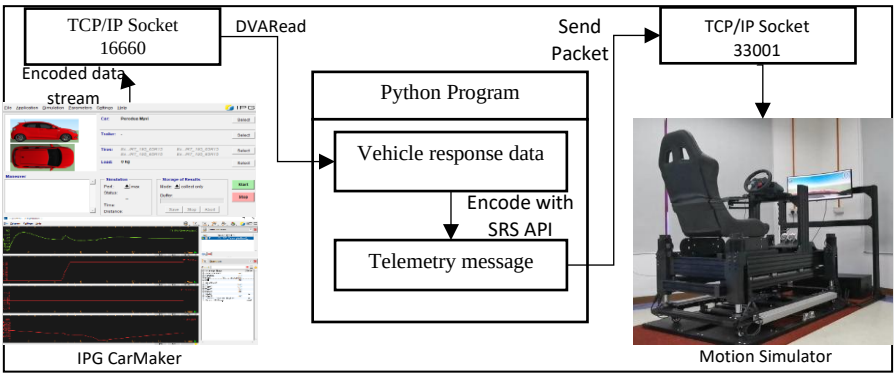


Figure 5-10 CarMaker-motion platform interface architecture

Table 5-1 Motion simulator integration signals

CarMaker Signals	Motion Simulator Signals	Description
Env.WindVel	speed	Wind speed in m/s
Vhcl.Engine.rotv	rpm	Rotation per minute of the engine
PT.GearBox.GearNo	gear	Gear position
Vhcl.Pitch	pitch	Pitch angle in degrees -180 to +180
Vhcl.Roll	roll	Roll angle in degrees -180 to +180
Vhcl.Yaw	yaw	Yaw angle in degrees -180 to +180
Car.vy	lateral_velocity	Lateral velocity in m/s used for traction loss simulation
Car.ax	lateral_acceleration	Lateral direction g-force in values between -10 to 10
Car.az	vertical_acceleration	Vertical direction g-force in values between -10 to 10
Car.ay	longitudinal_acceleration	Longitudinal direction g-force in values between -10 to 10
Car.DampFL.l	suspension_travel_front_left	Suspension travel of front left wheel in value between -10 to 10
Car.DampFR.l	suspension_travel_front_right	Suspension travel of front right wheel in value between -10 to 10
Car.DampRL.l	suspension_travel_rear_left	Suspension travel of rear left wheel in value between -10 to 10
Car.DampRR.l	suspension_travel_rear_right	Suspension travel of rear right wheel in value between -10 to 10

As shown in Table 5-1, the signals from the IPG CarMaker need to be mapped to the corresponding signals for the motion simulator. However, the convention used by the two platform is different such as the value of the pitch angle from the IPG CarMaker is in radian whereas the value accepted by the motion simulator is in degrees. Therefore, the Python program also need to convert the signals received from the IPG CarMaker to the correct format before the values can be passed to the motion simulator. In order to convert the pitch, roll and yaw angle from radian to degrees, a standard mathematical conversion formula for converting angular measurements from radians to degrees [329] is shown in Equation 5.1 and applied to the angles' value obtained from the CarMaker.

$$angle_{degrees} = angle_{radians} \times \frac{180}{\pi} \tag{5.1}$$

Whereas for converting the lateral acceleration, vertical acceleration and longitudinal acceleration which has value in m/s^2 to the corresponding g force, the value from IPG CarMaker is divided by gravitational acceleration, g which is 9.81 m/s^2 to obtain the g force experience by the virtual vehicle. To convert the suspension travel length in metres to suspension travel scaling from range 0 to 10, a normalization method that is commonly used in vehicle dynamics simulation [301] as shown in Equation 5.2 is applied.

$$suspension\ travel_{motion\ platform} = \frac{l_{s_{CM}} - l_{s_{droop}}}{l_{s_{bump}} - l_{s_{droop}}} \times 10 \tag{5.2}$$

Where, $l_{s_{CM}}$ is the suspension travel length in metres obtained from CarMaker, and $l_{s_{bump}}$ and $l_{s_{droop}}$ is the suspension travel clearance allowed for bump and droop behaviour of the wheel respectively. Figure 5-11 shows bump and droop behaviour of suspension on an uneven terrain. From the CarMaker, the buffer of the vehicle is defined as a spring to limit the wheel travel in one direction. From Figure 5-12, the characteristic of the buffer shows that the maximum travel of the suspension spring defined for "Push"/Bump is 0.075 m, whereas the maximum travel of the suspension spring defined for "Pull"/Droop is 0.025 m. Therefore, the total suspension travel clearance for this vehicle is 1m. With all the parameters available, the python program developed then proceed to encode using SRS API and send the data to the motion simulator via TCP/IP network.

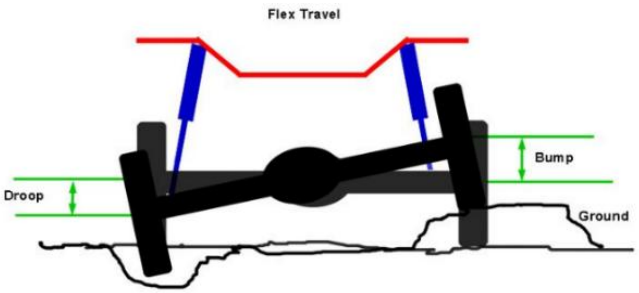


Figure 5-11 Bump and droop behaviour of a suspension [330]

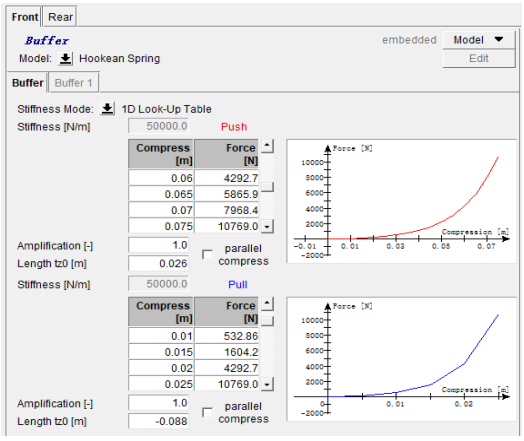


Figure 5-12 Buffer characteristic definition of CarMaker virtual vehicle.

In order to provide visual feedback to human driver that sits on the motion simulator, a camera is mounted on the driver seat. The view captured from the driver seat is displayed on the monitor's screen so that the driver has a feeling of sitting in the virtual vehicle. Figure 5-13 shows the position of a camera sensor mounted on the driver seat of the virtual vehicle and the view captured from the driver seat camera.

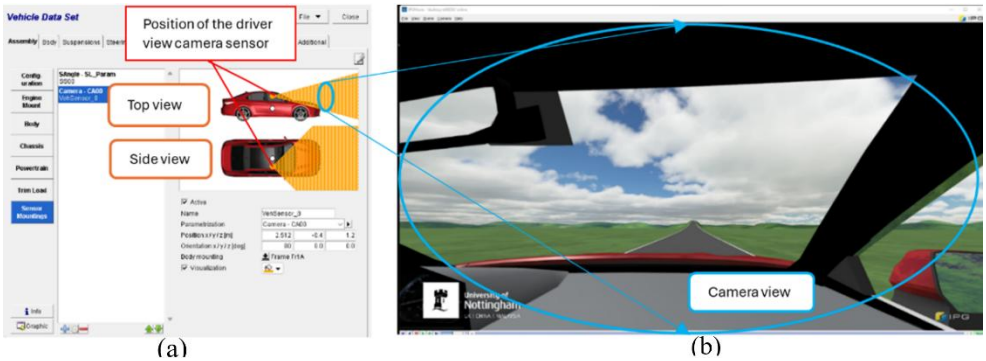


Figure 5-13 (a) Driver view camera mounted on the driver seat using IPG CarMaker and (b) camera view captured from camera mounted on driver seat.

Once the 6 DOF vehicle driving simulator is in fully functional mode, the driver-in-the-loop (DiL) simulator can be developed. The DiL can be designed using the Logitech G29 steering wheel and pedal kit by interfacing it using Simulink model configuration as shown in previous section. Human driver can drive the virtual vehicle by interacting with the steering wheel and pedals to provide steering wheel angle, acceleration or braking inputs to the virtual vehicle and receiving visual feedback from the monitor screen and motion feedback from the motion simulator. Figure 5-14 shows a human driver driving the virtual vehicle on the motion simulator.

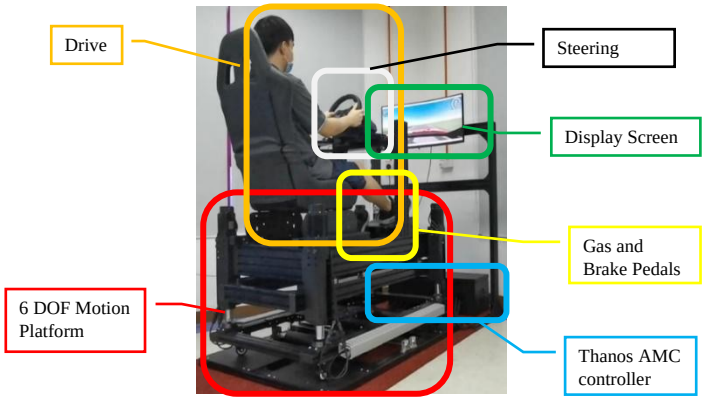


Figure 5-14 Driver-in-the-loop simulator with motion feedback.

5.4 Integration of Virtual Simulator with Virtual Reality

The introduction of virtual reality (VR) via an Oculus Quest 2 headset enriches the immersive quality of the simulator, bridging the gap between the real world and simulation. As the drivers put on the VR headset, they are transported into a virtual world that feels just like being inside a real car. The use of VR headset in driving simulator also offers a stereoscopic 3D view that mimics the depth perception experienced in a real world. This is achieved through

the independent screens for each eye, creating a sense of depth and space. It is an essential feature for realistic simulations because it helps drivers judge distances and speeds more accurately, just as they would on an actual road. Despite low cost, the Oculus Quest 2 head-mounted display (HMD) also equipped with a built-in visual and inertial based motion tracking system to provides precise movement tracking. Additionally, the integrated stereo speakers within the headset contribute to immersive sound feedback. Notably, the Quest 2's wireless capability enhances user freedom during operation, eliminating the need for wired connections. However, integrating VR with IPG CarMaker presents challenges due to the absence of native VR support. In order to implement VR with IPG CarMaker, development of an interface between the IPG CarMaker and the VR headset is required. This interface is required to engage virtual reality headset with IPG CarMaker simulation platform.

5.4.1 Virtual Reality Headset Interface

During the development of the interface process between Oculus Quest 2 and IPG CarMaker. a few challenges have been identified. Firstly, the orientation and position of the camera sensor in the IPG CarMaker is fixed and cannot be rotated and moved during the simulation. Secondly, the IPG CarMaker does not have an interface to receive input from the Oculus Quest 2 headset. Therefore, a customize plugin are required to transmit the orientation of the headset to the IPG CarMaker. This is to control the position and orientation of the camera so that the camera turns according to the orientation of the headset. To solve the first challenge, a free moving traffic geometry object with 3D model of a driver is generated using the CarMaker's Traffic editor as shown in Figure 5-15. This is because the position $[tx, ty, tz]$ and orientation $[rx, ry, rz]$ of the free moving geometry can be accessed via Direct Variable Access (DVA) of the CarMaker. Next, the driver camera view is configured by attaching to the geometry object in order to make the camera move and rotate along with the geometry object. Hence, by using Simulink or Python, the geometry object can be manipulated during the simulation which resulting the camera view to rotate and move together.

Next, to solve the second challenge, third party software such as Simulink and Python is used to develop the interface with the Oculus headset using OpenVR API. OpenVR API is an open-source API to interface with VR headset with other simulation platform. Therefore, by using the OpenVR API, not only the Oculus headset will be supported but any VR headsets that are compatible with OpenVR API can be used. Initially, Python is chosen to set up a bridge between the Quest 2 and the CarMaker. However, it has been identified that there is a lot of delay using this method which caused the driver view to always be lagged and could not move with the virtual vehicle smoothly. This is because Python is an interpreted language where the code is interpreted during runtime but not compiled to native code. Therefore, Python is slower than native programming language such as C/C++. To make the driver geometry object move smoothly with the virtual vehicle, Simulink is used as a bridge to interface the Oculus Quest 2 headset with the IPG CarMaker. This is mainly because the function blocks in the Simulink were compiled before it starts to run the simulation process. Figure 5-16 shows the design of the interface for using virtual reality with CarMaker.

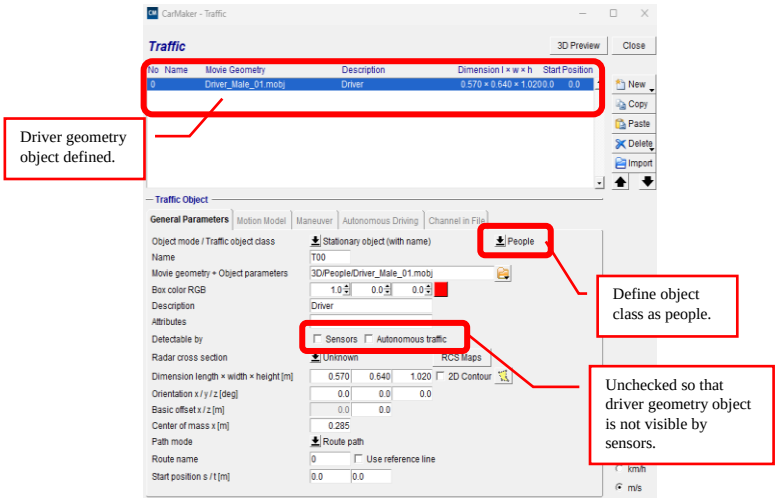


Figure 5-15 Virtual driver geometry object generated using Traffic editor.

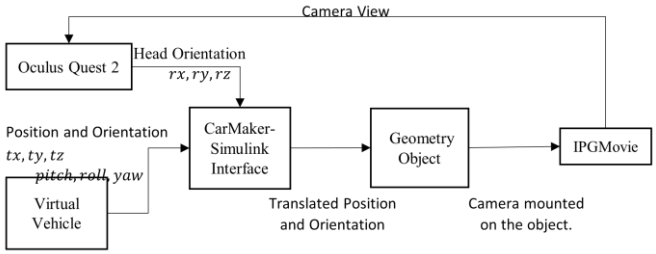


Figure 5-16 Overview design of VR-CarMaker interface developed.

In CarMaker, the position of all the geometry objects and vehicles are relative to the origin position of the map. In order to identify the driver's position, kinematic model that describe the relationship between the driver object and the vehicle in 2D planar are designed as shown in Figure 5-17. This position is identified so that the geometry object can be placed on the driver seat of the virtual vehicle.

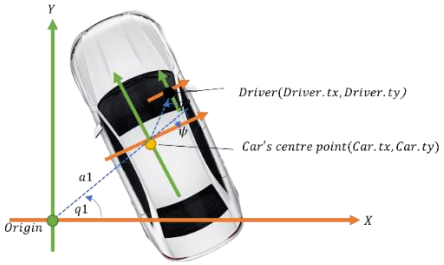


Figure 5-17 Kinematic model of the driver object in 2D planar.

To find the driver position relative to the origin, a transformation matrix, T that describe the translation and rotation required to transform the position of the driver from origin to the desired position is defined in Equation 5.3.

$$T = R_{z_O}(q1)T_{x_O}(a1)R_{Z_C}(\psi)R_{Z_C}(q2)T_x(a2)$$

$$T = \begin{pmatrix} \cos(q1+q2-\psi) & \sin(q1+q2-\psi) & a2\cos(q1+q2-\psi)+a1\cos q1 \\ \sin(q1+q2-\psi) & \sin(q1+q2-\psi) & a2\sin(q1+q2-\psi)+a1\sin q1 \\ 0 & 0 & 1 \end{pmatrix} \quad (5.3)$$

Where,
$a1$ is the displacement from the virtual vehicle's centre point to the origin,
$a2$ is the displacement from the driver's centre point to the virtual vehicle's centre point,
$q1$ is the angle between the virtual vehicle's frame and the global frame,
ψ is the yaw angle of the virtual vehicle relative to the Y axis, and
$q2$ is the angle between the driver's frame and the virtual vehicle's frame.

Since this is a homogeneous transformation matrix which combined rotation matrix, R and translation matrix, P into a single matrix H which is shown in Equation 5.4, the equations that describe the displacement of the driver object from the origin is defined in Equation 5.5.

$$H = \begin{pmatrix} R & P \\ 0 & 1 \end{pmatrix} \tag{5.4}$$

$$H = \begin{pmatrix} C1 & -S1 & x_d \\ S1 & C1 & y_d \\ 0 & 0 & 1 \end{pmatrix} \tag{5.5}$$

By comparing Equation 5.3 and Equation 5.5, Equation 5.6 can be found as below:

$$\begin{pmatrix} C1 & -S1 & x_d \\ S1 & C1 & y_d \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos(q1+q2-\psi) & \sin(q1+q2-\psi) & a2\cos(q1+q2-\psi)+a1\cos q1 \\ \sin(q1+q2-\psi) & \sin(q1+q2-\psi) & a2\sin(q1+q2-\psi)+a1\sin q1 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = \begin{pmatrix} a2\cos(q1+q2-\psi)+a1\cos q1 \\ a2\sin(q1+q2-\psi)+a1\sin q1 \end{pmatrix} \tag{5.6}$$

The yaw angle, ψ can be obtained from the IPG CarMaker's "Car.Yaw" quantity via DVA. To obtain $q1$ and $a1$ from the instantaneous position of the virtual vehicle, Equation 5.7 and Equation 5.8 can be used to calculate the two values respectively, based on basic trigonometry.

$$q1 = \tan^{-1}\frac{Car.tx}{Car.ty} \tag{5.7}$$

$$a1 = \frac{Car.tx}{\cos q1} \tag{5.8}$$

The value of $q2$ and $a2$, which describe the displacement of the driver seat to the centre point of the vehicle body, remain constant because the driver seat is fixed to the chassis of the vehicle.

The final step is to provide the head tracking signals of the Oculus headset to the driver geometry object so that the camera mounted on the object will rotate according to the orientation of the VR headset. Figure 5-18 shows the model configuration using the VR headset with the CarMaker. A Python program is developed to obtain head tracking signals from the VR headset through OpenVR API. The Python code then normalise and mapped the signals to an emulated virtual joystick using a virtual joystick device driver (vJoy). The reason for mapping the head tracking data to a virtual joystick is because the Simulink does not have native method to read the tracking data, but it has a built-in joystick function block to obtain signals from external device. Next, the VR headset orientation in pitch, roll and yaw angles can be converted from the joystick signal. To make sure the driver object always facing at the same direction of the car when there is no head movement, the car's instantaneous orientation obtained from CarMaker is added to the driver object's orientation. Figure 5-19 shows the view of a scene inside the virtual vehicle in the CarMaker.

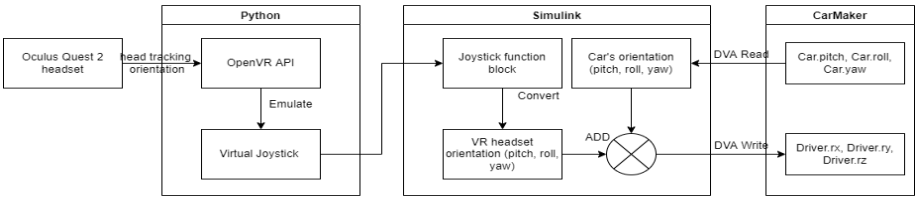


Figure 5-18 Implementation of Oculus's head tracking in CarMaker

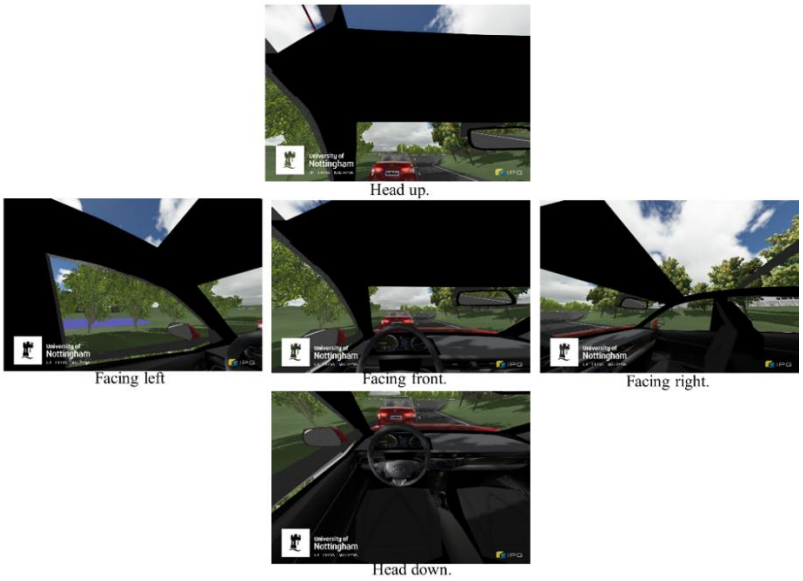


Figure 5-19 View of inside the virtual simulator with virtual reality where each frame represents the driver's view as the driver turn their head left, right, up and down, illustrating 360-degree view

5.4.2 Motion Compensation

Driving inside a fully equipped cockpit with VR headset and head tracking gives drivers the feeling of driving like in a real car. However, when combine the use of VR with motion simulator, there is a serious flaw where the seat movement of the motion platform will be captured and recognized by the VR headset as head movements. An example of a driver using the VR headset with motion simulator is shown in Figure 5-20. It can be observed from the figure below that without motion compensation, when the driver perform double lane change, the motion simulator performed yaw and roll movements to simulate a force that indirectly disengage the driver's body outside the seat. The VR headset identified this movement as the driver's head movements and processes them in the normal way which resulting the driver's view tilted and looks like it is moving toward the side windows which is unpleasing.

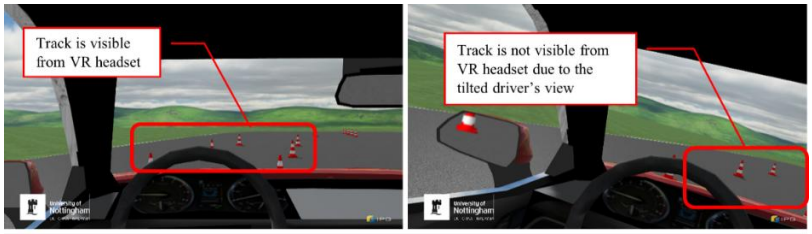


Figure 5-20 Driver's view during double lane changes with motion compensation (left) and without motion compensation (right).

In order to eliminate the issue, motion compensation is required to cancel the motion generated by the motion simulator from the VR headset's head tracking value. The process of motion compensation is shown in Figure 5-21. The response generated by the motion simulator is captured and used to calculate the correction needed to apply to the head tracking data captured by the VR headset at runtime.

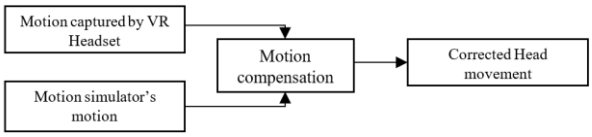


Figure 5-21 Process of motion compensation for head tracking.

In order to capture the movement of the motion simulator, one of the touch controllers of the Oculus Quest 2 is mounted on the motion simulator to move with the simulator as shown in Figure 5-22. The Oculus Quest 2's touch controller has built-in three DOF IMU sensor that can be used to measure the motion of the simulator. By making the controller as a reference tracker, VR headset's head tracking is first calibrated to the IMU sensor reading of the controller. Hence, when the driver is facing the same direction as the touch controller, the motion captured by the VR headset is not counted as the head movement of the driver. Only when the captured head tracking values are more than or less than the value of the controller's sensor value, the difference between the values is considered as the head movement of the driver. With this motion compensation, the driver's view will not be affected by the platform's movements and can provide a more immersive and authentic driving experience to the drivers. Figure 5-23 shows the system overview of the virtual simulator developed with integration of VR and motion platform.

Figure 5-22 Virtual autonomous vehicle simulator with virtual reality and motion simulator.

It is worth noting that many driving simulators utilize either the Stewart or compact configuration for their motion base. The compact type of simulator, as the name suggests, offers advantages such as lower cost and a compact design, making it easily transportable for events and customer demonstrations. However, compact simulators have limited stroke, which makes them unsuitable for high-fidelity motion simulation. On the other hand, the Stewart type, also known as the hexapod type, originates from aircraft simulators. In this study, the wedge type simulator, specifically designed to simulate vehicle motion with lower latency, is utilized. The motion and visual latency tolerated in aerospace simulators typically range from 100 ms to 150 ms [331]. In contrast, experienced automotive test drivers have limited tolerance for latencies around 30 ms or even as low as 10 ms for racing drivers. The driving simulator developed in this study employs the wedge configuration, which helps reduce mass, size, and complexity,

lower the centre of gravity, maximize the range of motion, and independently and linearly control the axes of motion to replicate short and sharp movements. While recent novel techniques such as driving simulators with rail and movement concepts and omnidirectional wheels can provide high-fidelity translational motion, they are expensive and require significant space for the simulator to move. In comparison, the wedge simulator in this study offers advantages in terms of reduced mass, size, and complexity, as well as the ability to replicate short and sharp movements. Additionally, the Stewart simulator requires motion inputs as if they are originating from the driver's head position. In contrast, the simulator used in this study utilizes the suspension travel of the four wheels to simulate vehicle motion. This design is based on the 14 DOF mathematical vehicle model developed in Chapter 3, employing four vertical direction actuators that independently emulate the suspension travel of the wheels. Consequently, the simulator in this study can provide a more realistic driving experience, especially in events such as bumps, braking, accelerating, and cornering, compared to the Stewart type, which has limited stroke. Moreover, the simulator in this study is capable of emulating skid and traction loss situations, such as wet road conditions and locked wheels, which are challenging to achieve in the Stewart simulator.

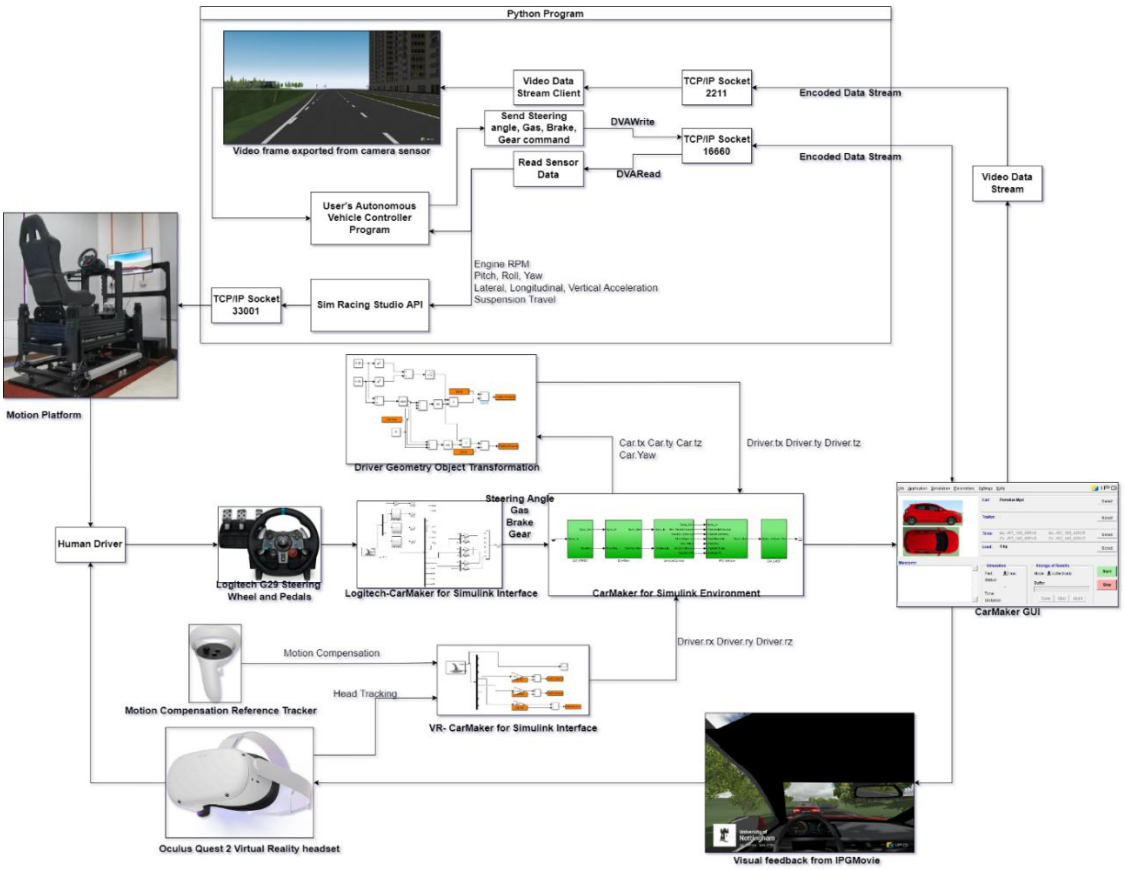


Figure 5-23 System architecture of the virtual simulator based-on IPG CarMaker with integration of VR and motion platform.

In terms of the visualization interface and software utilized, no implementation of a driving simulator integrating a VR headset with IPG CarMaker software is available. This limitation arises because IPG CarMaker was not designed with the intention of being used for virtual reality driving simulators and lacks the necessary features for interfacing with VR

headsets. On the other hand, game engines like Unity3D provide tools for interfacing with VR equipment and motion platforms more easily. Additionally, game engines can offer high-fidelity visualization of the driving environment, unlike IPG CarMaker, where reflections in IPGMovie are computed on the Central Processing Unit (CPU), resulting in low resolution and framerate when reflections are enabled [332]. However, these simulators that utilize game physics engines for vehicle dynamics and sensor models are not as accurate as the vehicle dynamic model and realistic sensor models provided by IPG CarMaker. Furthermore, by utilizing a workstation with higher computational CPU power, this study enables the simulation of the driving environment with high-fidelity graphics, including enabled reflections, while maintaining a high framerate. To compare the driving simulator developed with other driving simulator with motion base, the degree of freedom, type, software used, fidelity of the simulator and visualization interface were compared in Table 5-2.

Table 5-2 Comparison of existing driving simulator.

Driving Simulator	Degree of Freedom	Type of Motion Platform	Software	Fidelity		Visualization interface
				Textures resolution	Reflection	
Tractor driving simulator (TDS) [333]	3	Stewart	Unity 5	Low	No	VR headset
Compact 3 DOF Driving Simulator using Immersive Virtual Reality [334]	3	Compact	Unity3D	High	Yes	VR headset
VR DS [335]	6	Compact	Unity3D	High	Yes	VR headset
CASTER [332]	6	Stewart	IPG CarMaker	Low	No	Triple monitor setup
Dynamic driving simulator with a novel rail and movement concept [336]	6	Rail and movement concept	IPG CarMaker	Yes	Yes	CAVE
Wheeled mobile driving simulator [337]	6	Omnidirectional wheel based	IPG CarMaker	High	Yes	Not implemented
This study	6	Wedge	IPG CarMaker	High	Yes	VR headset

5.5 Scenario Identification and Classification for Scenario-Based Virtual Testing

From the real-world data captured using the instrumented vehicle, the relevant scenarios are gathered based on the location for the deployment of the autonomous vehicle's operation. However, not all test scenarios can be used for test cases to conduct simulation and physical assessment. This is primarily because the test cases need to be designed based on necessary characteristics required for the assessment, such as test objectives, input data, test procedures, unique identifiers, testing platform capabilities, and expected results. Therefore,

scenario identification is required based on international standards such as ISO 21488 (Safety of the Intended Functionality or SOTIF) and ISO 26262 (Road Vehicles - Functional Safety), which are fundamental for the safety assessment of autonomous vehicles in physical environments. Additional standards, such as BSI Flex 1889, ISO 34501, ISO 34502, ISO 34503, and ISO 34504, provide comprehensive guidelines for scenario-based testing, scenario categorization, and the specification of the operational design domain (ODD), as presented in Chapter 2.

Based on these standards, the raw data collected using the instrumented vehicle in the previous section must undergo manual pre-processing to extract longitudinal scenarios such as vehicle following, vehicle overtaking, and pedestrian jaywalking. Next, machine learning algorithms are used to identify specific events, such as near misses or accidents, from the pre-processed data. Features that characterize the scenarios, including vehicle speed, distance between objects, weather conditions, and actor actions, are extracted from the data. These features are used to classify the scenarios into different categories relevant to the ODD and testing objectives. This classification process should be aligned with the ISO standards for scenario-based testing and classification. Finally, the classified scenarios are subjected to risk and hazard assessment to evaluate the potential safety risks associated with each scenario. This assessment involves qualitative and quantitative analysis methods. The scenario extraction process is discussed in detail in the following section and illustrated in Figure 5-22.

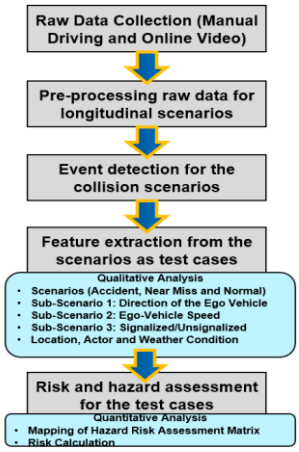


Figure 5-24 Process of data extraction and scenario classification in autonomous vehicle

5.5.1 Development of Malaysian Road Scenario database (MaRSeD)

In order to extract and classify the scenarios from the raw data, this study proposes the Malaysian Road Scenario Database (MaRSeD), a scenario classification database. One primary function of MaRSeD is to develop a generic database that identifies potential test scenarios from the Malaysian environment for autonomous vehicle safety testing and maps them to international standards. Three main input data types have been considered for the classification process in MaRSeD. The first data type focuses on normal driving scenarios, which are low-risk but frequently occurring in daily Malaysian driving conditions, such as pedestrian jaywalking, motorbike cut-ins, traffic vehicle cut-outs, and construction scenarios. The second type of scenario classification focuses on near-miss accident scenarios in the selected location.

Video data from instrumented vehicles are inputs that can lead to potential critical test scenarios common in the selected location. However, important scenarios from areas outside of the selected locations might be missed, crucial for test scenarios. Therefore, online videos uploaded by Malaysian road users on social media were used to re-create potential test scenarios as well.

Additionally, road accident videos were obtained from the Malaysian Institute of Road Safety Research (MIROS), responsible for investigating road accidents in Malaysia. The collected raw videos were analysed to identify and label potential scenario classifications according to their severity. The test scenarios were further classified into four categories: Functional Scenario, Abstract Scenario, Logical Scenario, and Concrete Scenario, discussed in the following section. These categories were used to parameterize test scenarios for autonomous vehicle safety assessment into basic behaviour testing and critical test scenarios, discussed in the next section. The process flow is shown in Figure 5-25.

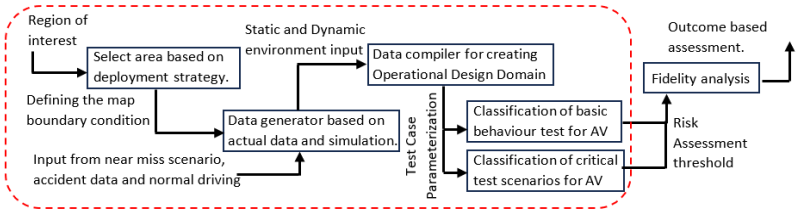


Figure 5-25 Scenario classification process flow based on input data from instrumented vehicle

Machine vision was considered in this study for scenario identification and classification based on Malaysian driving scenarios from raw video inputs, classified as Functional Scenarios. This raw video input was used to determine the severity encountered by the ego-vehicle (the testing vehicle for autonomous vehicles) during driving conditions. By measuring the distance between target actors (movable or static traffic objects) and the ego-vehicle, the severity of a scenario can be determined. In this study, YOLOv8 tracking capability was used to determine the distance between target actors and the ego-vehicle. YOLOv8 was used to identify the object and determine its ID in the frame. Once the object reached a certain threshold, the severity index was triggered as part of the visualization stage. In the visualization, only the highest severity index for a scenario is displayed, while lower severity indexes from previous detections are not displayed. The machine vision flowchart using YOLOv8 for object detection and visualization is presented in Figure 5-26.

The scenarios from the video database were fed into the machine vision algorithm once the configuration stage was completed. YOLOv8 utilized non-maximum suppression to eliminate unwanted overlapping detections. Based on this suppression approach, the remaining bounding boxes and class labels were extracted from the input frame using the prediction from the detection model. To demonstrate the detected objects through bounding boxes in the frame, the OpenCV library was used in this study. YOLOv8 utilized OpenCV's cv2.rectangle function to configure the rectangular shape around the object for each frame during the iteration process. Based on the coordinates for the bottom-right to top-left corners of the bounding box, the rectangle shape was drawn using the object detection model. In the next stage, a set of trackers was configured for each detected object to track objects in the frame. The trackers were linked with each bounding box, updating the position and size of the bounding box based on the

object's motion. Regions of Interest (ROIs) were established based on the distance between actors and the ego-vehicle. Based on the proximity between the ego-vehicle and the actor, a severity index was assigned to each ROI, increasing as the proximity range increased. The object IDs were gathered into the local database to eliminate repeated counts. The severity associated with the ROI was appended to a variable that stores the scenario severity. However, a new severity index would be considered for storage if an object appeared with a higher severity value in the ROI. The algorithm continued processing every frame until the final input frame provided in the machine vision algorithm. The outputs from the recorded data are based on severity rating, region of interest boundaries, and object detection bounding boxes.

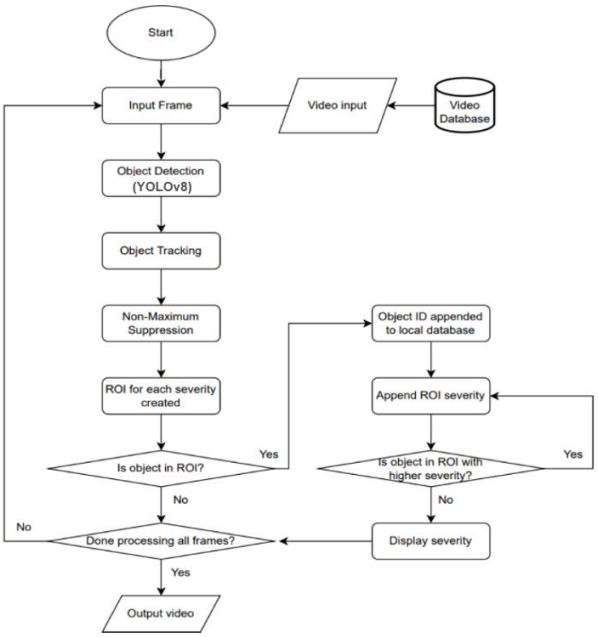
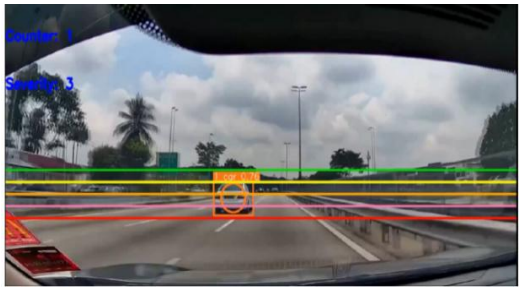
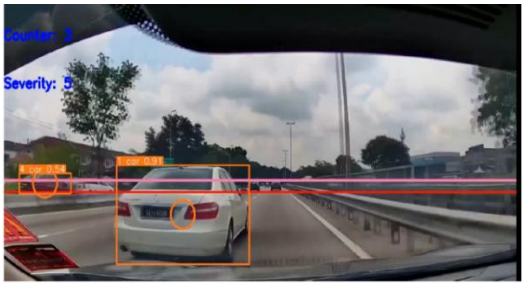


Figure 5-26 Flow chart for the objection detection using machine vision

Using the YOLOv8 algorithm, two main tasks were completed. The first task focused on detecting target actors, and the second on quantifying object IDs for each target actor from the video. The region of interest was displayed in the frame for visual analysis purposes, and five regions were designed for each level of severity. Figure 5-27(i) shows the representation of the five regions for a sample scenario on a highway. Five horizontal lines display the ROI that the vehicle presides in once it crosses the region corresponding to the severity levels. The severity levels are color-coded as follows: Level 1 in green, Level 2 in yellow, Level 3 in orange, Level 4 in pink, and Level 5 in scarlet. The highest severity index value measures the potential risk of each target actor, multiplied by other factors such as controllability and frequency. The initial object detection process and identification of severity within ROI are illustrated in Figure 5-27(i). Whereas Figure 5-27(ii) illustrates the updated severity index based on changes in the ROI. Based on the ROI, the scenarios are tagged and categorized under Abstract Scenarios, defined into several sub-categories as described in Table 5-3.



(i) Machine Vision Far Point



(ii) Machine Vision Near Point

Figure 5-27 Region of interest detected using machine vision

Table 5-3 Description of scenario

Tag	Description
Main Scenario	Primary scenario categorizing the level scenarios that commonly occurred on Malaysian driving environment. There are three main scenario tags as describe below: • Normal Driving Scenario (NORM) • Near Miss Scenario (NM) • Accident Scenarios (ACCD) A normal scenario classified for a vehicle driving without any hazard level and following all the traffic rules and regulation. A near miss addressing the certain level of hazardous conditions which has high potential for road accidents. The third scenario focusing on the accident scenario which addressing the high level of hazard situation.
1st sub-scenario	The first sub-scenario addressing the driving direction of the ego-vehicle that commonly occurred in Malaysian road traffic. This scenario can be tagged as • Driving in straight direction • Left turn driving • Right-turn driving
2nd sub-scenario	The second sub-scenario focusing on the ego-vehicle driving speed which is tagged into three level such as: • Longitudinal high speed • Longitudinal moderate speed • Longitudinal low speed
3rd sub-scenario	The third sub-scenario are addressing the signalling or non-signalling conditions on Malaysian routes. The scenario tags for third sub-scenario are: • Signalled routes • Non-signalled routes
Operational Design Domain	The ODD focusing on the road infrastructure that occurred at the designated testbeds
Actor	Addressing the potential actors that might interact with the ego-vehicle in a scenario
Day/Night	This scenario tags emphasizing the scenario occurred during night and during day.
Weather	This scenario tag focusing on different whether conditions which is generally classified as good or bad weather. Bad weather encompasses rain, mist, and haze while good weather addressing on sufficient day light, no rain and clear skies.
Severity	Displays the severity due to proximity of Ego vehicle and target actor via machine vision application.
Expected Severity	Displays the expected severity from looking at all the videos.
Scenario Variation	Provides an in-depth description towards scenario.
Characteristic of ego vehicle	Description of ego-vehicle behaviour during specific scenario

The Automotive Safety Integrity Level (ASIL) is a risk classification system that specifies automotive safety standards under the ISO 26262 standard. Based on severity, exposure, and controllability, the risk analysis of a potential hazard and assessment of risk

factors are conducted to assign the ASIL level. The risk index is measured using Equation 5.9 based on ISO 26262:

$$Risk = Frequency \times Severity \times Controllability \tag{5.9}$$

Table 5-4 presents the ASIL distribution used for comparison with calculated risk. The original ISO 26262 allocation table includes three severity classes (S1, S2, S3), four probability classes (E1, E2, E3, E4), and three controllability classes (C1, C2, C3). This study, however, expands the risk allocation table to five severity classes: 1-No Injuries, 2-Light Injuries, 3-Moderate Injuries, 4-Severe Injuries, and 5-Life Threatening Injuries, and five frequency classes: 1-Very Low, 2-Low, 3-Medium, 4-High, and 5-Very High. The controllability classes which depend on the capability to avoid dangerous situations remain as C1 Easy, C2 Moderate, and C3 Difficult. These expanded classes allow for a more detailed classification of scenarios. In Table 2, different colour codes correlate with ASIL values: ASIL A (green) represents the lowest hazard level, while ASIL D (red) represents the highest.

