UNIVERSIDADE FEDERAL DO PARANÁ

ISABELLA FERNANDA WOSNIACK

AUTOMATED COMPARISON OF DIFFERENT SIMULATION SETUPS FOR SCENARIO-BASED TESTING BASED ON SIMILARITY METRICS

CURITIBA

2023

ISABELLA FERNANDA WOSNIACK

AUTOMATED COMPARISON OF DIFFERENT SIMULATION SETUPS FOR SCENARIO-BASED TESTING BASED ON SIMILARITY METRICS

Dissertação apresentada ao Programa de Pós Graduação em Engenharia Elétrica, Departamento de Engenharia Elétrica, Setor de Tecnologia, Universidade Federal do Paraná, como parte das exigências para a obtenção do título de Mestre em Engenharia Elétrica .

Orientador: Prof. Dr. Eduardo Gonçalves de Lima
Co-Orientador: Prof. Dr. Alessandro Zimmer

CURITIBA

2023

DADOS INTERNACIONAIS DE CATALOGAÇÃO NA PUBLICAÇÃO (CIP)
UNIVERSIDADE FEDERAL DO PARANÁ
SISTEMA DE BIBLIOTECAS – BIBLIOTECA DE CIÊNCIA E TECNOLOGIA

Wosniack, Isabella Fernanda
Automated comparison of different simulation setups for scenario-based testing based on similatary metrics / Isabella Fernanda Wosniack. – Curitiba, 2023.
1 recurso on-line : PDF.

Dissertação (Mestrado) - Universidade Federal do Paraná, Setor de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica.

Orientador: Eduardo Gonçalves de Lima

1. Indústria automobilística. 2. Simulação (computadores). 3. Criticalidade (Engenharia nuclear). I. Universidade Federal do Paraná. II. Programa de Pós-Graduação em Engenharia Elétrica. III. Lima, Eduardo Gonçalves de. IV. . V. Título.

Bibliotecário: Leticia Priscila Azevedo de Sousa CRB-9/2029

MINISTÉRIO DA EDUCAÇÃO
SETOR DE TECNOLOGIA
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO ENGENHARIA ELÉTRICA - 40001016043P4

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação ENGENHARIA ELÉTRICA da Universidade Federal do Paraná foram convocados para realizar a arguição da dissertação de Mestrado de **ISABELLA FERNANDA WOSNIACK** intitulada: **AUTOMATED COMPARISON OF DIFFERENT SIMULATION SETUPS FOR SCENARIO-BASED TESTING BASED ON SIMILARITY METRICS**, sob orientação do Prof. Dr. EDUARDO GONÇALVES DE LIMA, que após terem inquirido a aluna e realizada a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.
A outorga do título de mestra está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

Curitiba, 24 de Outubro de 2023.

Assinatura Eletrônica
27/10/2023 16:51:29.0
EDUARDO GONÇALVES DE LIMA
Presidente da Banca Examinadora

Assinatura Eletrônica
27/10/2023 15:37:56.0
GUILHERME DE SANTI PERON
Avaliador Externo (UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ)

Assinatura Eletrônica
27/10/2023 14:54:47.0
ANDRÉ AUGUSTO MARIANO
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

THANKS

I would like to start by expressing my gratitude to my family. My dad, Edilton Wosniack, my mom, Susana Wosniack, my sisters, Marina Frigo and Laís Mariana, my brother André Wosniack, my grandma, Iolanda Bonato, my uncle, Gilmar Bonato and my dog Cisco. Without you, I wouldn't be able to finish my Master's.

Special thanks to Viktor Pusz for all the support during my Master's thesis and for making my life much happier and lighter.

Thanks to my friend Redjon, with whom I shared many laughs during this period.

Special gratitude to Professor Eduardo Lima for his guidance and motivation since my bachelor's.

Thanks to my supervisor Marcel Langer, for all the guidance and advice regarding my research.

“However difficult life may seem,
there is always something you can do
and succeed at.“
(Stephen Hawking)

RESUMO

O uso de simulação computacional na indústria automotiva tem se tornado uma realidade cada vez mais comum para teste e validação de Sistemas Avançados de Assistência ao Condutor (ADAS) e Sistemas de Diração Autônoma (ADS). Atualmente, diversos simuladores estão presentes no mercado e padrões foram definidos para unificar arquivos utilizados por tais ferramentas. No entanto, nem todas as plataformas suportam todas as funcionalidades do arquivo de cenário, e a dinâmica do veículo implementada em cada plataforma pode afetar a forma como a simulação é executada. Tendo em vista garantir a qualidade dos arquivos de cenário, este trabalho tem como objetivo implementar uma ferramenta capaz de comparar e analisar simulações executadas em diferentes plataformas. O trabalho propõe um conjunto de métricas de similaridade que podem ser utilizadas em conjunto para avaliar a porcentagem de correspondência entre as execuções de cenários. Primeiramente, uma análise de trajetória é proposta com base na similaridade de cosseno. Na sequência, uma análise de criticalidade é proposta com base no tempo de colisão entre dois veículos. Finalmente, um novo conjunto de manobras relacionadas a objetos é definida de forma a obter uma análise de manobras. Combinadas, as três métricas descritas são utilizadas para calcular a equivalência entre as simulações. Para isso, uma soma ponderada entre as métricas é realizada. Três plataformas foram utilizadas para implementação e teste da ferramenta de comparação. A primeira ferramenta, esmini, é a ferramenta base de simulação, compatível com arquivos OpenSCENARIO, mas que possui uma dinâmica veicular simples e por vezes não realista. A segunda e terceira ferramentas, Carla e CarMaker, possuem uma dinâmica veicular mais realista e suportam arquivos OpenSCENARIO. No entanto, nem todas as funcionalidades são 100% compatíveis. Utilizando-se a ferramenta esmini como base, foram comparados os resultados de diferentes cenários quando simulados nas ferramentas Carla e CarMaker. Resultados mostram que a ferramenta é capaz de obter uma comparação precisa dos resultados da simulação, identificando ações não intencionais e gerando um relatório de compatibilidade entre simulações para cada tipo de cenário. A ferramenta é capaz de realizar análise de trajetória, identificar diferentes manobras executadas durante a simulação e realizar o cálculo de criticalidade em cenários que possuem mais de um veículo.

Palavras-chaves: simulação, ADAS, teste baseado em cenário, criticalidade, trajetória, manobras

ABSTRACT

The use of computer simulation in the automotive industry has become a common reality for testing and validating Advanced Driver Assistance Systems (ADAS) and Autonomous Driving Systems (ADS). Currently, several simulation tools are present in the market, and standards have been developed to unify the patterns used by those tools. However, not all platforms support all the functionalities in the scenario file, and the vehicle dynamics implemented in each platform may affect how the simulation is executed. To ensure the quality assessment of scenario files, this work aims to implement a tool able to compare and analyze simulations running on different platforms. The work proposes a set of similarity metrics that can be used together to evaluate the percentage match between scenario executions. A trajectory analysis is proposed based on cosine similarity, and a criticality analysis is proposed based on time-to-collision. Also, a new set of object-related maneuvers was defined for a maneuver analysis. Combined, the three individual metrics are used to calculate the final score between simulations based on the weighted sum of the individual metrics. To implement and test the comparison tool, three platforms were used. The first one, esmini, is the reference simulation tool, compatible with OpenSCENARIO files but with simple vehicle dynamics. The second and third ones, Carla and CarMaker, have more realistic vehicle dynamics and support OpenSCENARIO files. However, not all features are 100% compatible, and the dynamics of each tool are implemented differently. Using esmini as the basis for comparison, the results of different scenarios simulated in Carla and CarMaker were compared. Results show that the tool can accurately compare simulations, identifying unintentional actions and generating a compatibility report between simulations for each scenario type. The tool can perform a trajectory analysis, identify different maneuvers performed during the simulation, and perform the criticality calculation in scenarios with more than one vehicle.

Key-words: simulation, ADAS, scenario-based testing, criticality, trajectory, maneuver

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ACC Adaptative Cruise Control

ADAS Advanced Driving Assistance Systems

ADS Autonomous Driving Systems

AEB Autonomous Emergency Breaking

ASAM Association for Standardization of Automation and Measuring Systems

ASIL Automotive Safety Integrity Level

CCCscp Car-to-Car Crossing straight crossing path

CCRs Car-to-Car Rear Stationary

DiL Driver-in-the-Loop

FEP Functional Engineering Platform

FOV Field of View

HiL Hardware-in-the-Loop

IEC International-Electrotechnical Commission

MiL Model-in-the-Loop

ODD Operational Design Domain

SAE Society of Automotive Engineers

SiL Software-in-the-Loop

SuT System under Test

TCL Tool Command Language

THW Time-Headway

TTC Time-to-Collision

XML eXtensible Markup Language

CONTENTS

1 INTRODUCTION

According to the German Motor Vehicle Monitoring Association (DEKRA), driver mistakes are the primary cause of accidents in Europe. Only in Germany, almost 90% of accidents resulting in personal injury in 2019 were caused by driver's error (E.V., 2021). Some reasons include driving too close to the vehicle in front, driving under the influence of alcohol, and use of electronic devices while driving.

To increase safety, optimize traffic flow, reduce fuel consumption, and improve the mobility of elderly and disabled people, an increasing number of vehicles are being developed with automated driving capabilities (WATZENIG; HORN, 2016). But before these vehicles can be approved for production, they must undergo rigorous tests, including covering a distance of millions of miles (KLÜCK et al., 2018). This extensive testing is necessary to assess the vehicle's response to different traffic situations.

To ensure vehicle safety behavior, the use of computer simulation technology has become a relevant tool in the field of the automotive industry. According to Chen *et al.* (CHEN et al., 2022), the simulation approach allows the verification and evaluation of Advanced Driver Assistance Systems (ADAS) and Autonomous Driving Systems (ADS) with high-test efficiency, strong repeatability, low cost, and process safety. Since most of the situations encountered by ADAS/ADS in daily traffic situations are repetitive and non-critical, simulation-based tests focus on the evaluation of relevant traffic scenarios under a range of conditions and circumstances, providing more flexibility to test various traffic situations that may occur (RIEDMAIER et al., 2020), (STEVIĆ et al., 2020). The traffic scenarios can either be defined manually based on system requirements and expert knowledge or can be generated from real-world traffic data.

To test ADAS/ADS functions using simulation, different platforms can be used. IPG CarMaker and Carla, for example, are two commercial solutions in which it is possible to include sophisticated vehicle dynamics models. Environment Simulator Minimalistic (esmini), on the other hand, is an open-source solution to play OpenSCENARIO files with simple dynamics.

For the simulation to be valid, it must follow a certain quality level, ensuring that all the System Under Test (SUT) specifications are met and that the scenario runs as intended on the simulation test bench.

When executing the same scenario file on different platforms, different results can be obtained. Because each software might implement distinct vehicle dynamics, it can cause discrepancies in how the scenario runs. The issue occurs when the behavior is different to a point where the scenario executes wrong actions or triggers events at inappropriate times, compromising the accuracy of reliable tests of ADAS functionalities. This happens especially when the tool fails to read the scenario file correctly because it follows a different standard or has another primary type of file format.

This study proposes an evaluation tool based on a set of criticality metrics designed to assess different scenario executions, comparing the expected behavior with the behavior when running the file on different platforms. The goal is to simplify the validation process by automatically determining if a scenario is performing as intended, eliminating the need for manual observation during each execution.

The analysis is based on a set of three similarity metrics defined in this work. The first metric is a deviation analysis, which compares the vehicle trajectory in two simulations and returns a matching percentage between the trajectories. The second metric is a maneuver analysis, which intends to compare the different maneuvers executed by the ego vehicle during the simulation. The final metric is a criticality analysis based on time-to-collision. This metric calculates the minimum time before a possible collision happens between two vehicles and can be used to determine the time difference between simulations. The tool also verifies if an unexpected collision is happening in the scenario file using a new approach for modeling the vehicles.

After calculating the percentage match between each of the similarity metrics, an overall score is calculated based on the weighted sum of the metrics. The overall score tells the user how similar the simulations are and if the scenario is behaving as expected in different tools.

1.1 OBJECTIVES

The aim of this thesis is to create a framework for comparison of scenario execution in different tools. The framework is implemented to meet the following four objectives:

1. Develop and implement a tool able to analyze and compare different simulations recorded as CSV files;
2. Create a user interface that allows the user to select different scenario test cases for comparison;
3. Define a set of metrics that describes the similarity between simulations;
4. Generate a final report with the comparison results of different tools.

1.2 STRUCTURE

This document is organized as follows. Chapter 2 describes the main concepts related to scenario-based testing, in-the-loop simulations, traffic assessment metrics, criticality analysis, maneuver analysis, and trajectory analysis, as well as presenting different software used for testing.

Chapter 3 describes the methodology applied to achieve this work's objectives. Starting with the main concepts related to Functional Engineering Platform (FEP), system workflow, user interface, test cases applied, and how the similarity metrics were defined and implemented.