Table 5-4 Risk Allocation Table

Severity	Frequency	Controllability		
		C1 Easy	C2 Moderate	C3 Difficult
1 No Injuries	1 Very Low	1	1	1
	2 Low	2	2	2
	3 Medium	3	3	3
	4 High	4	4	4
	5 Very High	5	5	5
2 Light Injuries	1 Very Low	2	2	2
	2 Low	4	4	4
	3 Medium	6	6	6
	4 High	8	8	8
	5 Very High	10	10	10
3 Moderate Injuries	1 Very Low	3	3	3
	2 Low	6	6	6
	3 Medium	9	9	9
	4 High	12	12	12
	5 Very High	15	15	15
4 Severe Injuries	1 Very Low	4	4	4
	2 Low	8	8	8
	3 Medium	12	12	12
	4 High	16	16	16
	5 Very High	20	20	20
5 Life Threatening Injuries	1 Very Low	5	5	5
	2 Low	10	10	10
	3 Medium	15	15	15
	4 High	20	20	20
	5 Very High	25	25	25

A Graphical User Interface (GUI) is developed to measure and analyse the risk, as shown in Figure 5-28. The GUI helps obtain the abstract scenario based on initial scenario classification. This study develops the logical scenario category, as stated in ISO 34504, focusing on the parameterization of scenario sets. Logical scenarios emphasize parameter ranges and distribution in scenario classification, which are defined quantitatively. These parameters can vary based on the Operational Design Domain of the vehicle and the deployment location of the autonomous vehicle. Parameterization includes ego-vehicles, dynamic traffic objects, and static traffic objects based on the scenario category.

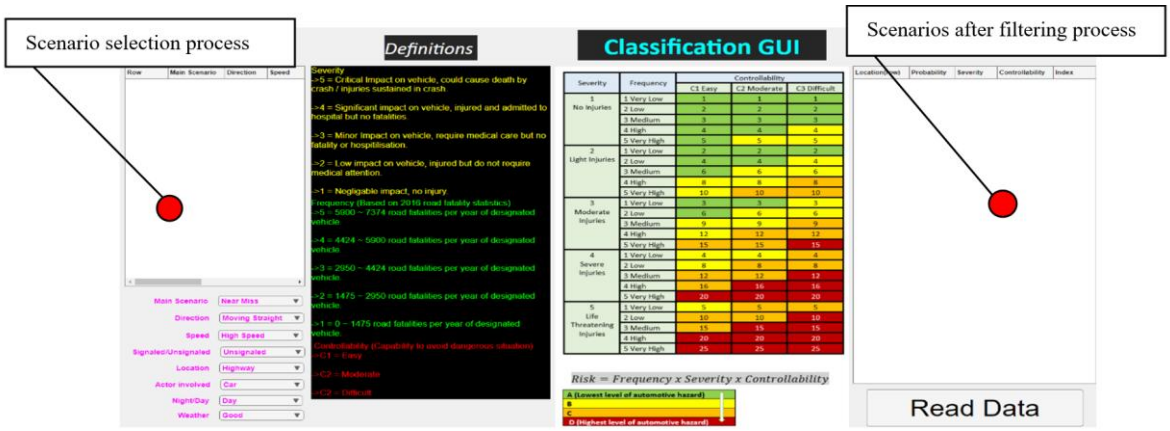


Figure 5-28 Graphical User Interface (GUI) for Scenario Classification

5.6 Development 3D Environment Model for Scenario-Based Testing

According to the process flow proposed, in order to develop scenarios for the scenario-based testing, the data collected using the instrumented vehicle must first be analysed and identified. The video recordings were feed to a deep learning neural network which is YOLOv8 for object detection. Based on the traffic object detection result, the critical scenarios occurred in the campus can be identified and extracted. Six different road actors were found in the datasets including car, truck, bus, bicycle, motorcycle, pedestrian. Figure 5-29 shows examples of the scenes identified. Figure 5-30 shows the road actors identified from data recorded at the selected road network.

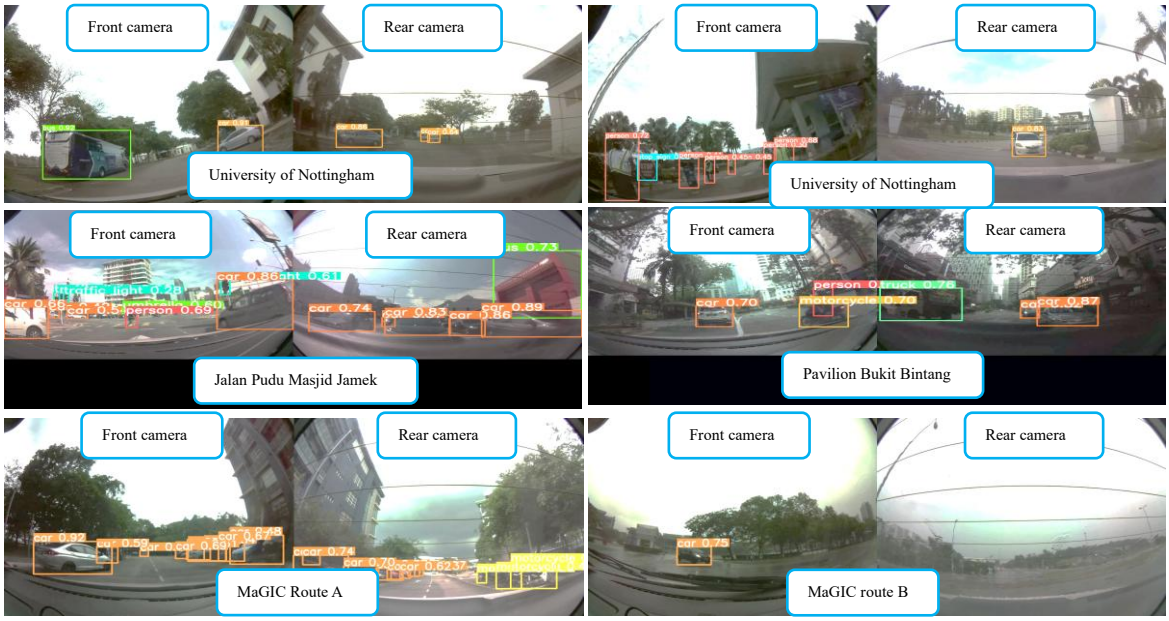


Figure 5-29 Examples of scenarios captured by the instrumented vehicle.

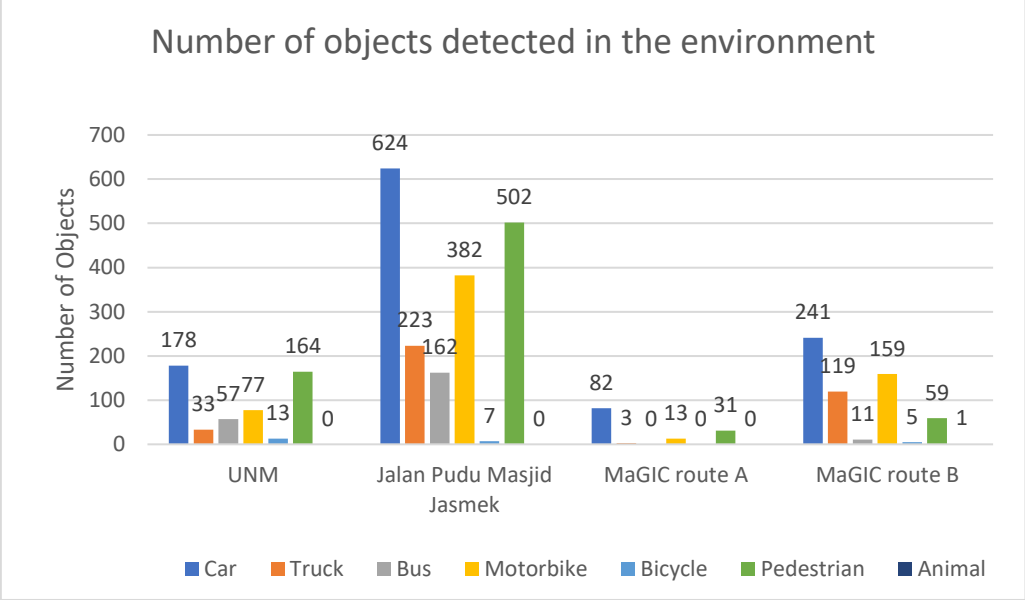


Figure 5-30 Distribution of object category for each route.

5.6.1 Development of 3D environment model

The 3D environment models, serving as a digital twin of the selected road environment for scenario development, are created using IPG CarMaker. Upon running the program, the software menu is displayed, where the *Scenario/Road* option can be found under the *Parameters* toolbar, as highlighted in Figure 5-31. As shown in Figure 5-32, the *Scenario Editor* function is used to build a virtual driving environment, allowing roads to be modelled without needing actual road data. Any road network can be created as desired.

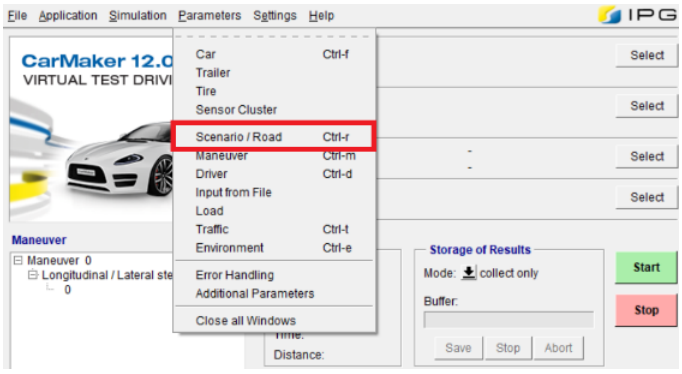


Figure 5-31 IPG CarMaker main GUI menu

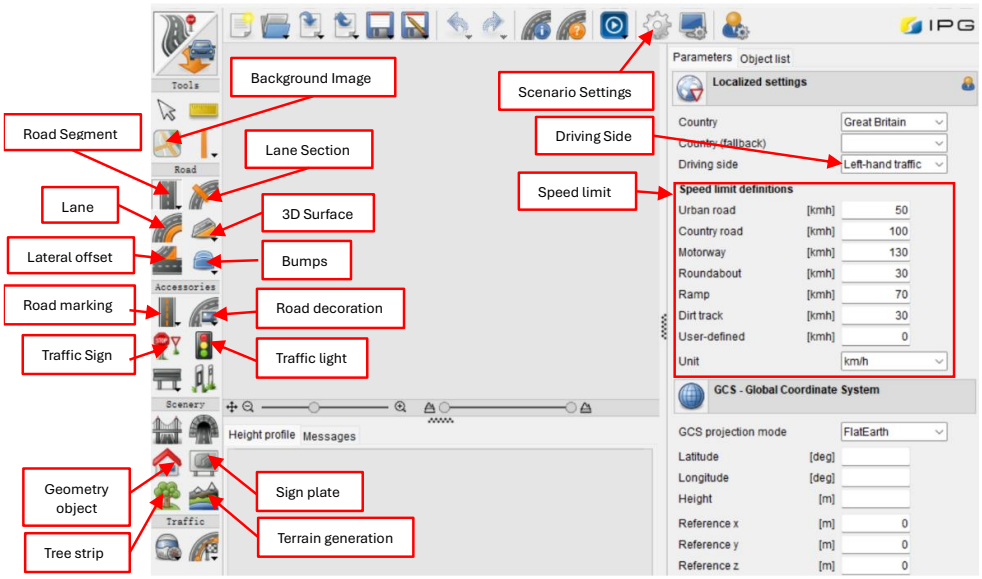


Figure 5-32 Scenario Editor tools for development of 3D environment model

In *Scenario Editor*, it is important to ensure the settings are adjusted appropriately. *Scenario Settings* should be selected to access *Localised Settings* in the right sidebar. The default *Driving Side* should be changed from Right-hand to Left-hand traffic to match Malaysian driving regulations. The speed limit for each road type can also be adjusted here. To replicate the road network of a specific location, *Background Image* should be selected under *Tools* in the left sidebar to import a map from Google Maps. This map will serve as a reference for constructing the road network. The orientation and scale of the map image can be modified as needed. The *Road* features in the left sidebar include *Road Segment*, *Lane Section*, *Lane*, *3D Surface*, *Lateral Offset*, and *Bumps*, which are used to create and modify the road network.

The road network is defined by *Links*, consisting of road segments that can be straight, turns, ramp, or junction. The *Lane Section* feature allows a Link to be split into sections, with modifications affecting only the selected section. The *Lane* feature enables the modification of existing lanes or addition of new lanes, and the adjustment of lane width. The *3D Surface* feature allows modifications to lateral, longitudinal, and camber gradients to illustrate slopes. The *Lateral Offset* feature allows the reference line to be moved laterally, which is useful for creating exit and entrance turns. The *Bumps* feature adds various road bumps, commonly used to illustrate sidewalks. The *Road Marking* feature allows lines to be drawn on the road, such

as no parking areas and zebra crossings. The *Road Decoration* feature configures road textures and adds lane or bus lane markers. The *Traffic Sign* feature allows the insertion and modification of traffic signs, while the *Traffic Light* feature adds and configures traffic lights. The *Traffic Barrier* feature places traffic barriers like guardrail and Jersey barriers. The *Guideposts* feature adds guideposts along the road. The *Bridge* and *Tunnel* features add these structures to the road network. The *Geometry Object* feature adds objects like houses, street furniture, animals, and people. The *Sign Plate* feature inserts images on sign plates, typically for road directions or advertisements. The *Tree Strip* feature adds multiple trees along the road, and *Terrain Generation* generates terrain around the created scenery.

Besides, creating a realistic testing environment involves including static objects such as buildings and train stations. Using Google Maps Street View, buildings and landmarks were surveyed, and 3D models were sourced from 3D Warehouse. These models were converted from ".skp" to ".kmz: format using SketchUp before being imported into IPG CarMaker. For examples, the 3D model of Sultan Abdul Samad Building, as shown in Figure 5-33, was imported into IPG CarMaker for use in the Light Rapid Transit (LRT) Masjid Jamek location. The building, located opposite Merdeka Square, is an iconic structure featuring domes and a clock tower. It is typically crowded with tourists, especially on weekends, as it is one of Kuala Lumpur's most famous landmarks. Another 3D model that was imported into IPG CarMaker for use in the LRT Masjid Jamek location is the 3D model of train station, as shown in Figure 5-34. The station, situated near Masjid Jamek in central Kuala Lumpur, is named after the area. It serves as an interchange station between the Ampang Line and Kelana Jaya Line, allowing commuters to switch between lines via a connected walkway. Using these tools, the 3D model of the selected road network is developed with reference to Google Maps as shown in Figure 5-35. Consequently, 3D virtual road models based on the actual dimensions of the selected road networks were created.

1. Figure 5-36 shows the 3D virtual road model developed based on University of Nottingham Malaysia campus route.
2. Figure 5-37 depicts the 3D virtual road model developed for Jalan Pudu Masjid Jamek.
3. Figure 5-38 demonstrates the 3D virtual road model developed for Pavilion Bukit Bintang.
4. Figure 5-39 presents the 3D virtual road model developed for MaGIC route A.
5. Figure 5-40 displays the 3D virtual road model developed for MaGIC route B.

Figure 5-33 3D model of Sultan Abdul Samad Building

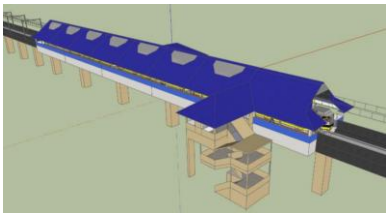


Figure 5-34 3D model of train station and track

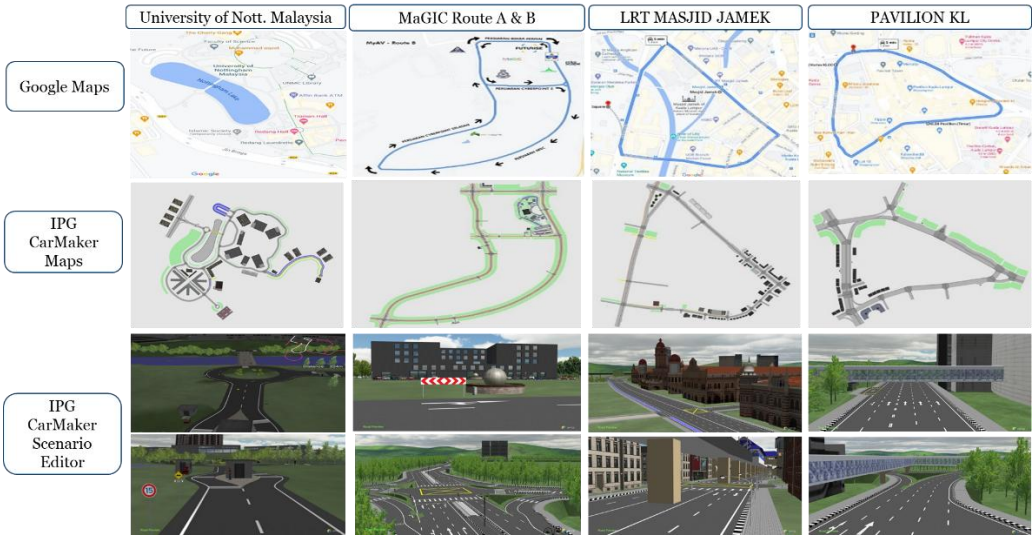


Figure 5-35 3D preview of the road environments developed in IPG CarMaker

Figure 5-36 3D virtual road model of the University of Nottingham Malaysia

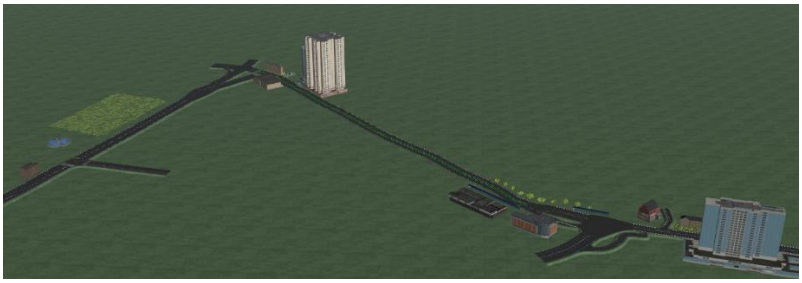


Figure 5-37 3D virtual road model of the Jalan Pudu Masjid Jamek area

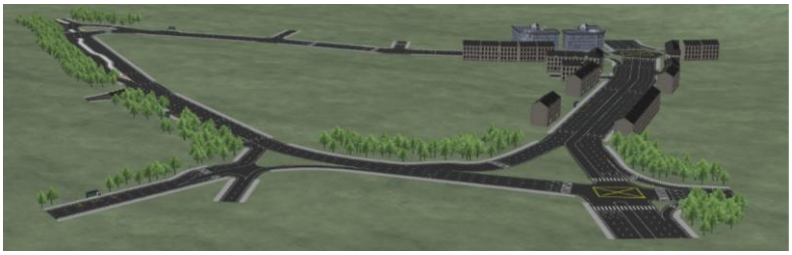


Figure 5-38 3D virtual road model of the Pavilion Bukit Bintang area

Figure 5-39 3D virtual road model of the MaGIC route A

Figure 5-40 3D virtual road model of the MaGIC route B

5.6.2 Development of scenarios

Based on the MaRSeD scenario classification tool developed, a total of 30 critical test scenarios were selected from the classified dataset and used to develop virtual scenarios. While numerous scenarios were created, only a subset from each route is presented here. The first two scenarios occurred at the University of Nottingham Malaysia. The first scenario involves avoiding a road obstacle, typically represented by a hole in the road with traffic pylons placed around it. This scenario mirrors common situations in Malaysia where road hazards require drivers to manoeuvre around them. The second scenario depicts a road crossing pedestrians' scenario, where drivers must slow down and stop to allow pedestrians to cross safely.

Moving on, the third and fourth scenarios took place in Kuala Lumpur's capital city area. The third scenario unfolded at the Pavilion Bukit Bintang area, simulating congested traffic with slow-moving vehicles. This scenario includes a sudden cut-in by a motorcycle, a common occurrence in congested Malaysian traffic. The fourth scenario occurred at Jalan Pudu Masjid Jamek area, where congestion and frequent stops by vehicles loading or unloading along the road are typical. Drivers must change lanes to avoid these stopped vehicles.

The fifth and sixth scenarios are set around the MaGIC area. The fifth scenario, at MaGIC route A, is a car-following scenario where the vehicle must follow a lead vehicle that is constantly changing speed. This scenario was chosen to add variety to the MaGIC route A, which sees little traffic and requires all vehicles to adhere to speed limits. While not critical, including scenarios like this is important for testing autonomous vehicles on this route, which is certified for such testing. Lastly, the sixth scenario at MaGIC route B involves an animal crossing the road, a scenario often leading to accidents in Malaysia as drivers attempt to avoid colliding with animals. Including such scenarios is crucial for comprehensive testing.

(a) Road Object Avoidance

This scenario took place at the University of Nottingham Malaysia's route between the Sport Complex and student accommodation hall. In this scenario, the ego vehicle needs to travel forward however there are obstacles on the way in front, so the ego vehicle needs to apply manoeuvre to avoid collision with the obstacles. The obstacles are traffic pylons which are put on the centre of the junction as there is a hole on the road. At the beginning of this scenario, there is a road bump on the road, so the vehicle needs to slow down to cross the bump. There are also pedestrians walking along the right lane, so the ego vehicle can only travel on the left lane apply manoeuvre adjustment to the left side of the road to avoid collision with the obstacles. Furthermore, there are many vehicles parked at the side of the road, so the vehicle also needs to prevent collision with these vehicles when avoiding the obstacles. Figure 5-41 shows the road object avoidance test scenario developed using IPG CarMaker and Figure 5-42 shows the comparison between the video frame captured from the instrumented vehicle and the video frame captured from the virtual vehicle.

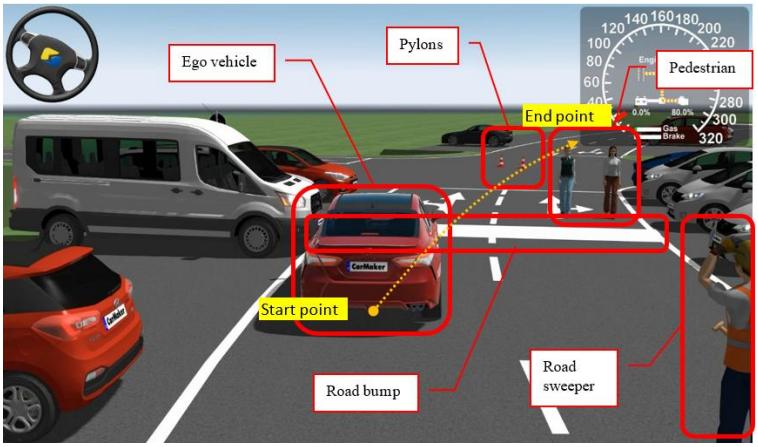


Figure 5-41 Road object avoidance test scenario developed using IPG CarMaker

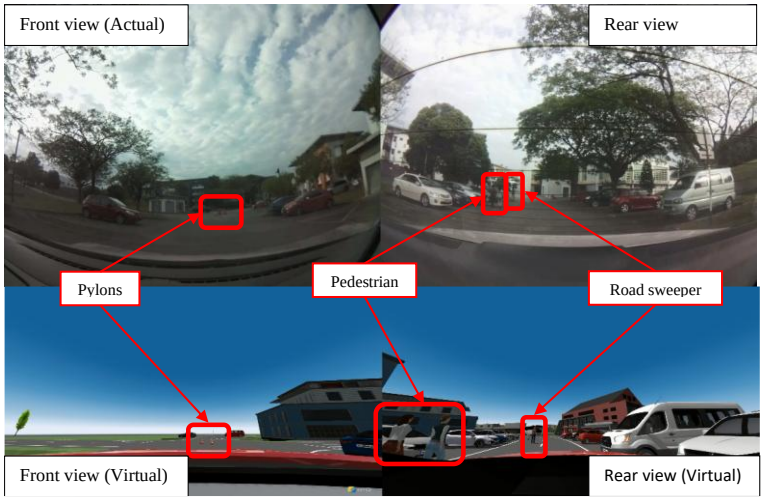


Figure 5-42 Onboard camera comparison. Top figure shows the video frame recorded from instrumented vehicle and the figure below is the video frame from IPG CarMaker. The frame on the left is the front view and the right is the rear view.

(b) Pedestrian Road Crossing

The second scenario took place at the pedestrian crossing in front of the University of Nottingham Malaysia's Trent building. In this scenario, the ego vehicle is following a red car in front with vehicle speed of 20 km/h. However, there is a group of pedestrians crossing the road in front. The red car in front applied brake and stopped to let the pedestrians to cross. Therefore, the ego vehicle needs to apply brake to slow down while maintaining distance with the vehicle in front and stop the vehicle to wait the pedestrians to cross. Once the pedestrians crossed the road, the red car in front will start moving and the ego vehicle will follow the red car. Figure 5-43 shows the pedestrian road crossing virtual scenario developed and Figure 5-44 shows the onboard camera comparison between real-world and virtual simulation.

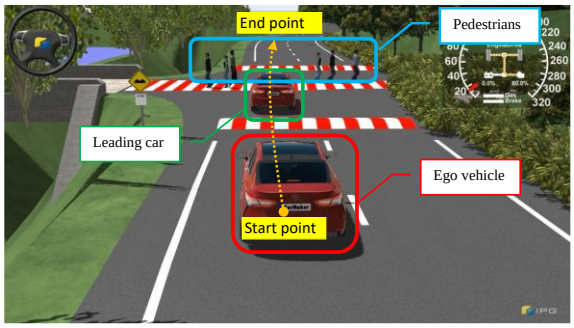


Figure 5-43 Pedestrian Road crossing test scenario developed using IPG CarMaker

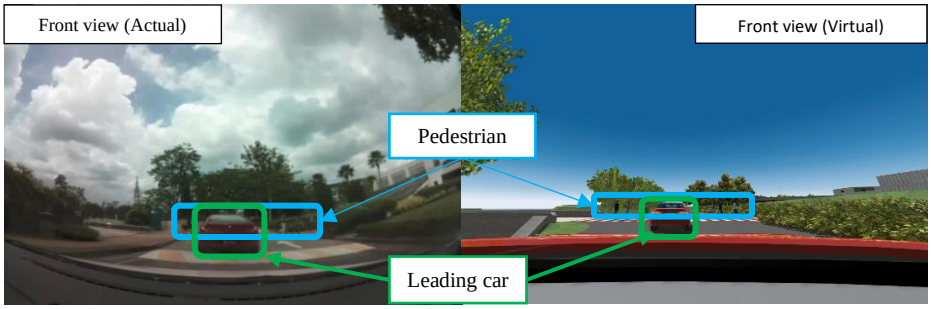


Figure 5-44 Onboard camera comparison

(c) Motorcycle Cut-in

This scenario occurred at one of the right turn corners near the Pavilion, Bukit Bintang, Kuala Lumpur. In this scenario, the ego vehicle followed the stop-and-go rhythms of the congested traffic flow, adjusting its speed to match the nearby vehicles. As the ego vehicle was about to make its manoeuvre to close-up the distance from the front vehicle, a motorcyclist suddenly cut into its lane from the left as shown in Figure 5-45. To avoid a collision, the ego vehicle had to apply emergency braking. After the motorcyclist passed by, the ego vehicle continued its path through the congested road.

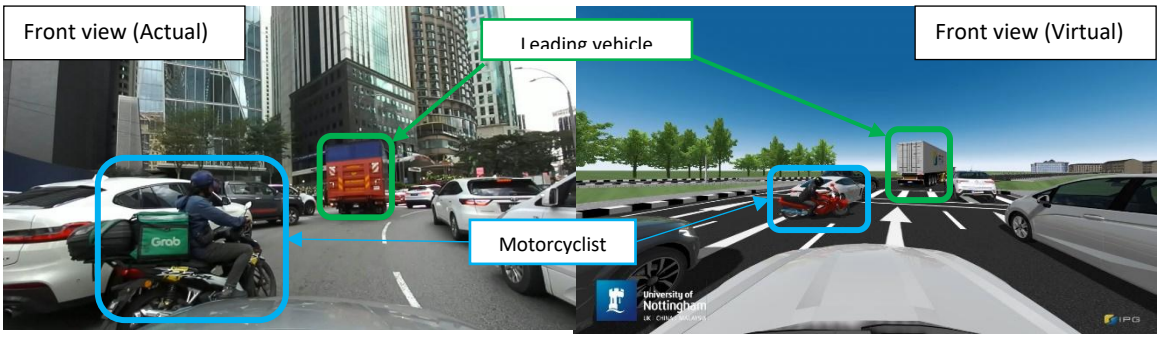


Figure 5-45 Moment of the motorcycle cut-in occurred captured by onboard camera.

(d) Lane Change Overtaking

This scenario occurred at Jalan Pudu area, where the ego vehicle needs to overtake a stopped vehicle ahead by changing lane. However, due to the congested traffic, there are vehicles coming from the fast lane as shown in Figure 5-46. So, the driver needs to make sure no vehicle coming from the fast lane, and it is safe to perform overtaking. Once no more vehicle coming from the right-hand side, the ego vehicle performs lane change to overtake the stopped vehicle ahead.

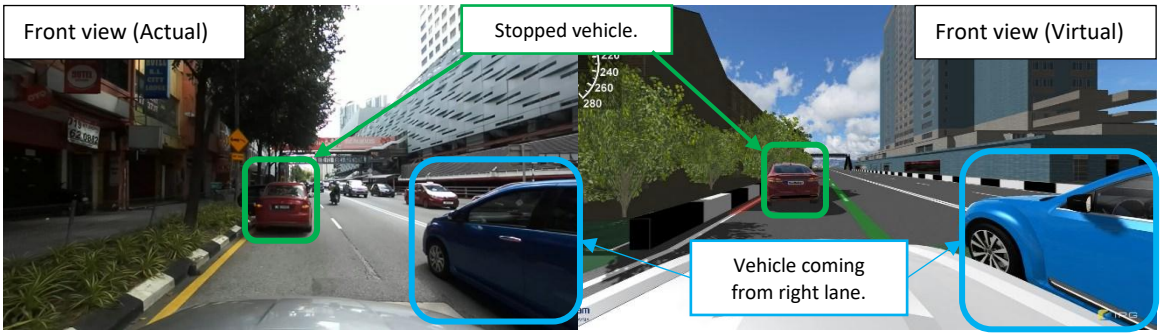


Figure 5-46 Ego vehicle waiting until safe to overtake.

(e) Car Following

In everyday driving scenarios, the “car-following” scenario is the most encountered situation. In this scenario, the ego vehicle maintains a consistent speed while ensuring a safe distance from the leading vehicle ahead. The scenario occurred at MaGIC route A as shown in Figure 5-47. The ego vehicle adjusts its acceleration in response to changes in the speed of the vehicle ahead in the "car-following" scenario. Figure 5-48 shows how the leading vehicle looks like from the ego vehicle perception both in real world and simulation. The drivers are required to follow the car while maintaining a safe distance from it. This will generate a valuable dataset for training the driver model to regulate speed when following another vehicle.

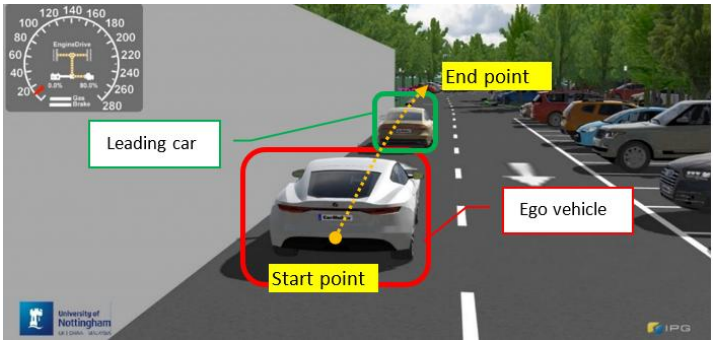


Figure 5-47 Ego vehicle following car while maintaining a safe gap.

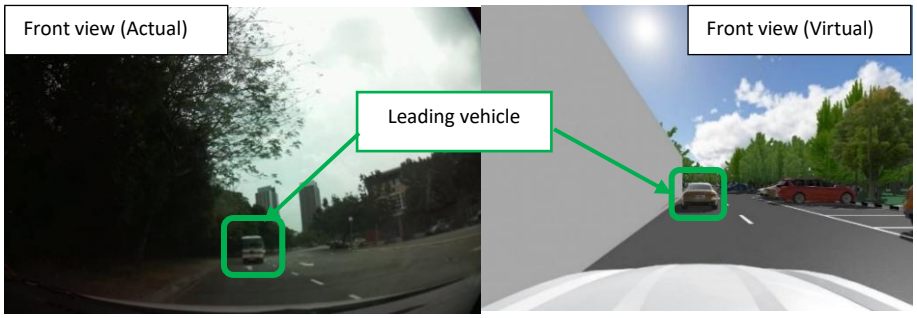


Figure 5-48 Front camera view from the ego vehicle.

(f) Animal Road Crossing

This scenario took place on MaGIC route B. While the ego vehicle was moving at approximately 30 km/h, an animal suddenly appeared in the middle of the road, crossing it, as shown in Figure 5-49. The ego vehicle had to brake to avoid colliding with the animal. After the animal crossed the road, the ego vehicle accelerated and resumed its path. Figure 5-50 shows the appearance of the animal from the perspective of the ego vehicle, both in the real world and in the simulation. It can be observed that the animal is very small, making it difficult for drivers to notice it from a far distance while driving, which poses a challenging test for the driver's reaction.

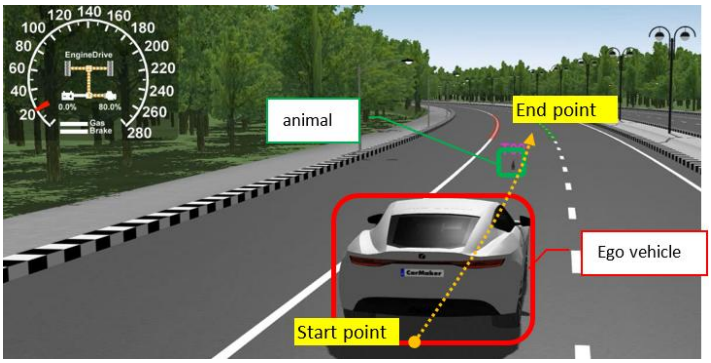


Figure 5-49 Animal road crossing scenario developed in the IPG CarMaker

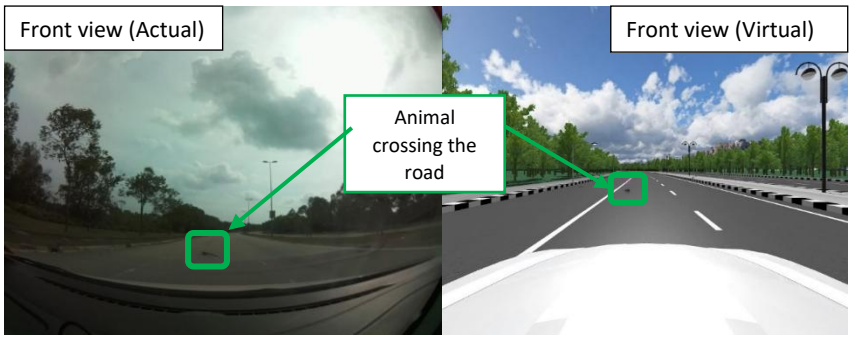


Figure 5-50 Animal front camera view of the ego vehicle.

5.6.3 Comparing built-in IPGDriver against human driver.

Virtual scenarios were developed in IPG CarMaker based on actual scenarios captured from the instrumented vehicle. However, to test a safety system for an autonomous vehicle, a driver model is required to manoeuvre the vehicle by providing driving inputs to the virtual vehicle in the simulator. The accuracy of the driver model in replicating the driving style of human drivers is crucial to ensure the reliability of the test results. Therefore, in this section, the driving data recorded from the instrumented vehicle is compared with the data simulated in IPG CarMaker. In this experiment, the driver in the instrumented vehicle is a human driver, whereas the driver in the IPG CarMaker virtual vehicle is the built-in IPGDriver, designed to replicate the driving style of average drivers. The first two scenarios that were mentioned in the previous section were selected to evaluate the performance of the IPGDriver in reproducing the driving input of the human driver.

(a) Road obstacle avoidance

Figure 5-51 shows the screenshots of virtual ego vehicle manoeuvring to avoid the road obstacle in front by sliding to the left side of the road. Figure 5-52 to Figure 5-56 shows the vehicle response obtained from the instrumented vehicle and the IPG CarMaker simulation. The vehicle responses that are analysed in this scenario are the pitch rate and yaw rate of the vehicle. Meanwhile, driving inputs that are used for the analysis are steering angle, gas pedal input and brake pedal input. From the data plots, it can be observed that the driving input produced by the IPGDriver and the vehicle response from the IPG CarMaker is similar to the actual driving input and vehicle response. From Table 5-5, it can be observed that the steer angle, brake input, gas input, pitch rate and yaw rate show the percentage difference of RMS errors about 10.42%, 30.67%, 3.18%, 51.33% and 17.55% respectively.

Figure 5-55 and Figure 5-56 show noticeable differences in yaw rate and pitch rate between the real-world and the virtual scenario. These differences can be attributed to several factors. Firstly, simulations often rely on idealized representations of road geometry and surface conditions, which may not capture the variability and unpredictability inherent in real-world environments. This disparity can lead to differences in vehicle dynamics and sensor responses between simulated and actual conditions. A study comparing real and simulated performance for an off-road autonomous ground vehicle in obstacle avoidance highlighted such

discrepancies, noting that simulations might not fully account for the complexities of real-world terrains and obstacles [338].

Secondly, the simplifications inherent in sensor models within simulations can contribute to these errors. Simulated sensors often operate under controlled conditions, lacking the noise, environmental interferences, and occlusions present in real-world scenarios. This can result in simulated sensor data that is less variable and more predictable than actual sensor outputs, leading to discrepancies when algorithms trained or tested in simulation are deployed in real-world applications. Research on the transferability of virtual versus physical-world testing of autonomous driving systems has emphasized the challenges posed by such "reality gaps," where the performance in simulation does not fully translate to real-world effectiveness [339]. Besides, the driving style of a human driver and the virtual IPGDriver is not the same. From Figure 5-53, it can be observed that at around 1.5 seconds, the IPGDriver pressed the brake pedal later than the human driver. So, the IPGDriver needs to apply braking force to the vehicle at a shorter time as shown by the gradient of the pedal inputs. This caused the vehicle to stop at a higher deceleration. As a result, the vehicle pitched down at a higher rate due to higher inertial value.

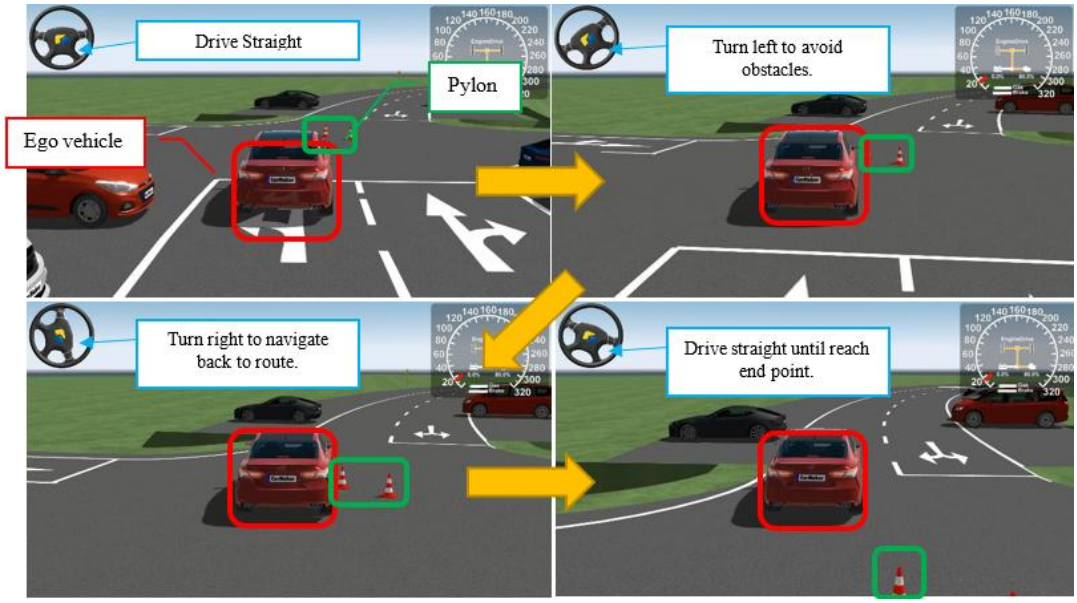


Figure 5-51 Road obstacle avoidance scenario manoeuvre

Similar to the braking scenario, the IPGDriver also applied gas input to the vehicle at a higher rate as shown in Figure 5-54. Therefore, the vehicle pitched up at a higher rate as shown in Figure 5-55. Whereas the large yaw rate difference at around 2 seconds of the experiment is due to the higher steer angle provided by the IPGDriver at 2 seconds. However, it can still be observed that the pattern of the vehicle response and driving data obtained from the virtual environment closely followed the data captured from real world environment. Furthermore, the large RMS errors also show that the driving behaviour between a human driver and the default virtual driver are different. Therefore, the IPGDriver is not suitable to be used as driver model for evaluation of safety system of autonomous vehicles for Malaysia driving conditions.

While no simulation can perfectly replicate the real-world complexities, in order to address these issues, one proposed solution involves enhancing the simulation environment to better mimic real-world variability [340]. This includes incorporating more detailed road surface models and dynamic obstacle representations to reduce the gap between simulated and actual conditions. Additionally, improving sensor models to account for noise and environmental uncertainties can make simulated sensor data more representative of real-world scenarios. Techniques such as domain randomization, which involves varying environmental parameters during simulation, have been shown to improve the robustness of models when transferred to real-world applications [341], [342].

Despite the high percentage of error observed, the testing platform's results are considered acceptable and reliable for the current work. The identified discrepancies are largely due to known limitations in the simulation's ability to replicate all aspects of real-world environments. By acknowledging these limitations and implementing the proposed solutions, the simulation platform can serve as a valuable tool for preliminary testing and validation of autonomous vehicle systems [342]. Furthermore, simulations offer a controlled environment to test scenarios that may be challenging to reproduce consistently in the real world, providing insights that are crucial for the iterative development of advanced driver assistance systems.

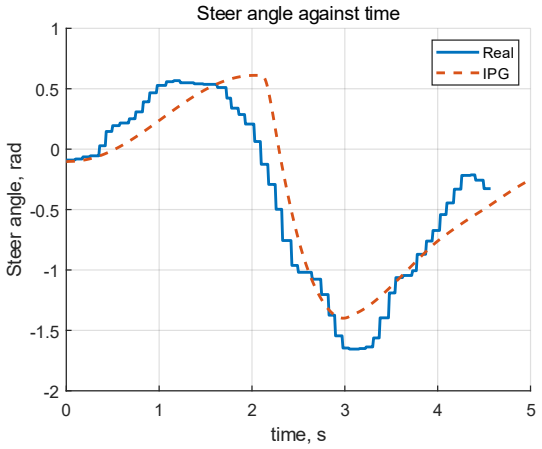


Figure 5-52 Steering angle input for road obstacle avoidance

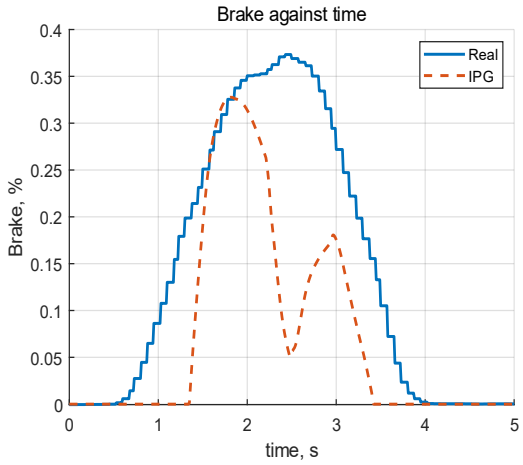


Figure 5-53 Brake pedal input for road obstacle avoidance

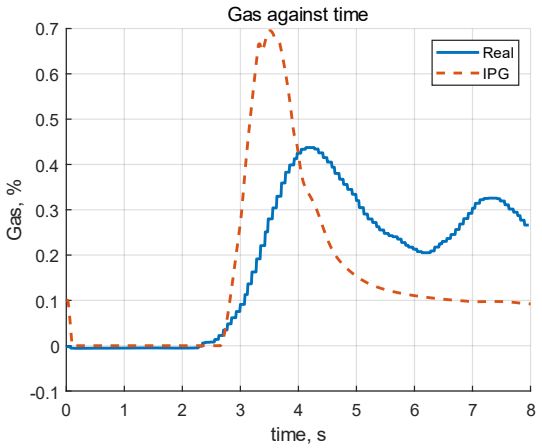


Figure 5-54 Gas pedal input for road obstacle avoidance

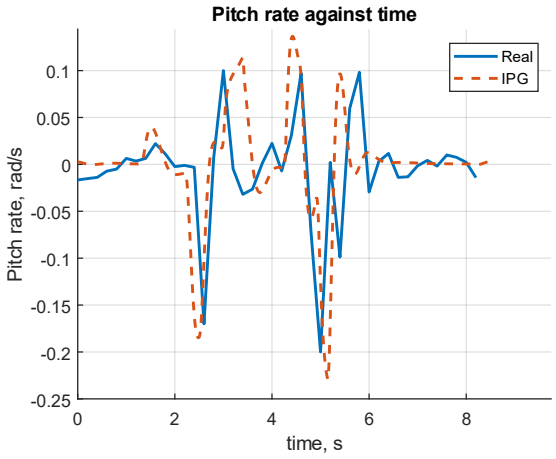


Figure 5-55 Pitch rate response for road obstacle avoidance

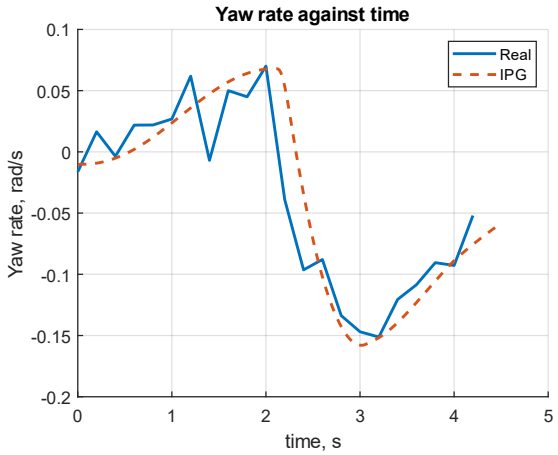


Figure 5-56 Yaw rate response for road obstacle avoidance.

Table 5-5 Percentage of RMS error for road obstacle avoidance test

Testcase	Observation data	Root Mean Square (RMS)		Percentage of error (%)
		IPG	Real	
Road obstacle avoidance	Steering angle, rad	0.6481	0.7235	10.42
	Brake input, %	0.1006	0.1451	30.67
	Gas input, %	0.1590	0.1541	3.18
	Pitch rate, rad/s	0.05752	0.03801	51.33
	Yaw rate, rad/s	0.07661	0.06517	17.55

(b) Pedestrian at zebra crossing

For pedestrian road crossing scenario, the ego vehicle followed the traffic vehicle in front and stop behind the red vehicle while waiting for the pedestrians to cross the road in lateral direction. After the pedestrians crossed the road, the leading traffic vehicle start moving from the current positions. Then, the ego vehicle starts moving following the leading traffic vehicle by maintaining the longitudinal distance. *Figure 5-57* shows the screenshots of the lateral trajectory for the pedestrian road crossing scenario. *Figure 5-58* to *Figure 5-60* show the vehicle response obtained from the instrumented vehicle and the virtual vehicle model from IPG CarMaker simulation. The vehicle responses such as pitch rate response are evaluated in this test scenario since the vehicle driving in a straight direction without any steering input. Meanwhile, gas pedal input and brake pedal input are used to analyse as the driving inputs of the vehicle.

From the data plots, it can be observed that the driving input applied by the driver and the vehicle response from the IPG CarMaker simulation are similar to the actual driving inputs and vehicle response. In Figure 5-58 , both the human driver and IPGDriver applied up to 46.57% and 42.53% of the brake pedal input, respectively, at the start of the experiment. However, the human applied a large braking force only for the first two seconds, then reduced the braking input to around 30% after the car stopped moving. In contrast, the IPGDriver maintained a higher braking force until the distance from the leading car exceeded the default safe distance of 2 meters.

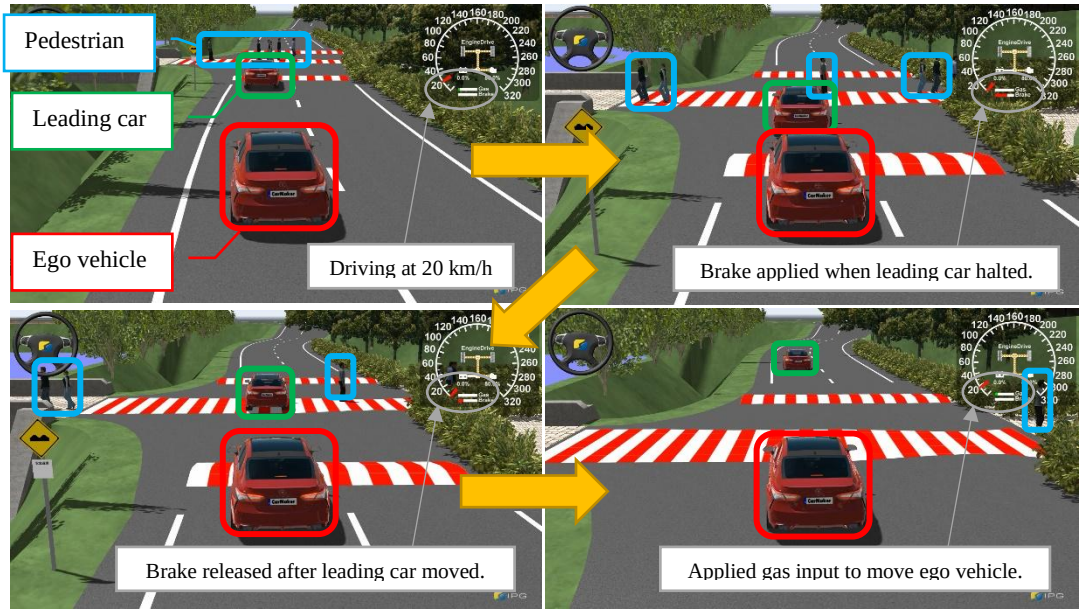


Figure 5-57 Pedestrian crossing road scenario's manoeuvre.

In Figure 5-59, it can be observed that once there was enough gap from the leading car, both the human driver and IPGDriver applied gas input to follow the leading car. However, the IPGDriver demonstrated a more aggressive driving style compared to the human driver, as shown by the higher gradients in its inputs. As a result, the brake input, gas input, and pitch rate show percentage differences in RMS errors of about 0.1388%, 12.83%, and 14.75%, respectively. The large RMS errors indicate that the IPGDriver is unable to replicate human driving behaviour accurately in this test case. Therefore, a driver model that can reproduce human driving behaviours is needed.

Additionally, the pitch rate of the vehicle obtained from the real-world and IPG CarMaker showed noticeable differences. Several factors contributed to this discrepancy, primarily related to differences in road-surface modelling, variations in vehicle dynamics, and the contrasting driving styles between human drivers and the IPGDriver simulation model. One key factor leading to the high RMS errors is the difference in road surface representation between the virtual and real-world environments. The real-world road surface exhibits minor undulations, variations in texture, and localized irregularities, which affect vehicle motion, particularly in longitudinal dynamics and pitch angle fluctuations. In contrast, the virtual road model is based on an idealized, mathematically smooth surface that does not account for these small-scale road imperfections. This disparity influences both vehicle acceleration and braking responses, leading to differences in pitch motion, as real-world suspension and tire compliance interact with uneven terrain in a way that is not fully replicated in the simulation.

Another significant cause of the high RMS errors is the difference in driving styles between human drivers and the IPGDriver model. Human drivers exhibit adaptive, experience-based decision-making, adjusting acceleration, braking, and steering input based on external cues, perceived pedestrian behaviour, and environmental feedback. For instance, the human driver applied a gradual acceleration strategy when the front car start moving, incorporating real-time visual cues and personal judgment regarding pedestrian movement. The IPGDriver,

however, operates based on predefined mathematical models that lack the intuitive variability of human behaviour. It often follows an algorithmically optimized driving profile, which can be more abrupt or overly conservative compared to the smoother, anticipatory driving patterns exhibited by human drivers. This difference in accelerating and braking patterns leads to inconsistencies in the vehicle's pitch response, as sudden acceleration in the simulation causes a more pronounced backward pitch motion compared to the real-world vehicle, which may exhibit more progressive suspension dynamics, as shown in Figure 5-60. Consequently, more precise measurement of the real environment is needed for the development of the environment model to ensure that the vehicle response in the real and virtual environments is identical. The summary of RMS percentage errors for the pedestrian crossing test is shown in Table 5-6.

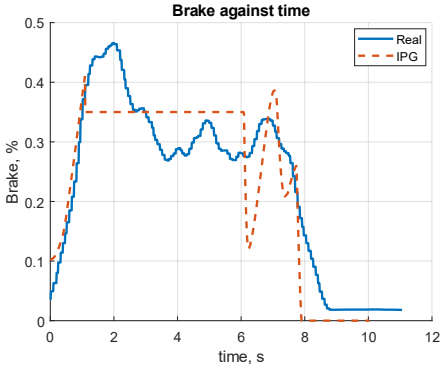


Figure 5-58 Brake pedal input for pedestrian crossing scenario

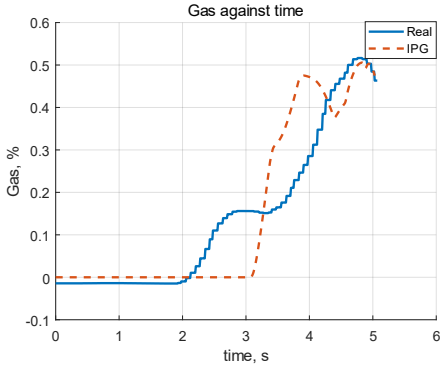


Figure 5-59 Gas pedal input for pedestrian crossing scenario

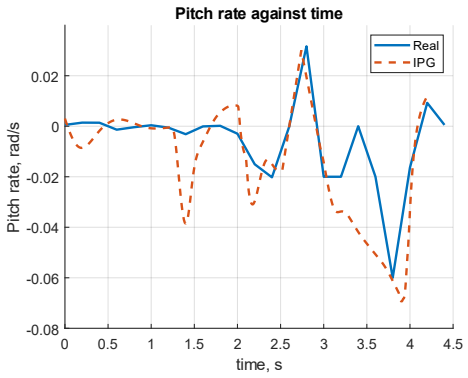


Figure 5-60 Pitch rate response for pedestrian crossing scenario.

Table 5-6 Percentage of RMS error for pedestrian crossing scenario.

Testcase	Observation data	Root Mean Square (RMS)		Percentage of error (%)
		IPG	Real	
Pedestrian crossing	Brake input, %	0.1441	0.1439	0.1388
	Gas input, %	0.2022	0.1792	12.83
	Pitch rate, rad/s	0.01648	0.01405	14.75

5.7 Summary

In this chapter, a motion simulator was developed and integrated with IPG CarMaker to provide realistic motion feedback for an immersive driving experience. To enhance immersion further, a virtual reality (VR) headset was incorporated, allowing drivers to experience a 360-degree cockpit view within the simulated environment. The development of this simulation platform aimed to bridge the gap between real-world driving and virtual testing, providing a high-fidelity environment for evaluating driving behavior and vehicle responses.

A systematic approach was established to extract and classify real-world driving scenarios, ensuring adherence to international safety testing standards. The Malaysian Road Scenario Database (MaRSeD) was developed as a comprehensive tool for identifying key traffic scenarios unique to Malaysia's driving conditions. Using video data recorded from the instrumented vehicle, road users and objects were detected and classified with the YOLOv8 deep learning object detection algorithm, enabling a structured approach to analysing real-world traffic interactions. From this dataset, 30 unique driving scenarios were identified, and a subset of them was selected for 3D environment modelling within IPG CarMaker. These virtual scenarios were designed to closely replicate real-world road environments, allowing for controlled and repeatable testing conditions.

To evaluate the accuracy of the virtual models and the performance of IPG CarMaker's built-in driver model (IPGDriver), comparative analyses were conducted between real-world driving data and simulated vehicle responses. The results demonstrated that the virtual simulations were able to reproduce the key characteristics of real-world scenarios, capturing the overall trend of vehicle dynamics and interactions. However, significant discrepancies were identified between human driver behaviour and the default IPGDriver model, particularly in braking strategies, acceleration profiles, and steering input adaptation. These differences resulted in noticeable variations in vehicle pitch, yaw rate, and speed regulation, making the default IPGDriver unsuitable for safety testing in Malaysian traffic conditions.

Recognizing these limitations, the study emphasized the need for a driver model optimized for Malaysia's traffic environment, ensuring reliable and representative safety testing for autonomous vehicles. To facilitate the development of this data-driven driver model, a human-in-the-loop approach was implemented, where participants were invited to drive on the simulator under different traffic scenarios. Their driving inputs, reaction times, and behavioural patterns were recorded to construct a dataset reflecting natural driving tendencies. The collected data served as the foundation for training a deep learning-based driver model, which will be used to enhance the accuracy and realism of virtual driver behaviour within the simulation platform.

The findings of this chapter reinforce the validity of using a high-fidelity simulation environment for vehicle safety assessment, while also highlighting the need for region-specific adaptations in driver modelling. The development of a Malaysia-optimized driver model will contribute to more reliable, representative, and scalable autonomous vehicle testing, ensuring that future AV technologies are evaluated under realistic, locally relevant driving conditions.

Chapter 6: Development of An Artificial Intelligence Driver Prediction Model for Autonomous Vehicle Testing Platform

6.1 Overview

Deep learning requires a large amount of labelled driving dataset to train. As discussed in Chapter 2, capturing and labelling driving dataset from actual driving in real world is time consuming and expensive. Therefore, another method that can generate high quality driving dataset easily and at low cost is using synthetic dataset. However, as the aim of this study is to train the driver model to reproduce the driving behaviour of Malaysian drivers, an artificial intelligence-based driver model is proposed in this chapter.

The chapter begins with the development of a driver model designed to replicate human driving styles. The first section covers the collection of human driving data for training the model. This involves inviting participants to drive on a simulator, evaluating the simulator's performance based on participant feedback, and recording their driving data. The recorded data is classified by driving style, and data exhibiting normal driving behaviour is extracted to construct the training dataset. The second section details the preprocessing and preparation of the dataset for training. The third section discusses the construction and training of the driver model's neural network. Following this, the chapter examines the model's performance using both simulation and real-world testing data. The final section provides a summary of the chapter.

6.2 User Experience Evaluation of the Driving Simulator

To assess the immersive nature and realism of the VR Driving simulator and gather feedback for enhancement, a user test involving 30 participants was conducted. In accordance with the University of Nottingham's Code of Research Conduct and Research Ethics, this study underwent a comprehensive ethical review to ensure adherence to the highest standards of research integrity and participant welfare. The ethical review process was conducted by the University's Research Ethics Committee, which operates under the guidelines established in the University's Code of Practice for Research Ethics Committees. This framework aligns with the principles outlined in the Declaration of Helsinki, emphasizing respect for individuals, informed consent, and the prioritization of participant welfare.

The ethical review encompassed a thorough evaluation of the study's aims, methodologies, and data management procedures. Key considerations included ensuring that participants' dignity, rights, and safety were upheld throughout the research process. The committee assessed the procedures for obtaining informed consent, measures for maintaining confidentiality, and strategies for minimizing potential risks associated with participation. Participants were provided with detailed information about the study's objectives, procedures, and any potential risks or benefits involved. This transparency allowed individuals to make informed decisions regarding their involvement. Consent was obtained prior to participation,

ensuring that individuals were aware of their rights, including the option to withdraw from the study at any point without any negative consequences.

The study's design included measures to protect participant confidentiality and data security. All personal information and responses were anonymized and securely stored, accessible only to authorized research personnel. This approach aligns with the University's commitment to maintaining high standards of data protection and participant privacy. By adhering to the University's ethical guidelines and obtaining the necessary ethical clearance, the study ensured that all research activities were conducted responsibly and ethically. This rigorous ethical oversight not only protected the participants but also enhanced the credibility and integrity of the research findings.

The evaluation employed both pre-Test (pre-TQ) and Post-Test (po-TQ) questionnaires. The pre-TQ comprised four sections: personal data, 3D gaming and HMD familiarity, and driving experience. Participants completed this questionnaire before the test session to gauge susceptibility to motion sickness. Users with minimal or no exposure to 3D gaming or prior HMD usage were cautioned beforehand. The po-TQ, comprising 23 questions, was derived from expert evaluation using the ITC-Sense of Presence Inventory (ITC-SOPI) [343] to gauge presence in immersive VR environments. Focused on user experience with the prototype, this questionnaire was completed by participants after the driving session to assess the perceived realism of the driving experience. It mainly queried aspects related to typical virtual reality systems, contrasting them with real-world experiences.

6.2.1 Participant Composition

The study comprised 30 participants, consisting of 23 males and 7 females. The test drivers were selected without prior knowledge of their backgrounds. All participants were affiliated with the University of Nottingham Malaysia, including lecturers, staff, and students. The age of the participants varied between 18 to 60 years, averaging 32.7 years old. All participants possessed a minimum of secondary school education, with 10 individuals holding university degree with an average of 16.1 years of education. The majority, constituting 70% of the sample (21 individuals), identified as Chinese, while 13% were Malay (4 individuals), 10% Indian (3 individuals), and 7% classified themselves as other (2 individuals). During testing, 83% (25 individuals) reported being active drivers, whereas 17% (5 individuals) had limited or discontinued driving. In this study, “active” driving was defined as engaging in at least single driving experience per week. Among those who were not active drivers, the primary reasons for discontinued driving were the lack of personal vehicle ownership, due to financial constraints or reliance on public transportation. The average driving experience, defined as the total number of years of holding a driver's license, was 15 years for the group. Except for five users, all participants had experience with 3D content, including 3D movies using regular 3D glasses and virtual reality. An average duration of 45 minutes was allotted for each participant.

The sample size for the driving simulator experiment was determined based on an empirical method proposed by Wang et al. [344], who used the mean squared error (MSE) curves of significant variables to infer the proper sample size for a driving simulator experiment

to evaluate the safety of freeway driving design. The experiment indicates that 30 was an acceptable sample size. Besides, lengthy experimental procedures (providing information, training, etc.) and high cost of running simulator studies also lead to small samples were tested which is also supported by other studies that used similar or smaller sample sizes for simulator studies [345]. Therefore, same method was applied to this study, which involved testing the effects of integrating VR to the driving simulator system.

6.2.2 Execution of the User Test

First, the participants completed the pre-TQ questionnaire, followed by an orientation on the simulator's basic operations. Information about simulator sickness, including its occurrence and symptoms, was provided before the formal testing commenced. Following that, participants wore the Oculus Quest 2 to experience an immersive visual simulation of sitting in the cockpit of a virtual vehicle within the system. During the test session, various driving scenarios representing Malaysian traffic conditions were presented. These included typical maneuvers such as curve driving, car following, emergency braking, and lane departure warnings, as well as unexpected events like an approaching motorcycle or a pedestrian crossing the road to simulate avoidance of head-on collisions. Upon completing all driving tasks, participants were then asked to fill out the post-TQ questionnaire.

6.2.3 User Test Result

The post-test questionnaire (po-TQ) comprised 23 items utilizing a 5-point Likert scale, designed to evaluate the concept of presence. Through factor analysis, four key factors emerged: controllability of the system, spatial presence, naturalness, and negative effects. Table 6-1 displays the items alongside the average scores provided by participants for both the driver-in-the-loop simulator with and without virtual reality integration.

The quantitative analysis of po-TQ responses revealed intriguing findings: when queried about feeling present in the virtual environment, a significant 80% of participants reported a heightened sense of presence while using the virtual reality (VR) simulator. Notably, with the VR headset, users could freely move and rotate their heads within the cockpit, contributing to an improved view and higher scores regarding difficulties in observing lane markers. Moreover, this enhanced cockpit view provided a sense of better control, reducing incidents of the vehicle steering off the road, particularly during sharp turns. These outcomes generally support the hypothesis that VR yields a more realistic driving experience compared to a fixed-screen setup.

However, some findings contrasted this trend. For instance, regarding the perception of controlling vehicle speed using the gas pedal and brake, most users favoured the fixed-screen setup, expressing challenges with control inside the VR headset. Users reported difficulties in perceiving their body's movement, pedals, and steering wheel due to limitations of the IPG CarMaker, making vehicle control more challenging within the VR environment. Another noteworthy discovery arose from the question "I am still aware of the real world." Surprisingly, more users acknowledged awareness of the real world within the VR simulator (80%), with

20% showing disengagement and 2% remaining neutral. This phenomenon could stem from heightened sensory sensitivity when the brain perceives a non-real environment, causing users to become more attuned to external stimuli. Additionally, users noted that the lack of a realistic surround sound system reduced the overall immersive experience within the VR setting.

Table 6-1 po-TQ questionnaire and the corresponding responses collected.

po-TQ item		**Average Score**	
		Without VR	With VR
Controllability	I can control the speed of the vehicle using the gas pedal and brake.	4.1	3.8
	I have difficulty seeing the lane markings in the simulator	3.6	2.9
	Driving with simulator feels almost like driving in a car cockpit.	3.3	4.2
	I think that the car will go off the road often when I drive with the simulator	3.5	3.1
	I feel like I am in control of the car when driving with the simulator	3.6	4.4
Spatial Presence	I was aware of the real world	3.5	3.3
	I wanted to see more of the space in the displayed environment than I was able to	3.9	3.3
	I found it easy to forget that I was watching a display	2.6	4.3
	I felt I was visiting the places in the displayed environment	3.9	4.3
	I had a sense of being in the scenes displayed	3.6	4.5
	I felt that the characters and/or objects could almost touch me	3.3	4.0
	The temperature of the real world distracted me	3.7	3.9
	I was distracted by the quality of the technology	2.8	3.1
Naturalness	The content seemed believable to me.	3.8	4.3
	The displayed environment seemed natural	3.7	4.1
	I had a strong sense that the characters and objects were solid.	3.5	3.8
	Did the 6 degree of freedom movements of the motion platform contribute to the realistic of the driving simulator	3.8	4.1
	How much did this experience seem consistent with your real-world experiences	3.5	4.7
	Were the motion and visual feedback synchronized together?	3.9	4.5
Negative Effects	I felt prone to vomiting	1.8	2.6
	I felt dizzy	2.1	3.2
	I felt nauseous	1.5	1.5
	I felt fear	1.6	1.7

Moreover, the analysis revealed that over half of the users experienced motion sickness during the VR tests, particularly on curvy routes. Three participants experienced simulation sickness within the initial five minutes and had to halt their sessions. Interestingly, most users experiencing motion sickness had minimal or no prior HMD experience, except for one user. Notably, none of the users experienced vomiting. The motion sickness experienced by users likely stemmed from the VR system's low refresh rate and resolution. This mismatch created a sensory conflict between the visual and motion feedback delivered by the simulator and the cognitive expectations of the users.

Furthermore, another limitation of the current setup is the lack of a surround sound system. Accurate auditory feedback is essential for drivers to gauge their speed, especially

when visual cues are limited or absent. Studies have shown that drivers tend to underestimate their speed in the absence of acoustic feedback [346]. In a simulator, a surround sound system can replicate the spatial and directional qualities of sound, providing cues about movement and speed. This auditory information, combined with visual feedback, helps create a more immersive experience that closely mimics real-world driving, allowing for more accurate speed perception and decision-making by the driver. Therefore, sound, and visual enhancements can be implemented to further improve the immersion of the current system.

6.2.4 Comparison of driving behaviour of test drivers

In this section, the driving behaviour of four participants, including an aggressive driver, two average drivers, and a careful driver, was selected for comparison out of thirty participants. Four scenarios, involving motorcycle cut-in, lane change, car following, and animal road crossing, were chosen for evaluation. Drivers were classified based on speeding behaviour, identified as the most prevalent aggressive action and the primary cause in 30.7% of fatal crashes according to a study in [347]. The percentage of time spent travelling at a speed exceeding 60 km/h by each driver was calculated using Equation 6.1:

$$\%time\ speeding = \frac{time\ spent\ travelling\ faster\ than\ 60\ kmph}{time\ spent\ travelling\ faster\ than\ 20\ kmph} \times 100 \tag{6.1}$$

The threshold of 60 km/h was chosen as it represented the 99th percentile of the vehicle speed distribution in the dataset, exceeding the 50 km/h speed limit on most urban roads [348]. Only time spent traveling faster than 20 km/h was considered, presuming speeds below 20 km/h were due to reasons other than driver aggressiveness, such as roadblock or traffic congestion. In Figure 6-1, the speed distribution profiles of both a non-aggressive and an aggressive driver are illustrated. It can be observed that more time at higher speeds was allocated by the aggressive driver compared to the non-aggressive driver. Using Equation 6.1, the percentage of time spent speeding for all participants was calculated, with results presented in Figure 6-2. The findings indicate that over 10% of driving time exceeding 60 km/h was spent by the top 10% of participants, equating to five individuals. Conversely, a percentage of time speeding below 4% was exhibited by the bottom 10% of participants.

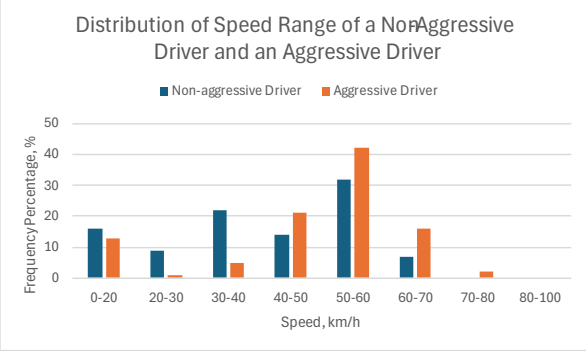


Figure 6-1 Speed distribution of a non-aggressive driver and an aggressive driver.

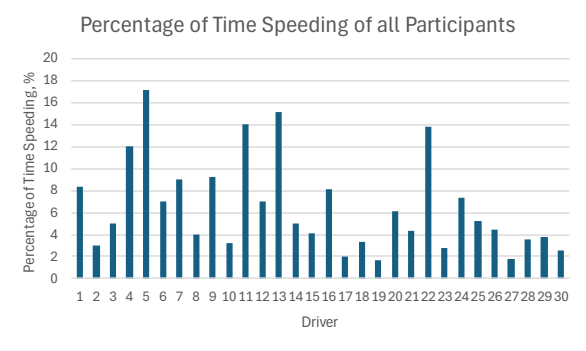


Figure 6-2 Percentage of time spent exceeding 60 km/h for all 30 participants.

In the second method, drivers were categorized based on aggressive tailgating behaviour. Among various aggressive behaviours as defined by NHTSA [349], tailgating was

chosen primarily because it can be quantitatively measured more conveniently compared to behaviours like erratic driving or failure to yield. The percentage of time spent following the vehicle ahead with a time to collision less than 2.56 sec was calculated using Equation 6.2.

$$\%of\ time\ tailgating = \frac{time\ to\ collision\ less\ than\ 2.56\ s}{time\ to\ collision\ less\ than\ 5\ s} \tag{6.2}$$

The threshold of 2.56 seconds was chosen based on research conducted by Yue et al. [350], which determined that the average time to collision without any accidents over a 45-day investigation period was 2.56 seconds. Additionally, Hirst and Graham [351] observed that a 4-second time to collision could differentiate between situations where drivers inadvertently find themselves in danger and scenarios where they maintain control. Saffarzadeh et al. [352] also noted that drivers' behaviour varies across different situations, suggesting that there is no absolute time to collision threshold to distinguish between safe and unsafe car-following situations. Therefore, only times with a time headway of less than 5 seconds were considered, presuming that a TTC greater than 5 seconds indicated free-flowing traffic rather than close following. After calculating these percentages for each driver, rankings were determined in descending order based on these percentages. The top 10% (N = 5) were categorized as "aggressive drivers," the bottom 10% as "slow drivers," and the remaining 20 participants as "normal drivers". The driving style, reaction time and steering angle consistency of all the participants is shown in Table 6-2.

Table 6-2 Performance of all the participants

Driver ID	Driver Type	Reaction Time (s)	Steering Angle Consistency (%)
1	Normal	1.45	82
2	Slow	2.30	92
3	Normal	1.45	81
4	Aggressive	0.95	67
5	Aggressive	1.00	66
6	Normal	1.55	85
7	Normal	1.52	80
8	Normal	1.55	81
9	Normal	1.58	83
10	Slow	2.25	89
11	Aggressive	0.85	68
12	Normal	1.50	80
13	Aggressive	0.90	70
14	Normal	1.60	83
15	Normal	1.50	84
16	Normal	1.48	79
17	Slow	2.20	90
18	Normal	1.47	82
19	Slow	2.35	91
20	Normal	1.60	83
21	Normal	1.50	80
22	Aggressive	0.80	65
23	Slow	2.40	93
24	Normal	1.52	80
25	Normal	1.47	81

26	Normal	1.55	82
27	Slow	2.25	89
28	Normal	1.50	80
29	Normal	1.48	75
30	Normal	1.55	81

(a) Motorcycle Cut-in

First of all, it is important to clarify that all 30 participants successfully completed all test cases without any issues. However, for the purpose of detailed analysis, only four drivers were selected to represent different driving profiles in this section. These selected drivers were chosen to provide a representative comparison of different driving behaviours, including aggressive, normal, and cautious driving styles. Additionally, it should be noted that the Driver 1, Driver 2, Driver 3, and Driver 4 labels do not correspond to the same individuals across different test cases. Instead, they are used to illustrate distinct driving behaviours within each scenario. This approach ensures that the discussion remains focused on variations in driving styles rather than individual driver performance, providing a more generalizable and meaningful interpretation of the results.

In this scenario, participants drove until simulation time t=16 seconds, when a motorcyclist abruptly cut into the ego vehicle's lane, requiring the test drivers to apply the brakes to slow down and avoid a collision. From Figure 6-3, it is evident that driver 1 exhibited the most aggressive driving behaviour, reaching the highest speed. Consequently, driver 1 also experienced the highest deceleration during the emergency braking process to avoid colliding with the motorcyclist. Figure 6-4 also shows that driver 1 travelled the farthest among the four drivers, indicating that the vehicle came to a stop very close to the motorcycle, with only 3 meters of separation between them. On the other hand, driver 4 demonstrated extremely careful driving behaviour, as evidenced by starting to slow down from a distance when the motorcyclist was approaching from behind, resulting in no need for full braking. Drivers 2 and 3 exhibited behaviours between the extremes of driver 1 and driver 4. These drivers showed average acceleration and deceleration, enabling them to maintain a safety distance of up to 10 meters. Therefore, drivers whose graphs resemble those of driver 2 and driver 3 will be labelled as normal drivers, and their driving data will be used for the development of the driver model in the next chapter.

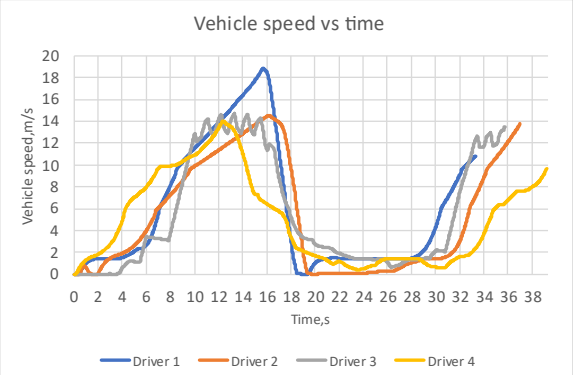


Figure 6-3 Vehicle speed of different drivers in the motorcycle cut-in test scenario.

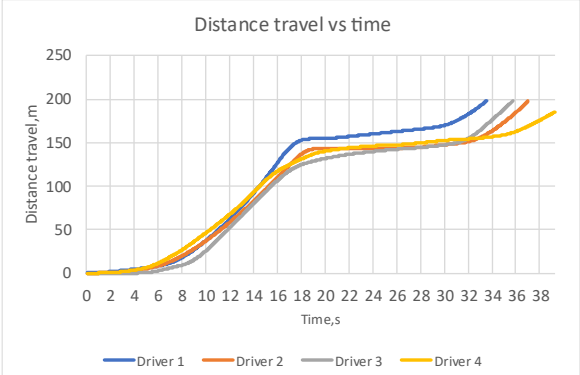


Figure 6-4 Distance gap between the vehicle driven by different drivers with the motorcycle.

(b) Lane Change Overtaking

In this scenario, a vehicle is stopped on the road, requiring drivers to perform a lane change to overtake it. This scenario exhibits the largest deviation in driving behaviours. There are three different types of lane-changing manoeuvres. First, drivers such as driver 2 and driver 4 notice the stopped vehicle from a distance, driving at a low speed and searching for opportunities to change lanes. However, driver 4, unlike driver 2, is less patient. When driver 4 sees a sufficient gap from the car behind, driver 4 quickly turns the steering wheel and accelerates to change to the right lane. Due to the low speed, driver 4 must apply a large steer angle, as shown in Figure 6-5. Meanwhile, driver 2 stops the vehicle and turns the steering wheel to prepare to change lanes, as shown in Figure 6-6. Once the vehicle from the right lane has passed, only driver 2 accelerates to perform the lane change. Therefore, driver 2 is considered a careful driver. An impatient driver like driver 3, driving at high speed, performs a lane change without carefully checking whether it is safe to do so. Driver 1 is between driver 3 and driver 4, exhibiting fast driving behaviour but ensuring it is safe to perform a lane change. Figure 6-7 shows the changes in latitude of each driver over time. It can also be observed that unlike other drivers who maintain at the right lane after changing lanes, driver 4 performs a double lane change in this scenario to return to the left lane.

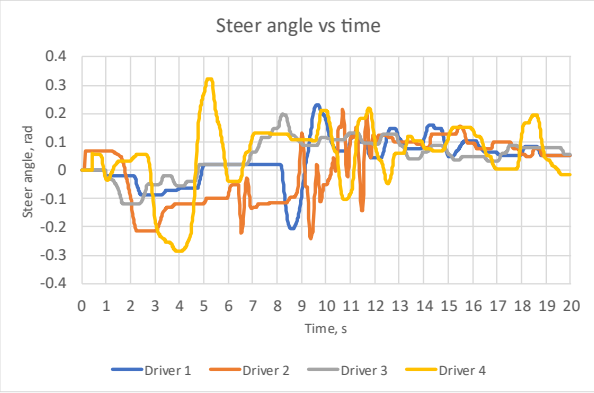


Figure 6-5 Steer angle applied by different drivers in the lane change test scenario.

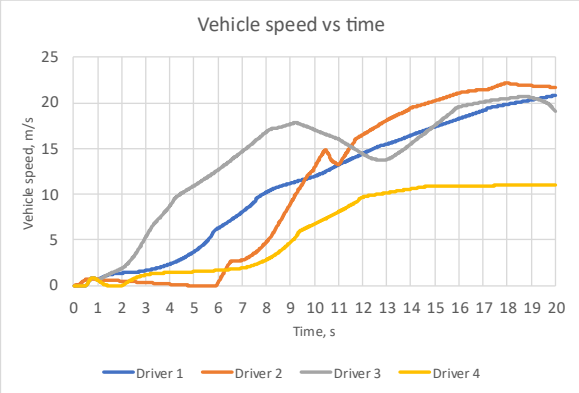


Figure 6-6 Vehicle speed of different drivers in the lane change test scenario.

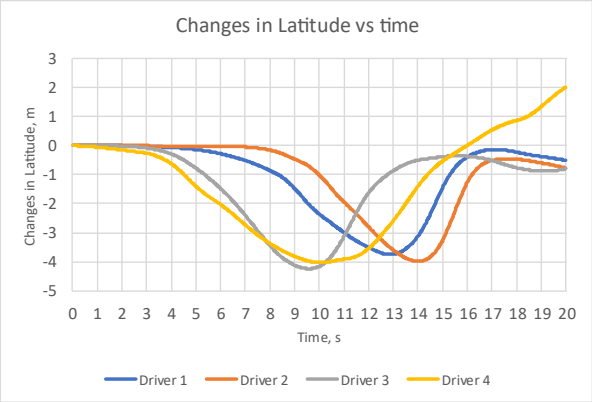


Figure 6-7 Changes in latitude of different drivers in the lane change test scenario.

(c) Car Following

In this scenario, drivers need to follow a leading vehicle with variable speed while maintaining a safe distance between the vehicles. Comparing the velocity of the drivers in Figure 6-8, it is evident that Driver 3's graph fluctuates the most, with the highest velocity reaching 20m/s, whereas Driver 1 is a careful driver who maintains an almost constant speed between 3m/s and 10m/s. Driver 2 and Driver 4 exhibit similar graphs, representing standard driving behaviour. Figure 6-9 shows the distance travelled over time by each driver. It can be observed that driver 1 and driver 3 have similar graphs in the first half of the simulation, which is consistent with their velocity graphs. However, in the second half of the simulation, driver 3 displays impatience by following the leading car more closely in an attempt to overtake it. This results in driver 3 accelerating and decelerating more frequently. In contrast, driver 4 is more aggressive, driving at a higher speed and reducing the gap from the leading vehicle, whereas driver 2 and driver 1 attempt to maintain the distance between the vehicles.

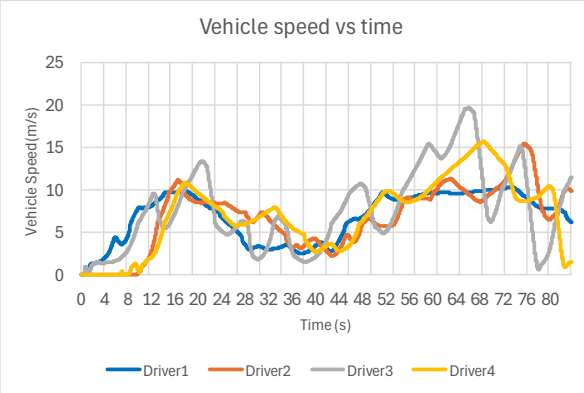


Figure 6-8 Vehicle speed of different drivers in the car following test scenario.

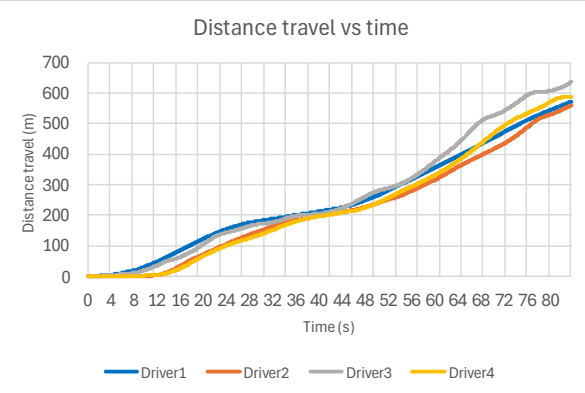


Figure 6-9 Distance travel of different drivers in the car following test scenario.

(d) Animal Road Crossing

In this scenario, a small animal is in the middle of the road, located 80 meters away from the starting point. Figure 6-11 reflects the reckless driving behaviour of driver 1, who stopped before the animal at the lowest distance to crossing among the eleven drivers, marked at -0.24m. Driver 4 drove very carefully, maintaining a low speed of around 7m/s as shown in Figure 6-10. Therefore, driver 4 did not have to apply a large braking force to decelerate the vehicle and could maintain a distance of more than 10 meters from the small animal. Meanwhile, driver 2 and driver 3 exhibited similar driving behaviours, both driving at around 10m/s with only slight differences in braking and accelerating behaviours. Driver 3 tended to accelerate and decelerate faster than driver 2.

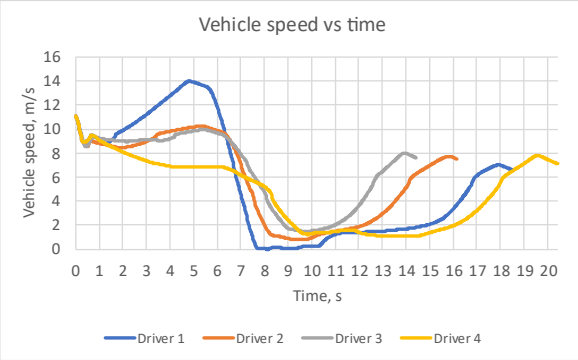


Figure 6-10 Vehicle speed of different drivers in the animal crossing test scenario.

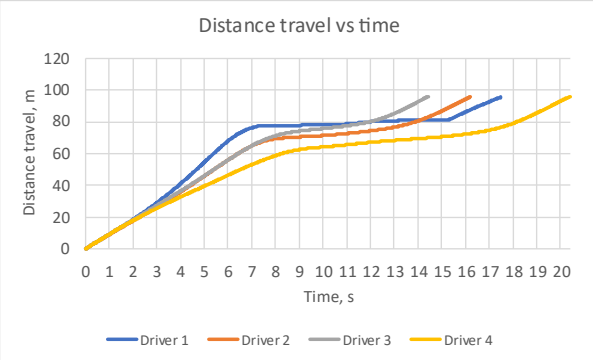


Figure 6-11 Distance gap between the vehicle driven by different drivers with the animal.

From the driving data recorded from the participants, it can be observed that for a driver model to exhibit normal driving behaviour, the difference in vehicle speed should not exceed 4m/s compared to a normal driver's data. This threshold range is crucial for indicating whether the driver model is performing normal driving during the performance evaluation of the driver model later.

6.3 Pre-processing and Preparation of Training Data

Based on previous research works [353], dataset generation for training, validation, and testing forms the cornerstone of every supervised machine learning application. Accuracy is crucial in achieving desired outcomes in supervised machine learning, which heavily relies on how input data is collected, fed into the learning agent, and extracted. Additionally, data science principles, particularly the extraction of relevant data, are applied in this study. Given the complexity involved, statistical analysis of the data is necessary to mitigate the risk of creating a flawed database. Hence, this section outlines the strategy for extracting and partitioning the dataset into three subsections. To facilitate training of a neural network model through supervised learning, the dataset is structured as an input array with dimensions $n \times 5$ every delta (Δ) time, where n represents the number of timestamps in the dataset. Each row in the matrix includes the following features, as shown in Equation 6.3:

$$dataset = [v, \delta, \rho, \tau, w_{t+\Delta t}] \tag{6.3}$$

Where, v is the vehicle velocity, δ is the steering angle, ρ is the throttle input, τ is the brake input, $w_{t+\Delta t}$ is the next waypoint of the test track.

The idea for including only the variables in Equation 6.3, while omitting other vehicle states, is to enable the neural network to develop a more complex representation of these parameters. This approach allows the network to predict odometry for various types of vehicles through training. To create the dataset for training of time-series neural network, first, the raw dataset as shown in Figure 6-12 is pre-processed where the driving inputs, vehicle speed and waypoint data were normalized using min max scaler to the range of 0 to 1. Besides, frame images are extracted from the videos recorded from front and rear facing cameras as shown in Figure 6-13 so that repeated frames are readily available for analysis. Moreover, this will

ensure that the repeated frames do not need to be extract from the videos repetitively and this could enhance the training process.

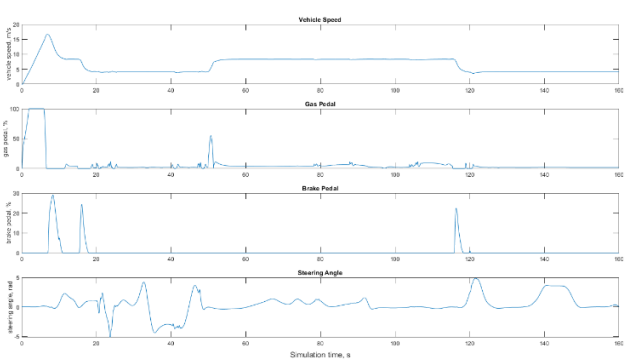


Figure 6-12 An example of raw driving data collected from the dataset.