Chapter 4 presents the achieved results for each one of the four test cases analyzed in this work: cut-in scenario, Car-to-Car Rear Stationary, Car-to-car Crossing Straight Crossing Path, and simple trajectory.

Chapter 5 presents the final conclusion of this work.

2 SCENARIO-BASED TESTING

With the development of automated vehicles (AVs), ensuring the safety of AVs has become a significant concern for automotive companies. According to the Society of Automotive Engineers (SAE), there are six levels of driving automation, described in Table 1.

Especially with SAE levels 4 and 5 of automated driving, the number of kilometers that need to be driven to validate all the AD/AV functionalities is not feasible. For this reason, simulation-based testing has gained more attention, as the data used to evaluate the performance of automated driving functions are generated artificially. This allows low cost, less time and energy, and provides more safety during the test of automated functions (WACHENFELD; WINNER, 2016) (SCHÜTT et al., 2022).

TABLE 1 – Different levels of driving automation according to SAE (YURTSEVER et al., 2020)

Driving Automation	Driving Automation System
Level 0	No driving automation
Level 1	Adaptive cruise control, anti-lock braking systems, and stability control; Disengages immediately upon driver’s request.
Level 2	Partial automation, including emergency braking or collision avoidance systems; Disengages immediately upon driver’s request.
Level 3	Conditional automation. The ADS allows the driver to focus on a different task while driving. The ADS operates in a limited Operational Design Domain (ODD); Disengages immediately upon the driver’s request.
Level 4	Human attention is not required, but the vehicle can only operate in a limited ODD.
Level 5	Human attention is not required. The vehicle can operate in any road network and any weather condition.

Because there are almost infinite situations that can occur in real-world traffic scenarios, the simulation-based approach intends to use relevant scenarios according to the test object, objectives, and requirements that can be used to validate ADS (MENZEL; BAGSCHIK; MAURER, 2018). The most significant advantage of this approach is the low cost compared to real-world tests, as individual traffic situations can be tested and evaluated using virtual simulations (RIEDMAIER et al., 2020).

2.1 SCENARIO TERMINOLOGY AND ABSTRACTION LEVELS

In scenario-based testing, a scenario refers to a sequence of scenes with temporal development of actions and events. As shown in Figure 1, each scenario starts with an initial scene that describes all the participant's representations. Next, there is a sequence of actions and events, such as different maneuvers executed by other traffic participants (ULBRICH et al., 2015), (ISO/PAS, 2019).

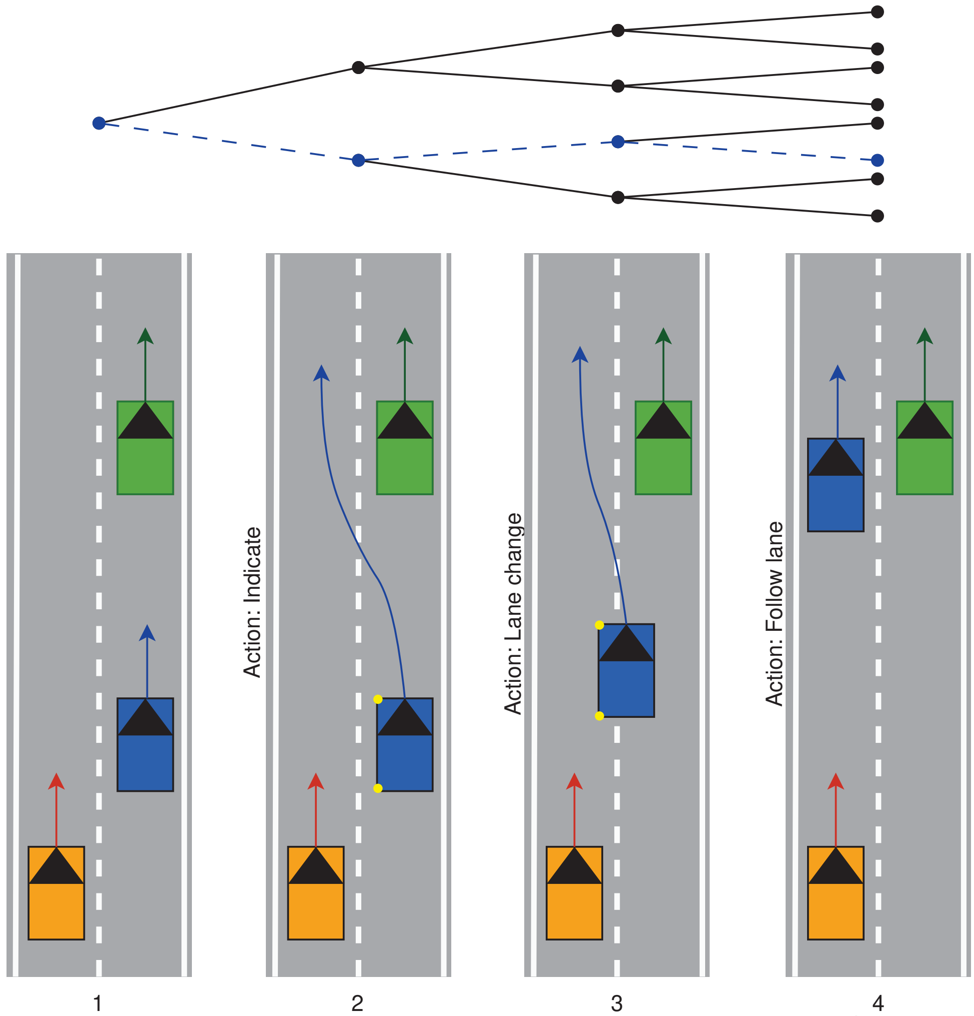


FIGURE 1 – Example of scenario representation adapted from (ULBRICH et al., 2015)

Menzel *et al.* (MENZEL; BAGSCHIK; MAURER, 2018) suggests three abstraction levels for scenarios: *functional scenarios, logical scenarios*, and *concrete scenarios*. The first one contains the most abstract level of scenario representations, where the entities (i.e., road type and movable objects) and interactions are described using a linguistic scenario notation.

The second type of scenario is defined as a logical scenario, and it uses a formal scenario notation, describing the entities and their interactions using a range of values of the state space variables.

The last scenario type is the concrete scenario, which represents the different entities and their relations using a concrete value for each parameter. This type can be used to generate test cases during the testing phase.

Bagschik *et al.* (BAGSCHIK; MENZEL; MAURER, 2018) proposes a five-layer model for scene representation, illustrated in Figure 2. The first level (L1) is called *road-level* and describes the road layout, topology, and markings. The second level (L2) is denominated *traffic infrastructure* and adds traffic infrastructure to the road level, such as signs or road markings. The third level contains classes defining how to mark, route, and secure construction sites within a time frame of one day, and it is called *temporary manipulation of L1 and L2*. The fourth level is denominated *objects*, and it's responsible for describing the traffic participants that don't belong to the traffic infrastructure, such as cars, trucks, and motorcycles. The fifth level is called *environment* and describes environmental circumstances, for example, weather and temperature.

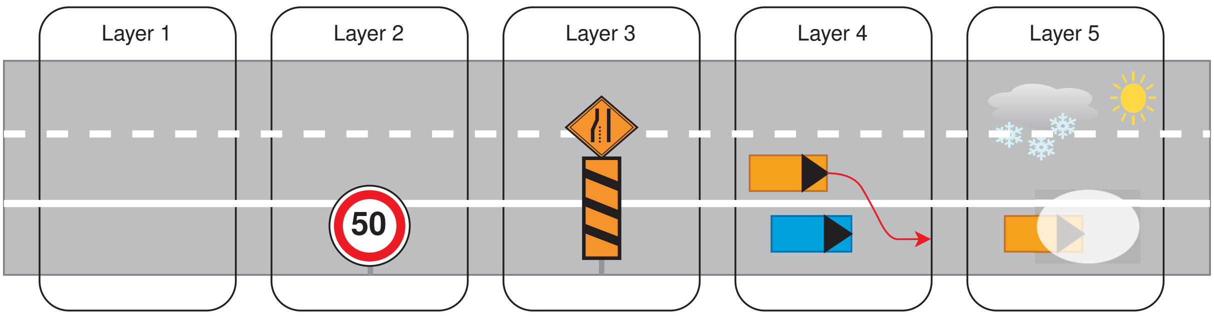


FIGURE 2 – 5 layer model proposed by (BAGSCHIK; MENZEL; MAURER, 2018)

2.2 SIMULATION STANDARDS

The Association for Standardization of Automation and Measuring Systems (ASAM) is a non-profit organization responsible for defining and standardizing protocols, interfaces, file formats, and data models to develop and test vehicles in the automotive industry (STANDARDIZATION OF AUTOMATION; SYSTEMS, s.d.). This section presents the OpenScenario and OpenDrive standards used for simulation-based testing.

2.2.1 OpenSCENARIO

ASAM OpenSCENARIO format is used to define the dynamics of a traffic simulation, such as the traffic participants' trajectories, actions, and the simulation environmental conditions. A scenario file can contain numerous maneuvers for one or more vehicles. Additionally, the conditions required for each maneuvers must be specified. For example, a lane change maneuver can be triggered by a reach position condition, or by the relative distance to another vehicle (EV, 2023b).

The ASAM OpenSCENARIO defines various coordinate systems. The *system coordinate*, for example, is based on the Cartesian coordinate system, and the orientation of the road objects are expressed by the roll, pitch, and heading (yaw) angles, as shown by Figure 3. The first one corresponds to the x coordinate, the second one to the y coordinate, and the last one to the z coordinate, respectively.

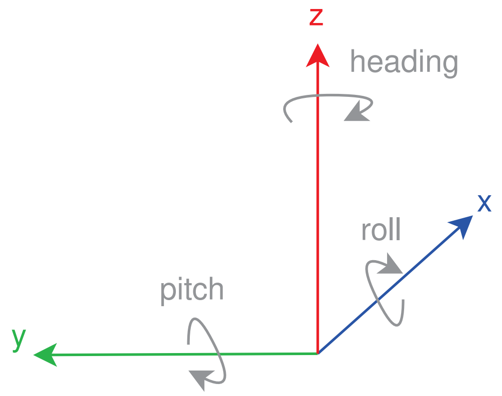


FIGURE 3 – System coordinate according to ASAM OpenSCENARIO (EV, 2023b)

The *road coordinate* system is defined by a *s/t*-type coordinate system assigned to each road in the road network definition. The s-axis belongs to the (X, Y)-plane of the world coordinate system, and its origin is fixed at the beginning of the road reference line. The t-axis, on the other hand, can be defined multiple times, and it's positioned to the left side of the s-axis, as shown by Figure 4.

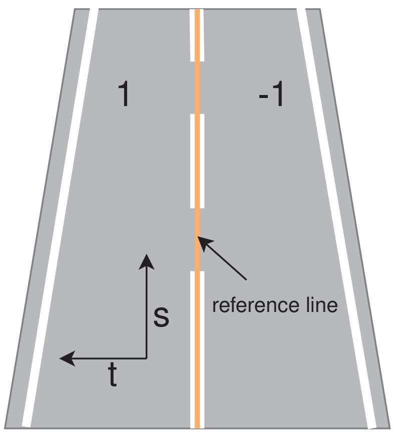


FIGURE 4 – Road coordinate system according to ASAM (EV, 2023b)

Finally, the *vehicle axis* system is defined by ISO 8855:2011 and follows the (x, y, z) coordinate system (STANDARDIZATION, 2011). The vehicle's center of reference is obtained by projecting the center of the vehicle's rear axis to the ground plane, as shown by Figure 5 (ASAM, 2022).

2.2.2 OpenDRIVE

OpenDRIVE format uses *extensible markup language* (XML) to describe the road network, the geometry of roads and lanes, lane marks, traffic signals, and other static objects (i.e., trees and buildings) present on the road network. The data used to

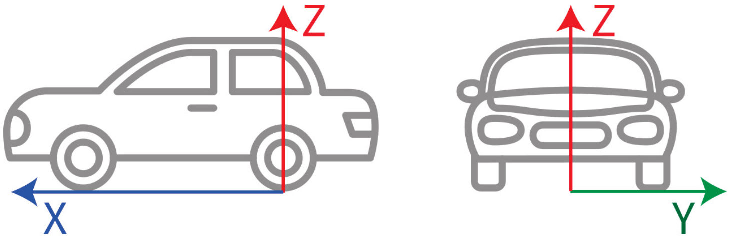


FIGURE 5 – Vehicle coordinate system according to ASAM standard (BAGSCHIK; MENZEL; MAURER, 2018)

create the road network can be either synthetic or based on real data (E.V, 2023a). The ASAM OpenDRIVE format is accepted by different simulation tools, reducing the costs for companies to convert the file between them.

2.3 IN-THE-LOOP SIMULATIONS

In-the-loop simulations are a testing method where one or more components of a system are replaced with simulations. The idea behind the method is to validate, test, and optimize the performance of the system under realistic conditions without having to rely on real-world tests.

Different in-the-loop techniques can be used during the design and development of ADAS systems. Software-in-the-loop (SiL), for example, is implemented during the early stages of development to validate the software, where the code runs on computers in a completely virtual environment. This approach is the cheapest and safest, as it allows full control of different test cases and offers no risk to the drivers. However, the higher level of abstraction affects the validation, and the simulation can only estimate the result once the test is performed on a real-world scenario (WAGNER, 2021).

On the other hand, hardware-in-the-loop (HiL) proposes to integrate parts of the actual hardware into the simulation loop (BACIC, 2005). This allows real-time testing and validation of AD functions with cheaper costs than real road test (WAGNER, 2021). Fathy *et. al.* (FATHY et al., 2006) mentions other advantages of HiL, such as a faster prototyping process, repeatability, safety, and comprehensiveness, as it is possible to develop and test different system parts concurrently.

Also, it is possible to test different driver assistant (AD) functions using a real vehicle immersed in a virtual environment or a mechanic structure with the basic vehicle interfaces (seat, paddles, and steering wheel) in a moving platform. A human driver is then responsible for testing the man-machine interfaces in an approach known as Driver-in-the-Loop (DiL). Not only is DiL used to test ADAS applications but also for driving monitoring systems, such as eye movement, fatigue, and driver attention (WAGNER, 2021), (PETERSSON; FLETCHER; ZELINSKY, 2005).

2.4 SIMILARITY METRICS

Different metrics can be used to compare scenario-based simulations. Stadler et al. (STADLER; GERMAN; DJANATLIEV, 2022) propose a multivariate scoring metric to evaluate the quality of virtual scenario-based resimulation when compared to real-world data. The authors use single metrics to compare trajectory deviation, driven maneuvers, and criticality of traffic situations. Based on single metrics, a multivariate metric level is obtained using a weighted sum, and the final value can vary from 0, indicating perfect resimulation, to 4, indicating high deviation between the resimulation and real-world data.

This work focuses on three main metrics to assess the similarity between simulations: criticality analysis, deviation analysis, and maneuver analysis.

2.4.1 Criticality Analysis

The criticality analysis is based on time-to-collision (TTC), introduced in 1972 by (HAYWARD, 1972). It corresponds to the time until a collision happens between two vehicles, given that both vehicles keep their current course and speed.

Figure 6 presents two situations where a collision between two vehicles is on course. First, in Figure 6a, the EGO vehicle and the Traffic Object (TO) travel along the same lane and in the same direction, with $v_{EGO} > v_{TO}$. Next, Figure 6b shows both vehicles traveling in the same lane but in opposite directions. For both circumstances, the TTC is given by Equation 2.1 (WAGNER, 2021):

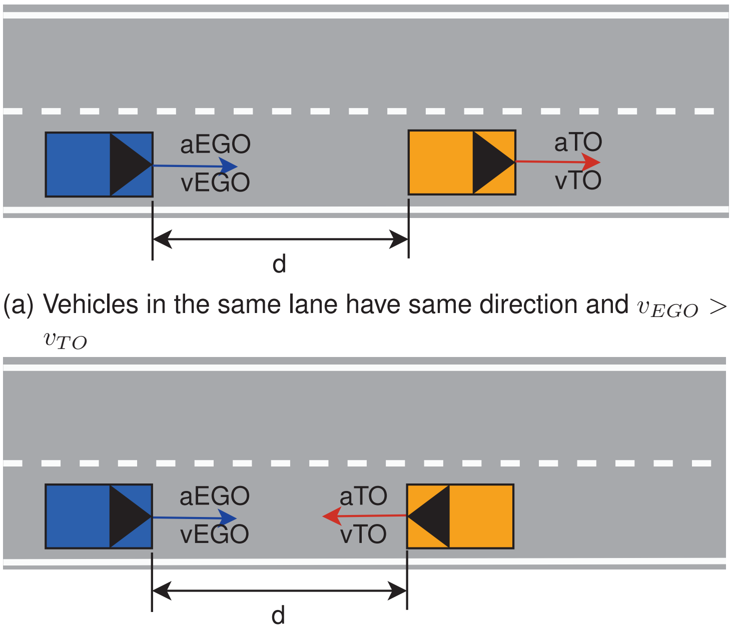
$$TTC = \frac{d}{v_{EGO} - v_{TO}}, \tag{2.1}$$

where v_{EGO} is the EGO velocity, v_{TO} is the traffic object velocity and d is the distance between both vehicles.

Another possible situation is presented in Figure 7, where both vehicles meet at an intersection. Supposing that both vehicles keep their trajectories without performing any turns or reducing their speed, the collision happens at a collision point P at a distance d from the vehicles. In this situation, the TTC is given by Equation 2.2 (HUBER et al., 2020):

$$TTC = \begin{cases} \dfrac{d_{EGO}}{v_{EGO}}, & \text{if } \dfrac{d_{TO}}{v_{TO}} > \dfrac{d_{EGO}}{v_{EGO}} > \dfrac{d_{TO} + l_{TO} + w_{EGO}}{v_{TO}} \\ \dfrac{d_{TO}}{v_{TO}}, & \text{if } \dfrac{d_{EGO}}{v_{EGO}} > \dfrac{d_{TO}}{v_{TO}} > \dfrac{d_{EGO} + l_{EGO} + w_{TO}}{v_{EGO}}, \end{cases} \tag{2.2}$$

where d_{EGO} is the distance of the Ego vehicle to the collision point, d_{TO} is the distance



(a) Vehicles in the same lane have same direction and $v_{EGO} > v_{TO}$

(b) Vehicles in the same lane with opposite directions

FIGURE 6 – Different cases of time-to-collision (WAGNER, 2021). Adapted from (HUBER et al., 2020)

of TO to the collision point, l_{EGO} and w_{EGO} are the length and width of the ego vehicle and l_{TO} and w_{TO} are the length and width of the TO, respectively.

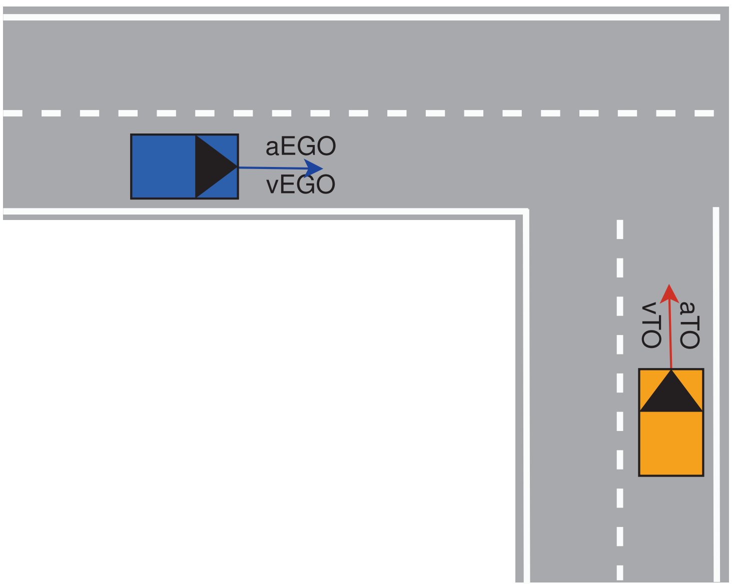


FIGURE 7 – Vehicles in an intersection. Adapted from (HUBER et al., 2020)

2.4.2 Deviation analysis

The analysis of driving trajectories has gained more attention in the field of traffic safety in the past few years. Various methods can be used to compare the similarity between different vehicle trajectories. According to (SOUSA; BOUKERCHE; LOUREIRO, 2020), when considering two paths A and B, and a distance function $D(A, B)$, the following conditions must be satisfied:

- *Uniqueness:* $D(A, B) = 0 \iff A = B$
- Nonnegativity: $D(A, B) \geq 0$
- Symmetry: $D(A, B) = D(B, A)$

Because two vehicles can perform the same trajectory at different speeds, for example, a vehicle performs a trajectory at 30 km/h and a second vehicle performs it at 60km/h, it's necessary for the similarity metric to consider local time shifts when computing the distance between trajectories. In other words, the time necessary to perform the trajectory should not be taken into account, only the trajectory itself.

Another important property is if the trajectories may have different lengths, the similarity metric needs to deal with the data having a different number of samples. Finally, the similarity metric compares only trajectories as a whole instead of segments (SOUSA; BOUKERCHE; LOUREIRO, 2020).

Two of the most common metrics for deviation analysis are the *Root Mean Square Error* (RMSE) and *Mean Absolute Error* (MAE). The MAE is given by Equation 2.3. It calculates the average magnitude of the error samples e in a subset of n samples. The error consists of the absolute difference between the expected value (y_j) and the observed value ($\hat{y_j}$), where all values have the same weight (CHAI; DRAXLER, 2014):

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |e_i| \tag{2.3}$$

The RMSE also measures the average magnitude of the error, but it calculates through the squared root of the average of the squared error. The RMSE calculation is given by Equation 2.4 (CHAI; DRAXLER, 2014):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2} \tag{2.4}$$

Another popular method is the *Fréchet Distance* (δ_F), a geometric shape-based measure commonly used to measure the similarity between two curves. The basic idea of this metric is to imagine a person taking his dog for a walk. Both walk in their respective curve with a leash between them. The man and the dog can change their speed but only go forward. The Fréchet distance is the minimum length of the leash that allows the person and the dog to walk their respective curve from the start to the endpoint (ARONOV et al., 2006). This metrics advantage allows comparing trajectories with different sizes and sample rates (SOUSA; BOUKERCHE; LOUREIRO, 2020).

Thomas Eiter and Heikki Mannila proposed a discrete variation of Fréchet distance called *coupling distance* δ_F. The variation for polygonal curves considers all the possible couplings between the endpoints of the line segments of the polygonal curves (EITER; MANNILA, 1994).

Another deviation metric is the cosine similarity, a measure used to determine the cosine of the angle between two non-zero vectors (HAN; PEI; TONG, 2022). It's defined as being the inner product of two vectors A and B divided by the product of their magnitude, as shown by Equation 2.5:

$$sim(A, B) = \frac{A \cdot B}{|A||B|}, \tag{2.5}$$

where $|A|$ is the magnitude of the vector A and $|B|$ is the magnitude of vector $|B|$.

The cosine similarity returns values between -1 and 1, where 1 means the vectors have the same orientation, 0 means the vectors are orthogonal (no correlation), and -1 indicates that the vectors are diametrically opposed (HAN; PEI; TONG, 2022).

2.4.3 Maneuver Analysis

Different actions can be performed during a driving task, and it's important for ADAS systems to detect and classify different maneuvers correctly. The driver can perform basic maneuvers, such as accelerating, braking, keeping the velocity, decelerating, and reversing, as well as more complex maneuvers that may involve more than one vehicle (HARTJEN et al., 2019).

Hartjen et al. defines a set of urban traffic maneuvers divided into three main categories: *vehicle state maneuver*, *infrastructure maneuver*, and *object-related maneuver*. The first one refers to all maneuvers that impact on vehicle's velocity, such as acceleration, deceleration, keeping velocity, and reversing. The second one refers to the possible interactions between the vehicle and the road infrastructure, like lane change, parking, and turns. Finally, object-related maneuvers are the ones in which the vehicle interacts with other traffic participants, either approaching an object, following, passing, or falling behind (HARTJEN et al., 2019). Table 2 presents the referred maneuvers that will be applied in the present work.

TABLE 2 – Urban maneuver model (HARTJEN et al., 2019)

Vehicle State Maneuver	**Infrastructure Maneuver**	**Object-related Maneuver**
Accelerate	Follow lane	Follow object
Keep velocity	Lane change (left, right)	Approach object
Decelerate	Turn (left, right)	Fall behind
		Passing

The same vehicle maneuver sequences are also described by Stadler et al., who compared the maneuvers executed during a real test drive with the ones executed during the simulation. For each timestamp t, it is verified if there is a match between both real-world and simulation data. A boolean value of 1 is assigned in case the same maneuver is executed in the same timestamp and 0 otherwise (STADLER; GERMAN; DJANATLIEV, 2022).

2.5 ISO 26262: ROAD VEHICLES – FUNCTIONAL SAFETY

ISO 26262 is the functional safety standard for electrical and electronic systems in road vehicles. It's based on the International Electrotechnical Commission (IEC) 61508 and adapted for automotive systems. The standard provides requirements for functional safety management and establishes four Automotive Safety Integrity Levels (ASILs). The levels are based on three criteria: severity, exposure, and controllability (STANDARDIZATION, 2018), defined below.

- **Severity level:** represents the hazard potential that the system can cause to its passengers.
- **Exposure level:** refers to the probability of occurrence.
- **Controllability level:** determines how the system can mitigate the damages.