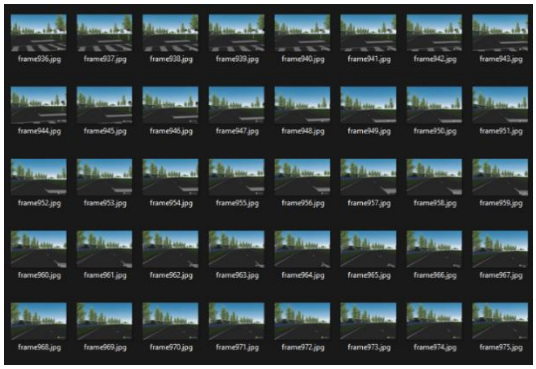


Figure 6-13 An example of front camera image frame from the dataset.

In order to initiate the training process for the driver model, the dataset needs further processing. The recorded driving data includes various scenarios typical of Malaysian traffic, such as driving forward, making turns, changing lanes, driving at normal speeds or moving slowly in traffic jams, and driving on straight or curved roads. Figure 6-14 represents the steering angle distribution, showing the number of data points recorded in each interval using a histogram. It can be observed that the dataset is highly unbalanced, with an overwhelming number of neutral and small steering angles. This imbalance occurs because roads are straight, and only a small portion requires high steering angles. A model trained on such unbalanced data might tend to drive straight, even while showing low mean square errors during training. To address this bias, data points with steering angles larger than 5 degrees and counts less than the mean of the intervals are resampled three times. The datasets are then randomly shuffled before initiating the training process for the machine learning model.

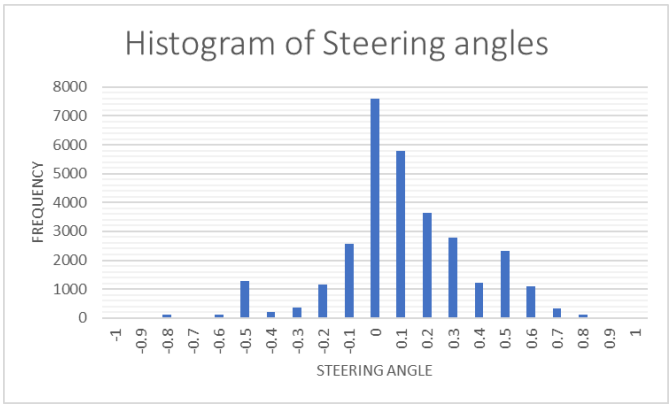


Figure 6-14 Histogram of steering angles in the dataset.

In deep learning training, it is common practice to split the dataset into three parts: training, validation, and testing datasets for cross-validation [354]. A training dataset is used to tune the weights of the neural network to fit the input data. A validation dataset is similar to the testing dataset, where the samples are not used to tune the weights of the neural network. However, the validation dataset provides continuous, unbiased evaluations of the model's performance on unseen data during training. This helps in tuning hyperparameters, selecting

the best model, and preventing overfitting by enabling early stopping. Meanwhile, a testing dataset provides an unbiased evaluation of the final neural network. This evaluation is based on the accuracy of the neural network to generalize the problem using samples that are not used to update the weights of the network. Unlike the validation dataset, the testing dataset is employed only after the model has been fully trained to offer a final, unbiased measure of the model's performance, assessing how well the model generalizes to completely unseen data. In this study, the dataset is split into training, validation, and testing datasets with a ratio of 70:20:10.

6.4 Neural Network Models

This section presents the architecture of a novel driver model for autonomous vehicles based on CNN-LSTM. The proposed model leverages the strengths of convolutional neural networks (CNNs) and long short-term memory (LSTM) networks to process multimodal input data, including RGB images, depth images, and vehicle state data. The model is configured to extract features from RGB images using a pre-trained ResNet-50, extract temporal dependencies using LSTM, and utilize custom CNNs to process depth images. The final predictions for steering angle, gas, and brake inputs are made through fully connected layers. This methodology emphasizes the novelty of integrating attention mechanisms and autoencoders, which enhance the model's performance and robustness compared to traditional CNN-LSTM models. The combination of these techniques addresses specific challenges in autonomous driving, providing significant advantages over traditional CNN-LSTM models. The model is trained using Red-Green-Blue-Depth (RGB-D) camera images and vehicle state data, ensuring a comprehensive approach to enhancing the accuracy and robustness of autonomous vehicle navigation. Figure 6-15 shows the model architecture of the driver model.

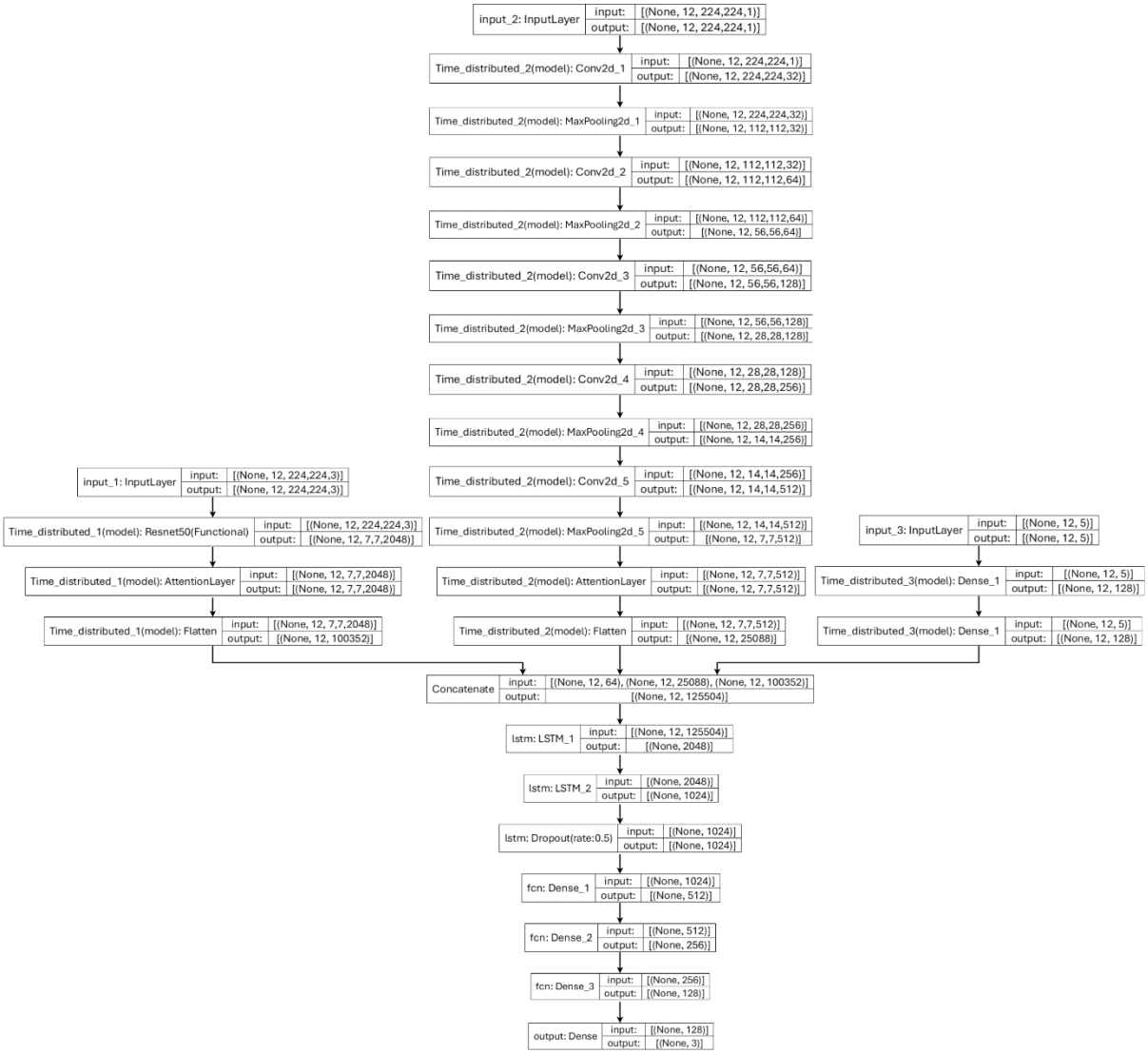


Figure 6-15 Architecture of the proposed driver model

6.4.1 RGB Image Processing with ResNet-50

The RGB image input consists of a sequence of 12 frames, each of size 224×224 pixels with 3 colour channels (RGB). To extract high-level features from these images, a pre-trained ResNet-50 model is used. ResNet-50 is known for its ability to capture intricate spatial hierarchies within images through its deep residual learning framework. Atliha et al. [355] demonstrated that ResNet performed better than other state-of-the-art CNN like the Visual Geometry Group Network (VGGNet) as encoders for image captioning. This is because the "shortcut-connections" in its architecture allows it to capture more complex features from the image. Hence, the output of its second last layer will be a high-quality representation of the input image.

The input to ResNet-50 is the sequence of RGB images, and the output is a feature map representing each frame. ResNet-50 is a deep convolutional neural network with 50 layers. The key innovation of ResNet is the introduction of residual learning through skip connections,

which allows for the training of very deep networks by addressing the vanishing gradient problem. A residual block in ResNet-50 is defined as follows:

$$y = F(x, \{W_i\}) + x \tag{6.4}$$

where x is the input to the block, $F(x, \{W_i\})$ represents the residual mapping to be learned, and y is the output. The residual mapping $F()$ is typically composed of two or three convolutional layers, and the identity connection x is added to the output of $F()$.

The architecture of ResNet-50 consists of the following components:

1. **Conv1**: 7 × 7 convolution, 64 filters, stride 2
2. **MaxPool**: 3 × 3 max pooling, stride 2
3. **Conv2_x**: 3 residual blocks, each with 3 × 3 convolutions
4. **Conv3_x**: 4 residual blocks, each with 3 × 3 convolutions
5. **Conv4_x**: 6 residual blocks, each with 3 × 3 convolutions
6. **Conv5_x**: 3 residual blocks, each with 3 × 3 convolutions
7. **Average Pooling**: Global average pooling
8. **FC**: Fully connected layer

Notably, the fully connected layer of ResNet-50 is removed, so that the feature maps can be further processing directly. The feature extraction process for each frame I_i in the RGB image sequence is represented as:

$$F_{RGB,i} = ResNet50(I_{RGB,i}) \tag{6.5}$$

where $I_{RGB,i}$ is the i-th frame in the RGB image sequence, and $F_{RGB,i}$ is the extracted feature map for the i-th frame.

In order to enhance the feature representation compared to the existing CNN-LSTM models, an attention mechanism is applied to the features extracted by ResNet-50. Traditional models treat all features equally, potentially missing critical information. The attention mechanism helps the model focus on the most relevant parts of the image, improving its ability to make accurate predictions. The attention mechanism computes the attention weights α_i for each frame's feature map $F_{RGB,i}$:

$$e_i = \tanh(W_{attn} F_{RGB,i} + b_{attn}) \tag{6.6}$$

$$\alpha_i = \frac{\exp(e_i)}{\sum_{j=1}^{12} \exp(e_j)} \tag{6.7}$$

where W_{attn} is the weight matrix for the attention layer, b_{attn} is the bias vector, and e_i is the attention score for the i-th frame.

The attended feature map $F_{RGB,Att,i}$ is then computed as:

$$F_{RGB,Att,i} = \alpha_i \odot F_{RGB,i} \tag{6.8}$$

where $\odot$ denotes element-wise multiplication.

6.4.2 Depth Image Processing with Custom CNN

The depth image input also consists of a sequence of 12 frames, each of size 224×224 pixels but with a single channel. A custom convolutional neural network (CNN) is designed to process these depth images. The custom CNN includes several convolutional layers followed by activation functions and pooling layers to capture important spatial features. The architecture of the custom CNN for depth image processing is as follows:

1. **Conv1**: 3×3 convolution, 32 filters, stride 1, ReLU activation
2. **MaxPool1**: 3×3 max pooling, stride 2
3. **Conv2**: 3×3 convolution, 64 filters, stride 1, ReLU activation
4. **MaxPool2**: 3×3 max pooling, stride 2
5. **Conv3**: 3×3 convolution, 128 filters, stride 1, ReLU activation
6. **MaxPool3**: 3×3 max pooling, stride 2
7. **Conv4**: 3×3 convolution, 256 filters, stride 1, ReLU activation
8. **MaxPool4**: 3×3 max pooling, stride 2
9. **Conv5**: 3×3 convolution, 512 filters, stride 1, ReLU activation
10. **MaxPool5**: 3×3 max pooling, stride 2

The feature extraction process for each frame i in the depth image sequence is represented as:

$$F_{Depth,i} = CNN\left(I_{Depth,i}\right) \tag{6.9}$$

where $I_{Depth,i}$ is the i-th frame in the depth image sequence, and $F_{Depth,i}$ is the extracted feature map for the i-th frame.

Similar to the RGB features, an attention mechanism is applied to the depth features to focus on the most important aspects of the depth images. The attention mechanism computes the attention weights β_i for each frame's feature map $F_{Depth,i}$:

$$d_i = \tanh\left(W_{Depth,attn} F_{Depth,i} + b_{Depth,attn}\right) \tag{6.10}$$

$$\beta_i = \frac{\exp(d_i)}{\sum_{j=1}^{12} \exp(d_j)} \tag{6.11}$$

Where $W_{Depth,attn}$ is the weight matrix for the attention layer, $b_{Depth,attn}$ is the bias vector and d_i is the attention score for the i-th frame.

The attended feature map $F_{Depth,Att,i}$ is then computed as:

$$F_{Depth,Att,i} = \beta_i \odot F_{Depth,i} \tag{6.12}$$

6.4.3 Vehicle State Data Processing with Dense Layers

The vehicle state data includes a sequence of 12 instances, each with 5 features: speed, steering angle, gas, brake, and waypoint data. Before preprocessing, the vehicle state data is flattened into a 1D vector:

$$V_{Flattened} = Flatten(V) \tag{6.13}$$

where V is the input vehicle state data matrix of shape 12×5, and $V_{Flattened}$ is the flattened vector of shape 60×1. Dense layers are then employed to process this data, with two dense layers of 128 and 64 units, respectively, each followed by ReLU activation functions. The transformation can be expressed as:

$$H_1 = ReLU(W_1 V_{Flattened} + b_1) \tag{6.14}$$

$$H_2 = ReLU(W_2 H_1 + b_2) \tag{6.15}$$

where W_1 and W_2 are the weight matrices, b_1 and b_2 are the biases, H_1 is the hidden representation after the first dense layer, and H_2 is the output of the second dense layer.

6.4.4 Feature Concatenation

The outputs from the RGB CNN with attention, depth CNN with attention, and dense layers processing vehicle state data are concatenated to form a combined feature vector:

$$F_{Combined} = \left[F_{RGB\,,Att}, F_{Depth,Att}, H_2\right] \tag{6.16}$$

where $F_{Combined}$ represents the concatenated features.

6.4.5 Temporal Dependency Extraction with LSTM

The combined feature vector is then fed into an LSTM network, which is designed to capture temporal dependencies within sequential data. The LSTM network consists of two layers, with 2048 and 1024 units, respectively. The LSTM processes the concatenated features over time, learning patterns and dependencies that are crucial for making accurate driving decisions. The LSTM transformations are given by:

$$h_k = LSTM(F_{Combined}) \tag{6.17}$$

where h_k is the hidden state at time step k. The LSTM equations for the hidden state updates can be represented as:

$$\begin{gathered} i_k = \sigma\left(W_i F_{Combined,k} + W_i h_{k-1} + b_i\right) \\ f_k = \sigma\left(W_f F_{Combined,k} + W_f h_{k-1} + b_f\right) \\ g_k = tanh\left(W_{Ig} F_{Combined,k} + W_c h_{k-1} + b_g\right) \\ c_k = f_k \odot c_{k-1} + i_k \odot g_k \\ o_k = \sigma\left(W_o F_{Combined,k} + W_o h_{k-1} + b_o\right) \\ h_k = o_k \tanh \odot (c_k) \end{gathered} \tag{6.18}$$

where f_k, i_k, g_k, o_k are forget, input, input modulation, and output gates, respectively; c_k is the cell state; σ is the sigmoid non-linearity activation function; and $\odot$ denotes matrix multiplication.

Dropout regularization is employed to enhance the robustness and generalization of the temporal feature learning within the LSTM layers. Dropout is a crucial technique used to prevent overfitting, a common issue where the model performs well on training data but fails to generalize to new, unseen data. In the context of LSTM networks, dropout can be applied both to the input units and the recurrent connections between the LSTM units. For input dropout, a fraction of the input units is randomly set to zero during each forward pass in the training phase. This is mathematically represented as:

$$\tilde{F}_{Combined,k} = dropout\left(F_{Combined,k}, p\right) \tag{6.19}$$

Where $F_{Combined,k}$ is the input at time step k, $\tilde{F}_{Combined,k}$ is the input after applying dropout, and p is the dropout rate indicating the probability of setting an input unit to zero. This process helps to prevent the network from becoming overly reliant on specific input features, thereby reducing the risk of overfitting.

Recurrent dropout, on the other hand, is applied to the connections between the hidden states of the LSTM units. This can be described by the equation:

$$\tilde{h}_{k-1} = dropout(h_{k-1}, p) \tag{6.20}$$

where h_{k-1} is the hidden state from the previous time step, $\tilde{h}_{k-1}$ is the hidden state after applying dropout, and p is the dropout rate for the recurrent connections. By regularizing these connections, recurrent dropout prevents the co-adaptation of recurrent units, encouraging the LSTM to learn more generalized and robust features. In the proposed driver model, the LSTM layer incorporates both input and recurrent dropout to address these challenges. Specifically, the LSTM layer with 256 units employs a dropout rate of 0.2 for both inputs and recurrent connections. This configuration is denoted as:

$$h_k = LSTM\left(F_{Combined}, dropout = 0.2, recurrent_{dropout} = 0.2\right) \tag{6.21}$$

By setting 20% of the input units and 20% of the recurrent connections to zero during training, the model is forced to learn more meaningful patterns and relationships in the data, leading to improved generalization and performance. The incorporation of dropout in the LSTM layers ensures that the proposed driver model can effectively handle the complexity and variability inherent in driving data. This results in a model that is more resilient to overfitting and better equipped to generalize to new driving scenarios, thus providing a significant advantage over models that do not utilize dropout regularization.

6.4.6 Fully Connected Layers for Driving Input Prediction

The final hidden state from the LSTM is passed through fully connected layers to predict the driving inputs. The first fully connected layer has 128 units with ReLU activation,

followed by a second layer with 64 units. The output layer consists of 3 units, corresponding to the steering angle, gas, and brake inputs, with linear activation functions to provide the final driving commands:

$$D_1 = ReLU(W_3 h_k + b_3)$$
$$D_2 = ReLU(W_4 D_1 + b_4) \quad (6.22)$$
$$D_3 = ReLU(W_5 D_2 + b_5)$$
$$[\delta, \rho, \tau] = W_6 D_3 + b_6$$

Where W_3, W_4,W_5 and W_6 are the weight matrices, b_3, b_4, b_5, and b_6 are the biases, D_1, D_2 and D_3 are the outputs of the first, second and third dense layers, respectively, δ is the predicted steering angle, ρ is the predicted gas input, and τ is the predicted brake input.

The novelty of this model configuration lies in its comprehensive integration of multimodal data using advanced neural network techniques. Several advanced features have been incorporated into the proposed driver model to address key issues found in traditional CNN-LSTM driver models, enhancing performance, robustness, and accuracy. A major issue with traditional models is their inability to focus on the most relevant parts of the input data, leading to suboptimal performance in complex environments. This issue is addressed by incorporating attention mechanisms, allowing the model to dynamically prioritize important features from both RGB and depth images. This ensures concentration on critical objects and regions in the driving environment, such as pedestrians, vehicles, and road signs, thereby improving situational awareness and decision-making. For instance, in scenarios with multiple vehicles, the attention mechanism enables the model to focus on vehicles directly in the path of the autonomous vehicle, ignoring irrelevant background details.

The use of the state-of-the-art ResNet-50 for RGB image processing addresses another significant issue: the limited ability of simple CNNs to capture intricate spatial hierarchies within images. ResNet-50, with its deep residual learning framework, effectively captures high-level spatial features, mitigating the vanishing gradient problem commonly encountered in deep networks. This leads to improved feature extraction, allowing accurate identification of traffic lights, signs, and lane markings under various lighting conditions, thus providing robust feature maps for subsequent processing. Similarly, the custom CNN designed for depth image processing addresses the challenge of fully exploiting the spatial information present in depth images, which traditional models often fail to do. The custom CNN captures vital depth information that complements the RGB data, enabling accurate detection of obstacles and determination of their distances. This is particularly important for safe navigation in environments with varying terrain and object proximity.

Vehicle state data has traditionally been processed using simple concatenation or basic feedforward networks, which may not capture the complex relationships within the data. This model uses dense layers to process vehicle state data, allowing for a better understanding and leverage of these relationships. This leads to more accurate predictions of driving inputs. For

example, the correlation between speed and steering angle can be analysed, enabling more informed control decisions, such as adjusting speed during sharp turns to maintain stability. Temporal dependencies in sequential data are captured by employing an LSTM network to process the concatenated features over time, addressing another challenge faced by traditional models. Temporal patterns and dependencies critical for driving tasks are captured by the LSTM. Patterns such as decelerating before a turn and accelerating after completing the turn are learned, mimicking human driving behaviour for smoother control.

Multimodal data from different sources (RGB, depth, and vehicle states) has traditionally been integrated inefficiently. By concatenating the attended features from RGB and depth images with the processed vehicle state data, a comprehensive representation of the driving environment is provided. This holistic integration helps the model make well-informed decisions, such as determining the best path to avoid obstacles while maintaining optimal speed and steering. Finally, the fully connected layer transforms the integrated features into precise driving inputs (steering angle, gas, and brake), addressing the issue of high-dimensional feature spaces leading to overfitting and increased computational complexity. The fully connected layer balances complexity and performance, ensuring accurate prediction of necessary control actions based on a comprehensive analysis of the input data, thereby ensuring smooth and safe vehicle operation.

In summary, critical issues found in traditional driver models are effectively addressed by the proposed CNN-LSTM model with attention mechanisms, ResNet-50, custom CNN, and dense layers. By focusing on relevant features, capturing high-level spatial and temporal dependencies, and processing vehicle state data efficiently, the model enhances performance, robustness, and accuracy, ensuring reliable operation in diverse and complex driving environments. This holistic approach distinguishes the proposed model from existing CNN-LSTM models, offering a more sophisticated and accurate solution for autonomous driving tasks.

6.4.7 Integration with a Validated 14 DoF Mathematical Vehicle Model

To enhance the training and testing processes of the final driver model, the validated 14 DoF mathematical vehicle model has been integrated. This step addresses significant challenges associated with using the vehicle model from the IPG CarMaker. The dataset utilized for training the driver model has been recorded from a virtual driving simulator, which provides detailed and realistic driving scenarios. However, testing the driver model necessitates generating vehicle state outputs, such as vehicle speed, longitudinal and lateral displacement, based on the predicted steering angle, gas, and brake outputs from the driver model. The vehicle model in the IPG CarMaker, although accurate, requires substantial time and hardware resources to run, making it impractical for extensive testing and iterative development.

To overcome these limitations, the validated 14 DoF mathematical vehicle model developed in the Chapter 3 has been employed as a replacement for the virtual simulator vehicle model. The integration process involves several key steps. Firstly, input data synchronization

ensures that the driver model, which processes RGB images, depth images, and vehicle state data, receives synchronized input data corresponding to the outputs and states of the 14 DoF vehicle model. This synchronization guarantees that the training and testing data reflect realistic driving scenarios. Secondly, the 14 DoF vehicle model generates realistic sensor data, such as vehicle speed, longitudinal and lateral displacement. This simulated data is fed into the driver model during training and testing, mimicking the data that would be collected from actual sensors on a real vehicle.

A feedback loop is established where the driver model's outputs (steering angle, gas, brake) are used to control the 14 DoF vehicle model. This loop ensures that the driver model learns to control the vehicle dynamics accurately in response to various driving scenarios. The integrated system undergoes extensive validation to ensure that the driver model performs well under different conditions. This includes testing the model in various simulated environments, such as urban, highway, and off-road scenarios, and refining it based on performance metrics. Figure 6-16 shows the training process of the proposed driver model. The integration of the driver model with the 14 DoF vehicle model offers several significant advantages. By using a detailed vehicle dynamics model, the driver model is tested in a highly realistic simulation environment, improving its performance in real-world conditions.

The 14 DoF mathematical vehicle model consumes fewer resources and less time to run compared to the virtual simulator vehicle model, making it more practical for extensive testing and iterative development. The integrated system allows for thorough validation of the driver model, ensuring it can handle a wide range of driving scenarios and conditions. The complex interactions and dynamics captured by the 14 DoF model help the driver model learn more sophisticated control strategies, leading to improved driving behaviour and safety. The feedback loop between the driver model and the vehicle dynamics model enables continuous refinement and improvement of the driver model based on simulated performance.

By integrating the final driver model with a validated 14 DoF mathematical vehicle model, the training and testing processes are significantly enhanced. The detailed vehicle dynamics model provides a realistic and resource-efficient simulation environment, ensuring the driver model is well-equipped to handle real-world driving scenarios, resulting in a robust and accurate autonomous driving system. This integrated approach addresses the limitations of traditional training methods and ensures that the driver model is capable of safe and efficient vehicle control in diverse and complex environments.

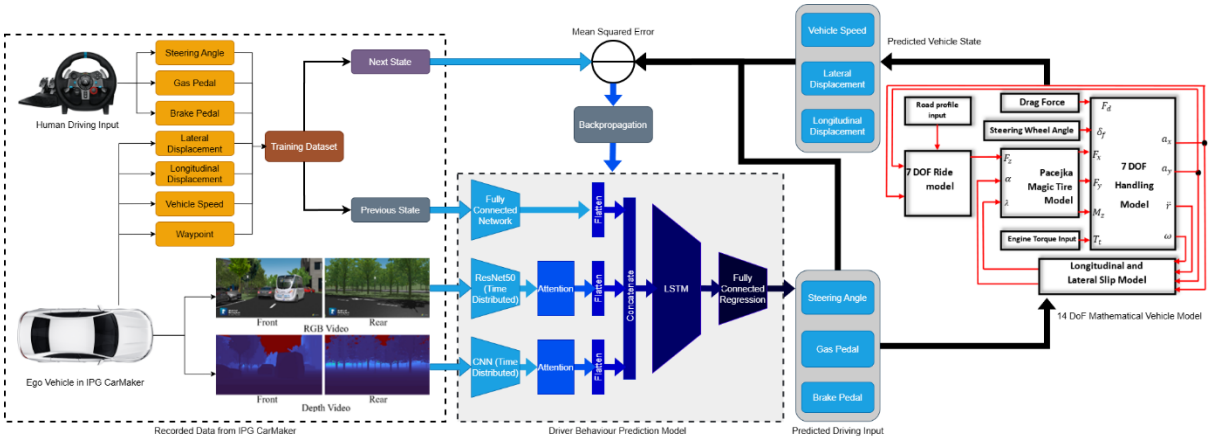


Figure 6-16 Training process of the proposed driver model

6.4.8 Comparison of Driver Model Performance with Baseline Models

To comprehensively evaluate the performance of the proposed driver model, two baseline models were developed for comparison: an CNN-only model and a traditional CNN-LSTM model. This comparative analysis aims to highlight the effectiveness and advantages of the novel driver model configuration.

(a) Baseline Model 1: CNN-Only Model

The CNN-only model is based on NVIDIA's PilotNet architecture, which is designed to process image data and directly predict the driving inputs. This model serves as a simpler baseline to assess the benefits of incorporating temporal modelling and attention mechanisms in the proposed model. Figure 6-17 shows the architecture of the baseline model 1. The architecture of the CNN-only model is as follows:

- **Input Layer**: Receives RGB images with dimensions (224, 224, 3).
- **Convolutional Layers**: Multiple convolutional layers are used to extract spatial features from the RGB images. The layers include convolution, ReLU activation, and pooling operations. This CNN module employs a structure akin to ResNet. Following the CNN layer, the output is flattened into flat data, leading to the last two layers, which are the dense layers and the output layer.
    - **Conv1**: 24 filters, kernel size 5×5, strides 2×2, ReLU activation.
    - **Conv2**: 36 filters, kernel size 5x5, strides 2x2, ReLU activation.
    - **Conv3**: 48 filters, kernel size 5x5, strides 2x2, ReLU activation.
    - **Conv4**: 64 filters, kernel size 3x3, strides 1x1, ReLU activation.
    - **Conv5**: 64 filters, kernel size 3x3, strides 1x1, ReLU activation.
- **Flatten Layer**: Flatten the 2D data Conv5 output into 1D.
- **Dense Layers**: Fully connected layers are used to map the extracted features to the driving inputs.

- **Dense1**: 100 units, ReLU activation.
- **Dense2**: 50 units, ReLU activation.
- **Dense3**: 10 units, ReLU activation.

- **Output Layer**: Predicts the steering angle, gas, and brake values.

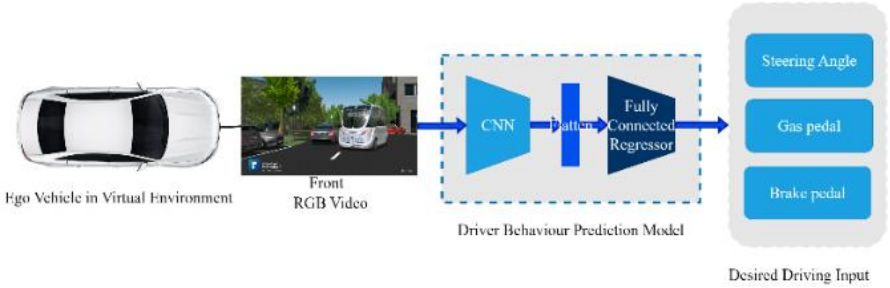


Figure 6-17 Architecture of the end-to-end baseline model 1

(b) Baseline Model 2: Traditional CNN-LSTM Model

The traditional CNN-LSTM model incorporates convolutional layers to process image data and LSTM layers to handle the sequential nature of the data. This model serves as a benchmark to demonstrate the improvements gained by introducing the novel components in the proposed model. Figure 6-18 shows the architecture of the baseline model 2. The architecture of the traditional CNN-LSTM model is as follows:

- **Input Layers**: There are two input layers. One of the input layers receives RGB images, while the other input layer receives vehicle state data such as vehicle speed, the driver's applied steering wheel angle, and the intensity of throttle and brake pedal usage.

- **CNN Layers**: Processes the RGB images to extract spatial features. The CNN typically includes convolutional, ReLU, and pooling layers.
  - **Conv1**: 24 filters, kernel size 5x5, strides 2x2, ReLU activation.
  - **Conv2**: 36 filters, kernel size 5x5, strides 2x2, ReLU activation.
  - **Conv3**: 48 filters, kernel size 5x5, strides 2x2, ReLU activation.
  - **Conv4**: 64 filters, kernel size 3x3, strides 1x1, ReLU activation.
  - **Conv5**: 64 filters, kernel size 3x3, strides 1x1, ReLU activation.
- **Flatten Layer**: Flatten the 2D data Conv5 output into 1D

- **LSTM Layers**: Captures temporal dependencies from the sequential data, integrating features from both the CNN and the vehicle state inputs.
  - **LSTM1**: 256 units. Drop out 0.5.

- **Dense Layer**: Maps the combined features to the driving inputs.
  - **Dense1**: 100 units, ReLU activation.
  - **Dense2**: 50 units, ReLU activation.
  - **Dense3**: 10 units, ReLU activation.

- **Output Layer**: Predicts the steering angle, gas, and brake values.

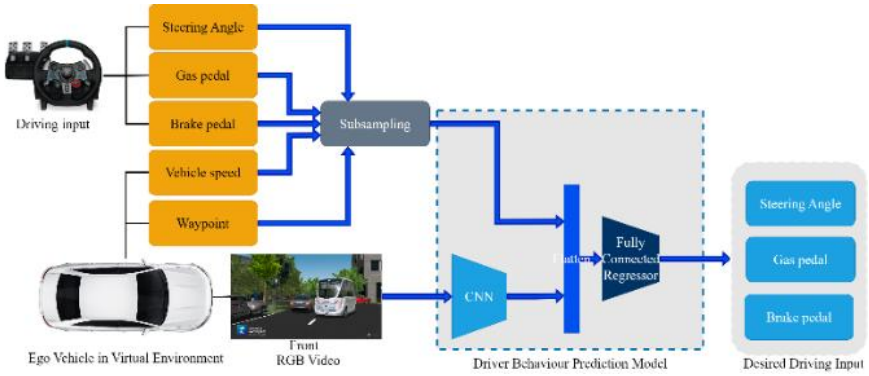


Figure 6-18 Architecture of the end-to-end baseline model 2

6.5 Experiments and Results using Simulated Data

Each model undergoes evaluation using identical test data sampled from IPG CarMaker. Mean Squared Error (MSE), presented in Equation 6.23, serves as the primary metric for assessing the prediction accuracy of the trained model. MSE is computed as the average of squared errors across all video clips in the test dataset. MSE is advantageous because it effectively captures differences between predicted and true values, which aids in comparing model performance. However, MSE may diminish these differences, particularly for small driving inputs typical of steady AV movement on congested roads where absolute values generally remain below 0.5.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}|p_i - g_i| \tag{6.23}$$

Where p_i represents the predicted outcome, and the g_i denotes the ground truth.

Since the focus of this study is to minimize differences between model output and ground truth, not necessarily to achieve exact replication. Therefore, a small error threshold ts is established to evaluate driving manoeuvres (DM). Predicted outcomes with MSE below ts indicate valid driving manoeuvres (VDM), while those above ts denote false driving manoeuvres (FDM). This distinction between VDM and FDM is quantified in Equation 6.24.

$$\begin{cases} |p_i - g_i| \geq ts, DM = VDM \\ |p_i - g_i| < ts, DM = FDM \end{cases} \tag{6.24}$$

Then, the system performance score is defined as $PESC$ in Equation 6.25 to evaluate the system:

$$PESC = \frac{VDM}{VDM + FDM} \tag{6.25}$$

As mentioned earlier, to make sure the driving behaviour is normal driving, it is crucial to make sure the error is within the 4 m/s for the speed. Hence the threshold is set as 4 degrees for steering angle. Given that control decisions are made every half second, positional errors in the vehicle's direction are 0.4m horizontally and 0.06m vertically. Such minor deviations are typically accommodated within safety margins designed into roads and traffic systems, thereby minimizing risk of accidents or damage [356].

6.5.1 Evaluation of Sequential Data Input Interval

In order to ensure the driver model retains essential information, the optimal interval between input data is determined. This interval aims to prevent an excess of redundant data, which would otherwise slow down the inference speed of the driver model. In this experiment, the interval x is varied while keeping other parameters constant. Given a data sampling rate of 30, each vehicle control decision corresponds to $x/30$ seconds. Values of x are tested from $\{2,4,6,8,10,12,14,16,18,20,22,24,26,28\}$, and the x that yields the best performance across steering angle and speed MSE metrics is identified. Figure 6-19 illustrates the MSE results for different x values, clearly showing that the interval between input data significantly impacts system performance. From the results, $x = 12$ emerges as optimal for both steering angle and speed, indicating that decisions made with input data every 0.4 seconds achieve the best overall performance.

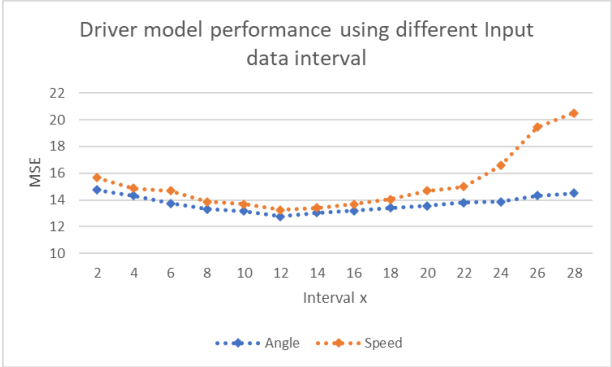


Figure 6-19 Driver model performance using different parameter x.

6.5.2 Driver Model Performance Results using Simulated Data

From Table 6-3, Baseline Model 1 (CNN-only) and Baseline Model 2 (CNN-LSTM) serve as benchmarks for evaluating the effectiveness of different driver model architectures. Baseline Model 1 (CNN-only) utilizes a convolutional neural network (CNN) architecture, which extracts spatial features from images but lacks temporal awareness. This model performs significantly worse than models incorporating long short-term memory (LSTM) networks, as it is unable to capture the sequential dependencies essential for driving behaviour prediction. The mean squared error (MSE) for steering and speed in Baseline Model 1 is the highest, indicating that relying solely on CNN-based spatial feature extraction is insufficient for modeling complex driver behaviors that depend on historical driving states.

Baseline Model 2 (CNN-LSTM) introduces LSTM layers to handle temporal dependencies in driving sequences, leading to substantial performance improvements. The CNN extracts spatial features from camera images, while the LSTM captures sequential information from past frames, enabling better driving input prediction. This is reflected in the lower MSE values and a higher PESC score compared to Baseline Model 1. However, despite the improvement, Baseline Model 2 does not integrate additional attention mechanisms or leverage multi-camera information effectively, limiting its ability to adapt to complex, real-world driving scenarios. Furthermore, it was observed that the CNN-LSTM with attention

mechanism model, as proposed in [223] , did not achieve satisfactory performance even after being retrained using the Malaysian driving dataset developed. While the model demonstrated improved speed prediction accuracy, with a lower mean squared error (MSE) compared to Baseline Model 2, its performance in steering angle prediction was significantly weaker.

The primary reason for this limitation is that the model lacks a navigation mechanism, which is crucial for decision-making at intersections and junctions. Without waypoints to guide its trajectory, the model struggles to determine the correct path, often selecting incorrect exits or failing to make appropriate turns, leading to higher steering prediction errors. To address this issue, the driver model proposed in this study incorporates a waypoint-based navigation system, ensuring that the model follows a predefined route toward the intended destination. By integrating waypoints as part of the input features, the model is able to improve trajectory planning and steering accuracy, resulting in more reliable and realistic driving behaviour in simulation and real-world testing.

The proposed model using 5 features, the front and rear camera images have outperformed most of the other models. This demonstrates that adding information about the current driving input to the input layer significantly improved the model's performance compared to baseline models and the model using only front camera image information. However, the model using all cameras performed similarly to the model using only the front and rear cameras, contrary to expectations. Several factors contribute to this unexpected result. Firstly, the additional side camera images introduced redundant or irrelevant information, increasing noise in the input data and making it harder for the model to extract meaningful features. Secondly, the spatial misalignments or inconsistencies between the side, front, and rear camera images confused the model, as it had to reconcile different perspectives and orientations.

In contrast, the significant performance improvement of the model with front and rear camera images over the model with only a front camera is attributed to the complementary nature of the information provided by these two cameras. The front camera captures the immediate driving environment, while the rear camera provides additional context, such as vehicles behind the driver. By combining both views, the model gains a more comprehensive understanding of the surrounding traffic, enabling better driving decisions. Overall, the failure of the model with all four camera images to outperform the model with only front and rear camera images indicate that using all four camera images (front, rear, left, and right) does not necessarily improve performance over the model using only front and rear cameras. This outcome underscores the importance of careful feature selection, as adding more input data does not always lead to better performance due to:

- Introduction of Redundant or Irrelevant Information: When incorporating left and right camera images, the model encounters additional visual data that may not always be relevant for decision-making. Unlike the front and rear cameras, which provide direct information about immediate and trailing traffic conditions, the side cameras capture peripheral views that may not always contribute critical information for tasks such as steering and speed control. The presence of irrelevant features can dilute the model’s

ability to focus on key decision-making elements, leading to higher prediction error and degraded overall performance.

- Spatial Misalignment and Perspective Variability: The integration of images from multiple camera angles introduces orientation mismatches that the model must reconcile. The front and rear cameras have a clear and linear relationship, making it easier for the network to learn the temporal transitions between observed traffic and driving inputs. However, the side camera views are angled differently, capturing information from a different perspective, making it harder for the model to extract consistent patterns. This spatial inconsistency confuses the model, reducing its ability to generalize and make accurate predictions.
- Noise Amplification and Increased Computational Complexity: Including all four cameras increases the dimensionality of the input space, leading to greater computational demands and potential overfitting. The model must process a larger feature set, which not only increases training time but also amplifies noise if certain camera views contribute conflicting or less relevant information. Instead of improving performance, the additional side views create ambiguity in learned representations, leading to higher error rates.

Therefore, the result demonstrated that feature selection must be guided by relevance and data efficiency. While adding complementary sources of information can improve performance, including excessive or less useful features can introduce noise, redundancy, and model confusion. The best-performing configuration which used only front and rear cameras, demonstrated that combining a primary viewpoint (front) with a supplementary, context-providing viewpoint (rear) enhances predictive accuracy without unnecessary complexity. This insight is critical for future work in autonomous driving and driver modelling, reinforcing the need for careful sensor fusion strategies to ensure optimal performance without unnecessary computational overhead.

Table 6-3 Performance of different driver model architecture with or without LSTM

Models	MSE Steering	MSE Speed	PESC
CNN Only baseline model 1	0.4014	0.4312	33.17
CNN-LSTM baseline model 2	0.2985	0.2802	64.25
CNN-LSTM with attention mechanism proposed in [223]	0.3296	0.2619	61.04
Proposed (front camera)	0.1781	0.1863	75.48
Proposed (front + rear cameras)	0.1278	0.1327	84.63
Proposed (front + rear + right + left cameras)	0.1379	0.1308	83.59

A comparison of different CNN architectures was also carried out through experiments. Specifically, three different CNN-LSTM combination architectures were implemented: NVIDIA PilotNet, VGGNet, and ResNet-50 CNN architectures. Furthermore, since the introduction of Transformers for natural language processing, Transformers have become increasingly popular not only in language processing but also in other field like Vision processing. Two different Transformer models were implemented and subjected to comparison in this experiment as well: Vision in Transformer (ViT) [357] and Compact Convolutional

Transformer (CCT) [358]. Table 6-4 shows the results, where MSE for the lateral and longitudinal controls for the five architectures are listed.

Table 6-4 Performance of different image feature extractor network.

Image Feature Extractor	MSE Steering	MSE Speed	PESC	#Params	MACs
NVIDIA	0.1828	0.2089	67.73	348.82 k	0.82 G
VGGNet	0.1356	0.1468	78.91	134.7 M	15.61 G
ResNet-50	0.1278	0.1327	84.63	25.56 M	4.14 G
Vision in Transformer (ViT-Base/16)	0.0927	0.0989	88.27	86.8 M	49.35 G
Compact Convolutional Transformer (CCT)	0.1363	0.1269	84.26	22.51 M	15.02 G

It can be noted that the ViT-LSTM architecture have the best performance among all the five architectures. ViT, based on the Transformer architecture originally designed for natural language processing task, employs self-attention mechanisms to model dependencies between words in text. Similarly, in ViT, input images were represented as a series of patches which act like a series of word embeddings used when using Transformer for text. However, training and deploying ViT required vast computational resources due to high computational complexity compared to those older and well-established CNN models. To counter this issue, instead of patching, CCT uses convolutions to perform sequence pooling which have been proved to be efficient in CNN networks for extracting and encoding features in images. This enable CCT to increase performance while reducing the size of the model due to smaller number of parameters compared to ViT. Despite having a smaller number of parameters, the performance of the CCT is on par with or slightly better than the ResNet-50 in some cases. However, due to the architecture of the Transformer, the Multiply-Accumulate Operations (MACs) of the CCT are more than three times those of ResNet-50, requiring significantly more hardware resources to deploy the model for real-time detection.

6.5.3 Visualization of the Driver Model Result

Upon completion of the training phase, the driver model is integrated into the IPG CarMaker to evaluate its performance in a realistic and controlled driving environment. This integration is crucial for ensuring that the driver model is reliable and robust enough for safety testing of autonomous vehicles, which require a highly accurate and responsive driver model for validation and verification purposes. Within this integration, the driver model receives real-time input data, including RGB and depth images from both front and rear cameras, along with the current vehicle states data provided by IPG CarMaker. These inputs encompass crucial information about the vehicle's speed, steering angle, gas throttle, brake status, and waypoints. The driver model processes the RGB and depth images through its pre-trained ResNet-50 and custom CNNs, respectively. Simultaneously, it analyses the vehicle states data using dense layers. The integration of attention mechanisms ensures that the model focuses on the most relevant features from the visual and state inputs. After extracting and combining these features, the LSTM network captures temporal dependencies to predict the driving control outputs.