ASIL level A is the lowest level, in which the system can cause minor injuries to the passengers. For example, rear lights are classified as ASIL A and need minor safety requirements. ASIL level B, for example, are the headlights and brake lights, while ASIL C can be ADAS functionalities such as automotive cruise control (XIE et al., 2023). Finally, ASIL level D can cause major injuries to the passengers (WAGNER, 2021) and needs to be implemented according to the most stringent safety requirements. For example, if the brakes fail during a high-speed maneuver, it can seriously harm the passengers.

One of the consequences of ISO 26262 was the creation of the V-model process, which establishes a process for developing safe-critical systems in the automotive sector. Figure 8 shows a simplified version of the V-model process according to Pegasus (PEGASUS, s.d.).

2.6 SIMULATION SOFTWARES

This section presents different simulation software used during this work to specify, create and visualize scenarios.

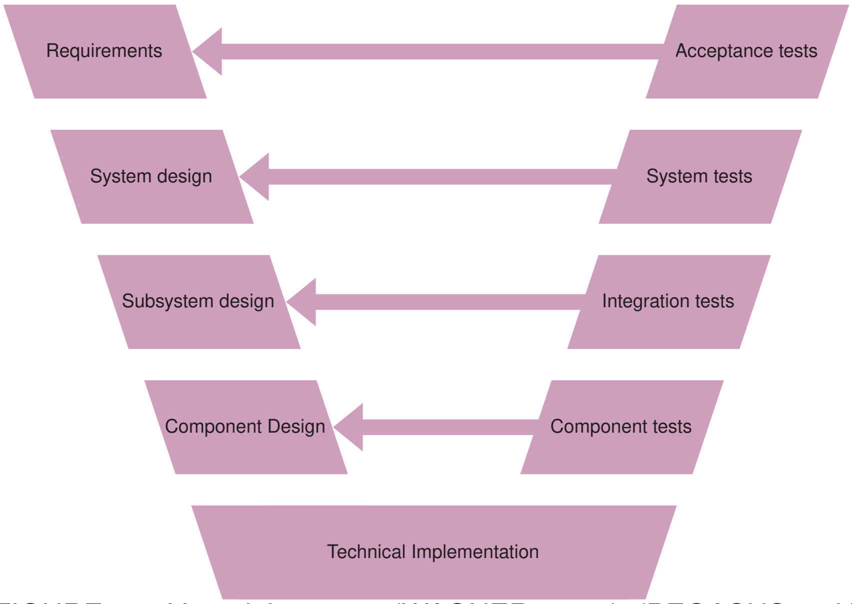


FIGURE 8 – V-model process (WAGNER, 2021), (PEGASUS, s.d.)

2.6.1 Environment Simulator Minimalistic (esmini)

Environment Simulator Minimalistic (esmini) is a basic OpenScenario player that allows the design and analysis of traffic scenarios (OPENSCENARIO PLAYER, 2023). It contains two main libraries: RoadManager and ScenarioEngine. The first one is responsible for providing an interface to OpenDRIVE format and the second one provides an interface to OpenSCENARIO format (ESMINI, 2023). Figure 9 presents an example of a simulation running on the esmini interface, where the ego vehicle is on the collision curse of a pedestrian.

FIGURE 9 – Example of a pedestrian collision simulated on esmini (ESMINI, 2023)

2.6.2 Road Runner

RoadRunner is a software designed to simulate and test automated driving systems (INC., 2023). It allows the creation of 3D scenes, scenarios, and roadway tracks, from the road specification to road signs, marks, signals, buildings, and other 3D models. Also, the tool supports the import and export of OpenDRIVE files.

Figure 10 shows the scenario editor of Road Runner, where the user can select different vehicles and define their trajectories, velocities, and actions. The tool also offers scene creation options to determine the road network.

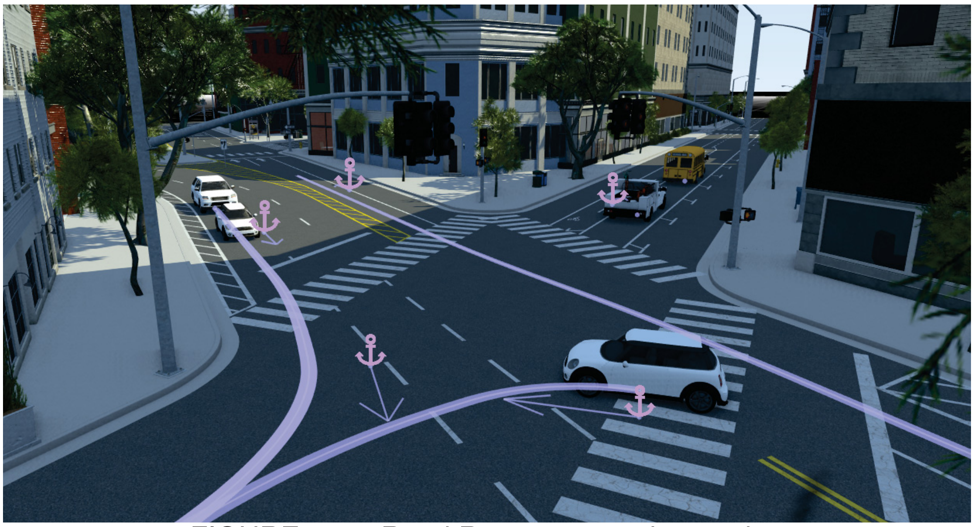


FIGURE 10 – Road Runner scenario creation

2.6.3 CarMaker

Carmaker is a simulation software designed to develop and test solutions for automated driving, ADAS functions, power train, and vehicle dynamics. It was developed by the company IPG Automotive, and it allows real-time simulations, vehicle modeling, and adaptive driver model, and it's suited for hardware-in-the-loop (HiL) tests.

The two main components are the IPGMovie, which allows a 3D visualization of the simulation, and the IPGControl for data analysis (AUTOMOTIVE, 2023). Figure 11 presents the CarMaker user interface.

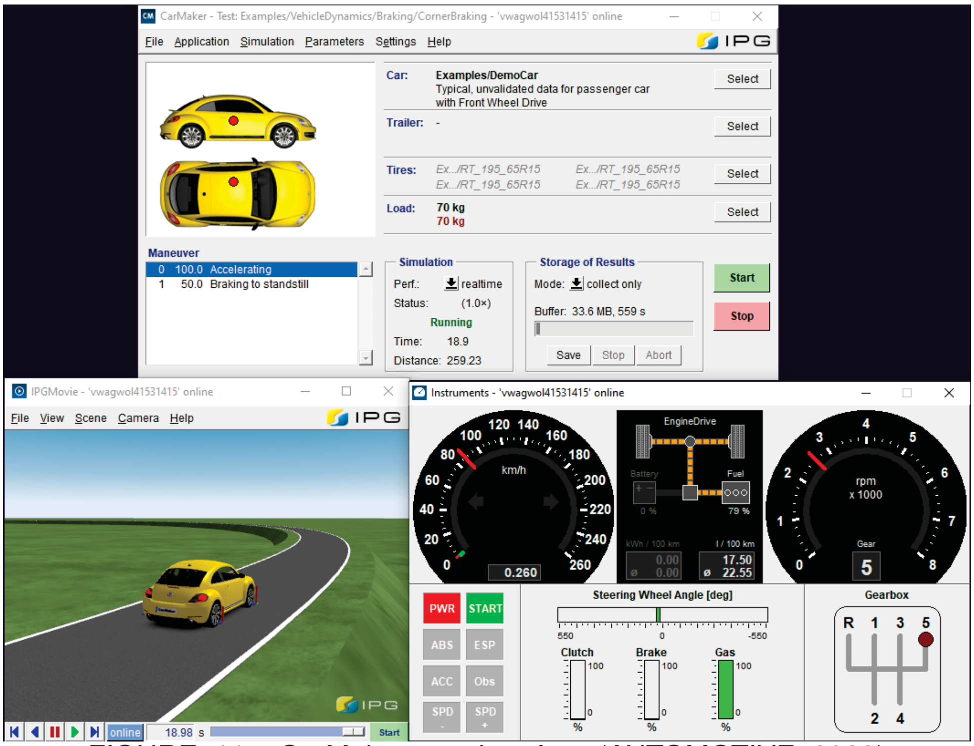


FIGURE 11 – CarMaker user interface (AUTOMOTIVE, 2023)

The interface allows the user to select multiple vehicle models, mount different sensors (i.e., radars, cameras, and lidars), and implement ADAS for emergency braking and ACC controller.

2.6.4 CARLA

CARLA is an open-source simulator designed to support the development of autonomous driving systems. It provides open-source code, protocols, and digital assets (such as urban layouts, buildings, and vehicles). Also, it makes it possible to run simulations with different sensor suites, environmental conditions, and static and dynamic actors (DOSOVITSKIY et al., 2017). Figure 12 shows a simulation running on Carla.

FIGURE 12 – Example of Carla simulation

3 METHODOLOGY

This chapter describes all the important steps taken throughout this work to achieve the proposed objectives. Section 3.1 describes the Functional Engineering platform adopted in this work. Section 3.2 presents the system workflow. Section 3.3 explains the analysis of *FEP Record* data. Section 3.4 presents the user interface. Section 3.5 describes the four different use cases adopted in this work. Section 3.6 describes each one of the metrics for evaluation (deviation analysis, maneuver analysis, and criticality analysis). Finally, Section 3.7 describes how the overall score is calculated based on similarity metrics.

3.1 FUNCTIONAL ENGINEERING PLATFORM (FEP)

As this work intends to compare simulations that run on different tools with different setups, it was decided to use the Functional Engineering platform (FEP) concept, developed by AUDI AG, to deal with the complexity of vehicle function networking. The FEP is a middleware between different platforms that work together and use different input/output data types. The model offers an FEP library to support the development and tests through the whole V-Model (GÜHMANN; RIESE; VON RÜDEN, 2016) and can be applied to communicate different simulation participants with each other.

The FEP participants used in this work are the *FEP Provider*, *FEP Visualization*, *FEP Record* and *FEP Control*, the last one responsible for controlling all the other FEP participants through a state machine. Figure 13 presents an overview of the different states of an FEP system state machine. After setting the participant properties (i.e., OpenSCENARIO path), the system can be loaded, initialized, and started.

The *FEP provider* interface provides all the simulation data, such as world state, vehicle initial and current position, cartesian speed, acceleration, object info (vehicle model, year, dimensions), simulation time, and other data necessary for future analysis of the simulation.

The *visualization* participant is a viewer for the simulation states received from the *FEP provider* and allows the user to see and analyze the simulation in real time. Finally, *the FEP record* saves all the simulation data provided by *the FEP provider* into a CSV file, allowing future data analysis.

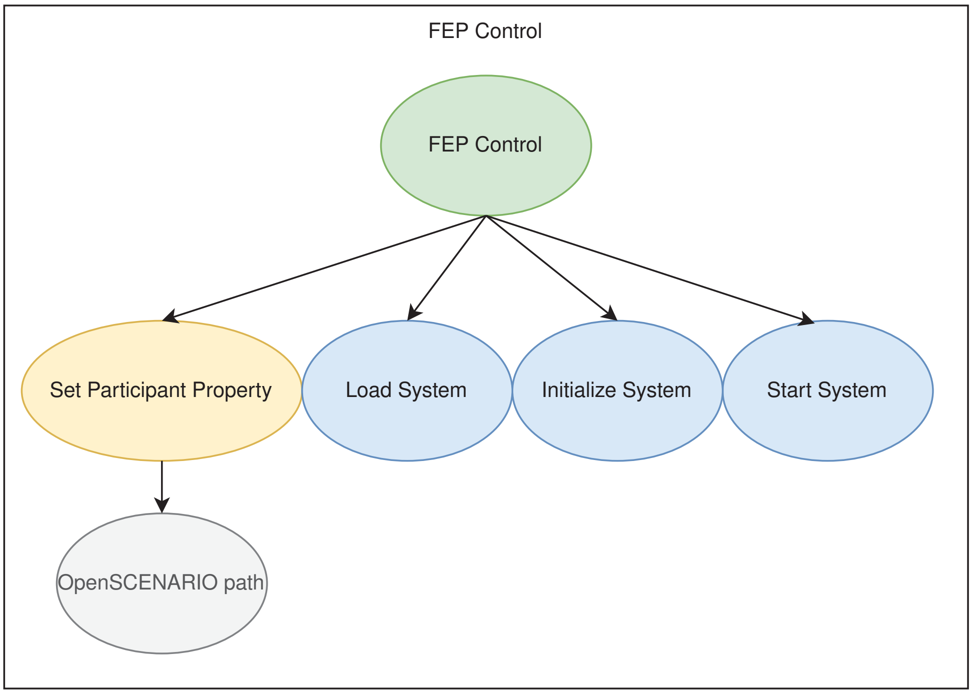


FIGURE 13 – FEP Control state machine

3.2 SYSTEM WORKFLOW

The main workflow is presented in Figure 14. The input data consists of simulation data stored as CSV files by *FEP record*. After selecting the simulation files for comparison, the user needs to specify which type of scenario is being evaluated (cut-in scenario, CCRs, CCCscp, or Ego-Only). As an optional field, selecting the OpenDRIVE file to plot the road network under the vehicle's trajectories is possible. Finally, the user can define the weighting factors for each of the metrics of interest (deviation, maneuver and criticality analysis) and the threshold for the TTC analysis. In case the user doesn't fill any value, there are pre-defined values used to calculate the final overall Score.

Based on the input, the Evaluation Tool processes the data, filtering and comparing both simulations according to the user-specified scenario type. The Cut-in, CCRs, and CCCscp scenarios consist of collision detection, criticality analysis, trajectory analysis, and maneuver analysis, while the ego-only scenario consists only of the trajectory and maneuver analysis.

The trajectory analysis is given by the cosine similarity between the two ego trajectories. The maneuver analysis compares the maneuvers performed by the ego vehicle in both simulations, as well as the maneuver duration and the time shift between maneuvers. The criticality analysis consists of the TTC comparison in percentage and the time for minimum TTC in both simulations. Once the overall score is calculated, the user can visualize the results and plot different data for more detailed analysis. The plotting options are described in Section 3.4.

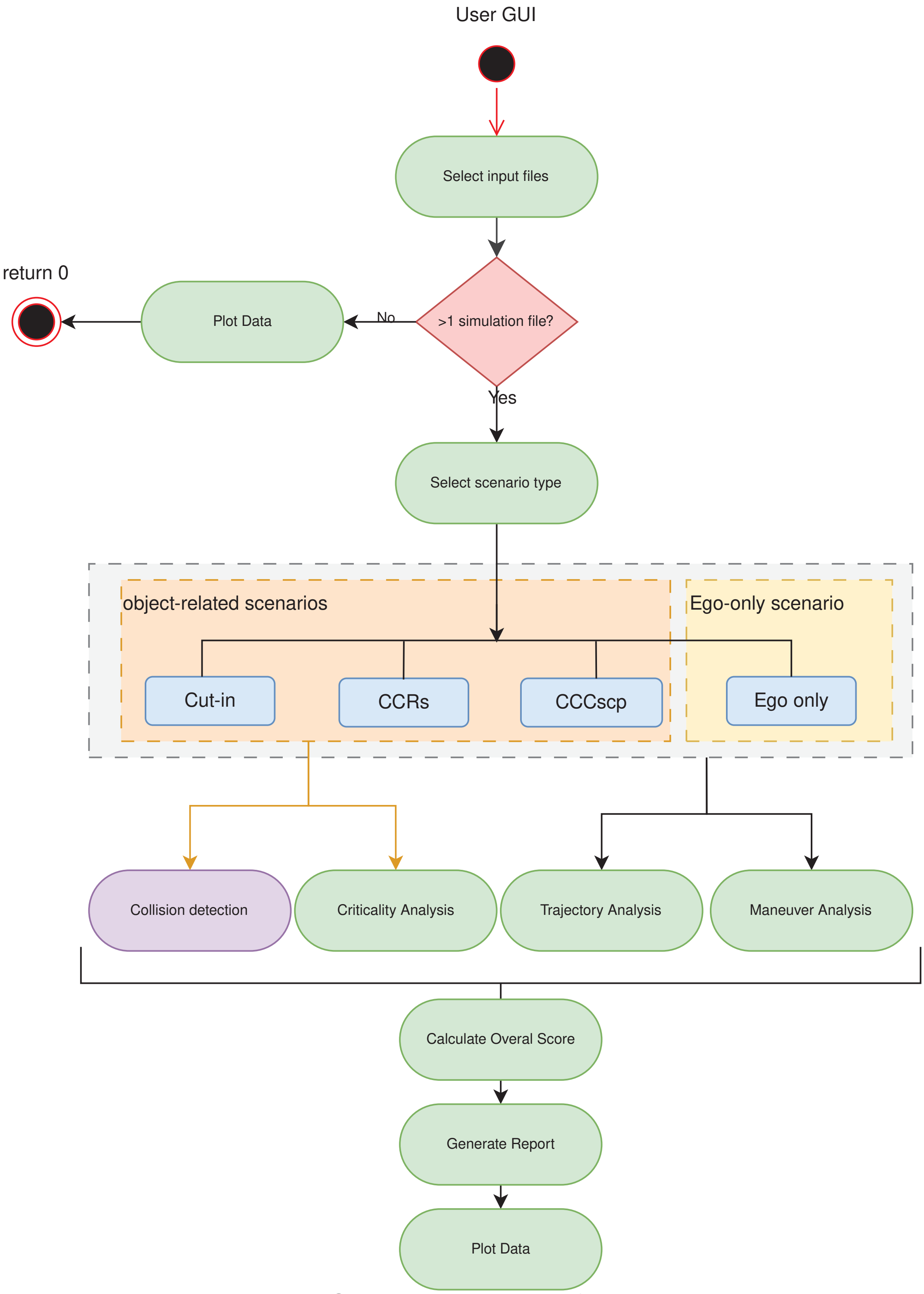


FIGURE 14 – Main workflow

3.3 DATA ANALYSIS

The first step to compare the two simulation files was to analyze the data recorded from *FEP Record*. For this, it was used a Python package called *Pandas* (MCKINNEY et al., 2010), which is an open-source data analysis tool used to work with data sets.

The data used in this work was the object ID (*object_id*), vehicle position *x* and *y* (*pos_cartesian.x*, *pos_cartesian.y*), the heading angle *h* (*pos_cartesian.h*), the lane ID (*pos_road.lane_id*), and the vehicle longitudinal speed (*speed_longitudinal*).

3.4 USER INTERFACE

As this work intends to automate the comparison between different scenario files based on similarity metrics, a user interface was developed using *CustomTkinter* Python library, which provides an interface to Tcl/Tk GUI toolkit, able to run across both Windows and Unix platforms (CUSTOMTKINTER. . . , s.d.). Figure 15 shows the main user interface.

The user can manually select two CSV files of recorded simulations for comparison, specify the simulation names, select the openDRIVE file, choose the scenario type, and specify the weights for the similarity metrics and threshold for the time-to-collision analysis. The required fields are the first file and the scenario type. The other fields have pre-defined values. If there is no second file for comparison, the interface will only analyze and present the data regarding the first file.

After clicking on the *compare* button, the simulations will be compared, and the results will be presented in the *Results* window. Figure 16 shows an example of a cut-in scenario being analyzed. The interface first specifies the expected result (first simulation file added for comparison) and the achieved result (second file added for comparison). Next, four main sections are presented: the deviation analysis, maneuver analysis, criticality analysis, and the final overall score.

The Deviation analysis shows the match between the trajectories in both simulations. The maneuver analysis shows if a collision between the ego vehicle and the traffic object was detected during the simulation. Also, the maneuver duration for both simulations is shown, which allows calculating the time shift between simulations. In the example, no object-related maneuver was detected for the second simulation, and an unexpected collision happened. Finally, the final maneuver match score is displayed.

The next main section is the criticality analysis, which allows the user to have an overview of object-related events for scenarios containing more than one traffic participant. The interface shows the minimum TTC for both simulations and at which

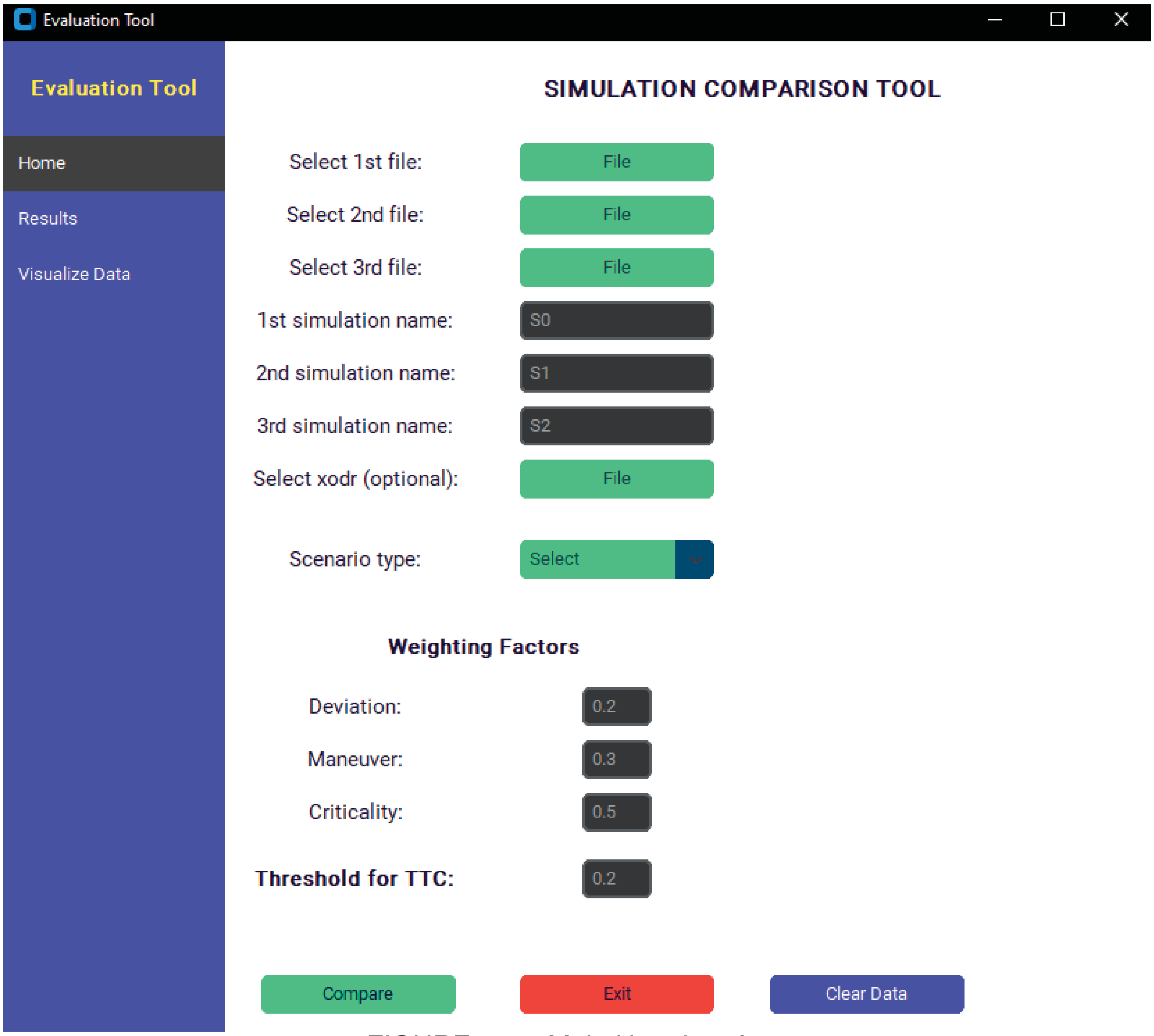


FIGURE 15 – Main User Interface

time the minimum TTC values were detected, allowing the calculation of the time shift between simulations, if there is any. If the minimum TTC value is below the threshold, probably a collision between objects happened, and the simulation failed, as shown in the example given. The interface then shows the matching percentage for the TTC between both simulations.

The final section is the overall score between simulations. As the goal is to compare if the scenario behaves the same when running on different platforms, a 100% match indicates that the scenario is behaving as expected, and no problems were found in the simulation.