The final outputs from the driver model are the steering angle, brake, and gas controls, which are used to manoeuvre the virtual vehicle within the IPG CarMaker simulator. This setup

allows for a comprehensive assessment of the model's capability to handle dynamic driving scenarios, respond to real-time environmental changes, and make precise driving decisions. The use of IPG CarMaker provides a robust platform to simulate various driving conditions and validate the model's effectiveness in an environment that closely mimics real-world driving. By integrating the driver model into the IPG CarMaker simulator, it is possible to perform a direct comparison with the built-in driver model provided by IPG CarMaker. This comparison involves evaluating the performance of both models under identical driving scenarios, enabling an assessment of the novel driver model's advantages in terms of accuracy, responsiveness, and robustness. Key performance metrics such as steering precision, brake and gas control, and overall driving behaviour can be measured and analysed. The integration and comparison process ensures that the developed driver model is rigorously tested and benchmarked against established standards. Figure 6-20 illustrates the driver model integrated into the IPG CarMaker, replacing the built-in IPGDriver to control the ego vehicle.

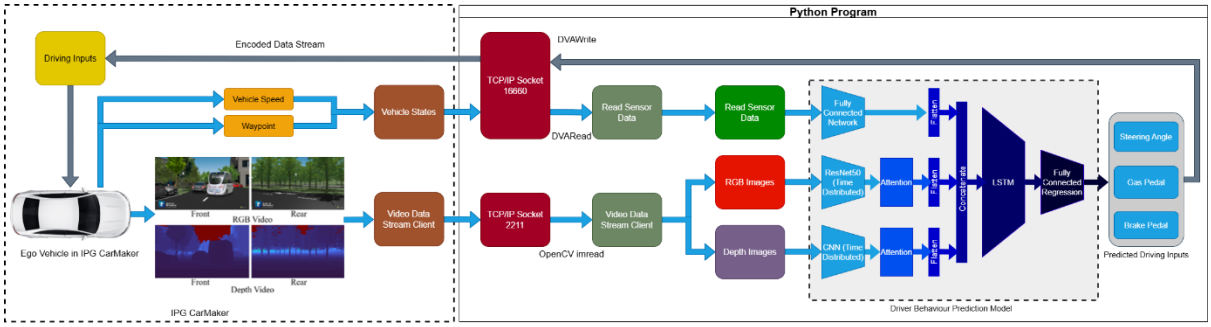


Figure 6-20 Integration of driver model into IPG CarMaker

To further ensure the reliability of the driver model for safety testing of autonomous vehicles, the driver model's performance is visually assessed by feeding predicted steering angles and pedal inputs into IPG CarMaker to control the virtual vehicle and generating videos as output. The visualized results were analysed across several common driving scenarios:

- **The "vehicle-following" scenario**: In this scenario, the ego vehicle maintained a safe distance from the leading car and travelled at a steady speed. This evaluation highlighted the model's capability to accurately replicate acceleration trends in response to speed changes of the preceding vehicle. Among all tested scenarios, "vehicle-following" showed the highest level of accuracy, with the prediction curve closely aligning with the ground-truth curve. This indicated the model's effectiveness in replicating real-world driving behaviours in this common driving situation.

- **The "emergency braking" scenario**: During the deceleration phase, the driver model detected the motorcycle cutting in and responded by releasing the throttle and applying the brakes, mirroring the human driver's actions to maintain a safe distance. Once the motorcycle left the lane, the ego vehicle accelerated. Both the ground truth and predicted acceleration curves rise, though the predicted curve peaked slightly lower than the ground truth.

- **The "pedestrian crossing scenario"**: When the pedestrian crossed the road, the driver model needed to apply the brakes to halt the car. Once the pedestrian had left the road, the ego vehicle accelerated and resumed its planned path.
- **The "overtaking" scenario**: The driver model needed to overtake traffic vehicle ahead by changing lane. The driver model needed to ensure no vehicles were coming from the right lane to safely perform overtaking.

In the data plots, the vertical axis represents the values of driving inputs, whereas the horizontal axis indicates the timestamp. The vertical axes are scaled to [−1.0,1.0] for steering input and [0.0,1.0] for throttle and brake inputs to ensure accurate interpretation. For horizontal axes, due to the extensive length of the recorded data, only key timestamps relevant to specific driving events are extracted and presented in this thesis. This selective presentation ensures that critical moments of interest are highlighted while reducing redundancy. Values of the driving inputs for both ground-truth and prediction were displayed on the generated video, identified as "ground truth" and "prediction" in the legends, respectively. By combining the data plots with actual video frames, it provides an intuitive method to examine prediction deviations across continuous spatial and temporal ranges. Figure 6-21 illustrates sample frames extracted from the simulation video, showcasing the steering angle and speed of the ground truth, the driver model, and the default IPGDriver model used in IPG CarMaker.

Looking at the example frames of a "pedestrian crossing" scenario as shown in Figure 6-21, it can be observed that the driver model able to produce steering angle and vehicle speed close to the ground truth human driver. As comparison, the IPGDriver provided by IPG CarMaker applied brake and bring the car to idle at 0km/h speed once pedestrians are crossing the road. Whereas the driver model could reproduce the driving style of human drivers where a human driver will slow down the vehicle instead of applying large braking force to stop the vehicle immediately, like the IPGDriver. The steering and braking profiles of the human driver and the driver model are illustrated in Figure 6-24. Small peaks and troughs in both the prediction and ground truth lines indicate slight variations in steering behaviour. The close alignment between the two profiles suggests that the driver model accurately mimics human steering actions during the pedestrian crossing scenario. Similarly, the prediction and ground truth lines exhibit fluctuations in the braking profile. The model's ability to replicate human braking patterns is evident from the consistent alignment between the two lines.

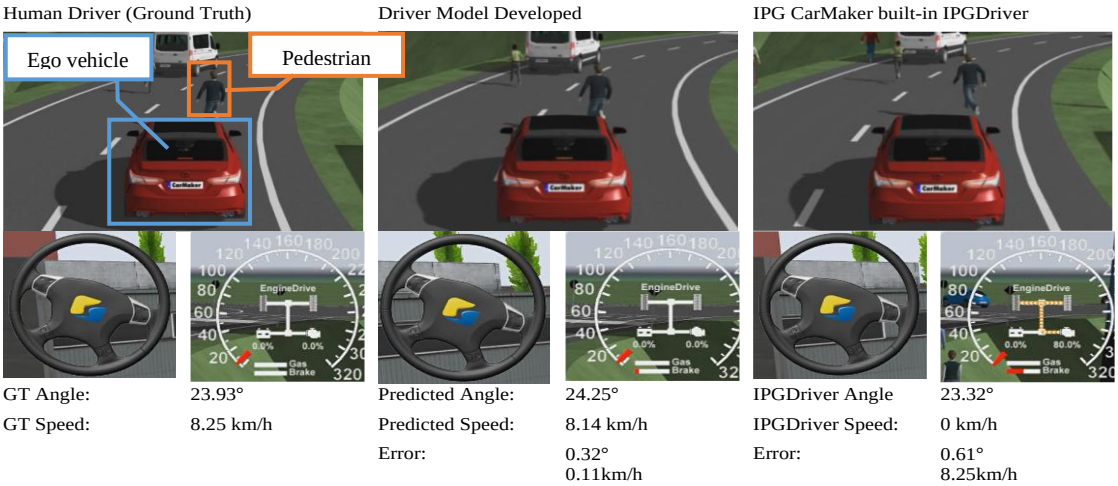
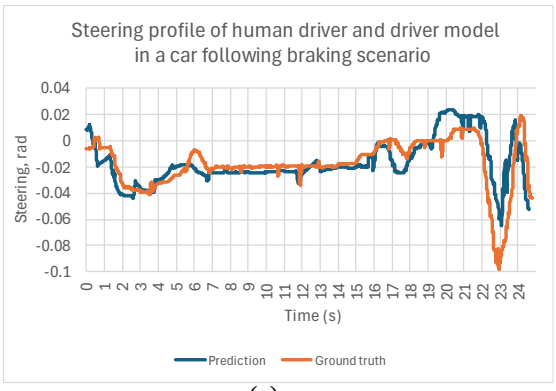
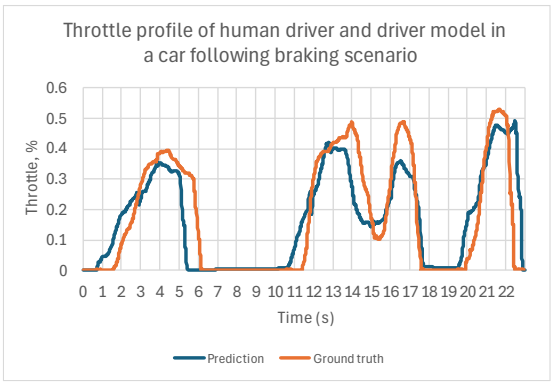


Figure 6-21 Example frames with the driving inputs from the ground truth, prediction of the driver model and generated by IPGDriver.

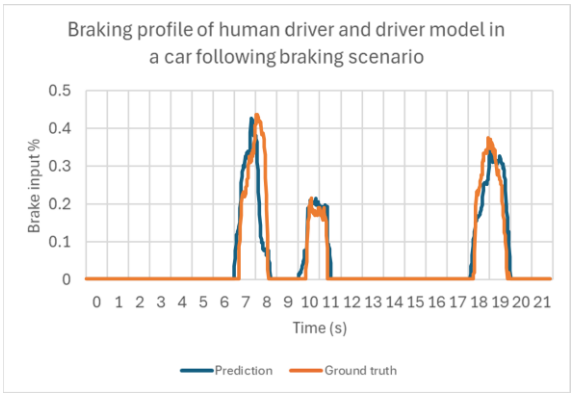
The steer angle, throttle and brake control predicted by the driver model can be seen in Figure 6-22, Figure 6-23, Figure 6-24 and Figure 6-25. As previously mentioned, the predicted curves in "car-following" scenario as shown in Figure 6-22 have least error when compared to the ground truth. This is because in vehicle following scenarios, the driving data mostly depends on the speed of the vehicle in front and the road shape, there were no large variation between the training data of different drivers. Therefore, the driver model can provide a good prediction of the maneuverer in this scenario. Whereas, in critical scenarios such as "pedestrian crossing", "emergency braking" and "overtaking", this is where the driving style of a human driver will affect a lot in the driving data. Since every driver has different driving style, it is difficult to make an average driver prediction model to predict driving input that can fit every single driver without personalization of the driver model tailored for a single driver. This can be seen clearly from the braking profile in Figure 6-23 "emergency braking" scenario where there are some obvious differences between the predicted curve and the ground truth curve. However, the throttle profile in "emergency-braking" scenario shown that the model able to produce throttle control closely match the ground truth. Similarly, this behaviour is evident in the steering and speed control profile during the "overtaking" scenario depicted in Figure 6-25, where the predicted curve closely mirrors the overall trend of the ground truth curve, resulting in minimal prediction error. Thus, it is observed that the model exhibits driving behaviour closely resembling that of average Malaysian drivers, despite with slightly more conservative predictions in acceleration and deceleration. Other than the difference in shape of the curve, there is also noticeable difference between the response time in a human driver and the driver model. In most of the time, the driver model reacts to the scenario much faster than human driver. Hence, although the driver model can predict the driving input at high accuracy, it still lacks the ability to accurately reproduce the response time of human driver with the current architecture proposed.



(a)

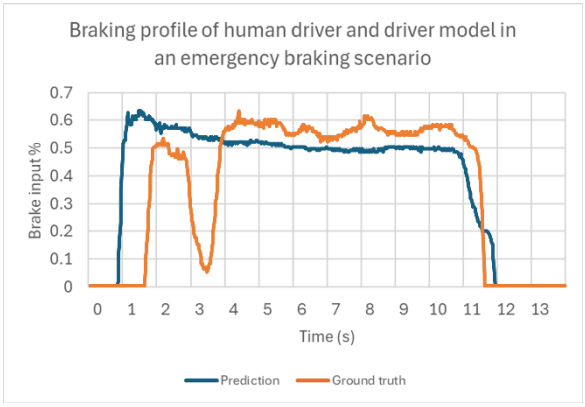


(b)

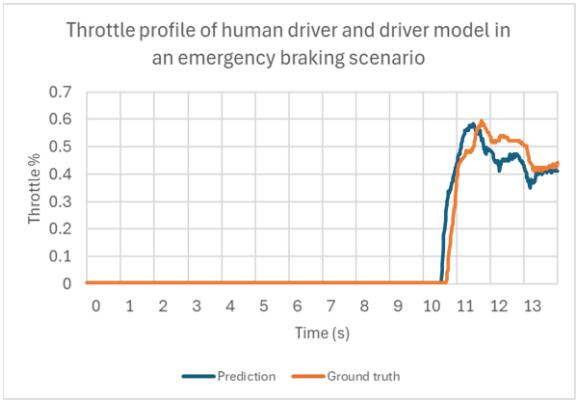


(c)

Figure 6-22 Driving profile of human driver and driver model developed in a car following scenario.

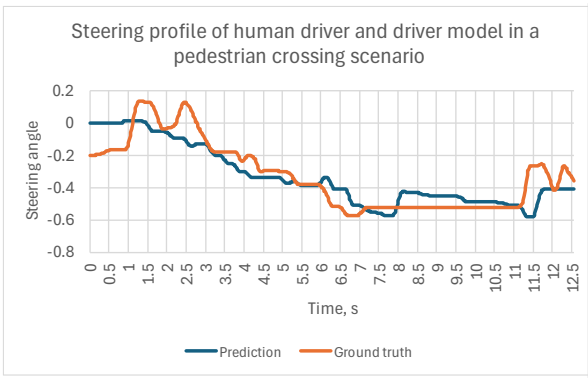


(a)

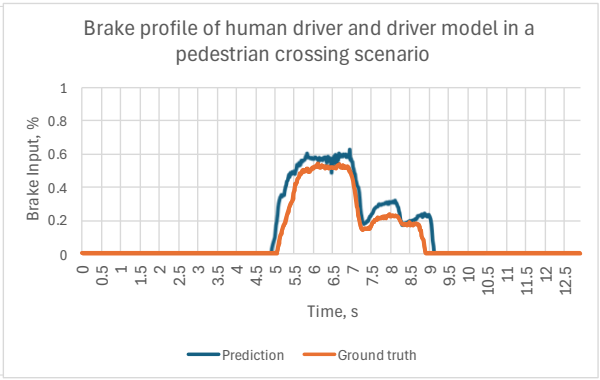


(b)

Figure 6-23 Driving profile of human driver and driver model in an emergency braking scenario.

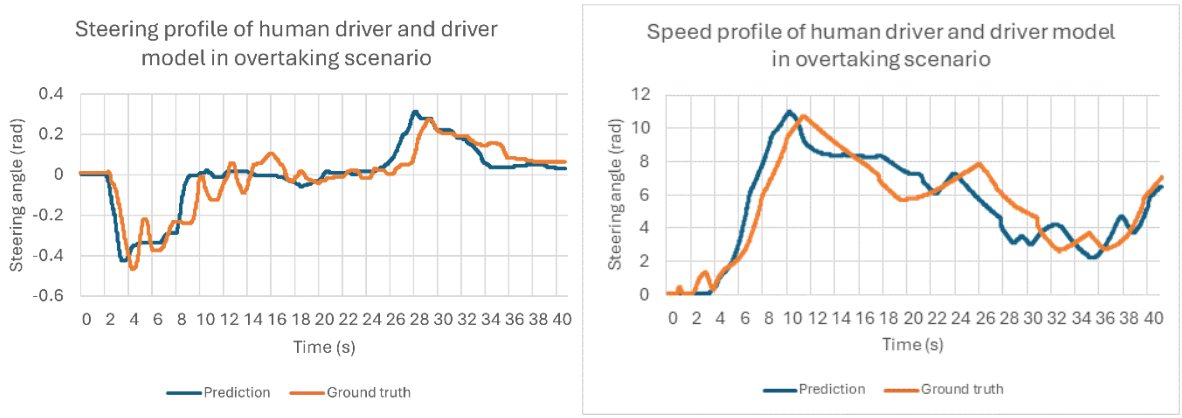


(a)



(b)

Figure 6-24 Driving profile of human driver and driver model developed in a pedestrian crossing scenario



(a) (b)

Figure 6-25Driving profile of human driver and driver model developed in an overtaking scenario.

6.6 Experiments and Results using Real World Data

Synthesized data is a valuable resource for training and validating the deep neural networks especially for tasks that require high fidelity ground truth labels and rare cases. However, synthesized data may not accurately capture the characteristics and variations of real-world data, such as sensor noise, lens artifacts, lightning conditions. Therefore, after the performance of the driver model is evaluated using synthesized data, the driver model is evaluated using real world data captured with the instrumented vehicle as well. However, the raw data captured from the instrumented vehicle cannot be fed directly into the driver model since it lacks several input parameters such as vehicle speed and waypoint to guide the driver model. Therefore, the preprocessing of the real-world raw data is crucial to convert the data to the format that is compatible with the driver model. The following section presents the data conversion methodology and then follow by the evaluation of the driver model performance using the real-world data.

6.6.1 Real-world Data Conversion

The driver model developed required waypoint and vehicle speed as input to generate the steering angle, throttle, and braking driving inputs. The raw data obtained from the sensors mounted on the instrumented vehicle did not have such parameters. To obtain the waypoint in terms of Cartesian (x,y,z) like the IPG CarMaker from the real data, the GPS data recorded is used. Assuming the GPS has no positional error, the XYZ Cartesian coordinates can be obtained as shown in Equation 6.26.

$$X = R \times \cos(lat) \times \cos(lon)$$

$$Y = R \times \cos(lat) \times \sin(lon) \tag{6.26}$$

$$Z = R \times sin(lat)$$

$$R = altitude + R_{earth} \tag{6.27}$$

Where, *lat* and *lon* are the lateral and longitudinal coordinates obtained from the GPS. R is defined as in Equation 6.27. Meanwhile, altitude is the height above sea level obtained from GPS. R_{earth} is the radius of the earth which is approximately 6371km. Next, to obtain the

vehicle speeds, computer vision calculation for optical flow is used to estimate the speed of the vehicle from the front facing camera data. Optical flow is the vector for each pixel that defines the relative motion between two sequential images. First, to obtain the optical flow frames from the recorded video using the instrumented vehicle, state-of-the-art deep learning model Recurrent All-Pairs Field Transforms for Optical Flow (RAFT) is used. After obtaining the optical frames from the video, the speed of the instrumented vehicle can be estimated using a pre-trained CNN network, Video Odometry using Optical Flow (VOOF) model which the architecture is based on Efficient Net. The system architecture to estimate the instrumented vehicle speed is as shown in Figure 6-26. Based on the system architecture estimation, the result of the speed estimation model is shown in Figure 6-27. With these pre-processed data, the dataset of the real-world data can be fed into the developed driver model to evaluate its performance and robustness while handling different types of data.

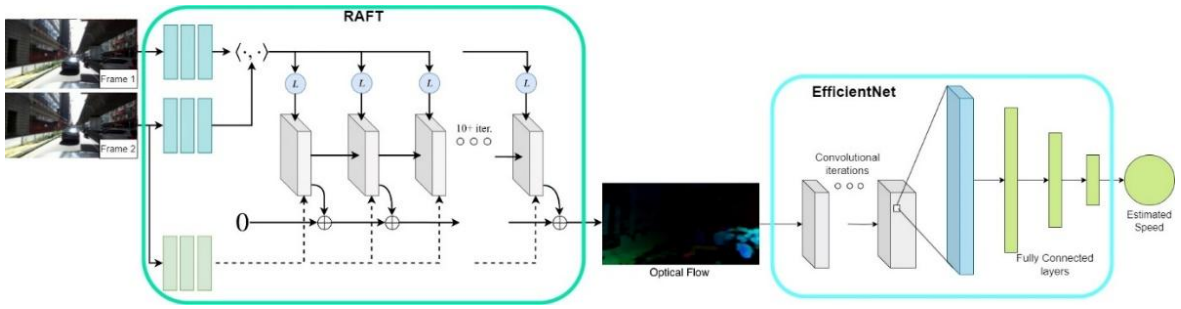


Figure 6-26 Optical flow frame extracted from video captured.

Figure 6-27 Estimated vehicle speed(left) and combined output with optical flow frame overlay(right).

6.6.2 Evaluation of Driver Model Performance using Real-world Data

In order to assess the robustness of the driver model, three models trained on different datasets were compared: the first model trained exclusively on simulation data, the second model trained solely on real-world data, and the third model trained on a combination of simulation and real-world data. Table 6-5 presents the performance metrics for these three models. As indicated in the table, the performance of the driver model trained solely on simulation data significantly decreased when evaluated with real-world data. This decline can be attributed to a domain gap between the two datasets. Simulation data tends to be more idealized, uniform, and less noisy compared to real-world data, which exhibits greater complexity, variability, and noise. Consequently, a model trained exclusively on simulation

data may struggle to generalize well to real-world scenarios, leading to reduced robustness and performance.

Table 6-5 Performance of driver model trained with different type of data tested with real-world data.

Model training data	MSE Steering	MSE Speed
Simulation only	0.4251	0.3623
Simulation + Real-world	0.3278	0.3108
Real-world only	0.5278	0.4937

On the other hand, the driver model trained exclusively on real-world data exhibits the poorest performance among the three models. In contrast, the model trained using both simulation and real-world data achieved the highest accuracy when evaluated with real-world data. This limitation arises from the limited diversity and quantity of critical test scenarios present in real-world data. This scarcity of data is insufficient to effectively train deep neural networks, which typically require extensive datasets for optimal performance. As a consequence, the driver model trained solely on real-world data tends to underfit, meaning it fails to adequately capture the complexity and variability inherent in real-world driving scenarios. This underfitting results in the model's inability to learn the underlying patterns and relationships within the data. Underfitting occurs when a model is too simplistic to capture the complex structures present in the data. Consequently, such models may exhibit poor performance and lack robust generalization capabilities, affecting their efficacy on both training and test datasets.

Whereas simulation data can provide more diversity and coverage of the input space, while real-world data provide more realism and variability of the input conditions. By combining both types of data, the driver model can learn from different sources of information and generalize better to new data. Therefore, the driver model trained with both data can obtain a higher accuracy. However, it is also noted that the accuracy of this model is lower than the accuracy when tested with simulation data. This could be caused by several reasons such as:

- The mixed data is not well balanced. For example, if the mixed data contains too little real-world data, or the real-world data is not diverse or relevant enough, the driver model may not learn the features and patterns of the data properly.
- The mixed data introduces noise or inconsistency that confuses the driver model. For example, if the real-world data has different quality, resolution, format, or annotation than the synthesized data, the model may not be abler to handle the variation and discrepancy between the two types of data.
- The mixed data requires more complex or flexible models or training methods. For example, if the mixed data has a larger or more heterogeneous input space than the simulated data, the model may need more layers, parameters, or regularization techniques to fit the data well.

To address these issues, some possible solutions are:

- Improve the quality and diversity of the mixed data, such as by using data augmentation, domain adaption, or domain randomization techniques.

- To align the characteristics and distributions of the real-world and synthesized data, such as by using data preprocessing, normalization, or domain translation techniques.
- To use more advanced or customized models or training methods, such as by using multi-task learning or meta-learning techniques.

Several scenarios from the real-world dataset were selected to evaluate the driving behaviour of the driver model compared to a human driver. The first testcase is a vehicle cut-in scenario occurred at the Jalan Pudu area during peak hours as shown in Figure 6-28. In the first test case, the ego vehicle follows a white van in congested traffic at Jalan Pudu. At a junction, a black passenger car cuts in from the left, entering the ego vehicle's lane. The driver model must decide whether to brake or change lanes, considering the available space on the right and the oncoming traffic, as shown in Figure 6-28(b) and Figure 6-28(c). Figure 6-29 and Figure 6-30 compare the vehicle speed and steering angle applied by the driver model trained with different data types against the ground truth. The model trained with mixed data (simulation + real-world) closely matches the ground truth, particularly when braking to slow down for the cut-in vehicle. In contrast, the simulation-only model applies a much stronger brake, resulting in a higher deceleration. This discrepancy suggests the simulation-only model is not fully adapted to real-world vehicle responses. The real-world-only model performs poorly due to insufficient data, driving slowly initially and accelerating abruptly later. Given the straight road scenario, deviations in steering angles between the models and the ground truth are minimal.

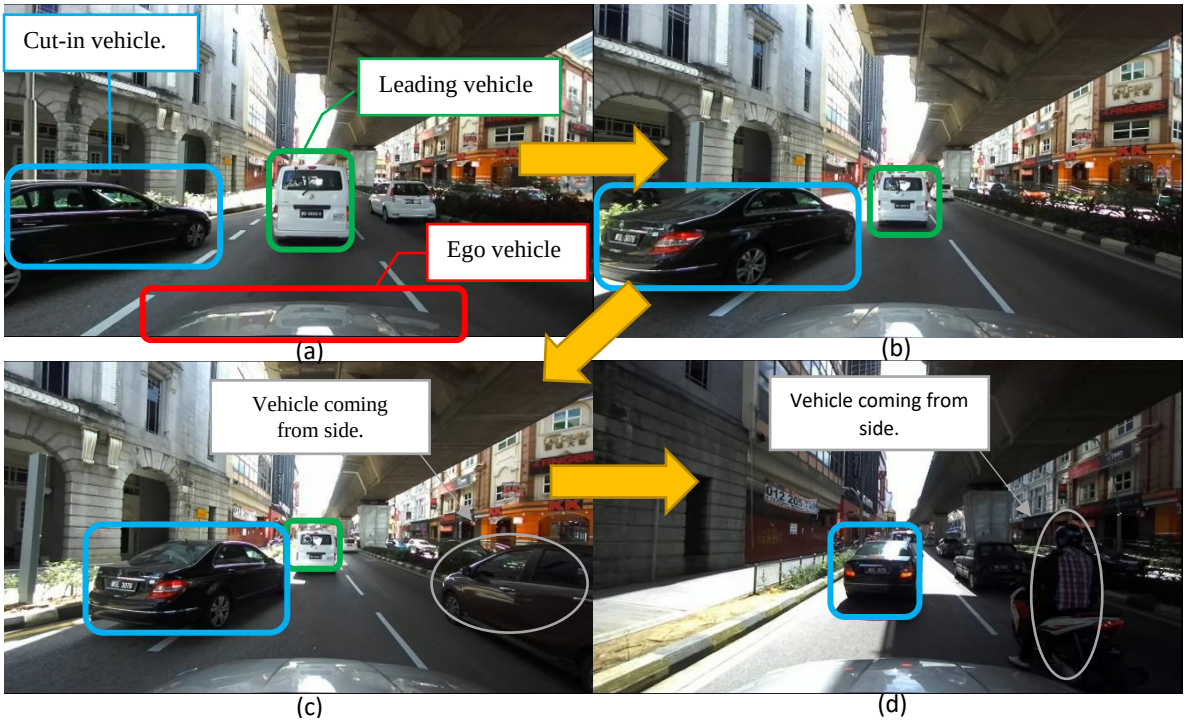


Figure 6-28 Extracted frames of a vehicle cut-in scenario from tested real-world data.

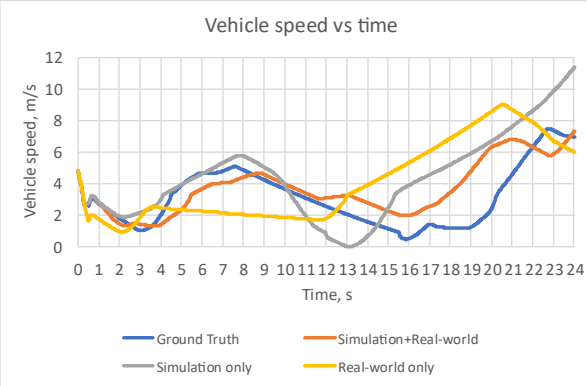


Figure 6-29 Comparison of vehicle speed in real-world vehicle cut-in scenario

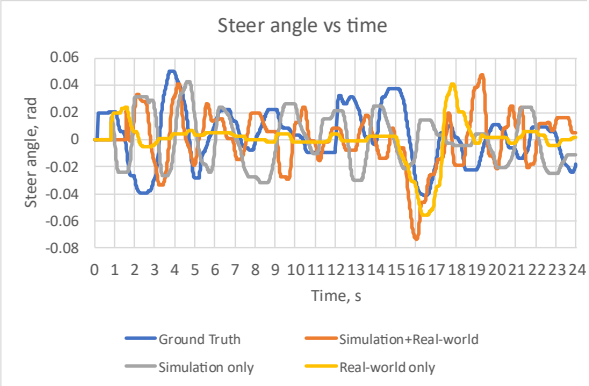


Figure 6-30 Comparison of steer angle in real-world vehicle cut-in scenario

The second scenario involves a pedestrian crossing at the Pavilion Bukit Bintang area during peak hours, as shown in Figure 6-31. The ego vehicle is driven slowly in congested traffic. A pedestrian begins to cross the road despite the green light, as the vehicles ahead are moving very slowly, as shown in Figure 6-31(b). Consequently, the ego vehicle must be stopped to avoid colliding with the pedestrian. Once the pedestrian has safely crossed, the brake of the ego vehicle is released, allowing it to continue moving forward. The braking and steering angle profile applied by the driver model trained by different training data were compared with the ground truth data in Figure 6-32 and Figure 6-33. From the braking percentage against time graph, it can be observed that the model trained with mixed data has similar response to the ground truth, indicating a more balanced response to a pedestrian crossing scenario. Similarly, the steer angle profile of the driver model trained with mixed data has a profile closer to that of the ground truth. In contrast, the simulation-only model exhibits a more aggressive braking and steering pattern, suggesting an overestimation of the required deceleration and steer angle. This could be attributed to the model's limited exposure to real-world vehicle dynamics, which are challenging to replicate precisely in a simulated environment. On the other hand, the real-world-only model underperforms, due to insufficient training data. Overall, the graph underscores the importance of integrating real-world data into driver model training to achieve a more accurate and reliable representation of vehicle behaviour.

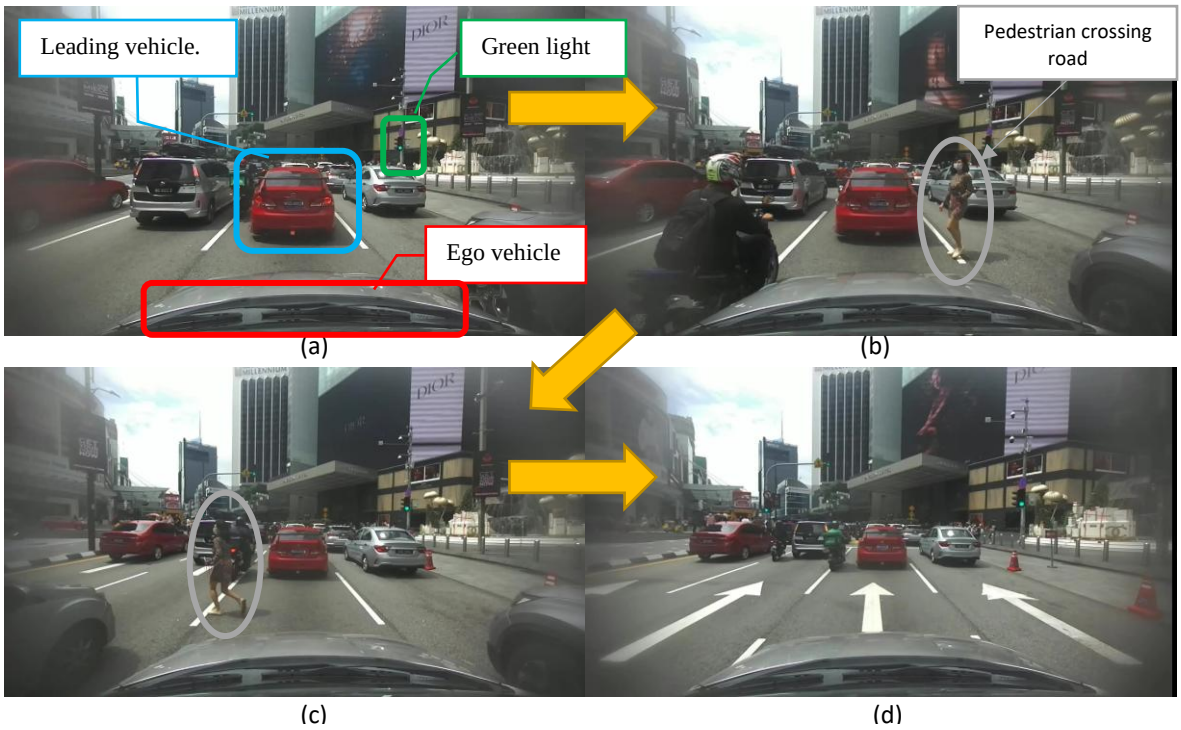


Figure 6-31 Extracted frames of a pedestrian crossing scenario from tested real-world data

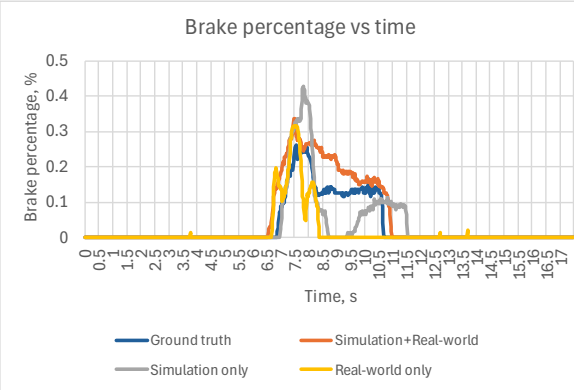


Figure 6-32 Comparison of braking profile in real-world pedestrian crossing scenario

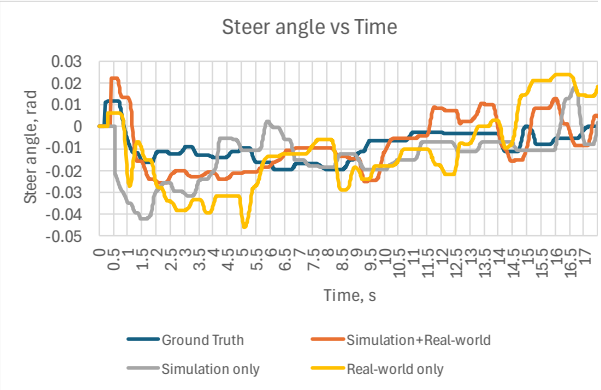


Figure 6-33 Comparison of steer angle between in real-world pedestrian crossing scenario

The third scenario is an overtaking manoeuvre that occurred in the Jalan Pudu area during off-peak hours, as illustrated in Figure 6-34. In this scenario, the ego vehicle is driving along the road when the leading vehicle stops at the side to drop off a passenger. The ego vehicle slows down upon approaching the stopped vehicle and comes to a halt while waiting for an opportunity to overtake. This is shown in Figure 6-34(b). Once it is safe to overtake, the ego vehicle moves into the right lane, as shown in Figure 6-34(c). Finally, Figure 6-34(d) shows the ego vehicle successfully overtake the stopped vehicle. the ego vehicle successfully overtaking the stopped vehicle. The braking, throttle, and steering angle profiles applied by the driver model, which was trained using different datasets, were compared with the ground truth data, as shown in Figure 6-35, Figure 6-36 and Figure 6-37. From the braking profile, it can be observed that the simulation-only model shows a higher degree of braking, particularly at around 11 to 13 seconds, indicating an overly aggressive response compared to the real-world data. This suggests that the simulation-only model may not adequately capture the subtleties of human braking behaviour. The real-world only model underestimates braking in several

instances, pointing to a lack of sufficient training data. The model trained with a mix of simulation and real-world data closely aligns with the ground truth, albeit with some deviations. This indicates that incorporating real-world data enhances the model's realism, though further tuning may be necessary to refine the braking response.

For the throttle profile, once the ego vehicle starts to overtake the vehicle ahead, the ground truth data shows a smooth increase and decrease in throttle application, reflecting an average human driver control. The simulation-only model exhibits abrupt changes in throttle application, particularly noticeable at the beginning and around 6 to 7 seconds. This abruptness suggests that the simulation data alone may not fully encapsulate the variability in human driving behaviour. The real-world only model shows inconsistent throttle application, due to insufficient training data. The mixed data model again demonstrates the closest alignment with the ground truth, but with some over- and under-estimations at different points. This reinforces the benefit of using a combination of data sources to train the model, improving its fidelity to real-world behaviour. In terms of steering angle, the ground truth indicates balanced control. The simulation-only model's frequent sharp adjustments suggest an overly sensitive response. The real-world-only model's instability highlights the challenge of training with limited data. The mixed data model is closest to the ground truth, though deviations during rapid corrections suggest a need for fine-tuning.

Overall, the results indicate that driver models trained with mixed data (simulation + real-world) perform better in replicating real-world human driving behaviour compared to models trained solely on simulation or real-world data. The simulation-only model tends to exaggerate responses, both in braking and throttle applications, due to potential discrepancies between simulated and real-world vehicle dynamics. The real-world only model, constrained by limited data, struggles to generalize across different scenarios, leading to less reliable performance. By integrating real-world data into the training process, the mixed data model benefits from the realism and variability inherent in human driving, leading to more accurate and nuanced control responses. However, this model still exhibits some deviations from the ground truth, highlighting the need for further refinement and potentially more sophisticated data augmentation techniques to bridge the remaining performance gap. In conclusion, training deep learning models for autonomous driving requires a balanced approach that leverages the strengths of both simulation and real-world data. While simulation provides a controlled environment for extensive scenario coverage, real-world data ensures the model captures the complexities of human driving behaviour. The combined approach offers a robust path towards developing driver models that can more accurately mimic human driving in diverse real-world scenarios.

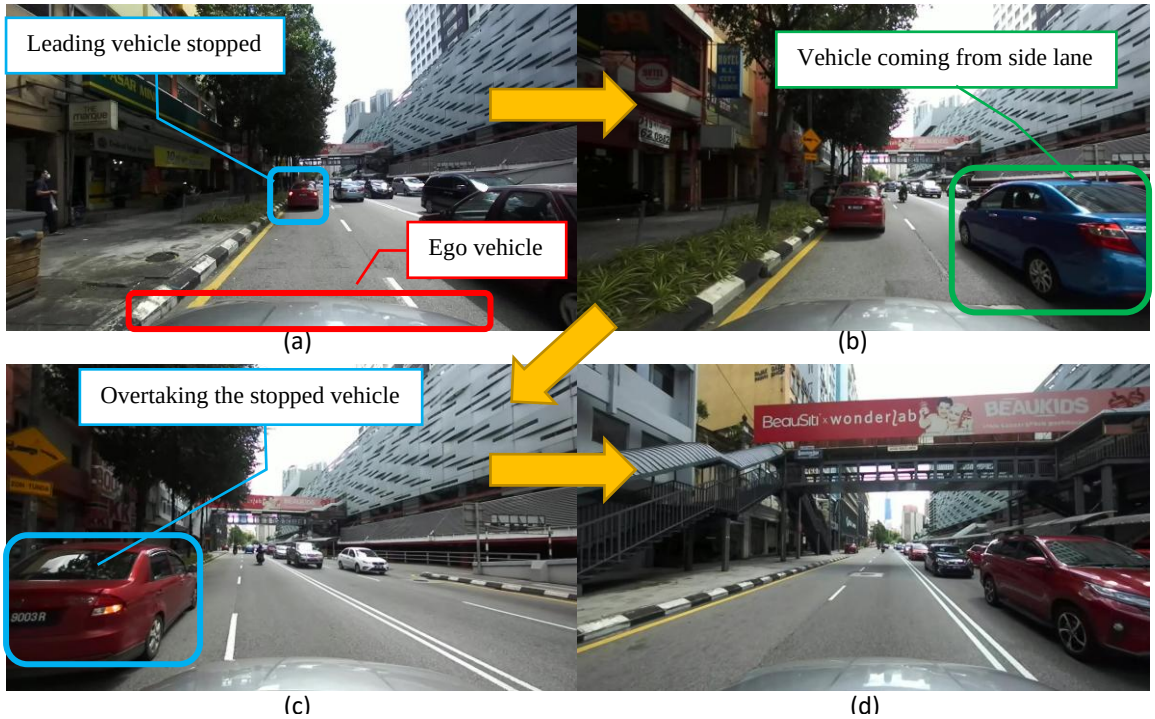


Figure 6-34 Extracted frames of an overtaking scenario from tested real-world data

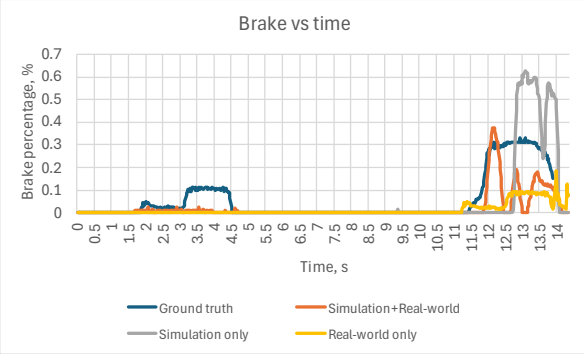


Figure 6-35 Comparison of braking profile in real-world overtaking scenario

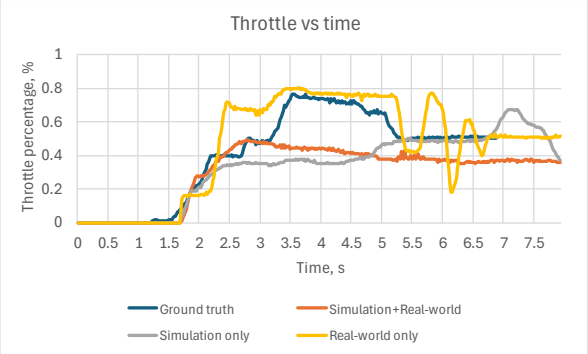


Figure 6-36 Comparison of throttle profile in real-world overtaking scenario

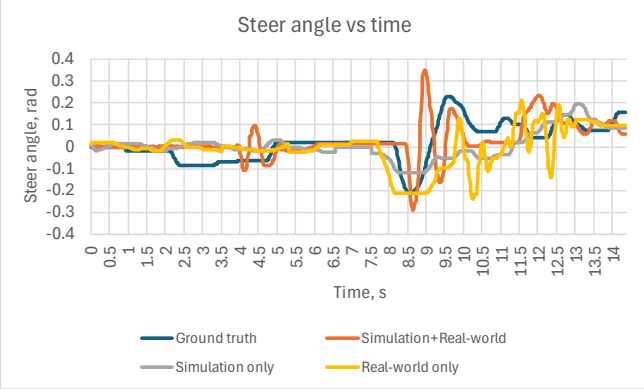


Figure 6-37 Comparison of steer profile in real-world overtaking scenario

6.7 Summary

In this chapter, a driver model based on CNN-LSTM is developed to adapt to normal driving behaviour of Malaysian driver. End-to-end learning approach is used for the development of the driver model that can automatically produce steering angles and throttle and brake inputs from image frames capture by the stereo front-view and rear-view cameras. The driver model is trained and evaluated using Malaysian road dataset, which contains image frames and driving data captured from human on-road driving. The test results show that the model can produce relatively accurate driving manoeuvre of average Malaysian drivers up to 82.17% when compared to the default IPGDriver model provided by the IPG CarMaker. After evaluated with simulation data, the driver model also evaluated using real-world data which collected using instrumented vehicle in the previous chapter. The driver model is retrained with mixed data and real-world data. By comparing the result with the ground truth data, the trained driver model with the mixed data obtained the highest accuracy when compared with the driver model trained only with simulation or real-world data. This is because the simulation data is ideal and do not contain complexity such as sensor noise and fidelity issues. Whereas the driver model trained with real-world data only had limited numbers of data variation and diversity to help the driver model to generalise the problem. Mixed data help the driver model to adapt to both variation of data and complexity in the real-world. However, the driver model with mixed data still only able to obtain accuracy up to 67.23% only. This is due to noise and inconsistent of the real-world data confused the driver model and would need a more advanced driver model architecture to adapt to these data.