Finally, Figure 17 shows the *Visualize Data* tab. The section allows the user to plot the data recorded by *FEP Record* in case a more detailed scenario analysis is required. It also allows plotting the criticality value for both simulations, the Euclidean distance between objects, and the percentage match over time for each one of the similarity metrics. This can be used to better analyze the scenario and check at which

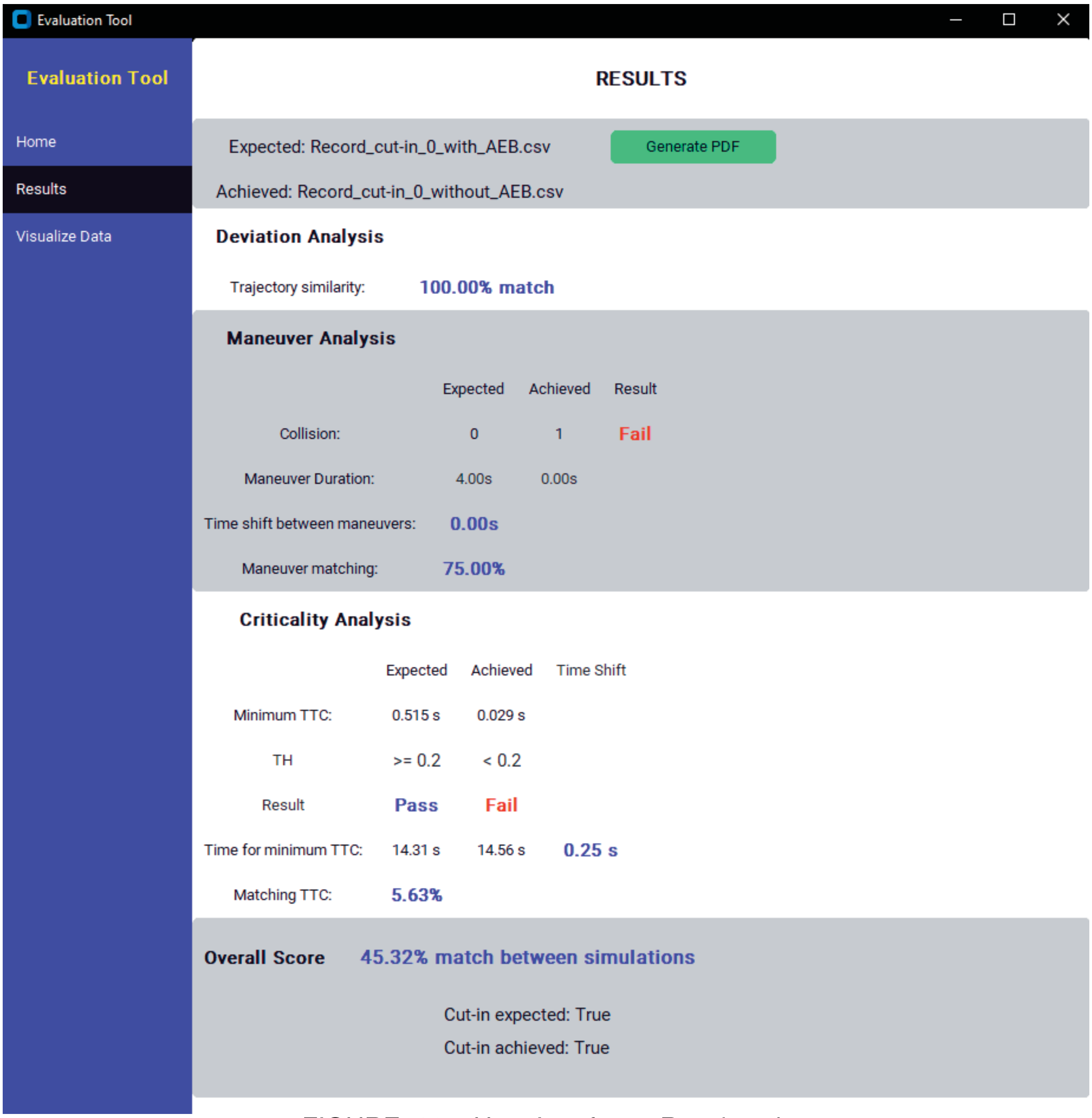


FIGURE 16 – User Interface - Results tab

point the scenario stopped behaving as expected in case there is a low match between simulations.

3.5 SCENARIO TEST-CASES

This section describes the different scenario types and use cases evaluated in this work: cut-in scenario, Car-to-Car Rear Stationary (CCRs), Car-to-car Crossing Straight Crossing Path (CCCscp), and Ego-Only. For each test case, a different scenario was created using OpenScenario Editor, and the scenario was then tested on three platforms: esmini, Carla, and CarMaker.

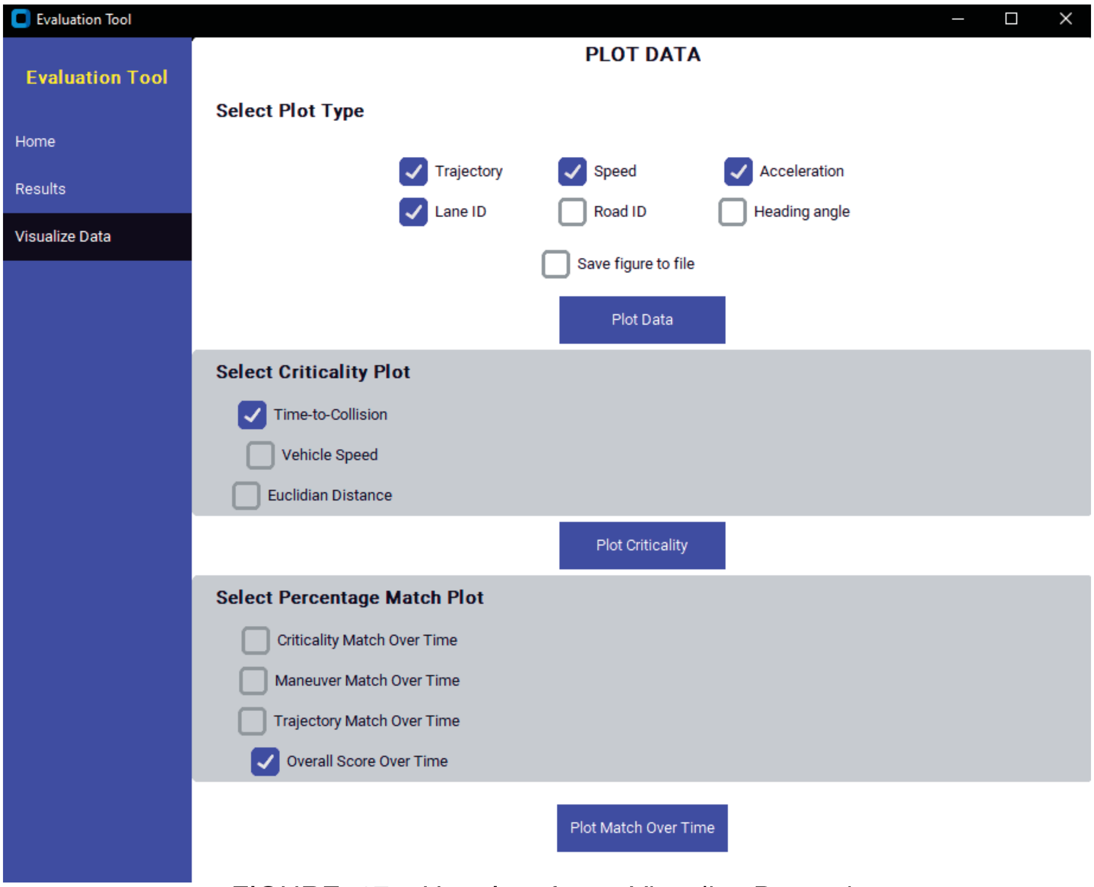


FIGURE 17 – User Interface - Visualize Data tab

3.5.1 Case Study: Cut-In Scenario

The first use case to be analyzed is a cut-in scenario, in which an adjacent vehicle performs a lane-change maneuver in front of the ego vehicle within a short distance, forcing the ego vehicle to decelerate to avoid a collision. Figure 18 shows an example of a cut-in maneuver.

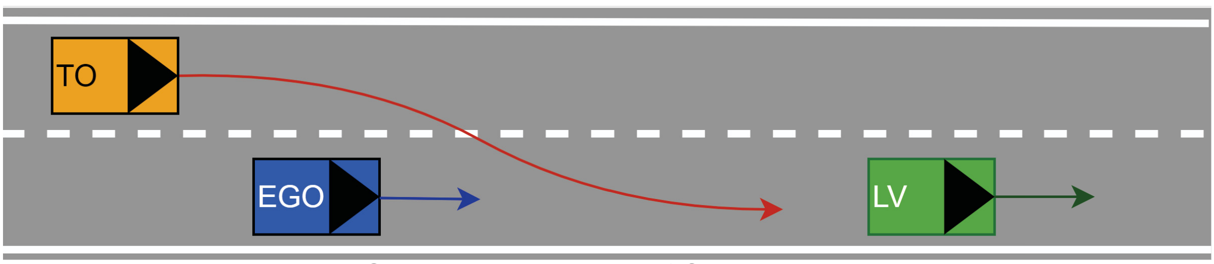


FIGURE 18 – Example of Cut-In scenario

To evaluate the cut-in scenario, the following actions and conditions were defined in the Open Scenario Editor:

1. The ego vehicle and a TO are positioned in adjacent lanes, with the TO in front of the ego vehicle. Both vehicles have zero initial speed.

2. The ego vehicle accelerates to 70km/h.

3. The TO accelerates to 60km/h.

4. If the time-to-collision between the ego vehicle and TO is less than 2.5s, the TO will perform a lane change in front of the ego, forcing the ego to decelerate to 50km/h to avoid a collision.

The same scenario was evaluated on esmini, Carla, and CarMaker. The results are shown in the section 4.1.

3.5.2 Case Study: Car-to-Car Rear Stationary (CCRs)

The second use case is based on the Euro NCAP test protocol for Car-to-Car systems (EURO, 2022) to test Autonomous Emergency Break (AEB) systems. The scenario is illustrated in Figure 19. The ego vehicle starts accelerating in a straight line, and a traffic object with zero speed is positioned in front of the ego vehicle, forcing the ego to activate the AEB system in order to avoid a collision.

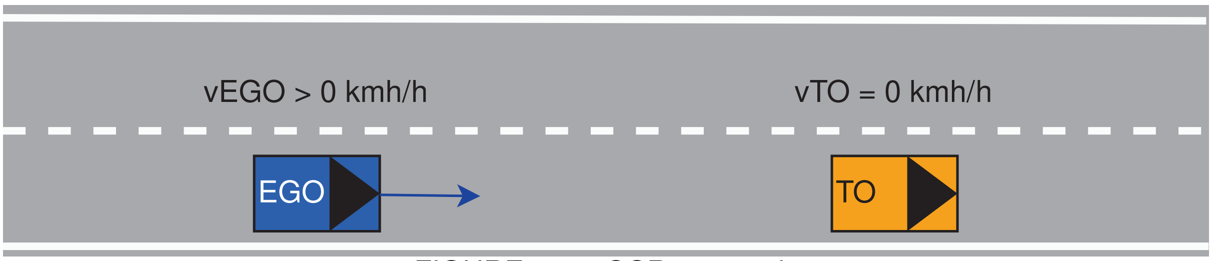


FIGURE 19 – CCRs scenario

To test this use case, the following actions and conditions were defined in the Open Scenario Editor:

1. The ego vehicle starts with zero initial speed and accelerates to 30 km/h.

2. The TO stays stationary in the same lane as the ego vehicle.

3. If the TTC between both vehicles is less than 1s, the ego vehicle decelerates to 0km/h.

The results for the CCRs use case are presented in section 4.2.

3.5.3 Case Study: Simple Trajectory

The third proposed scenario consists of three simple trajectories to be performed by the ego vehicle without any other scenario object. The goal is to evaluate how the dynamics affect the simulation, how each software interprets the scenario file, and if the evaluation tool is capable of detecting different behaviors in the simulations.

The first case is a simple straight line where the ego vehicle varies the speed five times between 0 and 80km/h. The second case is a three-lane scenario in which the vehicle changes lanes four times at different timestamps.

The third scenario is presented in Figure 20. In this case, the goal is to evaluate driver behavior and vehicle dynamics on different platforms when running a more complex scenario that involves both turns and lane changes.

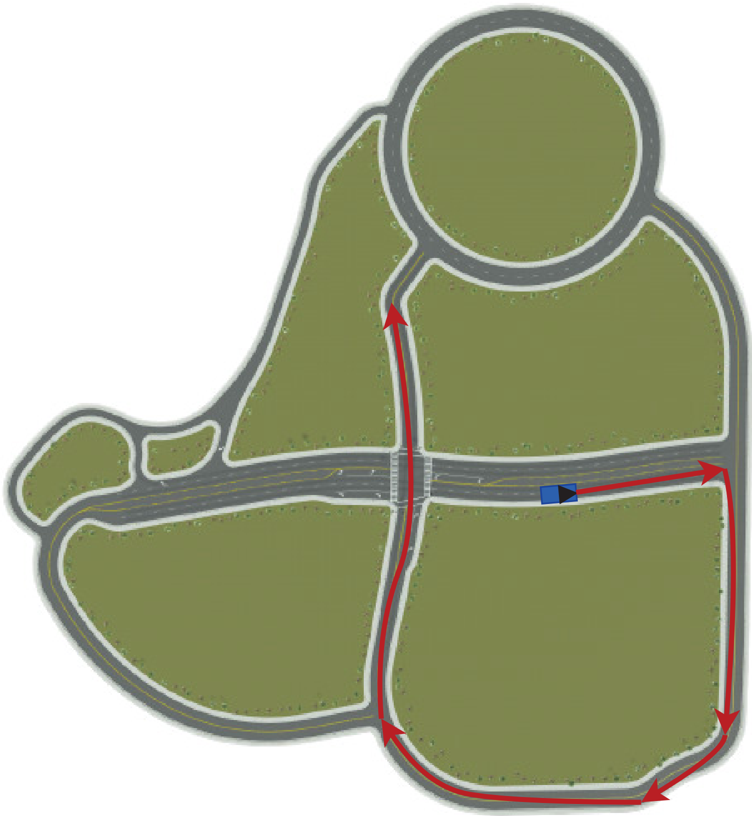


FIGURE 20 – Simple Trajectory

The results for this test case are presented in Section 4.3.

3.5.4 Case Study: Car-to-car Crossing Straight Crossing Path (CCCscp)

The last scenario is also described in the Euro NCAP test protocol. In this scenario, the ego vehicle travels in a straight line across a junction at 20km/h. Meanwhile, a target object (TO) also moves in a straight line at 20km/h, perpendicular to the path of the ego vehicle, as illustrated in Figure 21.

The results for this test case are presented in Section 4.4.

3.6 SIMILARITY METRICS

This section presents the similarity metrics defined in this work to calculate the match percentage between simulations.

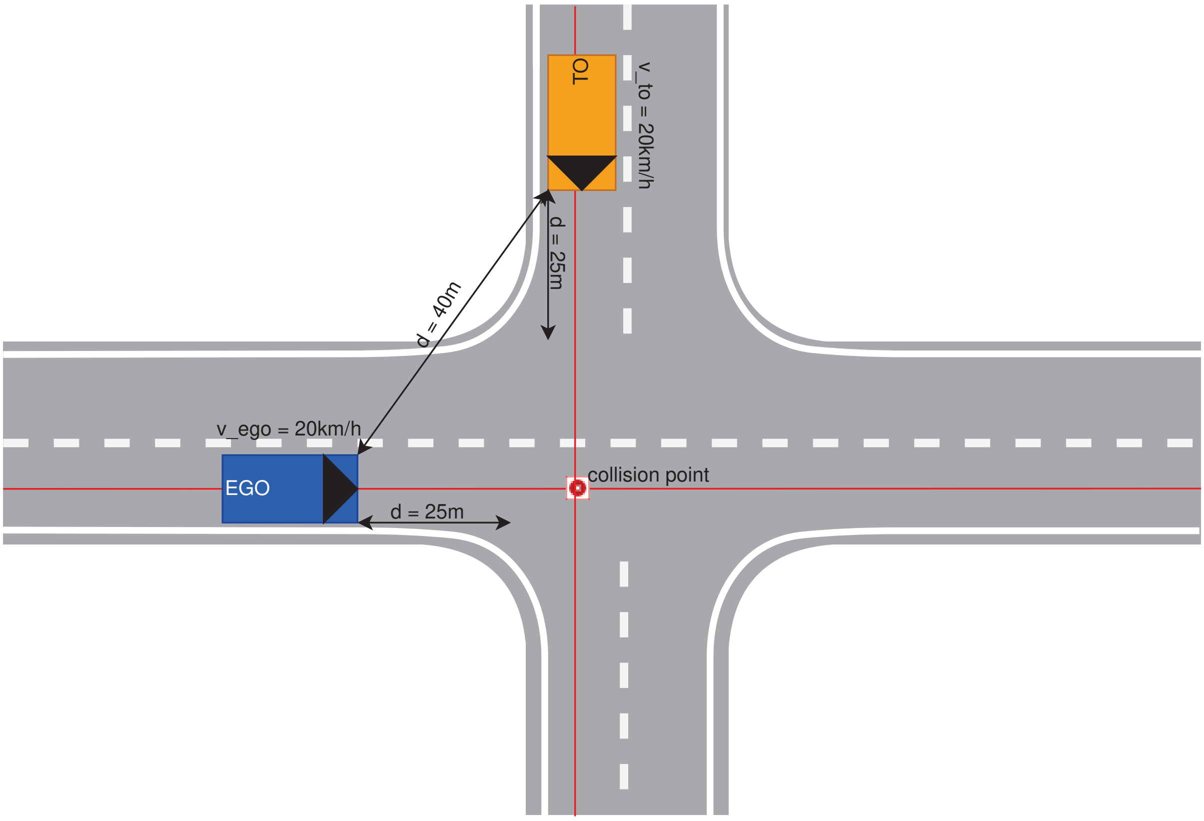


FIGURE 21 – CCCscp scenario

3.6.1 Deviation Analysis

The deviation analysis is based on cosine similarity and computed as the normalized dot product of two trajectories *A* and *B*, as described in Section 2.4.2.

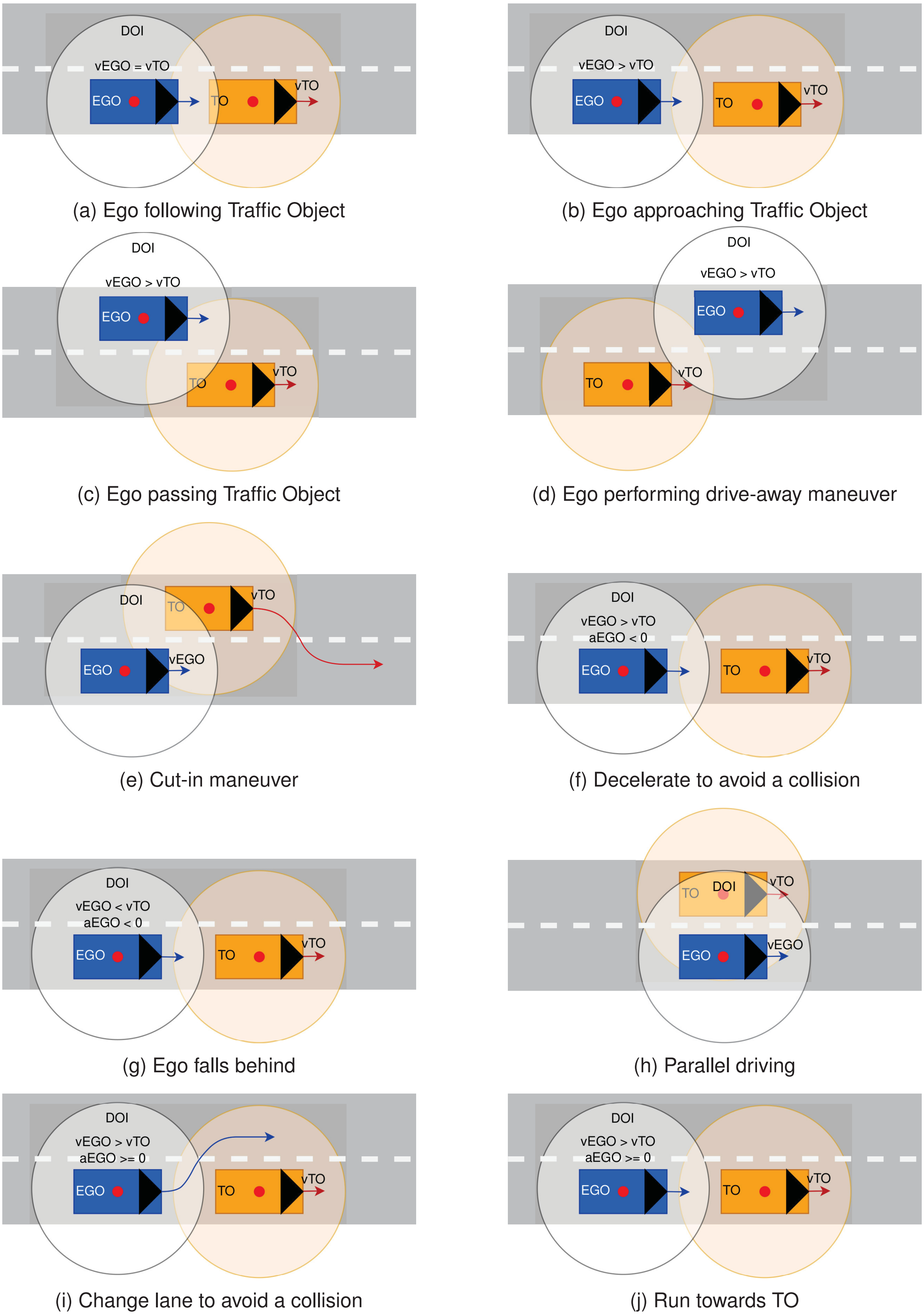
First, both vehicle trajectories are saved in an array containing the x and y coordinates. Next, because one simulation may run for longer than another, both arrays are adjusted to the size of the shortest simulation in such a way that the magnitude of both simulations doesn't affect the similarity. This is done by cutting-off the data of the longest array. Then, the similarity is computed according to equation 2.5.

It's important to mention that this metric focuses on the direction of both trajectories instead of the magnitude. In other words, it doesn't consider that a trajectory *B* may be longer than *A*, for example.

3.6.2 Maneuver Analysis

The maneuvers considered in this work are based on those shown in Table 2, adding the following definitions for object-related maneuvers: *drive away*, *parallel driving*, *cut-in*, *run towards ego/TO*, *decelerates to avoid collision*, and *change lane to avoid a collision*. The possible object-related maneuvers are shown in Figure 22.

For each timestamp, it was evaluated the vehicle state maneuvers, infrastructure



(a) Ego following Traffic Object

(b) Ego approaching Traffic Object

(c) Ego passing Traffic Object

(d) Ego performing drive-away maneuver

(e) Cut-in maneuver

(f) Decelerate to avoid a collision

(g) Ego falls behind

(h) Parallel driving

(i) Change lane to avoid a collision

(j) Run towards TO

FIGURE 22 – Object-related maneuvers defined in this work

maneuvers, and objected-related maneuvers. The last one only happens if the TO is in the range of the ego's domain of interest (DOI), set as a radius of 5m from the ego's center coordinates.

Because the vehicle can execute different maneuvers simultaneously (accelerate while changing lanes, for example), a list of maneuvers was obtained for each timestamp. Table 3 presents an example of maneuver extraction.

TABLE 3 – Example of ego's maneuver extraction

Time (s)	simuation 1	simulation 2	Intersection	Total matches
0.1	keep velocity, keep lane, -	keep velocity, keep lane, -	keep velocity, keep lane, -	3
0.5	decelerate, keep lane, fall behind	keep velocity, keep lane, follow	keep lane	1
1	decelerate, keep lane, fall behind	decelerate, keep lane, fall behind	decelerate, keep lane, fall behind	3

Notice that no objected-related maneuver is being executed at the simulation's beginning, as no previous states are known.

Next, it's necessary to check if the maneuvers are executed in the same order. This is done by removing all the duplicated sequences of maneuvers. To calculate the maneuver similarity, the achieved match between simulations is divided by the ideal match. In the example above, the perfect match would be 9, as there are 3 possible maneuvers for each time stamp, and the achieved value is 7. The final maneuver percentage match is then $7/9 * 100 = 77.78\%$ match.

3.6.3 Criticality Analysis

The criticality analysis consists of calculating the TTC for both simulations as described in 2.4.1, obtaining the time for the minimum TTC, and checking for possible collisions between the ego vehicle and other traffic participants.

For the collision detection algorithm, instead of representing the vehicles with a rectangular bounding box, it was used a different approach in which the vehicle is represented with three circles with the same radius, as shown in Figure 23. Each circle's radius equals half of the vehicle width ($r = w/2$).

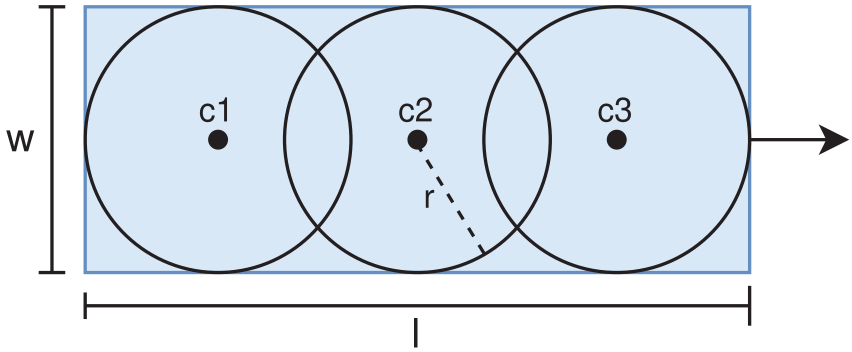


FIGURE 23 – Vehicle model used for detecting possible collisions

To simplify the model, vehicles with the same dimensions were used in the simulation, with a length of $4.7m$ and a width of $2m$. With the vehicle dimensions and the radius value, it's possible to obtain the distance between the three centers as being 1.5 the value of the radius. In Figure 23, C_1 is the x and y coordinates of the vehicle given by *FEP record* interface. As the interface also provides the heading angle of the vehicle (h), it's possible to calculate the x and y coordinates of centers c_2 and c_3 using trigonometry. Equations 3.1 and 3.2 present how to calculate the variation in x and y and Equations 3.3 and 3.4 present how to obtain the coordinate of c_2 and c_3 based on c_1, respectively.

$$\Delta y = \sin(h) \cdot 1,5r \tag{3.1}$$

$$\Delta x = \cos(h) \cdot 1,5r \tag{3.2}$$

$$c2(x, y) = c1(x, y) + (\Delta x, \Delta y) \tag{3.3}$$

$$c3(x, y) = c2(x, y) + (\Delta x, \Delta y) \tag{3.4}$$

If the distance of each one of the centers of the first vehicle to the centers of the second vehicle is less than 2 times the value of the radius, a collision is detected. In other words:

$$d(C_{1n}, C_{2m}) <= 2 \cdot r,$$

where C_{1n} is the center C_n of the first vehicle and C_{2m} is the center C_m of the second vehicle, with n, m varying from 0 to 3.