Chapter 7: Conclusion and Recommendations for Future Works

7.1 Overview

In this study, the use of deep learning algorithm in development of a driver model for predicting the lateral and longitudinal driving controls was explored. The study begins with the modelling of a non-linear mathematical model of 14 DOF vehicle. By developing the mathematical model, the optimum position to mount the sensor can be identified by comparing the data output produced by the sensors mounted on the virtual vehicle in IPG CarMaker and the vehicle response produced by the mathematical model developed.

In order to validate the vehicle model developed, an instrumented vehicle is developed using sensors such as camera, IMU, GPS, pedal sensors, and steering angle sensor. The developed mathematical model is validated using the instrumented vehicle in terms of handling, accelerating and braking behaviour tests based on ISO standards. Once the model is validated, the instrumented vehicle is used for data recording of driving inputs and road environment at several road network selected based on the traffic condition and at the autonomous vehicle testing road provided by the law makers in Malaysia to increase the variation and diversity of the dataset. The record data is then analysed and classified into critical and normal driving scenarios for development of dataset.

Based on the road environment captured by the instrumented vehicle, digital twin of the road network is developed using IPG CarMaker simulator software. In the virtual environment, virtual test scenarios were developed based on the actual critical scenarios identified. In order to obtain driving reaction data of Malaysian drivers in these scenarios, a 6 DOF motion driving simulator integrated with virtual reality is developed to provide both visual and motion feedback to the drivers while driving in the virtual environment. The driving simulator closed the gap between the virtual and reality world to ensure the driving controls of the drivers aligned with the real-world. Total of thirty participants with different background and driving skills were invited to drive on the driving simulator with different test scenarios. The driving data of the participants were recorded and analysed to extract the driving data of the normal driving drivers to develop the training dataset of the driver model.

Then the driver model is developed with the dataset obtained from the participants. First, the training dataset undergo a series of preprocessing so that the dataset is normalized and not biased so that the driver model can generalized well. Three different driver model is developed based on CNN and LSTM architecture so that the CNN layer can capture the underlying spatial features from the image whereas the LSTM layers can capture the time series features from the input data. The performance of the driver models developed is evaluated using different sets of testing data that were not used for training process so that the robustness of the model can be tested. The driver model with the best performance is selected for the evaluation.

Moreover, the performance selected driver model is compared with other more advanced large model to identify the performance improvement by using a larger neural network with the cost of computing resources. Finally, the driver model also tested using real-

world data recorded from the instrumented vehicle to investigate the performance of the driver model when implementing on an actual vehicle in the future research.

7.2 Conclusion

Based on the results and discussion from the previous chapters, the conclusion that can be drawn from the overall study are presented in the following paragraphs. Firstly, it can be concluded that the non-linear 14 DOF vehicle model developed in this study represents well the actual behaviour of vehicle in the form of handling characteristics as well as the behaviour of the vehicle in the presence of the riding characteristics. The accuracy and validity of the developed model was also shown by comparing the yaw rate, yaw angle, lateral acceleration, and lateral displacements at various testing conditions. Generally, it can be seen that the trends between simulation results and experimental data are almost similar, but slightly different in magnitude. This is due to the simplifications and assumptions considered in the vehicle dynamics modelling.

Secondly, an instrumented vehicle equipped with affordable off-the-shelf sensors was developed to collect road environment and driving data. This setup enables near real-time performance without the need for a stringent real-time OS or bare-metal embedded system. The instrumented vehicle underwent field tests, including SAE standard manoeuvres like double lane changes, step steering, and emergency braking, to evaluate the data recording system. The recorded vehicle response data was compared against responses from IPG CarMaker and a Simulink 14 DOF mathematical model. The results showed that the sensors captured the vehicle's responses with a root mean square error of less than 5%, indicating the virtual vehicle's ability to accurately replicate actual vehicle responses. Data collection took place on various road networks, such as the University of Nottingham Malaysia campus route, Jalan Pudu and Masjid Jamek route, and Cyberjaya MaGIC routes, during morning, afternoon, and evening sessions, resulting in a total of 245 hours of driving data recorded.

Thirdly, a motion simulator was developed and seamlessly integrated with IPG CarMaker, providing precise motion feedback. To deepen the immersive experience, a virtual reality headset was employed, offering drivers a complete 360-degree view of the cockpit. The recorded data from the instrumented vehicle underwent thorough analysis using the YOLOv8 deep learning object detection algorithm, allowing for the precise identification of road users and objects. Building on these identified scenarios, virtual scenarios were meticulously reproduced in IPG CarMaker. Once the system is configured, thirty participants were invited to take the virtual wheel and test the driving simulator. Their reactions and driving inputs were carefully recorded as they navigated through various scenarios. Their feedback was overwhelmingly positive, indicating that the simulator successfully bridged the gap between the virtual and real worlds, offering a remarkably lifelike driving experience. Each participant's driving data was meticulously analysed and classified into three categories: aggressive, normal, and slow drivers. Specifically, the driving data of the normal drivers was isolated and earmarked to create training dataset for the driver model. The results were promising, showcasing that the virtual simulations were adept at reproducing actual scenarios and vehicle responses with striking similarity. However, it was noted that the default driver model provided

by IPG CarMaker exhibited slightly different driving behaviour compared to that of human drivers. This observation highlighted the necessity of developing a driver model optimized for the Malaysian traffic environment, a crucial step to ensure the reliability of safety testing using the simulation platform.

Fourthly, the proposed model combined convolutional neural networks (CNNs) and long short-term memory (LSTM) networks to extract spatiotemporal features from camera images, vehicle dynamics, and waypoints. Evaluations conducted on a large dataset of Malaysian driving scenarios revealed that the driver model outperformed several baseline models, achieving high accuracy of up to 84.63% in predicting driving inputs. These findings suggest that the driver model effectively predicted driver actions and intentions, including steering, braking, accelerating, lane changing, and turning. Moreover, the model demonstrated high accuracy, robustness, and generalization across various driving scenarios and situations, surpassing existing models based on conventional machine learning or single deep learning methods such as FCN, CNN or LSTM. Additionally, the driver model was tested with real-world data recorded using the instrumented vehicle. Initially trained solely on simulation data, the model did not perform well when tested with real-world data due to fidelity and noise differences between the two datasets. However, after further training with real-world data, the model's accuracy improved significantly, reaching up to 67.23%. This indicates that providing a greater variety of real-world data to the driver model can enhance its ability to generalize and improve overall performance.

The limitations of this research were related to the data collection and the model design. The data collection was limited by the number of participants, the duration of driving, the diversity of road environments, and the ethical issues of informed consent and privacy protection. The model design was limited by the choice of input features, output variables, network parameters, and optimization algorithms. These limitations may affect the validity, reliability, and scalability of the driver model, and thus require further investigation and improvement. The implications of this research were significant for both theory and practice. For theory, this research contributed to the advancement of knowledge and understanding of driving behaviour and driver modelling, by applying the state-of-the-art deep learning method CNN-LSTM model, which can capture the spatial and temporal features of driving behaviour and predict the driver’s actions and intentions. For practice, this research provided a practical and effective solution for developing driver models, which can be used for various purposes, such as driver assistance systems, autonomous driving systems, driver training systems, and driver behaviour analysis systems.

The findings of this research present significant contributions to the fields of driver behaviour modelling, driving simulation, and autonomous vehicle validation. The development of a Malaysian-specific driver model using deep learning techniques marks a notable advancement in adaptive AI-driven driving systems, particularly in addressing the limitations of conventional models that do not fully capture region-specific driving behaviours and traffic conditions. Additionally, the integration of a high-fidelity motion simulator with virtual reality (VR) immersion provides a more realistic and interactive test environment, bridging the gap between traditional driving simulators and real-world driving experiences. The creation of the

Malaysian Road Scenario Database (MaRSeD) offers a novel, structured dataset for testing driver models under realistic and locally relevant conditions, which has been largely overlooked in existing studies focused on Western driving environments. Furthermore, the CNN-LSTM-based driver model has demonstrated superior performance over conventional approaches, showing its potential for applications in driver assistance systems, human-in-the-loop autonomous driving research, and intelligent vehicle safety systems. These advancements contribute to the next generation of AI-based driving technologies, reinforcing the importance of regionally optimized, data-driven solutions in enhancing road safety and autonomous vehicle development.

7.3 Recommendation and future works

The recommendations for future research were focused on enhancing the data collection and the model design. Some directions for future research are:

1. **Data Collection**: For data collection, the current data recorded is very limited due to time constraint to complete the research study which could have missed a lot of important scenarios that are happening in Malaysia. The dataset also lack data during nighttime, heavy rain, and fogging situation due to limitation of the sensors used in the instrumented vehicle having difficulty capturing data in the dark. Therefore, it is suggested to increase the number of participants, the duration of driving, the diversity of road environments, and the ethical standards of informed consent and privacy protection, to obtain more comprehensive and representative driving data. Furthermore, more sensors such as Lidar, Radar can be used on the instrumented vehicle to capture data during nighttime, heavy rain, and fogging where the visibility from the camera sensor is low.
2. **Model Design**: For model design, the current driver model only focuses on the lateral and longitudinal control of the vehicle due to action of other road actors such as pedestrians, car, motorcycle. However, in the real-world, there are other factors that need to be considered such as road signs and traffic lights. Therefore, it is suggested to incorporate more information sources into the model, such as road maps, traffic signs, and vehicle-to-vehicle communication, which can enrich the model's perception of the environment and improve its prediction accuracy.
3. **Model Architecture**: In this study, the driver model architecture is based on CNN-LSTM only. This is because due to difficulty to obtain a high-performance computing GPU for inferencing. This limited the size of the model that can be developed which can improve the performance of the driver model in predicting the driving controls more accurately. Therefore, it is also suggested to compare and integrate the driver model with other deep learning methods or hybrid methods, such as integration with Visual Transformers model, to further improve the accuracy, robustness, and generalization of the driver model. Furthermore, the model also can be extended to predict other aspects of driving behaviour, such as driver's intention, attention, and emotion, which can enhance the understanding of driver's cognitive state and decision-making process.

4. **Real-world deployment**: In this study, the driver model only tested with data recordings from the instrumented vehicle. This is due to the difficulty to obtain an actual autonomous vehicle for testing the algorithm developed on a real vehicle. The data recordings have limitation such as the data is fixed and will not respond to the manoeuvre produced by the driver model. Therefore, extending the evaluation of the driver model in real-world driving conditions using an actual vehicle is necessary to provide insights into practical challenges and risks.
5. **Safety Testing of Autonomous Vehicles**: The driver model can play a crucial role in the safety testing of AVs by simulating human driving behaviours and responses. It could be integrated into simulation platforms to replicate realistic driving scenarios and evaluate how an AV reacts to dynamic traffic conditions, such as sudden stops, pedestrian crossings, or erratic manoeuvres by other vehicles. This approach allows for a thorough examination of the AV's capabilities and limitations in a controlled environment, ensuring the vehicle can safely navigate real-world driving conditions before deployment. Additionally, the driver model could aid in developing human-machine interfaces (HMIs) by providing insights into how a human driver would interact with the AV's systems, which is critical for Level 3 and above autonomous vehicles where human intervention may be necessary. Overall, the driver model is an indispensable tool in the verification and validation framework for AV safety testing, contributing to the advancement of reliable and secure autonomous driving technology.
6. **Motion Platform and Teleoperation:** The 6DOF AVES Motion Driving Simulator (AMoDS) could play a vital role in the safety testing of AVs by providing a realistic driving experience for safety operators. It could allow for the simulation of various driving scenarios, including emergency conditions, to evaluate the operator's responses and the vehicle's systems. The integration of Virtual Reality (VR) could enhance immersion, aiding in the assessment of the Human-Machine Interface (HMI) and the operator's decision-making process. Teleoperation, particularly for Level 5 AVs, could be essential when the AV encounters scenarios not covered by its AI algorithms. The use of 5G technology ensures low latency and high reliability, allowing commands from the safety operator to be transmitted to the AV with minimal delay, which is crucial for the safety of both the vehicle and surrounding traffic.
7. **Malaysian Road Scenario Database:** The MaRSeD could play a crucial role in enhancing the safety testing of AVs in Malaysia. It is a comprehensive collection of test scenarios derived from real-world driving conditions that reflect the unique traffic patterns and road behaviours prevalent in Malaysia. By incorporating these scenarios into the safety testing framework, developers could rigorously assess the adaptability and capability of AVs to handle complex and unpredictable situations. This localized context ensures that AVs are evaluated against scenarios representative of actual challenges on Malaysian roads. MaRSeD aligns with international standards, promoting a standardized approach to scenario-based testing and facilitating the comparison of AV performance across different regions. It serves as a valuable resource for AV developers, researchers, and regulatory bodies in establishing safety guidelines and approval processes for AV deployment in Malaysia.

In summary, future research could focus on expanding data collection efforts, incorporating additional information sources and advanced model architectures, and thoroughly testing the driver model in real-world scenarios to enhance the performance, safety, and reliability of autonomous vehicles.

References

[1] X. Zhao, V. Robu, D. Flynn, K. Salako, and L. Strigini, “Assessing the safety and reliability of autonomous vehicles from road testing,” in *2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE)*, IEEE, 2019, pp. 13–23.

[2] S. A. Bagloee, M. Tavana, M. Asadi, and T. Oliver, “Autonomous vehicles: challenges, opportunities, and future implications for transportation policies,” *Journal of Modern Transportation*, vol. 24, no. 4, pp. 284–303, Dec. 2016, doi: 10.1007/s40534-016-0117-3.

[3] N. Kalra and S. M. Paddock, “Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?,” *Transp Res Part A Policy Pract*, vol. 94, pp. 182–193, 2016.

[4] K. Suzuki, K. Tang, W. Alhajyaseen, K. Suzuki, and H. Nakamura, “An international comparative study on driving attitudes and behaviors based on questionnaire surveys,” *IATSS research*, vol. 46, no. 1, pp. 26–35, 2022.

[5] R. C. McIlroy, K. A. Plant, M. S. Hoque, J. Wu, G. O. Kokwaro, V. H. Nam, and N. A. Stanton, “Who is responsible for global road safety? A cross-cultural comparison of Actor Maps,” *Accid Anal Prev*, vol. 122, pp. 8–18, 2019.

[6] S. Taxonomy, “Definitions for terms related to driving automation systems for on-road motor vehicles (j3016),” *Soc. Automot. Eng., Warrendale, PA, USA, Tech. Rep. J3016_201806*, 2016.

[7] R. Ferlis, “The dream of the automated highway,” *Public roads*, vol. 71, no. 1, pp. 42–47, 2007.

[8] J. Leonard, J. How, S. Teller, M. Berger, S. Campbell, G. Fiore, L. Fletcher, E. Frazzoli, A. Huang, and S. Karaman, “A perception-driven autonomous urban vehicle,” *J Field Robot*, vol. 25, no. 10, pp. 727–774, 2008.

[9] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, and C. Geyer, “Autonomous driving in urban environments: Boss and the urban challenge,” *J Field Robot*, vol. 25, no. 8, pp. 425–466, 2008.

[10] L. Lewis, B. Hare, H. Clyde, and R. Landis, “Vehicle Control History: Data from Driver Input and Pre-Collision System Activation Events on Toyota Vehicles,” SAE Technical Paper, 2019.

[11] S. Ingle and M. Phute, “Tesla autopilot: semi autonomous driving, an uptick for future autonomy,” *International Research Journal of Engineering and Technology*, vol. 3, no. 9, pp. 369–372, 2016.

[12] T. Yokoyama, “RoAD to L4, Advancing Autonomy: Research, Development, Demonstration, and Deployment of Level 4 Driving Automation and Enhanced Mobility Services in Japan,” in *Automated Road Transportation Symposium*, Springer, 2023, pp. 15–22.

[13] S. Gibbs, “Google sibling waymo launches fully autonomous ride-hailing service,” *The Guardian*, vol. 7, 2017.

[14] D. Kumari and S. Bha, “Accelerating the Race to Autonomous Cars–A Case Study,” *International Journal of Applied Engineering and Management Letters (IJAEML)*, vol. 5, no. 2, pp. 219–231, 2021.

[15] S. Singh, “Critical reasons for crashes investigated in the national motor vehicle crash causation survey,” 2015. Accessed: Jun. 05, 2022. [Online]. Available: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812115

[16] P. Koopman and M. Wagner, “Challenges in autonomous vehicle testing and validation,” *SAE Int J Transp Saf*, vol. 4, no. 1, pp. 15–24, 2016.

[17] A. M. Ivanov, S. S. Shadrin, and D. A. Makarova, “The analysis of international standards in the field of safety regulation of highly automated and autonomous vehicles,” in *2022 Systems of Signals Generating and Processing in the Field of on Board Communications*, IEEE, 2022, pp. 1–6.

[18] W. H. Widen and P. Koopman, “Autonomous vehicle regulation & trust: The impact of failures to comply with standards,” *UCLA JL & Tech.*, vol. 27, p. 169, 2022.

[19] Arberdeen, “Disruption Is Here in Automotive and Ground Transportation: Are You Ready?” Accessed: Jun. 22, 2021. [Online]. Available: https://www.ansys.com/resource-center/white-paper/disruption-in-auto-ground-transportation

[20] P. Sun *et al.*, “Scalability in Perception for Autonomous Driving: Waymo Open Dataset.” Accessed: Jun. 20, 2021. [Online]. Available: http://www.waymo.com/open.

[21] “Putting Zoox to the test: preparing for the challenges of the road.” Accessed: Jun. 09, 2022. [Online]. Available: https://zoox.com/journal/structured-testing/

[22] Y. Shi, M. Bordegoni, and G. Caruso, “User studies by driving simulators in the era of automated vehicle,” *Comput Aided Des Appl*, vol. 18, no. 1, pp. 211–226, 2020.

[23] G. P. Bertollini, C. M. Johnston, J. W. Kuiper, J. C. Kukula, M. A. Kulczycka, and W. E. Thomas, “The general motors driving simulator,” *SAE transactions*, pp. 54–67, 1994.

[24] J. Wong, “Inside GM’s 360-degree vehicle testing simulator.” Accessed: Jun. 20, 2021. [Online]. Available: https://www.cnet.com/roadshow/news/general-motors-gm-360-degree-simulator/

[25] J. Fadaie, “The state of modeling, simulation, and data utilization within industry: An autonomous vehicles perspective,” *arXiv preprint arXiv:1910.06075*, 2019.

[26] “Automated and Connected Mobility AVL.” Accessed: Jun. 22, 2021. [Online]. Available: https://www.avl.com/en/automated-and-connected-mobility

[27] G. Rong, B. H. Shin, H. Tabatabaee, Q. Lu, S. Lemke, M. Možeiko, E. Boise, G. Uhm, M. Gerow, and S. Mehta, “Lgsvl simulator: A high fidelity simulator for autonomous driving,” in *2020 IEEE 23rd International conference on intelligent transportation systems (ITSC)*, IEEE, 2020, pp. 1–6.

[28] SAE Media Group, “Simulation of Autonomous Vehicles,” Aug. 2019. Accessed: Jun. 20, 2021. [Online]. Available: https://www.techbriefs.com/component/content/article/34920-simulation-of-autonomous-vehicles

[29] S. Sovani, “Simulation accelerates development of autonomous driving,” *ATZ worldwide*, vol. 119, no. 9, pp. 24–29, 2017.

[30] OPAL-RT, “Maximum Flexibility for Co-Simulation of Heterogenous Models in Real-Time.” Accessed: Apr. 13, 2022. [Online]. Available: https://www.opal-rt.com/orchestra/

[31] NVIDIA, “AI-powered NVIDIA DRIVE Infrastructure.” Accessed: Jun. 20, 2022. [Online]. Available: https://www.nvidia.com/en-us/self-driving-cars/infrastructure/

[32] Hexagon, “System dynamics.” Accessed: Apr. 03, 2022. [Online]. Available: https://hexagon.com/solutions/system-dynamics

[33] FAAC, “ADAS Simulation Autonomous Vehicle Research.” Accessed: Apr. 03, 2022. [Online]. Available: https://www.faac.com/simulation-training/solutions/research-simulation-adas-autonomous-vehicles/

[34] rFpro, “Automotive Simulation • Driving Simulation • Autonomous Driving • rFpro.” Accessed: Apr. 03, 2022. [Online]. Available: https://rfpro.com/

[35] Vehicle Dynamics International, “Autonomous Vehicle Test & Development Symposium 2019,” May 2024. Accessed: May 03, 2023. [Online]. Available: https://www.vehicledynamicsinternational.com/event/autonomous-vehicle-test-development-symposium-2019

[36] G. Audi and A. G. Volkswagen, “About PEGASUS - pegasus-EN.” Accessed: Feb. 11, 2022. [Online]. Available: https://www.pegasusprojekt.de/en/about-PEGASUS

[37] Energy Research Institute @ NTU, “Centre of Excellence for Testing & Research of Autonomous Vehicles – NTU.” Accessed: Jan. 03, 2022. [Online]. Available: https://www.ntu.edu.sg/erian/research-capabilities/centre-of-excellence-for-testing-research-of-autonomous-vehicles-ntu

[38] NAVER LABS, “NAVER LABS.” Accessed: May 01, 2022. [Online]. Available: http://www.naverlabs.com/en/storyList

[39] MORAI, “MORAI Simulation Platform for Autonomous Vehicle & Urban Air Mobility.” Accessed: May 03, 2022. [Online]. Available: https://www.morai.ai

[40] S. Shah, D. Dey, C. Lovett, and A. Kapoor, “Airsim: High-fidelity visual and physical simulation for autonomous vehicles,” in *Field and Service Robotics: Results of the 11th International Conference*, Springer, 2018, pp. 621–635.

[41] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An open urban driving simulator,” in *Conference on robot learning*, PMLR, 2017, pp. 1–16.

[42] Uber, “Introducing AVS, an Open Standard for Autonomous Vehicle Visualization from Uber.” Accessed: Jun. 04, 2022. [Online]. Available: https://www.uber.com/en-SG/blog/avs-autonomous-vehicle-visualization/

[43] Apollo, “Apollo.” Accessed: Sep. 21, 2022. [Online]. Available: https://developer.apollo.auto/gamesim.html

[44] N. Koenig and A. Howard, “Design and use paradigms for gazebo, an open-source multi-robot simulator,” in *2004 IEEE/RSJ international conference on intelligent robots and systems (IROS)(IEEE Cat. No. 04CH37566)*, Ieee, 2004, pp. 2149–2154.

[45] CarSim, “Mechanical Simulation CarSim and TruckSim Brake System.” Accessed: Jun. 18, 2022. [Online]. Available: https://www.carsim.com/

[46] R. Donà and B. Ciuffo, “Virtual testing of automated driving systems. A survey on validation methods,” *IEEE Access*, vol. 10, pp. 24349–24367, 2022.

[47] D. Jianyu, Y. Wang, J. Ding, and W. Deng, “Digital Twin Test Method for Autonomous Vehicles Based on PanoSim,” SAE Technical Paper, 2023.

[48] IPG Automotive GmbH, “CarMaker IPG Automotive.” Accessed: Jan. 15, 2022. [Online]. Available: https://www.ipg-automotive.com/en/products-solutions/software/carmaker/

[49] M. Acosta and S. Kanarachos, “Tire lateral force estimation and grip potential identification using Neural Networks, Extended Kalman Filter, and Recursive Least Squares,” *Neural Comput Appl*, vol. 30, pp. 3445–3465, 2018.

[50] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “ROS: an open-source Robot Operating System,” in *ICRA workshop on open source software*, Kobe, Japan, 2009, p. 5.

[51] M. F. Drechsler, J. Peintner, F. Reway, G. Seifert, A. Riener, and W. Huber, “MiRE, A Mixed Reality Environment for Testing of Automated Driving Functions,” *IEEE Trans Veh Technol*, vol. 71, no. 4, pp. 3443–3456, 2022, doi: 10.1109/TVT.2022.3160353.

[52] M. Bruschetta, K. N. de Winkel, E. Mion, P. Pretto, A. Beghi, and H. H. Bülthoff, “Assessing the contribution of active somatosensory stimulation to self-acceleration perception in dynamic driving simulators,” *PLoS One*, vol. 16, no. 11 November, Nov. 2021, doi: 10.1371/journal.pone.0259015.

[53] P. M. T. Zaal, F. M. Nieuwenhuizen, M. Mulder, and M. M. Van Paassen, “Perception of Visual and Motion Cues during Control of Self-Motion in Optic Flow Environments,” in *AIAA Modeling and Simulation Technologies Conference and Exhibit*, Keystone, Colorado, Aug. 2006, p. 6627.

[54] R. Wilkie and J. Wann, “Controlling Steering and Judging Heading: Retinal Flow, Visual Direction, and Extraretinal Information,” *J Exp Psychol Hum Percept Perform*, vol. 29, no. 2, pp. 363–378, 2003, doi: https://doi.org/10.1037/0096-1523.29.2.363.

[55] J. R. Lackner and P. DiZio, “Vestibular, proprioceptive, and haptic contributions to spatial orientation,” *Annu Rev Psychol*, vol. 56, no. 1, pp. 115–147, 2005, doi: 10.1146/annurev.psych.55.090902.142023.

[56] Y.-S. Kim, H. Shi, N. Dagalakis, J. Marvel, and G. Cheok, “Design of a six-DOF motion tracking system based on a Stewart platform and ball-and-socket joints,” *Mech Mach Theory*, vol. 133, pp. 84–94, 2019, doi: https://doi.org/10.1016/j.mechmachtheory.2018.10.021.

[57] E. Zeeb, “Daimler’s New Full-Scale, High-dynamic Driving Simulator-A Technical Overview,” *Actes INRETS*, pp. 157–165, 2010.

[58] G. J. Heydinger, M. K. Salaani, W. R. Garrott, and P. A. Grygier, “Vehicle dynamics modelling for the National Advanced Driving Simulator,” *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 216, no. 4, pp. 307–318, 2002, doi: 10.1243/0954407021529138.

[59] Y. R. Khusro, Y. Zheng, M. Grottoli, and B. Shyrokau, “MPC-Based Motion-Cueing Algorithm for a 6-DOF Driving Simulator with Actuator Constraints,” *Vehicles*, vol. 2, no. 4, pp. 625–647, 2020, doi: 10.3390/vehicles2040036.

[60] C. Y. Shiong, M. K. A. Jalil, and M. Hussein, “Motion visualisation and control of a driving simulator motion platform,” in *2009 6th International Symposium on Mechatronics and its Applications*, 2009, pp. 1–5. doi: 10.1109/ISMA.2009.5164839.

[61] Y. R. Khusro, Y. Zheng, M. Grottoli, and B. Shyrokau, “MPC-Based Motion-Cueing Algorithm for a 6-DOF Driving Simulator with Actuator Constraints,” *Vehicles*, vol. 2, no. 4, pp. 625–647, Dec. 2020, doi: 10.3390/vehicles2040036.

[62] A. Adel, M. Mahmoud, N. Sayed, O. Hisham, O. Ossama, P. Adel, Y. Ayman, M. I. Awad, S. A. Maged, S. M. Umer, H. Iqbal, and H. F. Maqbool, “Design of A 6-DOF Hydraulic Vehicle Driving Simulator,” in *Proceedings of 2020 International Conference on Innovative Trends in Communication and Computer Engineering, ITCE 2020*, Aswan, Egypt: Institute of Electrical and Electronics Engineers Inc., Feb. 2020, pp. 170–175. doi: 10.1109/ITCE48509.2020.9047787.

[63] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 2001, pp. I–I. doi: 10.1109/CVPR.2001.990517.

[64] A. R. W. Huang and C. Chen, “A low-cost driving simulator for full vehicle dynamics simulation,” *IEEE Trans Veh Technol*, vol. 52, no. 1, pp. 162–172, 2003, doi: 10.1109/TVT.2002.807157.

[65] P. Pretto, M. Ogier, H. H. Bülthoff, and J.-P. Bresciani, “Influence of the size of the field of view on motion perception,” *Comput Graph*, vol. 33, no. 2, pp. 139–146, 2009, doi: https://doi.org/10.1016/j.cag.2009.01.003.

[66] G. Silvera, A. Biswas, and H. Admoni, “DReye VR: Democratizing Virtual Reality Driving Simulation for Behavioural & Interaction Research,” *2022 17th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 639–643, 2022.

[67] J. Pak and U. Maoz, “Compact 3 DOF Driving Simulator using Immersive Virtual Reality,” *ArXiv*, vol. abs/1909.05833, 2019.

[68] E. Tekin and S. Ertugrul, “Multi-input multi-output intelligent modelling techniques and application to human driver,” *Int. J. Vehicle Performance*, vol. 2, no. 4, pp. 390–417, 2016.

[69] C. J. Hong and V. R. Aparow, “System configuration of Human-in-the-loop Simulation for Level 3 Autonomous Vehicle using IPG CarMaker,” in *2021 IEEE International Conference on Internet of Things and Intelligence Systems (IoTaIS)*, 2021, pp. 215–221. doi: 10.1109/IoTaIS53735.2021.9628587.

[70] A. Riener Second Examiner Gary Burnett and J. Kepler, “Automated Driving: Towards Trustworthy and Safe Human-Machine Cooperation,” Dissertation (PhD), University Library Linz, 2020. Accessed: Dec. 03, 2021. [Online]. Available: www.jku.at

[71] T. Holdgrün, I. Doric, T. Brandmeier, T. Fuchs, J. Mïhlbauer, P. Steinert, and S. Peldschus, “A Virtual Reality based Approach for Researching Pedestrian to Vehicle Collisions,” in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 1318–1325. doi: 10.1109/IVS.2018.8500524.

[72] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The KITTI dataset,” *International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013, doi: 10.1177/0278364913491297.

[73] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha, “TrafficPredict: Trajectory Prediction for Heterogeneous Traffic-Agents,” *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 1, pp. 6120–6127, 2019.

[74] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, “Nuscenes: A multimodal dataset for autonomous driving,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, 2020, pp. 11618–11628. doi: 10.1109/CVPR42600.2020.01164.

[75] J. Mao, M. Niu, C. Jiang, H. Liang, J. Chen, X. Liang, Y. Li, C. Ye, W. Zhang, Z. Li, J. Yu, H. Xu, and C. Xu, “One Million Scenes for Autonomous Driving: ONCE Dataset,” *arXiv preprint arXiv:2106.11037*, 2021.

[76] M. Sheeny, E. De Pellegrin, S. Mukherjee, A. Ahrabian, S. Wang, and A. Wallace, “RADIATE: A Radar Dataset for Automotive Perception in Bad Weather,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, Xi’an, China, May 2021, pp. 1–7.

[77] K. Burnett, D. J. Yoon, Y. Wu, A. Z. Li, H. Zhang, S. Lu, J. Qian, W.-K. Tseng, A. Lambert, K. Y. K. Leung, A. P. Schoellig, and T. D. Barfoot, “Boreas: A Multi-Season Autonomous Driving Dataset,” *Int J Rob Res*, vol. 42, no. 1–2, pp. 33–42, Mar. 2023.

[78] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, and D. Ramanan, “Argoverse: 3d tracking and forecasting with rich maps,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 8748–8757.

[79] J. Geyer, Y. Kassahun, M. Mahmudi, X. Ricou, R. Durgesh, A. S. Chung, L. Hauswald, V. H. Pham, M. Mühlegg, and S. Dorn, "A2d2: Audi autonomous driving dataset," *arXiv preprint arXiv:2004.06320*, 2020.

[80] D. Bogdoll, F. Schreyer, and J. M. Zöllner, "Ad-datasets: a meta-collection of data sets for autonomous driving," *arXiv preprint arXiv:2202.01909*, 2022.

[81] Q.-H. Pham, P. Sevestre, R. S. Pahwa, H. Zhan, C. H. Pang, Y. Chen, A. Mustafa, V. Chandrasekhar, and J. Lin, "A 3D dataset: Towards autonomous driving in challenging environments," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 2267–2273.

[82] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2636–2645.

[83] L. Ding, J. Terwilliger, R. Sherony, B. Reimer, and L. Fridman, "MIT driveseg (manual) dataset for dynamic driving scene segmentation," in *Tech. Rep., Technical report*, Massachusetts Institute of Technology, 2020.

[84] H. Schafer, E. Santana, A. Haden, and R. Biasini, "A commute in data: The comma2k19 dataset," *arXiv preprint arXiv:1812.05752*, 2018.

[85] S. Agarwal, A. Vora, G. Pandey, W. Williams, H. Kourous, and J. McBride, "Ford multi-av seasonal dataset," *Int J Rob Res*, vol. 39, no. 12, pp. 1367–1376, 2020.

[86] J. Houston, G. Zuidhof, L. Bergamini, Y. Ye, L. Chen, A. Jain, S. Omari, V. Iglovikov, and P. Ondruska, "One thousand and one hours: Self-driving motion prediction dataset," in *Conference on Robot Learning*, PMLR, 2021, pp. 409–418.

[87] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The Oxford RobotCar dataset," *Int J Rob Res*, vol. 36, no. 1, pp. 3–15, 2017.

[88] Y. Dong, Y. Zhong, W. Yu, M. Zhu, P. Lu, Y. Fang, J. Hong, and H. Peng, "Mcity Data Collection for Automated Vehicles Study," *arXiv preprint arXiv:1912.06258*, Dec. 2019.

[89] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, D. A. R&d, and T. U. Darmstadt, "The Cityscapes Dataset for Semantic Urban Scene Understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 3213–3223.

[90] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang, "The apolloscape dataset for autonomous driving," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, IEEE Computer Society, Dec. 2018, pp. 1067–1073. doi: 10.1109/CVPRW.2018.00141.

[91] T. Roddick, A. Kendall, and R. Cipolla, "Orthographic Feature Transform for Monocular 3D Object Detection," *arXiv preprint arXiv:1811.08188*, Nov. 2018.

[92] B. Xu and Z. Chen, “Multi-Level Fusion based 3D Object Detection from Monocular Images,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2018, pp. 2345–2353.

[93] Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, and K. Q. Weinberger, “Pseudo-LiDAR from Visual Depth Estimation: Bridging the Gap in 3D Object Detection for Autonomous Driving,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Dec. 2019, pp. 8445–8453.

[94] H. Ritter and H. Rohling, “Pedestrian detection based on automotive radar,” in *2007 IET International Conference on Radar Systems*, Edinburgh, UK, 2007, pp. 1–4.

[95] K. Doman, D. Deguchi, T. Takahashi, Y. Mekada, I. Ide, H. Murase, and Y. Tamatsu, “Estimation of traffic sign visibility considering temporal environmental changes for smart driver assistance,” in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2011, pp. 667–672. doi: 10.1109/IVS.2011.5940467.

[96] T. Gandhi and M. M. Trivedi, “Pedestrian collision avoidance systems: A survey of computer vision based recent studies,” in *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, Institute of Electrical and Electronics Engineers Inc., 2006, pp. 976–981. doi: 10.1109/itsc.2006.1706871.

[97] HD Lidar, “High Definitian Lidar HDL-64E.” Accessed: Sep. 22, 2022. [Online]. Available: https://hypertech.co.il/wp-content/uploads/2015/12/HDL-64E-Data-Sheet.pdf

[98] M. Zhang, R. Fu, W. Cheng, L. Wang, and Y. Ma, “An approach to segment and track-based pedestrian detection from four-layer laser scanner data,” *Sensors (Switzerland)*, vol. 19, no. 24, Dec. 2019, doi: 10.3390/s19245450.

[99] M. Kutila, P. Pyykonen, H. Holzhuter, M. Colomb, and P. Duthon, “Automotive LiDAR performance verification in fog and rain; Automotive LiDAR performance verification in fog and rain,” in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, Maui, HI, USA, 2018, pp. 1695–1701. doi: 10.1109/ITSC.2018.8569624.

[100] Intelligente Transport- und Verkehrssysteme und -dienste Niedersachsen e.V. and ITS automotive nord e.V., “AAET - Automatisierungssysteme, Assistenzsysteme und eingebettete Systeme für Transportmittel Beiträge zum gleichnamigen 17. Braunschweiger Symposium vom 10. und 11. Februar 2016, Stadthalle, Braunschweig,” ITS Niedersachsen, Braunschweig, 2016.

[101] M. Zhu, X. Wang, and Y. Wang, “Human-Like Autonomous Car-Following Model with Deep Reinforcement Learning,” *Transp Res Part C Emerg Technol*, vol. 97, pp. 348–368, Dec. 2018.

[102] I. Bae, J. Moon, J. Jhung, H. Suk, T. Kim, H. Park, J. Cha, J. Kim, D. Kim, and S. Kim, “Self-Driving like a Human driver instead of a Robocar: Personalized comfortable driving experience for autonomous vehicles,” *arXiv preprint arXiv:2001.03908*, Jan. 2020.

[103] J. Svensson, “Design of a portable steering wheel angle measurement system,” Dissertation, KTH, 2015.

[104] M. Moussa, A. Moussa, and N. El-Sheimy, “Steering angle assisted vehicular navigation using portable devices in GNSS-denied environments,” *Sensors (Switzerland)*, vol. 19, no. 7, Apr. 2019, doi: 10.3390/s19071618.

[105] L. Fridman, D. E. Brown, W. Angell, I. Abdić, B. Reimer, and H. Y. Noh, “Automated Synchronization of Driving Data Using Vibration and Steering Events,” *Pattern Recognit Lett*, vol. 75, pp. 9–15, May 2016, doi: 10.1016/j.patrec.2016.02.011.

[106] I. Mohamad, M. A. M. Ali, and M. Ismail, “Abnormal driving detection using real time global positioning system data,” in *2011 IEEE International Conference on Space Science and Communication: “Towards Exploring the Equatorial Phenomena”, IconSpace 2011 - Proceedings*, 2011, pp. 1–6. doi: 10.1109/IConSpace.2011.6015840.

[107] A. Aljaafreh, N. Alshabatat, and M. S. Najim Al-Din, “Driving style recognition using fuzzy logic,” in *2012 IEEE International Conference on Vehicular Electronics and Safety, ICVES 2012*, 2012, pp. 460–463. doi: 10.1109/ICVES.2012.6294318.

[108] I. Mohamad, M. A. Mohd. Ali, and M. Ismail, “Availability, reliability and accuracy of GPS signal in Bandar Baru Bangi for the determination of vehicle position and speed,” in *2009 International Conference on Space Science and Communication, IconSpace - Proceedings*, Dec. 2009, pp. 224–229. doi: 10.1109/ICONSPACE.2009.5352632.

[109] Z. Yan, L. Sun, T. Krajnik, and Y. Ruichek, “EU long-term dataset with multiple sensors for autonomous driving,” in *IEEE International Conference on Intelligent Robots and Systems*, Institute of Electrical and Electronics Engineers Inc., Oct. 2020, pp. 10697–10704. doi: 10.1109/IROS45743.2020.9341406.

[110] A. Agnoor, P. Atmakuri, and R. Sivanandan, “Analysis of Driving Behaviour through Instrumented Vehicles,” in *2022 14th International Conference on COMmunication Systems and NETworkS, COMSNETS 2022*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 700–706. doi: 10.1109/COMSNETS53615.2022.9668532.

[111] V. Vaitkus, P. Lengvenis, and G. Žylius, “Driving style classification using long-term accelerometer information,” in *2014 19th International Conference on Methods and Models in Automation and Robotics, MMAR 2014*, Institute of Electrical and Electronics Engineers Inc., Nov. 2014, pp. 641–644. doi: 10.1109/MMAR.2014.6957429.

[112] S. M. LaValle, A. Yershova, M. Katsev, and M. Antonov, “Head tracking for the Oculus Rift,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 187–194. doi: 10.1109/ICRA.2014.6906608.

[113] V. Ilci and C. Toth, “High definition 3D map creation using GNSS/IMU/LiDAR sensor integration to support autonomous vehicle navigation,” *Sensors (Switzerland)*, vol. 20, no. 3, Feb. 2020, doi: 10.3390/s20030899.

[114] K. Huang, B. Shi, X. Li, X. Li, S. Huang, and Y. Li, “Multi-modal sensor fusion for auto driving perception: A survey,” *arXiv preprint arXiv:2202.02703*, 2022.

[115] S. Jagannathan, M. Mody, J. Jones, P. Swami, and D. Poddar, “Multi-sensor fusion for Automated Driving: Selecting model and optimizing on Embedded platform,” *Electronic Imaging*, vol. 2018, no. 17, pp. 251–256, 2018.

[116] D. J. Yeong, G. Velasco-Hernandez, J. Barry, and J. Walsh, “Sensor and sensor fusion technology in autonomous vehicles: A review,” *Sensors*, vol. 21, no. 6, p. 2140, 2021.

[117] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, and V. Pratt, “Towards fully autonomous driving: Systems and algorithms,” in *2011 IEEE intelligent vehicles symposium (IV)*, IEEE, 2011, pp. 163–168.

[118] J. Schlosser, C. K. Chow, and Z. Kira, “Fusing lidar and images for pedestrian detection using convolutional neural networks,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2016, pp. 2198–2205.

[119] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, “Joint 3d proposal generation and object detection from view aggregation,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 1–8.

[120] M. Velas, M. Spanel, M. Hradis, and A. Herout, “CNN for IMU assisted odometry estimation using velodyne LiDAR,” in *2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, IEEE, 2018, pp. 71–77.

[121] O. Schumann, M. Hahn, J. Dickmann, and C. Wöhler, “Semantic segmentation on radar point clouds,” in *2018 21st International Conference on Information Fusion (FUSION)*, IEEE, 2018, pp. 2179–2186.

[122] Z. Wei, F. Zhang, S. Chang, Y. Liu, H. Wu, and Z. Feng, “Mmwave radar and vision fusion for object detection in autonomous driving: A review,” *Sensors*, vol. 22, no. 7, p. 2542, 2022.

[123] F. Nobis, M. Geisslinger, M. Weber, J. Betz, and M. Lienkamp, “A deep learning-based radar and camera sensor fusion architecture for object detection,” in *2019 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, IEEE, 2019, pp. 1–7.

[124] V. Lekic and Z. Babic, “Automotive radar and camera fusion using Generative Adversarial Networks,” *Computer Vision and Image Understanding*, vol. 184, pp. 1–8, 2019, doi: https://doi.org/10.1016/j.cviu.2019.04.002.

[125] W. Farag, “Multiple Road-Objects Detection and Tracking for Autonomous Driving,” *Journal of Engineering Research*, vol. 10, no. 1A, pp. 237–262, 2022.

[126] A. Asvadi, L. Garrote, C. Premebida, P. Peixoto, and U. J. Nunes, “Multimodal vehicle detection: fusing 3D-LIDAR and color camera data,” *Pattern Recognit Lett*, vol. 115, pp. 20–29, 2018.

[127] Y. Zhou and O. Tuzel, “Voxelnet: End-to-end learning for point cloud based 3d object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4490–4499.

[128] M. Menze, C. Heipke, and A. Geiger, “Joint 3d estimation of vehicles and scene flow,” *ISPRS annals of the photogrammetry, remote sensing and spatial information sciences*, vol. 2, p. 427, 2015.

[129] Y. Choi, N. Kim, S. Hwang, K. Park, J. S. Yoon, K. An, and I. S. Kweon, “KAIST multi-spectral day/night data set for autonomous and assisted driving,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 3, pp. 934–948, 2018.

[130] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, “Multi-view 3d object detection network for autonomous driving,” in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 1907–1915.

[131] M. Rezaei and R. Klette, “Look at the driver, look at the road: No distraction! no accident!,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 129–136.

[132] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, “Semantickitti: A dataset for semantic scene understanding of lidar sequences,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9297–9307.

[133] Z. Huang, C. Lv, Y. Xing, and J. Wu, “Multi-modal sensor fusion-based deep neural network for end-to-end autonomous driving with scene understanding,” *IEEE Sens J*, vol. 21, no. 10, pp. 11781–11790, 2020.

[134] Y. Yue, C. Zhao, R. Li, C. Yang, J. Zhang, M. Wen, Y. Wang, and D. Wang, “A hierarchical framework for collaborative probabilistic semantic mapping,” in *2020 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2020, pp. 9659–9665.

[135] C. Laugier, “Situation Awareness & Decision-making for Autonomous Driving,” in *IROS 2019-IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2019, pp. 1–25.

[136] J. Nine, S. Manoharan, and W. Hardt, “Concept of the comprehension level of situation awareness using an expert system,” in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, 2021, p. 012103.

[137] H. A. Ignatious, H. El-Sayed, M. A. Khan, and B. M. Mokhtar, “Analyzing Factors Influencing Situation Awareness in Autonomous Vehicles—A Survey,” *Sensors*, vol. 23, no. 8, p. 4075, 2023.

[138] J. Dong, S. Chen, Y. Li, R. Du, A. Steinfeld, and S. Labi, “Space-weighted information fusion using deep reinforcement learning: The context of tactical control of lane-changing autonomous vehicles and connectivity range assessment,” *Transp Res Part C Emerg Technol*, vol. 128, p. 103192, 2021.

[139] J. Zhang, Y. Liao, S. Wang, and J. Han, “Study on driving decision-making mechanism of autonomous vehicle based on an optimized support vector machine regression,” *Applied Sciences*, vol. 8, no. 1, p. 13, 2017.

[140] P. Hang, C. Lv, C. Huang, Y. Xing, Z. Hu, and J. Cai, “Human-like lane-change decision making for automated driving with a game theoretic approach,” in *2020 4th CAA International Conference on Vehicular Control and Intelligence (CVCI)*, IEEE, 2020, pp. 708–713.

[141] A. Prakash, K. Chitta, and A. Geiger, “Multi-modal fusion transformer for end-to-end autonomous driving,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7077–7087.

[142] M. Nagiub and T. Beuth, “Towards Depth Perception from Noisy Camera based Sensors for Autonomous Driving.,” in *VEHITS*, 2022, pp. 198–207.

[143] Z. Liu, Y. Cai, H. Wang, L. Chen, H. Gao, Y. Jia, and Y. Li, “Robust target recognition and tracking of self-driving cars with radar and camera information fusion under severe weather conditions,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6640–6653, 2021.

[144] H. F. Nweke, Y. W. Teh, G. Mujtaba, and M. A. Al-Garadi, “Data fusion and multiple classifier systems for human activity detection and health monitoring: Review and open research directions,” *Information Fusion*, vol. 46, pp. 147–170, 2019.

[145] Y. Jin, N. Varia, and C. Wang, “The Importance of Autonomous Driving Using 5G Technology,” *arXiv preprint arXiv:2108.08966*, 2021.

[146] M. Attaran, “The impact of 5G on the evolution of intelligent automation and industry digitization,” *J Ambient Intell Humaniz Comput*, pp. 1–17, 2021.

[147] D. Hetzer, M. Muehleisen, A. Kousaridas, S. Barmpounakis, S. Wendt, K. Eckert, A. Schimpe, J. Löfhede, and J. Alonso-Zarate, “5G connected and automated driving: use cases, technologies and trials in cross-border environments,” *EURASIP J Wirel Commun Netw*, vol. 2021, no. 1, p. 97, 2021, doi: 10.1186/s13638-021-01976-6.

[148] M. Hasenjäger, M. Heckmann, and H. Wersing, “A survey of personalization for advanced driver assistance systems,” *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 2, pp. 335–344, 2019.

[149] N. Lin, C. Zong, M. Tomizuka, P. Song, Z. Zhang, and G. Li, “An overview on study of identification of driver behavior characteristics for automotive control,” *Math Probl Eng*, vol. 2014, no. 1, p. 569109, 2014.

[150] W. Wang, J. Xi, and H. Chen, “Modeling and recognizing driver behavior based on driving data: A survey,” *Math Probl Eng*, vol. 2014, no. 1, p. 245641, 2014.

[151] X. Ma, “Driver Modeling based on computational intelligence approaches: exploaration and Modeling driver-following data collected by an instrumented vehicle,” Dissertation, KTH, 2006.

[152] R. E. Chandler, R. Herman, and E. W. Montroll, “Traffic dynamics: studies in car following,” *Oper Res*, vol. 6, no. 2, pp. 165–184, 1958.

[153] C. Preusse, “A driver model for online control of virtual cars,” in *Proceedings of the 2001 IEEE International Conference on Control Applications (CCA’01)(Cat. No. 01CH37204)*, IEEE, 2001, pp. 1174–1178.

[154] K. Lidstrom and T. Larsson, “Model-based estimation of driver intentions using particle filtering,” in *2008 11th International IEEE Conference on Intelligent Transportation Systems*, IEEE, 2008, pp. 1177–1182.

[155] P. G. Gipps, “A behavioural car-following model for computer simulation,” *Transportation research part B: methodological*, vol. 15, no. 2, pp. 105–111, 1981.

[156] C. Miyajima, Y. Nishiwaki, K. Ozawa, T. Wakita, K. Itou, K. Takeda, and F. Itakura, “Driver modeling based on driving behavior and its evaluation in driver identification,” *Proceedings of the IEEE*, vol. 95, no. 2, pp. 427–437, 2007.

[157] D. A. Reynolds and R. C. Rose, “Robust text-independent speaker identification using Gaussian mixture speaker models,” *IEEE transactions on speech and audio processing*, vol. 3, no. 1, pp. 72–83, 1995.

[158] M. Bando, K. Hasebe, A. Nakayama, A. Shibata, and Y. Sugiyama, “Dynamical model of traffic congestion and numerical simulation,” *Phys Rev E*, vol. 51, no. 2, p. 1035, 1995.

[159] C. Miyajima, P. Angkititrakul, and K. Takeda, “Behavior signal processing for vehicle applications,” *APSIPA Trans Signal Inf Process*, vol. 2, p. e2, 2013.

[160] H. Chen and F. Allgöwer, “A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability,” *Automatica*, vol. 34, no. 10, pp. 1205–1217, 1998.

[161] D. Hrovat, S. Di Cairano, H. E. Tseng, and I. V Kolmanovsky, “The development of model predictive control in automotive industry: A survey,” in *2012 IEEE International Conference on Control Applications*, IEEE, 2012, pp. 295–302.

[162] C. C. MacAdam, “An optimal preview control for linear systems,” *Journal of Dynamic Systems, Measurement and Control, ASME*, vol. 192, pp. 188–190, Sep. 1980.

[163] T. Qu, H. Chen, Y. Ji, H. Guo, and D. Cao, “Modeling Driver Steering Control Based on Stochastic Model Predictive Control,” in *2013 IEEE International Conference on Systems, Man, and Cybernetics*, 2013, pp. 3704–3709. doi: 10.1109/SMC.2013.631.

[164] T. Qu, H. Chen, D. Cao, H. Guo, and B. Gao, “Switching-Based Stochastic Model Predictive Control Approach for Modeling Driver Steering Skill,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 1, pp. 365–375, 2015, doi: 10.1109/TITS.2014.2334623.

[165] C. C. Macadam, “Understanding and Modeling the Human Driver,” *Vehicle System Dynamics*, vol. 40, no. 1–3, pp. 101–134, Aug. 2003, doi: 10.1076/vesd.40.1.101.15875.

[166] D. H. Weir and D. T. McRuer, “Dynamics of driver vehicle steering control,” *Automatica*, vol. 6, no. 1, pp. 87–98, 1970, doi: https://doi.org/10.1016/0005-1098(70)90077-4.

[167] D. McRuer, “Human dynamics in man-machine systems,” *Automatica*, vol. 16, no. 3, pp. 237–253, 1980, doi: https://doi.org/10.1016/0005-1098(80)90034-5.

[168] D. C. Gazis, R. Herman, and R. W. Rothery, “Nonlinear follow-the-leader models of traffic flow,” *Oper Res*, vol. 9, no. 4, pp. 545–567, 1961.

[169] S. Panwai and H. Dia, “Comparative evaluation of microscopic car-following behavior,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 3, pp. 314–325, 2005, doi: 10.1109/TITS.2005.853705.

[170] H. Zheng, J. Zhou, Q. Shao, and Y. Wang, “Investigation of a longitudinal and lateral lane-changing motion planning model for intelligent vehicles in dynamical driving environments,” *IEEE Access*, vol. 7, pp. 44783–44802, 2019.

[171] S. Schnelle, J. Wang, R. Jagacinski, and H. Su, “A feedforward and feedback integrated lateral and longitudinal driver model for personalized advanced driver assistance systems,” *Mechatronics*, vol. 50, pp. 177–188, 2018.

[172] S. Schnelle, J. Wang, H. Su, and R. Jagacinski, “A driver steering model with personalized desired path generation,” *IEEE Trans Syst Man Cybern Syst*, vol. 47, no. 1, pp. 111–120, 2016.

[173] S. Schnelle, J. Wang, H.-J. Su, and R. Jagacinski, “A personalizable driver steering model capable of predicting driver behaviors in vehicle collision avoidance maneuvers,” *IEEE Trans Hum Mach Syst*, vol. 47, no. 5, pp. 625–635, 2016.

[174] J. Ni, Y. Chen, Y. Chen, J. Zhu, D. Ali, and W. Cao, “A survey on theories and applications for self-driving cars based on deep learning methods,” *Applied Sciences*, vol. 10, no. 8, p. 2749, 2020.

[175] J. E. Naranjo, M. A. Sotelo, C. Gonzalez, R. Garcia, and T. De Pedro, “Using fuzzy logic in automated vehicle control,” *IEEE Intell Syst*, vol. 22, no. 1, pp. 36–45, 2007.

[176] M. Van Ly, S. Martin, and M. M. Trivedi, “Driver classification and driving style recognition using inertial sensors,” in *2013 IEEE intelligent vehicles symposium (IV)*, IEEE, 2013, pp. 1040–1045.

[177] C. Miyajima, Y. Nishiwaki, K. Ozawa, T. Wakita, K. Itou, K. Takeda, and F. Itakura, “Driver modeling based on driving behavior and its evaluation in driver identification,” *Proceedings of the IEEE*, vol. 95, no. 2, pp. 427–437, 2007.

[178] X. Zhang, X. Zhao, and J. Rong, “A study of individual characteristics of driving behavior based on hidden markov model,” *Sensors & Transducers*, vol. 167, no. 3, p. 194, 2014.

[179] A. Sathyanarayana, S. O. Sadjadi, and J. H. L. Hansen, “Leveraging sensor information from portable devices towards automatic driving maneuver recognition,” in *2012 15th International IEEE Conference on Intelligent Transportation Systems*, IEEE, 2012, pp. 660–665.

[180] M. Acosta and S. Kanarachos, “Teaching a vehicle to autonomously drift: A data-based approach using neural networks,” *Knowl Based Syst*, vol. 153, pp. 12–28, 2018.

[181] E. Aria, J. Olstam, and C. Schwietering, “Investigation of automated vehicle effects on driver’s behavior and traffic performance,” *Transportation research procedia*, vol. 15, pp. 761–770, 2016.

[182] L. Müller, M. Risto, and C. Emmenegger, “The social behavior of autonomous vehicles,” in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, 2016, pp. 686–689.

[183] P. Steven, “Google AI expert explains the challenge of debugging machine-learning systems,” Network World. Accessed: May 23, 2022. [Online]. Available: https://www.networkworld.com/article/951837/google-ai-expert-explains-the-challenge-of-debugging-machine-learning-systems.html

[184] Z. Wu, C. Li, J. Chen, and H. Gao, “Learning driving behavior for autonomous vehicles using deep learning based methods,” in *2019 IEEE 4th International Conference on Advanced Robotics and Mechatronics (ICARM)*, IEEE, 2019, pp. 905–910.

[185] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, “A survey of deep learning techniques for autonomous driving,” *J Field Robot*, vol. 37, no. 3, pp. 362–386, 2020.

[186] C. Y. Low, H. Zamzuri, and S. A. Mazlan, “Simple robust road lane detection algorithm,” in *2014 5th international conference on intelligent and advanced systems (ICIAS)*, Ieee, 2014, pp. 1–4.

[187] A. Saudi, J. Teo, M. H. A. Hijazi, and J. Sulaiman, “Fast lane detection with randomized hough transform,” in *2008 international symposium on information technology*, IEEE, 2008, pp. 1–5.

[188] X. Wang, X. Hua, F. Xiao, Y. Li, X. Hu, and P. Sun, “Multi-object detection in traffic scenes based on improved SSD,” *Electronics (Basel)*, vol. 7, no. 11, p. 302, 2018.

[189] X. Zhang, W. Yang, X. Tang, and J. Liu, “A fast learning method for accurate and robust lane detection using two-stage feature extraction with YOLO v3,” *Sensors*, vol. 18, no. 12, p. 4308, 2018.

[190] W. Yang, X. Zhang, Q. Lei, D. Shen, P. Xiao, and Y. Huang, “Lane position detection based on long short-term memory (LSTM),” *Sensors*, vol. 20, no. 11, p. 3115, 2020.

[191] A. D. Dumbuya, R. L. Wood, T. J. Gordon, and P. Thomas, “An agent-based traffic simulation framework to model intelligent virtual driver behaviour,” Jan. 2002.

[192] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, and J. Zhang, “End to end learning for self-driving cars,” *arXiv preprint arXiv:1604.07316*, 2016.

[193] Z. Chen and X. Huang, “End-to-end learning for lane keeping of self-driving cars,” in *2017 IEEE intelligent vehicles symposium (IV)*, IEEE, 2017, pp. 1856–1860.

[194] Y. Pan, C.-A. Cheng, K. Saigol, K. Lee, X. Yan, E. Theodorou, and B. Boots, “Agile autonomous driving using end-to-end deep imitation learning,” *arXiv preprint arXiv:1709.07174*, 2017.

[195] J. Kocić, N. Jovičić, and V. Drndarević, “An end-to-end deep neural network for autonomous driving designed for embedded automotive platforms,” *Sensors*, vol. 19, no. 9, p. 2064, 2019.

[196] S. Yang, W. Wang, C. Liu, W. Deng, and J. K. Hedrick, “Feature analysis and selection for training an end-to-end autonomous vehicle controller using deep learning approach,” in *2017 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2017, pp. 1033–1038.

[197] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine, “Continuous deep q-learning with model-based acceleration,” in *International conference on machine learning*, PMLR, 2016, pp. 2829–2838.

[198] Z. Wu, C. Shen, and A. Van Den Hengel, “Wider or deeper: Revisiting the resnet model for visual recognition,” *Pattern Recognit*, vol. 90, pp. 119–133, 2019.

[199] S. Ha and S. Choi, “Convolutional neural networks for human activity recognition using multiple accelerometer and gyroscope sensors,” in *2016 international joint conference on neural networks (IJCNN)*, IEEE, 2016, pp. 381–388.

[200] Y. Yu, X. Si, C. Hu, and J. Zhang, “A review of recurrent neural networks: LSTM cells and network architectures,” *Neural Comput*, vol. 31, no. 7, pp. 1235–1270, 2019.

[201] C. Pan, H. Cao, W. Zhang, X. Song, and M. Li, “Driver activity recognition using spatial-temporal graph convolutional LSTM networks with attention mechanism,” *IET Intelligent Transport Systems*, vol. 15, no. 2, pp. 297–307, 2021.

[202] M. W. Gomaa, R. O. Mahmoud, and A. M. Sarhan, “A CNN-LSTM-based Deep Learning Approach for Driver Drowsiness Prediction,” *Journal of Engineering Research*, vol. 6, no. 3, pp. 59–70, 2022.

[203] N. Jain and S. Mittal, “Review of Computational Techniques for Modelling Eco-Safe Driving Behavior,” *International Journal of Automotive and Mechanical Engineering*, vol. 20, no. 2, pp. 10422–10440, 2023.

[204] O. Olabiyi, E. Martinson, V. Chintalapudi, and R. Guo, “Driver action prediction using deep (bidirectional) recurrent neural network,” *arXiv preprint arXiv:1706.02257*, 2017.

[205] F. J. Ordóñez and D. Roggen, “Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition,” *Sensors*, vol. 16, no. 1, p. 115, 2016.

[206] S. E. Forward, “The theory of planned behaviour: The role of descriptive norms and past behaviour in the prediction of drivers’ intentions to violate,” *Transp Res Part F Traffic Psychol Behav*, vol. 12, no. 3, pp. 198–207, 2009.

[207] M. Bojarski, C. Chen, J. Daw, A. Değirmenci, J. Deri, B. Firner, B. Flepp, S. Gogri, J. Hong, and L. Jackel, “The NVIDIA pilotnet experiments,” *arXiv preprint arXiv:2010.08776*, 2020.

[208] H. Xu, Y. Gao, F. Yu, and T. Darrell, “End-to-end learning of driving models from large-scale video datasets,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2174–2182.

[209] P. Selvaraj, K. P. Santhosam, and S. Nagarajan, “Lane detection and localization using hybrid deep neural network model,” in *AIP Conference Proceedings*, AIP Publishing, 2023.

[210] M. Gulzar, Y. Muhammad, and N. Muhammad, “A survey on motion prediction of pedestrians and vehicles for autonomous driving,” *IEEE Access*, vol. 9, pp. 137957–137969, 2021.

[211] S. Cheng, B. Yang, Z. Wang, and K. Nakano, “Spatio-temporal image representation and deep-learning-based decision framework for automated vehicles,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 24866–24875, 2022.

[212] U. M. Gidado, H. Chiroma, N. Aljojo, S. Abubakar, S. I. Popoola, and M. A. Al-Garadi, “A survey on deep learning for steering angle prediction in autonomous vehicles,” *IEEE Access*, vol. 8, pp. 163797–163817, 2020.

[213] A. O. Ly and M. Akhloufi, “Learning to drive by imitation: An overview of deep behavior cloning methods,” *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 2, pp. 195–209, 2020.

[214] A. Khanum, C.-Y. Lee, and C.-S. Yang, “Involvement of Deep Learning for Vision Sensor-based Autonomous Driving Control: A Review,” *IEEE Sens J*, 2023.

[215] A. Tampuu, T. Matiisen, M. Semikin, D. Fishman, and N. Muhammad, “A survey of end-to-end driving: Architectures and training methods,” *IEEE Trans Neural Netw Learn Syst*, vol. 33, no. 4, pp. 1364–1384, 2020.

[216] R. Valiente, M. Zaman, S. Ozer, and Y. P. Fallah, “Controlling steering angle for cooperative self-driving vehicles utilizing CNN and LSTM-based deep networks,” in *2019 IEEE intelligent vehicles symposium (IV)*, IEEE, 2019, pp. 2423–2428.

[217] H. M. Eraqi, M. N. Moustafa, and J. Honer, “End-to-end deep learning for steering autonomous vehicles considering temporal dependencies,” *arXiv preprint arXiv:1710.03804*, 2017.

[218] S. Hecker, D. Dai, and L. Van Gool, “End-to-end learning of driving models with surround-view cameras and route planners,” in *Proceedings of the european conference on computer vision (eccv)*, 2018, pp. 435–453.

[219] P. Qin, H. Li, Z. Li, W. Guan, and Y. He, “A CNN-LSTM car-following model considering generalization ability,” *Sensors*, vol. 23, no. 2, p. 660, 2023.

[220] W. Vijitkunsawat and P. Chantngarm, “comparison of machine learning algorithm’s on self-driving car navigation using Nvidia Jetson Nano,” in *2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, IEEE, 2020, pp. 201–204.

[221] Z. Yang, Y. Zhang, J. Yu, J. Cai, and J. Luo, “End-to-end multi-modal multi-task vehicle control for self-driving cars with visual perceptions,” in *2018 24th international conference on pattern recognition (ICPR)*, IEEE, 2018, pp. 2289–2294.

[222] Z. Gu, Z. Li, X. Di, and R. Shi, “An LSTM-based autonomous driving model using a waymo open dataset,” *Applied Sciences*, vol. 10, no. 6, p. 2046, 2020.

[223] J. Zhang, Z. Wu, F. Li, C. Xie, T. Ren, J. Chen, and L. Liu, “A deep learning framework for driving behavior identification on in-vehicle CAN-BUS sensor data,” *Sensors*, vol. 19, no. 6, p. 1356, 2019.

[224] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014.

[225] R. Dey and F. M. Salem, “Gate-variants of gated recurrent unit (GRU) neural networks,” in *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*, IEEE, 2017, pp. 1597–1600.

[226] X. Zhao, M. Qi, Z. Liu, S. Fan, C. Li, and M. Dong, “End-to-end autonomous driving decision model joined by attention mechanism and spatiotemporal features,” *IET Intelligent Transport Systems*, vol. 15, no. 9, pp. 1119–1130, 2021.

[227] E. Perot, M. Jaritz, M. Toromanoff, and R. De Charette, “End-to-end driving in a realistic racing game with deep reinforcement learning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 3–4.

[228] M. Jaritz, R. De Charette, M. Toromanoff, E. Perot, and F. Nashashibi, “End-to-end race driving with deep reinforcement learning,” in *2018 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2018, pp. 2070–2075.

[229] K. Makantasis, M. Kontorinaki, and I. Nikolos, “Deep reinforcement-learning-based driving policy for autonomous road vehicles,” *IET Intelligent Transport Systems*, vol. 14, no. 1, pp. 13–24, 2020.

[230] H. Van Hasselt, A. Guez, and D. Silver, “Deep reinforcement learning with double q-learning,” in *Proceedings of the AAAI conference on artificial intelligence*, 2016.

[231] X. Wang, C. Wu, J. Xue, and Z. Chen, “A method of personalized driving decision for smart car based on deep reinforcement learning,” *Information*, vol. 11, no. 6, p. 295, 2020.

[232] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” *arXiv preprint arXiv:1509.02971*, 2015.

[233] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, “Concrete Problems in AI Safety,” 2016, doi: 10.48550/ARXIV.1606.06565.

[234] Z. Huang, C. Lv, and J. Wu, “Modeling Human Driving Behavior in Highway Scenario using Inverse Reinforcement Learning,” *CoRR*, vol. abs/2010.03118, 2020, [Online]. Available: https://arxiv.org/abs/2010.03118

[235] B. Piot, M. Geist, and O. Pietquin, “Bridging the Gap Between Imitation Learning and Inverse Reinforcement Learning,” *IEEE Trans Neural Netw Learn Syst*, vol. 28, no. 8, pp. 1814–1826, 2017, doi: 10.1109/TNNLS.2016.2543000.

[236] S. E. Shladover and C. Nowakowski, “Regulatory challenges for road vehicle automation: Lessons from the California experience,” *Transp Res Part A Policy Pract*, vol. 122, pp. 125–133, 2019, doi: https://doi.org/10.1016/j.tra.2017.10.006.

[237] G. Audi and A. G. Volkswagen, “The PEGASUS Method.” Accessed: Jan. 19, 2022. [Online]. Available: https://www.pegasusprojekt.de/en/pegasus-method

[238] S. Lee, “Design and Verification Standard for Safety and Cybersecurity of Autonomous Cars: ISO/TR 4804,” *Journal of IKEEE*, vol. 25, no. 3, pp. 571–577, 2021.

[239] S. S. Shadrin and A. A. Ivanova, “Analytical review of standard Sae J3016 «taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles» with latest updates,” *Avtomobil’. Doroga. Infrastruktura.*, no. 3 (21), p. 10, 2019.

[240] B. Nahmias-Biran, J. B. Oke, N. Kumar, C. Lima Azevedo, and M. Ben-Akiva, “Evaluating the impacts of shared automated mobility on-demand services: an activity-based accessibility approach,” *Transportation (Amst)*, vol. 48, no. 4, pp. 1613–1638, 2021, doi: 10.1007/s11116-020-10106-y.

[241] R. Xu, H. Xiang, X. Han, X. Xia, Z. Meng, C.-J. Chen, C. Correa-Jullian, and J. Ma, “The OpenCDA Open-Source Ecosystem for Cooperative Driving Automation Research,” *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 4, pp. 2698–2711, 2023, doi: 10.1109/TIV.2023.3244948.

[242] P. Koopman and M. Wagner, “Toward a framework for highly automated vehicle safety validation,” SAE Technical Paper, 2018.

[243] A. J. Alnaser, M. I. Akbas, A. Sargolzaei, and R. Razdan, “Autonomous vehicles scenario testing framework and model of computation,” *SAE International Journal of Connected and Automated Vehicles*, vol. 2, no. 4, 2019.

[244] S. Behere and M. Törngren, “A functional reference architecture for autonomous driving,” *Inf Softw Technol*, vol. 73, pp. 136–150, 2016.

[245] C. Neurohr, L. Westhofen, M. Butz, M. H. Bollmann, U. Eberle, and R. Galbas, “Criticality analysis for the verification and validation of automated vehicles,” *IEEE Access*, vol. 9, pp. 18016–18041, 2021.

[246] A. Huang, X. Xing, T. Zhou, and J. Chen, “A safety analysis and verification framework for autonomous vehicles based on the identification of triggering events,” SAE Technical Paper, 2021.

[247] Y. Ma, C. Sun, J. Chen, D. Cao, and L. Xiong, “Verification and validation methods for decision-making and planning of automated vehicles: A review,” *IEEE Transactions on Intelligent Vehicles*, vol. 7, no. 3, pp. 480–498, 2022.

[248] M. Kamali, L. A. Dennis, O. McAree, M. Fisher, and S. M. Veres, “Formal verification of autonomous vehicle platooning,” *Sci Comput Program*, vol. 148, pp. 88–106, 2017.

[249] T. Bein, H. Atzrodt, R. Bartolozzi, S. Kupjetz, J. Millitzer, J. Nuffer, M. Rauschenbach, and G. Stoll, “Verification and validation of automated driving systems utilizing

probabilistic FMEA and simulation approaches," *Transportation research procedia*, vol. 72, pp. 470–477, 2023.

[250] International Organization for Standardization, "Pas 21448-road vehicles-safety of the intended functionality," 2019, *Geneva, Switzerland*.

[251] International Organization for Standardization, "Road Vehicles — Test Scenarios for Automated Driving Systems — Vocabulary," 2022, *Geneva, Switzerland*.

[252] International Organization for Standardization, "Road Vehicles — Test Scenarios for Automated Driving Systems — Scenario Based Safety Evaluation Framework," 2022, *Geneva, Switzerland*.

[253] International Organization for Standardization, "Road Vehicles — Test scenarios for automated driving systems — Specification for operational design domain," 2023, *Geneva, Switzerland*.

[254] British Standards Institution, "Formalised Natural Language Description of Scenarios for Automated Driving Systems – Specification," 2023, *United Kingdom*.

[255] International Organization for Standardization, "Road Vehicles — Test scenarios for automated driving systems — Scenario categorization," 2024, *Geneva, Switzerland*.

[256] National Science and Technology Development Agency (NSTDA), "The First Autonomous Vehicle Test in Thailand." Accessed: May 14, 2024. [Online]. Available: https://www.nstda.or.th/en/news/news-years-2020/the-first-autonomous-vehicle-test-in-thailand.html

[257] R. Mariani and K. Greb, "Recent Advances and Trends on Automotive Safety : (invited)," in *2022 IEEE International Reliability Physics Symposium (IRPS)*, 2022, pp. 7C.1-1-7C.1–6. doi: 10.1109/IRPS48227.2022.9764484.

[258] Euro NCAP, "The European New Car Assessment Programme| Euro NCAP." Accessed: Nov. 01, 2023. [Online]. Available: https://www.euroncap.com/en

[259] International Organization for Standardization, "ISO 3888-1:2018," *Geneva, Switzerland*. Accessed: May 11, 2022. [Online]. Available: https://www.iso.org/standard/67973.html

[260] S. D. Mazor and K. B. Matthews, "Vehicle testing to SAE acceleration and braking recommended practices," *SAE transactions*, pp. 612–639, 1986.

[261] International Organization for Standardization, "ISO 4138:2021," *Geneva, Switzerland*. Accessed: May 11, 2022. [Online]. Available: https://www.iso.org/standard/81710.html

[262] International Organization for Standardization, "ISO 7401:2011," *Geneva, Switzerland*. Accessed: May 11, 2022. [Online]. Available: https://www.iso.org/standard/54144.html

[263] International Organization for Standardization, "ISO 15037-1:2019," *Geneva, Switzerland*. Accessed: May 11, 2022. [Online]. Available: https://www.iso.org/standard/70164.html

[264] International Organization for Standardization, “ISO 22737:2021 - Intelligent transport systems — Low-speed automated driving (LSAD) systems for predefined routes — Performance requirements, system requirements and performance test procedures,” *Geneva, Switzerland*. Accessed: May 11, 2022. [Online]. Available: https://www.iso.org/standard/73767.html

[265] S. N. A. Halimi, S. T. Rasmana, D. Adiputra, W. J. Yahya, M. A. A. Rahman, M. H. M. Ariff, N. A. Husain, and K. A. A. Kassim, “A review of safety test methods for new car assessment program in Southeast Asian countries,” *Open Engineering*, vol. 13, no. 1, p. 20220501, 2023.

[266] M. Grieves, “Digital twin: manufacturing excellence through virtual factory replication,” *White paper*, vol. 1, no. 2014, pp. 1–7, 2014.

[267] E. H. Glaessgen and D. Stargel, “The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles,” 2012. [Online]. Available: https://api.semanticscholar.org/CorpusID:110572580

[268] D. Liu, H. Huang, B. Wang, T. Zhou, and S. Luo, “Operation paradigm for remanufacturing shop-floor based on digital twin,” *Jisuanji Jicheng Zhizao Xitong/Computer Integrated Manufacturing Systems, CIMS*, vol. 25, no. 6, pp. 1515–1527, 2019, doi: 10.13196/j.cims.2019.06.019.

[269] M. Grieves and J. Vickers, “Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems,” *Transdisciplinary perspectives on complex systems: New findings and approaches*, pp. 85–113, 2017.

[270] F. Tao and M. Zhang, “Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing,” *IEEE Access*, vol. 5, pp. 20418–20427, 2017, doi: 10.1109/ACCESS.2017.2756069.

[271] C. Zhuang, J. Liu, H. Xiong, X. Ding, S. Liu, and G. Weng, “Connotation, architecture and trends of product digital twin,” *Jisuanji Jicheng Zhizao Xitong/Computer Integrated Manufacturing Systems, CIMS*, vol. 23, no. 4, pp. 753–768, 2017, doi: 10.13196/j.cims.2017.04.010.

[272] K. M. Alam and A. El Saddik, “C2PS: A digital twin architecture reference model for the cloud-based cyber-physical systems,” *IEEE Access*, vol. 5, pp. 2050–2062, 2017, doi: 10.1109/ACCESS.2017.2657006.

[273] B. Björnsson, C. Borrebaeck, N. Elander, T. Gasslander, D. R. Gawel, M. Gustafsson, R. Jörnsten, E. J. Lee, X. Li, and S. Lilja, “Digital twins to personalize medicine,” *Genome Med*, vol. 12, pp. 1–4, 2020.

[274] W. Jia, W. Wang, and Z. Zhang, “From simple digital twin to complex digital twin Part I: A novel modeling method for multi-scale and multi-scenario digital twin,” *Advanced Engineering Informatics*, vol. 53, p. 101706, 2022, doi: https://doi.org/10.1016/j.aei.2022.101706.

[275] J. Guo, H. Hong, K. Zhong, X. Liu, and Y. Guo, “Production Management and Control Method of Aerospace Manufacturing Workshops Based on Digital Twin,” *Zhongguo*

Jixie Gongcheng/China Mechanical Engineering, vol. 31, no. 7, pp. 808–814, 2020, doi: 10.3969/j.issn.1004-132X.2020.07.006.

[276] P. Xu, W. Chen, L. Liao, Z. Zhang, and Y. Feng, "Research on digital workshop of ship pipe machining based on digital twin," *Ship Science and Technology*, vol. 41, no. 15, pp. 139–144, 2019.

[277] C. Fan, C. Zhang, A. Yahja, and A. Mostafavi, "Disaster City Digital Twin: A vision for integrating artificial and human intelligence for disaster management," *Int J Inf Manage*, vol. 56, p. 102049, 2021, doi: https://doi.org/10.1016/j.ijinfomgt.2019.102049.

[278] E. J. Tuegel, A. R. Ingraffea, T. G. Eason, and S. M. Spottswood, "Reengineering Aircraft Structural Life Prediction Using a Digital Twin," *International Journal of Aerospace Engineering*, vol. 2011, no. 1, p. 154798, Jan. 2011, doi: https://doi.org/10.1155/2011/154798.

[279] P. Kasey, "Strategic Technology Trends for 2020." Accessed: Sep. 13, 2023. [Online]. Available: https://www. gartner. com/smarterwithgartner/gartner-top-10-strategic-technology-trends-for-2020

[280] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Practical search techniques in path planning for autonomous driving," *Ann Arbor*, vol. 1001, no. 48105, pp. 18–80, 2008.

[281] B. Yu, W. Hu, L. Xu, J. Tang, S. Liu, and Y. Zhu, "Building the computing system for autonomous micromobility vehicles: Design constraints and architectural optimizations," in *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, IEEE, 2020, pp. 1067–1081.

[282] S. Singh, M. Weeber, and K.-P. Birke, "Advancing digital twin implementation: A toolbox for modelling and simulation," *Procedia CIRP*, vol. 99, pp. 567–572, 2021.

[283] M. J. Kaur, V. P. Mishra, and P. Maheshwari, "The convergence of digital twin, IoT, and machine learning: transforming data into action," *Digital twin technologies and smart cities*, pp. 3–17, 2020.

[284] G. Bhatti, H. Mohan, and R. R. Singh, "Towards the future of smart electric vehicles: Digital twin technology," *Renewable and Sustainable Energy Reviews*, vol. 141, p. 110801, 2021.

[285] B. Yu, C. Chen, J. Tang, S. Liu, and J.-L. Gaudiot, "Autonomous vehicles digital twin: A practical paradigm for autonomous driving system development," *Computer (Long Beach Calif)*, vol. 55, no. 9, pp. 26–34, 2022.

[286] J. Dong, Q. Xu, J. Wang, C. Yang, M. Cai, C. Chen, Y. Liu, J. Wang, and K. Li, "Mixed cloud control testbed: Validating vehicle-road-cloud integration via mixed digital twin," *IEEE Transactions on Intelligent Vehicles*, 2023.

[287] A. Niaz, M. U. Shoukat, Y. Jia, S. Khan, F. Niaz, and M. U. Raza, "Autonomous driving test method based on digital twin: A survey," in *2021 International Conference on Computing, Electronic and Electrical Engineering (ICE Cube)*, IEEE, 2021, pp. 1–7.

[288] C. Steinmetz, G. N. Schroeder, A. Rettberg, R. N. Rodrigues, and C. E. Pereira, "Enabling and supporting car-as-a-service by digital twin modeling and deployment," in *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, 2021, pp. 428–433.

[289] G. J. Tsinarakis, P. S. Spanoudakis, G. Arabatzis, N. C. Tsourveloudis, and L. Doitsidis, "Implementation of a petri-net based digital twin for the development procedure of an electric vehicle," in *2020 28th Mediterranean Conference on Control and Automation (MED)*, IEEE, 2020, pp. 862–867.

[290] Z. Wang, X. Liao, X. Zhao, K. Han, P. Tiwari, M. J. Barth, and G. Wu, "A digital twin paradigm: Vehicle-to-cloud based advanced driver assistance systems," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, IEEE, 2020, pp. 1–6.

[291] C. Yang, J. Dong, Q. Xu, M. Cai, H. Qin, J. Wang, and K. Li, "Multi-vehicle experiment platform: A digital twin realization method," in *2022 IEEE/SICE International Symposium on System Integration (SII)*, IEEE, 2022, pp. 705–711.

[292] M. Milton, C. De La O, H. L. Ginn, and A. Benigni, "Controller-embeddable probabilistic real-time digital twins for power electronic converter diagnostics," *IEEE Trans Power Electron*, vol. 35, no. 9, pp. 9850–9864, 2020.

[293] S. M. Sadek, "Power Electronic Converter Topologies used in Electric Vehicles," Dissertation (PhD), Shams University, 2016.

[294] M. Hirota, T. Baba, X. Zheng, K. Nii, S. Ohashi, T. Ariyoshi, and H. Fujikawa, "Development of new AC/DC converter for PHEVs and EVs," *Sei Technical Review*, no. 73, 2011.

[295] S. Korotunov, G. Tabunshchyk, and V. Okhmak, "Genetic algorithms as an optimization approach for managing electric vehicles charging in the smart grid.," in *CMIS*, 2020, pp. 184–198.

[296] Y. Liu, Z. Wang, K. Han, Z. Shou, P. Tiwari, and J. H. L. Hansen, "Sensor fusion of camera and cloud digital twin information for intelligent vehicles," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2020, pp. 182–187.

[297] R. Desai, A. Guha, and P. Seshu, "A comparison of quarter, half and full car models for predicting vibration attenuation of an occupant in a vehicle," *Journal of Vibration Engineering & Technologies*, vol. 9, pp. 983–1001, 2021.

[298] S. Palli, R. C. Sharma, and P. P. D. Rao, "Dynamic behaviour of a 7 DoF passenger car model," *International Journal of Vehicle Structures & Systems*, vol. 9, no. 1, p. 57, 2017.

[299] S. Yang, Y. Lu, and S. Li, "An overview on vehicle dynamics," *Int J Dyn Control*, vol. 1, no. 4, pp. 385–395, 2013, doi: 10.1007/s40435-013-0032-y.

[300] J. D. Setiawan, M. Safarudin, and A. Singh, "Modeling, simulation and validation of 14 DOF full vehicle model," in *International Conference on Instrumentation, Communication, Information Technology, and Biomedical Engineering 2009*, IEEE, 2009, pp. 1–6.

[301] R. Rajamani, *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[302] S. B. Abbas and I. Youn, "Performance Improvement of Active Suspension System Collaborating with an Active Airfoil Based on a Quarter-Car Model," *Vehicles*, vol. 6, no. 3, pp. 1268–1283, 2024.

[303] H. Pacejka, *Tire and vehicle dynamics*. Elsevier, 2005.

[304] E. Fiala, "Seitenkraffe am rollenden Luftreifen," *Z. VDI*, vol. 96, no. 29, 1954.

[305] W. Hirschberg, G. Rill, and H. Weinfurter, "Tire model tmeasy," *Vehicle System Dynamics*, vol. 45, no. S1, pp. 101–119, 2007.

[306] J. Svendenius and B. Wittenmark, "Brush tire model with increased flexibility," in *2003 European Control Conference (ECC)*, 2003, pp. 1863–1868. doi: 10.23919/ECC.2003.7085237.

[307] M. Gipser, "FTire–the tire simulation model for all applications related to vehicle dynamics," *Vehicle System Dynamics*, vol. 45, no. S1, pp. 139–151, 2007.

[308] T. Gillespie, "Fundamentals of Vehicle Dynamics. Society of Automotive Engineers, 1992," *Google Scholar Google Scholar Cross Ref Cross Ref*, 1992.

[309] M. Short, M. J. Pont, and Q. Huang, "Simulation of vehicle longitudinal dynamics," *Safety and Reliability of Distributed Embedded Systems*, pp. 1–4, 2004.

[310] K. Hudha, Z. A. Kadir, M. R. Said, and H. Jamaluddin, "Modelling, validation and roll moment rejection control of pneumatically actuated active roll control for improving vehicle lateral dynamics performance," *International Journal of Engineering Systems Modelling and Simulation*, vol. 1, no. 2–3, pp. 122–136, 2009.

[311] M. Short and M. J. Pont, "Assessment of high-integrity embedded automotive control systems using hardware in the loop simulation," *Journal of Systems and Software*, vol. 81, no. 7, pp. 1163–1183, 2008.

[312] Z. A. Kadir, K. Hudha, F. Ahmad, M. F. Abdullah, A. R. Norwazan, M. Y. Mohd Fazli, A. J. Khalid, and M. Gunasilan, "Verification of 14DOF full vehicle model based on steering wheel input," *Applied Mechanics and Materials*, vol. 165, pp. 109–113, 2012.

[313] V. R. Aparow, F. Ahmad, K. Hudha, and H. Jamaluddin, "Modelling and PID control of antilock braking system with wheel slip reduction to improve braking performance," *Int J Veh Saf*, vol. 6, no. 3, pp. 265–296, Jan. 2013, doi: 10.1504/IJVS.2013.055025.

[314] W. F. Milliken, D. L. Milliken, and L. D. Metz, *Race car vehicle dynamics*, vol. 400. SAE international Warrendale, 1995.

[315] V. R. Aparow, "Enhancements of armored vehicle stability using adaptive controller," Doctoral dissertation, Universiti Pertahanan Nasional Malaysia, 2018.

[316] F. Ahmad, K. Hudha, and M. H. Harun, "Pneumatically actuated active suspension system for reducing vehicle dive and squat," *Jurnal mekanikal*, vol. 28, no. 1, 2009.

[317] V. Rau Aparow, K. Hudha, M. Mohamad Hamdan Megat Ahmad, and H. Jamaluddin, "Development and verification of a 9-DOF armored vehicle model in the lateral and

longitudinal directions," *Jurnal Teknologi (Sciences & Engineering)*, vol. 78, no. 6, pp. 2180–3722, 2016, [Online]. Available: www.jurnalteknologi.utm.my

[318] Y. Zhang, Z. Guo, and Z. Sun, "Driving simulator validity of driving behavior in work zones," *J Adv Transp*, vol. 2020, no. 1, p. 4629132, 2020.

[319] S. H. Li and J. Y. Ren, "Driver Steering Control and Full Vehicle Dynamics Study Based on a Nonlinear Three-Directional Coupled Heavy-Duty Vehicle Model," *Math Probl Eng*, vol. 2014, no. 1, p. 352374, 2014.

[320] C. Vastrad, "Performance analysis of neural network models for oxazolines and oxazoles derivatives descriptor dataset," *arXiv preprint arXiv:1312.2853*, 2013.

[321] S. D. Mazor and K. B. Matthews, "Vehicle testing to SAE acceleration and braking recommended practices," *SAE transactions*, pp. 612–639, 1986.

[322] E. P. Ping, K. Hudha, and H. Jamaluddin, "Hardware-in-the-loop simulation of automatic steering control for double lane change and sine steer manoeuvres," *International journal of vehicle autonomous systems*, vol. 10, no. 1–2, pp. 67–104, 2012.

[323] W. Liu, Y. Gong, H. Wu, J. Zhai, and J. Jin, "Memory-Centric Communication Mechanism for Real-time Autonomous Navigation Applications," in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Aug. 2020. doi: 10.1145/3404397.3404459.

[324] Q. Zhang, H. Zhong, J. Cui, L. Ren, and W. Shi, "AC4AV: A Flexible and Dynamic Access Control Framework for Connected and Autonomous Vehicles," *IEEE Internet Things J*, vol. 8, no. 3, pp. 1946–1958, Feb. 2021, doi: 10.1109/JIOT.2020.3016961.

[325] M. Elias, "Performance Optimization Analysis of Robotic Operating System ROS Communication," *International Journal of Youth Science and Technology Innovation*, vol. 1, no. 1, 2023, [Online]. Available: https://robots.ros.org/

[326] World Health Organization, "Managing speed," World Health Organization, 2017. Accessed: Sep. 02, 2022. [Online]. Available: https://www.who.int/publications/i/item/managing-speed

[327] M. Garrosa, E. Olmeda, S. F. Del Toro, and V. Díaz, "Holistic vehicle instrumentation for assessing driver driving styles," *Sensors*, vol. 21, no. 4, pp. 1–28, Feb. 2021, doi: 10.3390/s21041427.

[328] Digital News Asia, "Now it gets real – Malaysia approves first autonomous vehicle testing routes in Cyberjaya by Futurise and Ministry of Transport," Digital News Asia. Accessed: Sep. 22, 2022. [Online]. Available: https://www.digitalnewsasia.com/digital-economy/now-it-gets-real-malaysia-approves-first-autonomous-vehicle-testing-routes-cyberjaya

[329] D. G. Zill, *Advanced engineering mathematics*. Jones & Bartlett Learning, 2020.

[330] Z. 'Timothy, "Glossary of Off-Road Terminology," CARiD. Accessed: Sep. 22, 2022. [Online]. Available: https://www.carid.com/articles/glossary-of-off-road-terminology.html

[331] K. Cammaerts, P. Morse, and K. Kidera, “Improving Performance through the Use of Driver-in-the-Loop Simulations,” *ATZ worldwide*, vol. 121, no. 1, pp. 52–57, 2019, doi: 10.1007/s38311-018-0198-1.

[332] A. K. Madhava Prakash, A. Garje Mohankumar, C. R. Misquith, D. P. Ganatra, I. A. K. Soudagar, and J. Johnsson, “Developing a Solution to Utilize Vehicle Dynamics Simulation Tools with a Motion Driving Simulator,” 2021.

[333] D. Ojados Gonzalez, B. Martin-Gorriz, I. Ibarra Berrocal, A. Macian Morales, G. Adolfo Salcedo, and B. Miguel Hernandez, “Development and assessment of a tractor driving simulator with immersive virtual reality for training to avoid occupational hazards,” *Comput Electron Agric*, vol. 143, pp. 111–118, 2017, doi: https://doi.org/10.1016/j.compag.2017.10.008.

[334] J. Pak and U. Maoz, “Compact 3 DOF driving simulator using immersive virtual reality,” *arXiv preprint arXiv:1909.05833*, 2019.

[335] Q. C. Ihemedu-Steinke, R. Erbach, P. Halady, G. Meixner, and M. Weber, “Virtual reality driving simulator based on head-mounted displays,” *Automotive User Interfaces: Creating Interactive Experiences in the Car*, pp. 401–428, 2017.

[336] T. Kersten, G. Ungemach, B. Schick, M. Böhle, M. Martynkewicz, and N. Harendza, “Study to Assess the Controllability after Chassis Component Damages on the Dynamic Driving Simulator,” in *12th International Munich Chassis Symposium 2021*, P. Pfeffer, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2022, pp. 11–42.

[337] A. Betz, A. Butry, P. Junietz, P. Wagner, and H. Winner, “Driving dynamics control of a wheeled mobile driving simulator utilizing an omnidirectional motion base for urban traffic simulation,” *Future Active Safety Technology toward zero-traffic-accident (FASTzero)*, vol. 22, p. 26, 2013.

[338] D. W. Carruth, C. Goodin, L. Dabbiru, N. Scherrer, M. N. Moore, C. H. Hudson, L. D. Cagle, and P. Jayakumar, “Comparing real and simulated performance for an off-road autonomous ground vehicle in obstacle avoidance,” *J Field Robot*, vol. 41, no. 3, pp. 798–810, May 2024, doi: https://doi.org/10.1002/rob.22289.

[339] A. Stocco, B. Pulfer, and P. Tonella, “Mind the gap! A study on the transferability of virtual versus physical-world testing of autonomous driving systems,” *IEEE Transactions on Software Engineering*, vol. 49, no. 4, pp. 1928–1940, 2022.

[340] F. C. Newman, A. C. Biondo, M. D. Mandelberg, C. C. Matthews, and J. R. Rottier, “Enhancing realism in computer simulations: environmental effects,” *Johns Hopkins APL Tech Dig*, vol. 23, no. 4, pp. 443–453, 2002.

[341] S. Pouyanfar, M. Saleem, N. George, and S.-C. Chen, “Roads: Randomization for obstacle avoidance and driving in simulation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, p. 0.

[342] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, “Domain randomization for transferring deep neural networks from simulation to the real world,” in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, IEEE, 2017, pp. 23–30.

[343] J. Freeman, E. Keogh, and J. Davidoff, “A Cross-Media Presence Questionnaire: The ITC-Sense of Presence Inventory,” *Presence: Teleoperators & Virtual Environments*, vol. 10, no. 3, pp. 282–297, 2001.

[344] X. Wang, S. Liu, B. Cai, D. Hurwitz, Q. Guo, and X. Wang, “Sample Size Study of Driving Simulator Experiment for Freeway Design Safety Evaluations,” *Transp Res Rec*, vol. 2677, no. 6, pp. 73–92, Jan. 2023, doi: 10.1177/03611981221144296.

[345] D. Goedicke, J. J. Li, V. Evers, and W. Ju, “VR-OOM: Virtual Reality On-rOad driving siMulation,” *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018, [Online]. Available: https://api.semanticscholar.org/CorpusID:5041099

[346] L. Bringoux, J. Monnoyer, P. Besson, C. Bourdin, S. Denjean, E. Dousset, C. Goulon, R. Kronland-Martinet, P. Mallet, T. Marqueste, C. Martha, V. Roussarie, J.-F. Sciabica, and A. Stratulat, “Influence of speed-related auditory feedback on braking in a 3D-driving simulator,” *Transp Res Part F Traffic Psychol Behav*, vol. 44, pp. 76–89, 2017, doi: https://doi.org/10.1016/j.trf.2016.10.006.

[347] F. Feng, S. Bao, J. R. Sayer, C. Flannagan, M. Manser, and R. Wunderlich, “Can vehicle longitudinal jerk be used to identify aggressive drivers? An examination using naturalistic driving data,” *Accid Anal Prev*, vol. 104, pp. 125–136, 2017, doi: https://doi.org/10.1016/j.aap.2017.04.012.

[348] Ministry of Transport Malaysia, “Road Safety Regulation in Malaysia.” Accessed: Dec. 11, 2021. [Online]. Available: https://www.mot.gov.my/en/land/safety/road-safety-regulation

[349] N. H. T. S. Administration, “2015 motor vehicle crashes: overview,” *Traffic safety facts: research note*, vol. 2016, pp. 1–9, 2016.

[350] Y. Yue, Z. Yang, X. Pei, H. Chen, C. Song, and D. Yao, “Will crash experience affect driver’s behavior? An observation and analysis on time headway variation before and after a traffic crash,” *Tsinghua Sci Technol*, vol. 25, no. 4, pp. 471–478, 2020.

[351] S. Hirst and R. Graham, “The format and presentation of collision warnings,” in *Ergonomics and safety of intelligent driver interfaces*, CRC Press, 2020, pp. 203–219.

[352] M. Saffarzadeh, N. Nadimi, S. Naseralavi, and A. R. Mamdoohi, “A general formulation for time-to-collision safety indicator,” in *Proceedings of the Institution of Civil Engineers-Transport*, Thomas Telford Ltd, 2013, pp. 294–304.

[353] C. J. Hong, V. R. Aparow, and H. Jamaluddin, “Real-time human search and monitoring system using unmanned aerial vehicle,” *International Journal of Vehicle Autonomous Systems*, vol. 17, no. 1–2, pp. 106–132, 2023.

[354] F. Maleki, N. Muthukrishnan, K. Ovens, C. Reinhold, and R. Forghani, “Machine learning algorithm validation: from essentials to advanced applications and implications for regulatory certification and deployment,” *Neuroimaging Clinics*, vol. 30, no. 4, pp. 433–445, 2020.

[355] V. Atliha and D. Šešok, “Comparison of VGG and ResNet used as Encoders for Image Captioning,” in *2020 IEEE Open Conference of Electrical, Electronic and Information Sciences (eStream)*, 2020, pp. 1–4. doi: 10.1109/eStream50540.2020.9108880.

[356] E. Hauer, “Safety in geometric design standards.” Accessed: Apr. 01, 2022. [Online]. Available: https://www.webpages.uidaho.edu/ce576/assignments/Assignment_04/SafetyinGeometricDesign.pdf

[357] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, and S. Gelly, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.

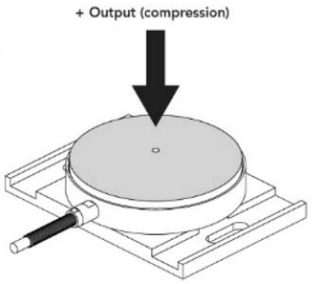
[358] A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi, “Escaping the big data paradigm with compact transformers,” *arXiv preprint arXiv:2104.05704*, 2021.

Appendix A: Sensor Specifications

Pedal Force Sensor

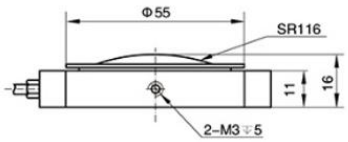
FEATURES

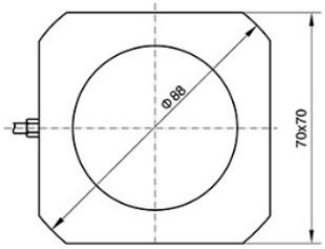
- Ideal for Automotive Applications involving brake testing
- Ideal for Aerospace Applications involving pedal force testing
- Spike resistant and highly resistant to off-axis loading
- Low Profile Height
- Removable Mounting Plate
- Light weight



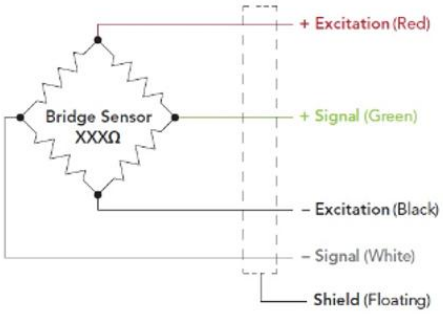
Technical Parameter	
Rated Load	0.5~1.5 kN
Comprehensive Precision	0.3% FS
Sensitivity	1.5 mV/V
Creep	±0.05% FS/30min
Zero Output	±1% FS
Temperature Effect On Zero	±0.05% FS/10°C
Temperature Effect On Output	±0.05% FS/10°C
Operating Temperature Range	-20~+65 °C
Input Resistance	380±10 Ω
Output Resistance	350± 3 Ω
Insulation Resistance	> 5000 MΩ
Safe Load Limit	150% FS
Bridge Voltage	Reference 10V DC
Material	Alloy Steel
Load Cell Mode Of Connection	Input+: Red
	Input-: Black
	Output+: Green
	Output-: White

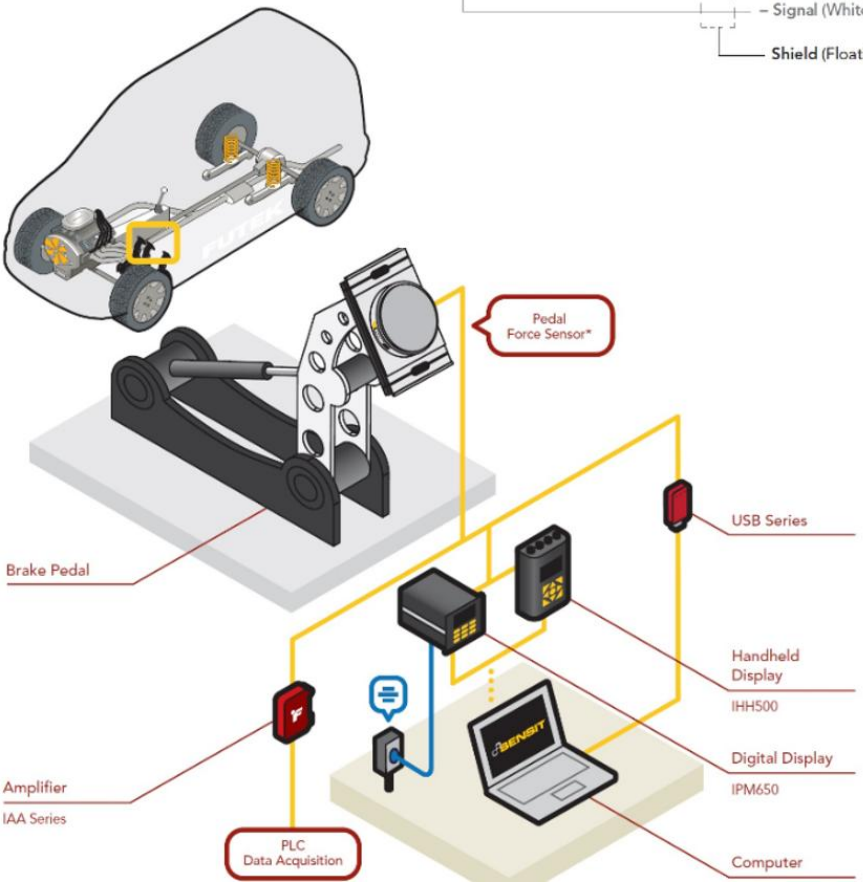
DIMENSIONS inches [mm]





WIRING CODE (WC1 WITH SHIELD)	
RED	+ EXCITATION
BLACK	– EXCITATION
GREEN	+ SIGNAL
WHITE	– SIGNAL
SHIELD	FLOATING



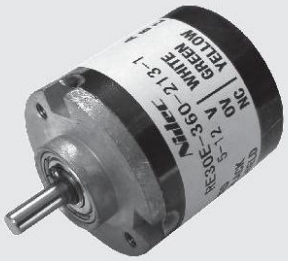


OPTICAL ENCODERS RE30E

FEATURES

- With temp. compensation circuit
- With zero index signal
- Power supply voltage is for 5 ~ 12 V, 24V
- Economy type
- RoHS compliant

RoHS compliant



STANDARD SPECIFICATIONS

Electrical characteristics

Input voltage		DC5~12 V ± 10 %	DC24 V ± 10 %
Input current		50 mA maximum	
Output wave form		Square wave	
Output phases		A, B, Z	
Resolution (P/R)		100, 200, 300, 360, 400, 500 600, 800, 900, 1000, 1024	
Phase difference of outputs A & B		90° ± 45°	
Maximum frequencys response		10 kHz (100 P/R), 20 kHz (200P/R) 25 kHz (300 ~ 500 P/R), 60 kHz (600 P/R) 80 kHz (800 P/R), 90 kHz (900P/R) 100 kHz (1000 ~ 1024 P/R)	
Output signal	"1 (High)"	(Vcc − 1) V min.	(Vcc − 2) V min.
	"0 (Low)"	+ 0.5 V max.	+ 1.0 V max.
Output impedance		2.2 kΩ	
Light source		LED	
Output Sink Current		80 mA maximum	

Mechanical characteristics

Starting torque		0.29 mN·m {3 gf·cm} maximum
Inertia		2 g·cm^2 maximum
Shaft loading (When mounting)	Radial	19.6 N {2 kgf} maximum
	Axial	9.81 N {1 kgf} maximum
Net weight		Approx. 70 g

Environmental characteristics

Operating temp. range	0 ~ 70 °C
Storage temp. range	– 20 ~ 80 °C
Protection grade	IP40

RELIABILITY TEST

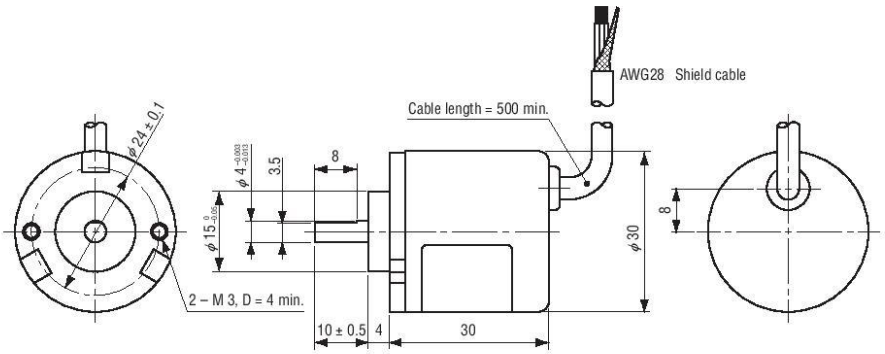
The output wave form shall satisfy the STANDARD SPECIFICATIONS after the following tests.

Test item		Test conditions	
Vibration	Power OFF	Amplitude : 1.52 mm or 98.1 m/s^2 (10 G) whichever is smaller. 10 ~ 500 Hz excursion 0.25 h/cycle, 8 cycles each for X, Y, Z, directions.	
Shock	Power OFF	3 times each in 6 directions (X, Y, Z) at 490 m/s^2 (50 G), 11 ms.	
High temperature exposure	Power OFF	80 °C 96 h	To be measured after leaving samples for 1 h at normal temperature and humidity after the test.
	Power ON	70 °C 96 h	
Low temperature exposure	Power OFF	− 20 °C 96 h	
	Power ON	0 °C 96 h	
Humidity	Power OFF	To be measured after wiping out moisture and leaving samples for 1 h at normal temperature and humidity after the test.	
Thermal shock	Power OFF	To be done 10 cycles with the following condition (To be measured after leaving samples for 1 h at normal temperature and humidity after the test.) 80 °C 1 h, − 20 °C 1 h	

RE30E
OPTICAL ENCODERS

■ OUTLINE DIMENSIONS

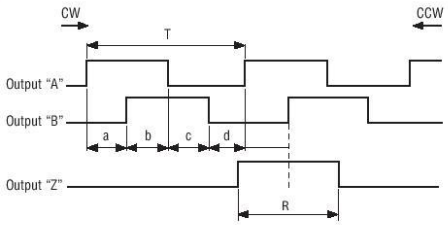
Unless otherwise specified, tolerance: ± 0.4 (Unit: mm)



■ ELECTRICAL WIRING

Red	Power ⊕
Black	Power 0 (V)
White	Output "A"
Green	Output "B"
Yellow	Output "Z"
Cable shield	NC

■ OUTPUT

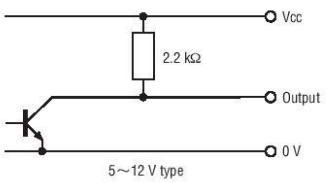


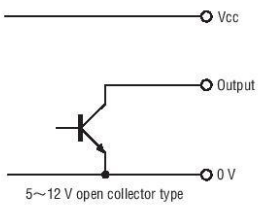
a, b, c, d = 1/4T1/8T
R= T3/4T

The "Z" phase, however, includes no more than two "B" phase startups (CW rotation)

■ OUTPUT CIRCUIT

● RE30E





Sink current of output circuit 80 mA maximum (at 25°C)

Appendix B: System and Environment Configuration

Deep learning is highly data-driven and requires significant computational resources. To expedite the training process, GPUs are preferred over CPUs due to their ability to perform parallel computations. While CPUs handle complex calculations sequentially, GPUs, with their numerous programmable streaming multiprocessors and floating-point capabilities, excel in parallelizing tasks such as matrix multiplications. Modern GPUs can deliver performance more than ten times that of general-purpose CPUs.

Choosing the Right GPU

Selecting an appropriate GPU is crucial for building a deep learning workstation. While AMD and NVIDIA are the main GPU manufacturers, NVIDIA's GPUs are preferred due to their support for CUDA architecture, cuBLAS, and cuDNN libraries. Although AMD has recently introduced the ROCm library for AI computing, its support and resources are limited compared to NVIDIA's established presence in the AI industry. NVIDIA's latest Ampere architecture features Tensor Cores, designed specifically for deep learning tasks, offering up to 20 times the speed of traditional CUDA cores.

VRAM Considerations

The size of a GPU's video RAM (VRAM) affects the capacity to train large neural networks and handle substantial batch sizes. For instance, a GPU with less than 6 GB of VRAM cannot train a Style GAN model with 256x256 pixel images. In this research, the NVIDIA GeForce RTX 3060 was chosen for its 3584 CUDA cores, 112 Tensor Cores, and 12GB GDDR6 VRAM, making it a cost-effective yet powerful option. To overcome the memory limitations for large models, two RTX 3060 GPUs were installed in the workstation. With model parallelism, large models that typically require a GPU with 24GB VRAM, such as the RTX 3090, can also be trained using this configuration at a significantly lower cost.

Workstation Specifications

1. **CPU**: Intel Core i9-11900KB (8 cores, 16 threads, up to 4.9GHz)
2. **Motherboard**: NUC11 Extreme Beast Canyon
3. **RAM**: 128GB (2x 64GB 3200MHz)
4. **GPU**: 2x NVIDIA GeForce RTX 3060 (12GB GDDR6 VRAM)
5. **Storage**:
    - 1TB XPG SX8200 Pro NVMe PCI-E 3.0 SSD (512GB partitioned for Windows, 512GB for Ubuntu)
    - 2x 4TB XPG S70 BLADE NVMe PCI-E 4.0 SSD (data storage)
    - 8TB Western Digital SN640 U.2 PCI-E 3.0 SSD (data storage)
6. **Connectivity**: Intel AX210 Wi-Fi 6E dual band with Bluetooth 5.3 PCI-E card
7. **Power Supply Unit**: Corsair 650W 80+ Gold

Operating System Selection

For deep learning applications, Linux is preferred over Windows for several reasons:

1. **Open Source**: Linux has a large open-source community, with most deep learning frameworks and libraries first available on Linux.
2. **GPU Support**: Installing CUDA toolkit and cuDNN library is simpler on Linux, with abundant online resources. Additionally, TensorRT, which optimizes deep learning performance, supports only Linux for Python API.
3. **Performance**: Linux performs better than Windows for deep learning due to lower resource utilization by the OS, leaving more resources for the applications.
4. **Ease of Use**: Installing packages or dependencies is straightforward on Linux with single commands.

Ubuntu Installation

Ubuntu is a popular Linux distribution for deep learning, supported by extensive documentation and community resources. Ubuntu 20.04 LTS was chosen for its compatibility with the RTX 3060 GPU and its long-term support until April 2025.

Installation Steps:

1. **Create Bootable Drive**: Use a USB flash drive (minimum 8GB) to create a bootable drive from the Ubuntu desktop image, downloadable from the official site. Use Etcher as shown in Figure 1 to flash the image.

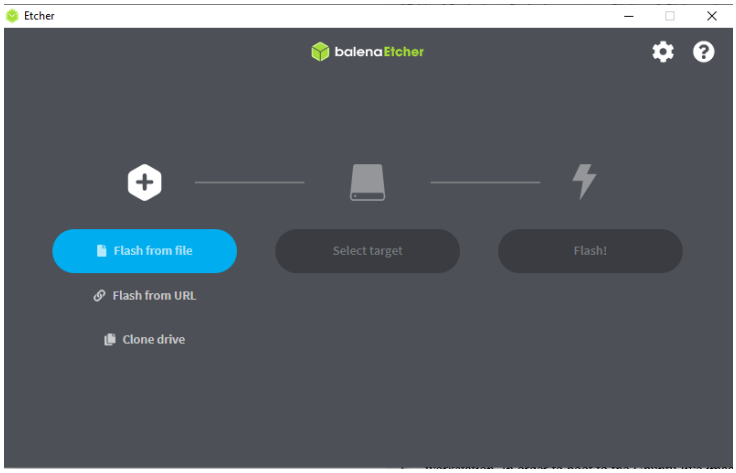


Figure 1: Etcher images flashing tool UI interface

2. **Boot from USB**: Insert the bootable drive, restart the system, and enter the boot menu (F12 for the motherboard). Select the USB drive to boot the Ubuntu live image.
3. **Install Ubuntu**: Follow the installation instructions to set up the installation drive, region, language, username, and password.
4. **Post-Installation**: After installation, remove the USB drive and restart the system. To install NVIDIA's driver, the following commands are used:

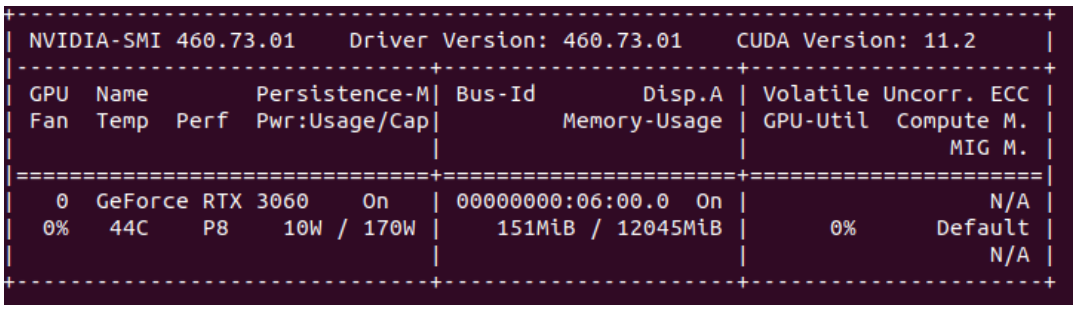
```
#Add NVIDIA package repository
$ sudo add-apt-repository ppa:graphics-drivers/ppa
```

```
#Install NVIDIA driver
$ sudo apt-install nvidia-driver-460

#Reboot the system
$ sudo reboot

#Verify installation
$ nvidia-smi

# Successful installation will display driver version and corresponding CUDA version.
```



NVIDIA driver information

Proper configuration of the workstation, including GPU selection and operating system setup, is essential for efficient deep learning research. By leveraging NVIDIA's GPUs and Ubuntu's robust support, this setup ensures optimal performance for deep learning tasks.

Deep Learning Framework

Deep learning frameworks provide libraries that simplify the design, building, training, and validation of deep learning models through high-level programming interfaces. Popular frameworks include TensorFlow, PyTorch, Caffe, Keras, and MATLAB. Both TensorFlow and PyTorch were utilized in this research.

TensorFlow and Keras

TensorFlow, integrated with Keras for high-level programming, offers numerous built-in functions that facilitate the early stages of neural network development. TensorFlow uses static computation graphs, requiring operations to be defined before running the model, which can limit flexibility during experimentation.

PyTorch

In contrast, PyTorch employs dynamic computation graphs, allowing seamless experimentation and fine-tuning of neural network hyperparameters. This makes PyTorch more adaptable for iterative research and development processes.

Python for Deep Learning

Python is the preferred programming language for deep learning research due to its interpreted nature, which compiles and executes code line-by-line, simplifying error debugging and shortening development time. Additionally, Python offers numerous built-in functions and libraries that accelerate deep learning research.

Compatibility Issues and Workarounds

During the setup of the deep learning workstation, it was found that the latest stable version of TensorFlow-GPU was incompatible with the RTX 3060 and NVIDIA Linux driver 460.56, which required CUDA 11.2 and cuDNN 8.0 libraries. As a workaround, NVIDIA's build of TensorFlow 1.15 was installed.

Anaconda Installation

Before installing TensorFlow, Anaconda was installed. Anaconda is a software distribution that includes essential packages for data science and deep learning, such as:

1. **Scikit-Learn**: Deep learning framework.
2. **Numpy, Scipy, Pandas**: Data processing and manipulation.
3. **Matplotlib**: Data visualization.

Anaconda also provides virtual environments to manage packages and specific library and Python versions without affecting other projects. The installation procedures for Anaconda on Ubuntu are detailed as follow:

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
bash Miniconda3-latest-Linux-x86_64.sh

conda update conda
conda update --all
```

TensorFlow Installation

Following the Anaconda installation, NVIDIA's build of TensorFlow was installed. A conda environment named "tf1-nv" with Python 3.8 was created. The procedures shown in the figure below were followed to complete the installation. Essential Python packages for image processing, such as OpenCV and Pillow, were installed using:

- pip install opencv-contrib-python
- pip install pillow

```
conda create --name TF1.15 python=3.8
conda activate TF1.15

pip install nvidia-pyindex
pip install nvidia-tensorflow[horovod]

conda install -c conda-forge openmpi
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$HOME/miniconda3/envs/TF1.15/lib/

mkdir tf-test
cd tf-test

wget https://github.com/dbkinghorn/NGC-TF1-nvidia-examples/archive/main/NGC-TF1-nvid
ia-examples.tar.gz

tar xf NGC-TF1-nvidia-examples.tar.gz
cd NGC-TF1-nvidia-examples-main/cnn/

python resnet.py --layers=50 --batch_size=64 --precision=fp32
```

PyTorch Installation

A separate conda environment was created for PyTorch to isolate its dependencies from the TensorFlow environment. The following commands were executed to create and activate the environment, and to install PyTorch, torchvision, and torchaudio:

```
conda create -n pytorch_env python=3.8
conda activate pytorch_env
conda install pytorch torchvision torchaudio pytorch-cuda=11.2 -c pytorch -c nvidia
```

Development Tools

Microsoft Visual Studio Code (VS Code) was installed as the source code editor to facilitate deep learning model development. After downloading the .deb file from the official website, a terminal was opened in the directory containing the file (usually the Downloads directory) and the following command was executed:

```
sudo apt install ./code_1.54.1-1614898113_amd64.deb
```

Since Anaconda is the default Python installation, VS Code uses Anaconda's Python interpreter to run Python code automatically.

Virtual Autonomous Vehicle Simulation Platform

In this research, the IPG CarMaker was selected as the virtual simulation platform due to its widespread use in the automotive industry, accurate vehicle dynamics model, and capability to personalize driver models and vulnerable road user behaviours. The installation procedures for IPG CarMaker on Linux are provided by IPG.

IPG CarMaker Installation

To install IPG CarMaker on Ubuntu:

1. Navigate to the CD-ROM containing the installation package using the command line terminal.
2. Initialize the IPG installer with the command:

```
sudo ./ipg-install
```

3. Follow the steps in the graphical user interface of the IPG installer.
4. Set the installation directory to the standard path /opt/ipg.
5. Allow the installer to scan the installation package for installable IPG products.
6. Select the IPG products to install, such as CarMaker 9.0.3 (64 bit), IPGMovie, IPGControl, and IPGGraph.
7. Once the installation is complete, place the license file obtained from IPG in the license directory of IPG CarMaker (/opt/ipg/etc/).

The IPG CarMaker will then be ready for developing the virtual simulation platform.

MATLAB and Simulink

MATLAB and Simulink are widely used for programming and numeric computing, essential for plotting functions and data, developing models, and interfacing CarMaker with external modules. This research relies heavily on MATLAB and Simulink for developing vehicle dynamic models and driver models.

To install MATLAB on Ubuntu:

1. Download the MATLAB installer zip file from the official site.
2. Unzip the installer into the matlab_2020a_installer folder with the command:

```
unzip -X -K matlab_R2020a_glnxa64.zip -d matlab_2020a_installer
```

3. Open a command line terminal in the matlab_2020a_installer directory.
4. Start the MATLAB installer using the command:

```
sudo ./install
```

5. Follow the steps in the graphical user interface, selecting necessary products such as the Deep Learning Toolbox and Automated Driving Toolbox.
6. After installation, MATLAB can be launched with the following commands:

```
cd /usr/local/MATLAB/R2020a/bin
./matlab
```

This setup enables the development of vehicle dynamic models and driver models essential for the research.

Appendix C: Questionnaire Form for AMoDS

Virtual Driving Simulator Questionnaire

Personal Information

1. Age: _____
2. Gender: _____
3. Total numbers of years driving: _____

Driving Style

1. Do you break the motorway speed limit?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

2. Do you drive fast?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

3. Do you exceed the 30km/h speed limit in built up areas (campus, urban, residential areas, cities and village)?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

4. Do you become flustered when faced with sudden dangers while driving?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

5. Do you remain calm when things happen very quickly and there is little time to think?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

6. Is your driving affected by pressure from other motorists?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

7. Are you happy to receive advice from people about your driving?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

8. Do you dislike people giving advice about your driving?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

9. Do you drive cautiously?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

10. Do you find it easy to ignore distractions while driving?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

11. Do you ignore passengers urging you to change your speed?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

12. How often do you set out on an unfamiliar journey without first looking at a map?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

13. Do you plan long journeys in advance including places to stop and rest?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

14. Do you overtake on the inside lane of a dual carriageway if you have the opportunity to do so?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

15. Do you ever drive through a traffic light after it has turned red?

Very infrequently or never					Very frequently or always
1	2	3	4	5	6

Driver-in-the-loop simulator with Fixed screen

Spatial Presence

1. I can control the speed of the vehicle using the gas pedal and brake.

Strongly disagree				Strongly agree
1	2	3	4	5

2. I have difficulty seeing the lane markings in the fixed screen driving system.

Strongly disagree				Strongly agree
1	2	3	4	5

3. Driving with fixed screen feels almost like driving in a car cockpit.

Strongly disagree				Strongly agree
1	2	3	4	5

4. I had a sense of being in the scenes displayed

Strongly disagree				Strongly agree
1	2	3	4	5

5. I felt I was visiting the places in the displayed environment.

Strongly disagree				Strongly agree
1	2	3	4	5

6. I felt that the characters and/or objects could almost touch me.

Strongly disagree				Strongly agree
1	2	3	4	5

7. To which extend do you feel present in the virtual environment, as if you were really there.

Strongly disagree				Strongly agree
1	2	3	4	5

8. I think that the car will go off the road often when I drive with fixed screen driving system

Strongly disagree				Strongly agree
1	2	3	4	5

9. I feel like I am in control of the car when driving with the fixed screen driving system.

Strongly disagree				Strongly agree
1	2	3	4	5

10. The content seemed believable to me.

Strongly disagree				Strongly agree
1	2	3	4	5

11. The displayed environment seemed natural

Strongly disagree				Strongly agree
1	2	3	4	5

12. I had a strong sense that the characters and objects were solid.

Strongly disagree				Strongly agree
1	2	3	4	5

13. I was aware of the real world.

Strongly disagree				Strongly agree
1	2	3	4	5

14. I wanted to see more of the space in the displayed environment than I was able to.

Strongly disagree				Strongly agree
1	2	3	4	5

15. I found it easy to forget that I was watching a display.

Strongly disagree				Strongly agree
1	2	3	4	5

16. The temperature of the real world distracted me.

Strongly disagree				Strongly agree
1	2	3	4	5

17. I was distracted by the quality of the technology.

Strongly disagree				Strongly agree
1	2	3	4	5

18. I felt I knew what was going to happen next.

Strongly disagree				Strongly agree
1	2	3	4	5

Motion Platform

1. Did the 6 degree of freedom movements of the motion platform contribute to the realistic of the driving simulator?

Strongly disagree				Strongly agree
1	2	3	4	5

2. How much did this experience seem consistent with your real-world experiences?

Strongly disagree				Strongly agree
1	2	3	4	5

3. How strong was your feeling of self-motion?

Strongly disagree				Strongly agree
1	2	3	4	5

4. Were the motion and visual feedback synchronized together?

Strongly disagree				Strongly agree
1	2	3	4	5

5. Was the system comfortable?

Strongly disagree				Strongly agree
1	2	3	4	5

6. How much did you enjoy using the system?

Strongly disagree				Strongly agree
1	2	3	4	5

Simulator Sickness

1. I felt prone to vomiting.

Strongly disagree				Strongly agree
1	2	3	4	5

2. I felt dizzy.

Strongly disagree				Strongly agree
1	2	3	4	5

3. I felt nauseous.

Strongly disagree				Strongly agree
1	2	3	4	5

4. I felt I had a headache.

Strongly disagree				Strongly agree
1	2	3	4	5

5. I had eyestrain.

Strongly disagree				Strongly agree
1	2	3	4	5

6. I felt mental pressure.

Strongly disagree				Strongly agree
1	2	3	4	5

7. I felt fear.

Strongly disagree				Strongly agree
1	2	3	4	5

8. I felt anxiety.

Strongly disagree				Strongly agree
1	2	3	4	5

Driver-in-the-loop simulator with Virtual Reality

Spatial Presence

1. I feel comfortable using the Virtual Reality Driving Motion Platform.

Strongly disagree				Strongly agree
1	2	3	4	5

2. I can control the speed of the vehicle using the gas pedal and brake.

Strongly disagree				Strongly agree
1	2	3	4	5

3. I have difficulty seeing the lane markings in the Virtual Reality Driving System.

Strongly disagree				Strongly agree
1	2	3	4	5

4. Driving in the Virtual Reality Driving System feels almost like driving in a car.

Strongly disagree				Strongly agree
1	2	3	4	5

5. I had a sense of being in the scenes displayed

Strongly disagree				Strongly agree
1	2	3	4	5

6. I felt I was visiting the places in the displayed environment.

Strongly disagree				Strongly agree
1	2	3	4	5

7. I felt that the characters and/or objects could almost touch me.

Strongly disagree				Strongly agree
1	2	3	4	5

8. To which extend do you feel present in the virtual environment, as if you were really there.

Strongly disagree				Strongly agree
1	2	3	4	5

9. I think that the car will go off the road often when I drive the VR Driving System.

Strongly disagree				Strongly agree
1	2	3	4	5

10. I feel like I am in control of the car when driving with the Virtual Reality Driving System.

Strongly disagree				Strongly agree
1	2	3	4	5

11. Wearing the Head Mount Display makes it difficult for me to drive in the Virtual Reality Driving System.

Strongly disagree				Strongly agree
1	2	3	4	5

12. The content seemed believable to me.

Strongly disagree				Strongly agree
1	2	3	4	5

13. The displayed environment seemed natural

Strongly disagree				Strongly agree
1	2	3	4	5

14. I had a strong sense that the characters and objects were solid.

Strongly disagree				Strongly agree
1	2	3	4	5

15. I was aware of the real world.

Strongly disagree				Strongly agree
1	2	3	4	5

16. I wanted to see more of the space in the displayed environment than I was able to.

Strongly disagree				Strongly agree
1	2	3	4	5

17. I found it easy to forget that I was watching a display.

Strongly disagree				Strongly agree
1	2	3	4	5

18. The temperature of the real world distracted me.

Strongly disagree				Strongly agree
1	2	3	4	5

19. I was distracted by the quality of the technology.

Strongly disagree				Strongly agree
1	2	3	4	5

20. I felt I knew what was going to happen next.

Strongly disagree				Strongly agree
1	2	3	4	5

Motion Platform

1. Did the 6 degree of freedom movements of the motion platform contribute to the realistic of the driving simulator?

Strongly disagree				Strongly agree
1	2	3	4	5

2. How much did this experience seem consistent with your real-world experiences?

Strongly disagree				Strongly agree
1	2	3	4	5

3. How strong was your feeling of self-motion?

Strongly disagree				Strongly agree
1	2	3	4	5

4. Were the motion and visual feedback synchronized together?

Strongly disagree				Strongly agree
1	2	3	4	5

5. Was the system comfortable?

Strongly disagree				Strongly agree
1	2	3	4	5

6. How distracting was the control mechanism?

Strongly disagree				Strongly agree
1	2	3	4	5

7. How much did you enjoy using the system?

Strongly disagree				Strongly agree
1	2	3	4	5

Simulator Sickness

1. I felt prone to vomiting.

Strongly disagree				Strongly agree
1	2	3	4	5

2. I felt dizzy.

Strongly disagree				Strongly agree
1	2	3	4	5

3. I felt nauseous.

Strongly disagree				Strongly agree
1	2	3	4	5

4. I felt I had a headache.

Strongly disagree				Strongly agree
1	2	3	4	5

5. I had eyestrain.

Strongly disagree				Strongly agree
1	2	3	4	5

6. I felt mental pressure.

Strongly disagree				Strongly agree
1	2	3	4	5

7. I felt fear.

Strongly disagree				Strongly agree
1	2	3	4	5

8. I felt anxiety.

Strongly disagree				Strongly agree
1	2	3	4	5

Thank you very much for your cooperation.

List of Publications

Journals

1. **Cheok Jun Hong**, Vimal Rau Aparow, Neng, J. N. Z., Cheah, J. L., & Leong, D. (2022). Development and Optimization of Formation Flying for Unmanned Aerial Vehicles Using Particle Swarm Optimization Based on Reciprocal Velocity Obstacles. *SAE International Journal of Aerospace*, *15*(01-15-02-0011), 171-184.
2. **Cheok Jun Hong**, Vimal Rau Aparow, & Hishamuddin Jamaluddin. (2023). Real-time human search and monitoring system using unmanned aerial vehicle. *International Journal of Vehicle Autonomous Systems*, *17*(1-2), 106-132.
3. **Cheok Jun Hong**, Lee Kah Onn, Vimal Rau Aparow, Amer, N. H., Peter, C. S. P., & Magaswaran, K. (2023). Validation of scenario-based virtual safety testing using low-cost sensor-based instrumented vehicle. *Journal of Mechanical Engineering and Sciences*, 9520-9541.
4. Lee Kah Onn, Colin Rei, Vimal Rau Aparow, **Cheok Jun Hong**, Wai, A. A. Y. L., & Jawi, Z. M. (2023). Analysis of Automated Emergency Braking System to Investigate Forward Collision Condition Using Scenario-Based Virtual Assessment. *International Journal of Integrated Engineering*, *15*(5), 273-285.
5. Ng Yuan Weun, Lee Kah Onn, **Cheok Jun Hong**, & Vimal Rau Aparow (2023). Sensor Fusion-Based Target Prediction System for Virtual Testing of Automated Driving System. In *Innovative Manufacturing, Mechatronics & Materials Forum* (pp. 13-29). Singapore: Springer Nature Singapore.
6. **Cheok Jun Hong**, Vimal Rau Aparow, Lee Kah Onn, Heng Lai Wai, Hishamuddin Jamaluddin (2024). Predicting Driving Behaviour using Hybrid CNN-LSTM Model: A Case Study in Malaysian Traffic. *Robotics and Autonomous Systems*. (Under Review)
7. Lee Kah Onn, Vimal Rau Aparow, **Cheok Jun Hong** (2024). Investigations on Lagrangian-Based AVES Motion Driving Simulator (AMoDS) for Virtual Testing in Autonomous Vehicles. *Automotive Innovation. (Under Review)*

Conferences

1. **Cheok Jun Hong**, & Vimal Rau Aparow (2021). System configuration of Human-in-the-loop Simulation for Level 3 Autonomous Vehicle using IPG CarMaker. In *2021 IEEE international conference on internet of things and intelligence systems (IoTaIS)* (pp. 215-221). IEEE.
2. Vimal Rau Aparow, **Cheok Jun Hong**, Ng Yuan Weun, Huei, C. C., Yen, T. K., Hong, L. C., ... & Wen, K. K. (2021, December). Scenario based simulation testing of autonomous vehicle using Malaysian road. In *2021 5th International Conference on Vision, Image and Signal Processing (ICVISP)* (pp. 33-38). IEEE.
3. Vimal Rau Aparow, **Cheok Jun Hong**, Lee Kah Onn, Jamaluddin, H., Kassim, K. A. A., & Jawi, Z. M. (2022, December). Challenges and Approach: Scenario-Based Safety Assessment of Autonomous Vehicle for Malaysian Environment and Driving

Conditions. In *2022 17th International Conference on Control, Automation, Robotics and Vision (ICARCV)* (pp. 386-391). IEEE.

4. **Cheok Jun Hong**, Lee Kah Onn, Vimal Rau Aparow (2024). Bridging Reality: A Driver-in-the-Loop Simulator with VR Headset and 6-DOF Motion Platform. *IEEE Transactions on Vehicular Technology*. (Under Consideration for Acceptance)

Copyrights

1. Vimal Rau Aparow, Alex Au, **Cheok Jun Hong**, Lee Kah Onn, Khairil Anwar Abu Kassim, and Zulhaidi Mohd Jawi (2023). Malaysian Traffic Scenario Classification and Identification Framework Based on Qualitative Analysis for Autonomous Vehicle Driving Conditions. CRLY2023W01446.

List of Magazine and Book Chapter

1. Vimal Rau Aparow, **Cheok Jun Hong**, Lee Kah Onn, Hishamuddin Jamaluddin, Khairil Anwar Abu Kassim, and Zulhaidi Mohd Jawi (2023). Virtual Safety Testing for Autonomous Vehicle in Malaysia. *MBOT Techies Magazine*, 6-7.
2. Vimal Rau Aparow, **Cheok Jun Hong**, Lee Kah Onn (2024). AV analysis framework for Malaysia. *Automotive Testing Technology International*, 13-14.
3. Ng Yuan Weun, Vimal Rau Aparow, Chai Chee Huei, Lee Chen Hong, Tiong Kai Yen, **Cheok Jun Hong** and Lee Kah Onn (2024). Scenario generation for safety testing of autonomous vehicles for Malaysian virtual environment. *UTM Publisher*. (Under Review)

List of Awards

1. Vimal Rau Aparow, **Cheok Jun Hong**, Lee Kah Onn, Ng Yuan Weun, Chai Chee Huei, Lee Chen Hong and Tiong Kai Yen, "Safety Assessment of Advanced Driving Assistance System based on Level 3 Autonomous Vehicle using Human-in-the-loop Interface" – **Silver Award**
2. Vimal Rau Aparow, Lee Kah Onn, **Cheok Jun Hong**, "Scenario Based Testing of Autonomous Emergency Braking System using Vehicle Driving Simulator," *Innovative Research, Invention and Application Exhibition* – **Gold Award and Special Award by IEEE System, Man and Cybernetics Society Malaysia Chapter**
3. Vimal Rau Aparow, **Cheok Jun Hong**, Lee Kah Onn, Aaruthiran Manoharan, Lee Wei Hong, "Motion Based Virtual Safety Testing of Autonomous Electric Vehicle using Malaysian Driving Scenarios," *34th International Invention, Innovation & Technology Exhibition (ITEX) 2023* – **Gold Award**
4. Vimal Rau Aparow, Lee Kah Onn, Cheok Jun Hong, "Scenario-based Safety Testing of Autonomous Vehicle using Malaysian Driving Conditions," *Society of Automotive Engineers Malaysia Invention and Innovative Exhibition 2023* – **Gold Award**