The time-to-collision was calculated according to Equation 2.1 and 2.2. First, the distance between the vehicles was calculated following the 3 circles approach described above. For each pair of circles, the distance was calculated as shown by Figure 24. After calculating all the possible distances between the centers, the minimum distance was used to calculate the time to collision (see Equation 2.1).

After calculating the TTC for each timestamp of the simulation, the minimum value and the simulation time in which the minimum value occurs are obtained. This information can be used to detect both collisions and time-shift between simulations.

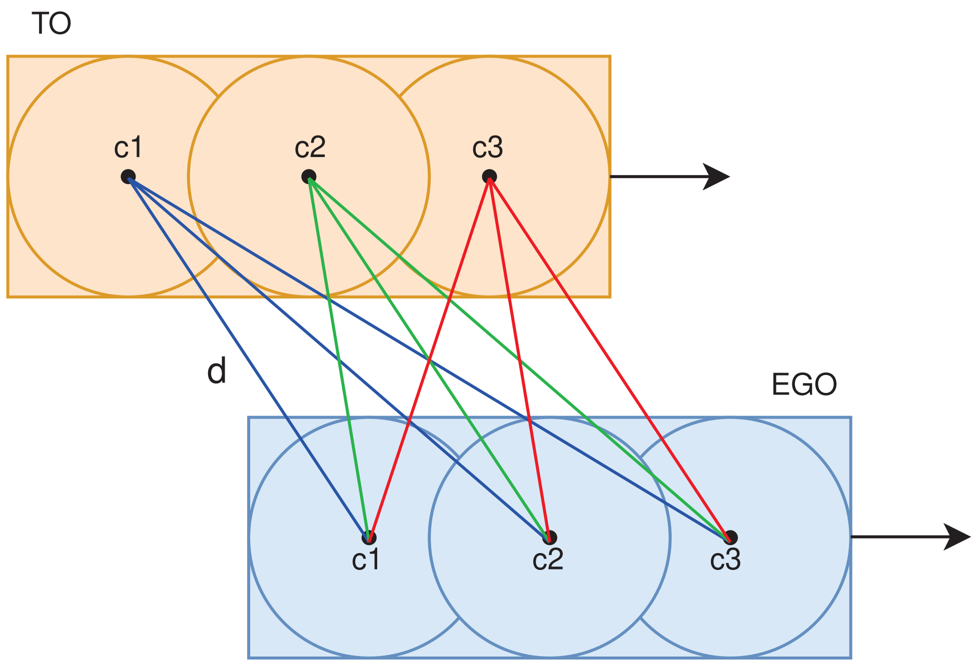


FIGURE 24 – Distance calculation between two vehicles based on 3 circles model

3.7 OVERALL SCORE

The overall score is the weighted sum of the deviation analysis, criticality analysis, and maneuver analysis. For each scenario type, different weights were defined based on which metrics are more relevant for the scenario. For example, in a cut-in scenario, the criticality and the maneuver are more relevant factors than the vehicle trajectory, thus having a higher value. But in an ego-only scenario, where the trajectory is the main factor, the criticality is not taken into account, as there is no other object in the simulation.

The weights assigned to each scenario type are detailed in Table 4. The values were obtained after testing and adjusting different weights for each scenario type. Additionally, the interface offers users the flexibility to adjust these weights as necessary.

TABLE 4 – Weights for Overall Score

	Deviation Weight	**Maneuver Weight**	**Criticality Weight**
Cut-in	0.2	0.3	0.5
CCRs/CCCscp	0.2	0.2	0.6
Ego-Only	0.6	0.4	0

After defining the weights, the overall score is calculated according to Equation 3.5:

$$\textit{OS} = W_{dev} \cdot \textit{deviation value} + W_{man} \cdot \textit{maneuver value} + W_{crit} \cdot \textit{criticality value} \quad (3.5)$$

4 RESULTS

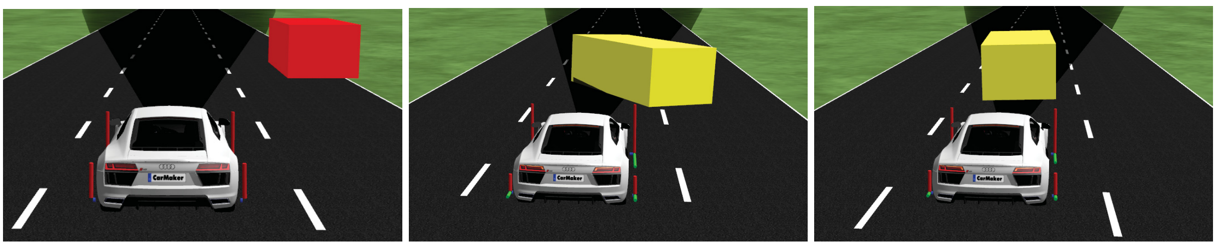
This section aims to present the results of the evaluation tool developed during this work. Section 4.1 presents the analysis of a cut-in scenario. Section 4.2 presents the study of the CCRs scenario. Section 4.3 analyses a scenario where there is only the ego vehicle without any other traffic participant. Finally, Section 4.4 analyses the CCCscp scenario.

4.1 CUT-IN ANALYSIS

The first scenario to be evaluated is the cut-in scenario, in which two test cases were defined. The first one compares the scenario with and without an Autonomous Emergency Braking (AEB) system. The second one models the scenario in OpenSCENARIO Editor to use actions and conditions.

4.1.1 Vehicle Equipped With and Without AEB System for Cut-in Analysis

For the first comparison, the CarMaker tool was used, and the ego vehicle was equipped with an AEB system consisting of a long-range radar of 150 meters with a FOV (Field of View) of 16 meters in height and 10 meters in width. As shown by Figure 25, the ego vehicle detects the second vehicle at a short distance and breaks to avoid a collision.



(a) Ego vehicle following lane (b) Cut-in maneuver (c) Ego vehicle breaks

FIGURE 25 – Ego uses AEB system to avoid collision during a cut-in maneuver

The same scenario was then executed without the AEB system and compared with the first case. The goal is to verify if the evaluation tool is able to detect the difference between both scenarios based on a set of similarity metrics and show the expected matches between maneuver, trajectory, and criticality.

Table 5 summarizes the result given by the evaluation tool. As expected, without an AEB system, the ego vehicle is unable to detect the TO, and there is a collision between both vehicles.

TABLE 5 – Comparison of Cut-in scenario - CarMaker with and without AEB system

	Carmaker with AEB system	**CarMaker without AEB system**
Minimum TTC (s)	0.515	0.029
Collision	No	Yes
Maneuver matching	78.26%	
Trajectory matching	100%	
Criticality matching	5.63%	
Overall score	46.29%	

Figure 26 shows the percentage match over time for both simulations. The interval in which the match is less than 80% is marked in yellow. If it's less than 60%, it's marked in red. Notice that both simulations have a good match until the AEB system is activated in one simulation and not activated in another. From the image, it is possible to conclude that the scenario behaves the same until the AEB system is triggered, at which point both scenarios diverge, causing the mismatch between maneuver and criticality.

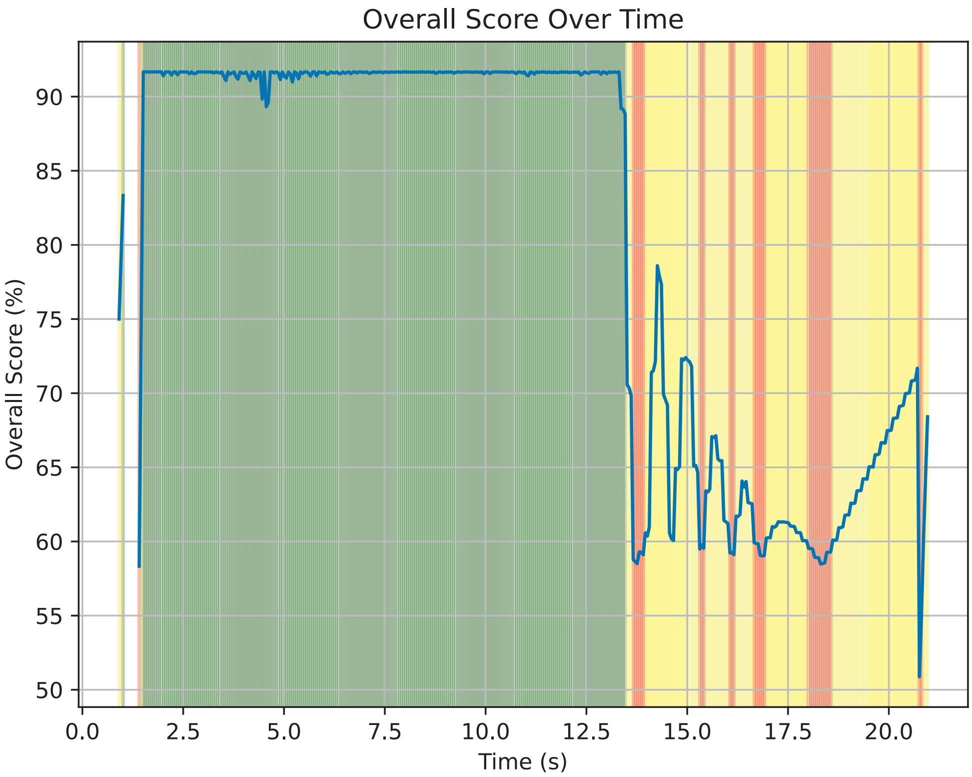


FIGURE 26 – Overall Score for CarMaker with and without AEB

4.1.2 Cut-in with brake reaction modeled with actions and conditions

The same cut-in scenario was then modeled using a time-to-collision condition instead of an AEB system, as shown in Figure 27.

Initially, both vehicles are positioned within a distance of 18 meters between each other, with the TO in front of the ego in an adjacent lane. The ego accelerates from 0 km/h to 70 km/h, and the TO accelerates from 0 km/h to 60 km/h. If the TTC is less than 2.5 seconds, the TO will perform a cut-in maneuver in front of the ego vehicle while the ego vehicle decelerates to 50 km/h in order to avoid a collision.

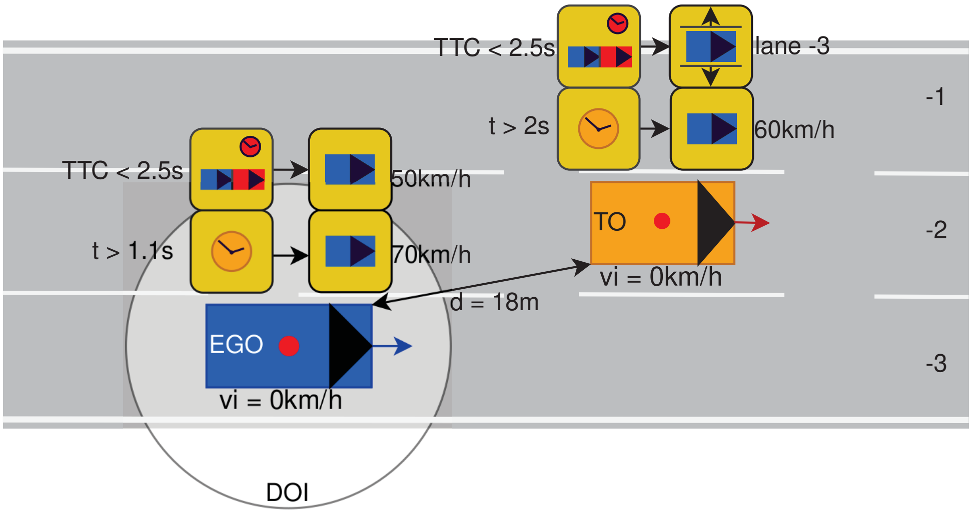


FIGURE 27 – Cut-in maneuver scenario defined in OpenScenario Editor

Table 6 presents the comparison between simulations when running the same scenario file on esmini (expected result), CarMaker, and Carla. Notice that in both CarMaker and Carla, the cut-in maneuver fails to happen.

TABLE 6 – Comparison between cut-in scenario: esmini, CarMaker and Carla

	Cut-inScenario			
	esmini	**CarMaker**	**esmini**	**Carla**
Minimum TTC (s)	1.014	4.096	1.014	2.778
Cut-in	Yes	No	Yes	No
Collision	No	No	No	No
Maneuver matching	20%		20%	
Trajectory matching	100%		100%	
TTC matching	24.76%		36.50%	
Overall score	38.38%		44.25%	

As the final score is calculated based on the values of maneuver matching and TTC, the final value is affected by the low values of both metrics, indicating a low match between the expected behavior and the achieved behavior for both Carla and CarMaker simulators.

During the simulation, the TO performed a lane change outside the ego's Domain of Interest (DOI), not triggering the deceleration action. This was detected by the evaluation tool, which was able to inform that the cut-in maneuver didn't happen. For this reason, the actions and conditions were adapted according to Figure 28.

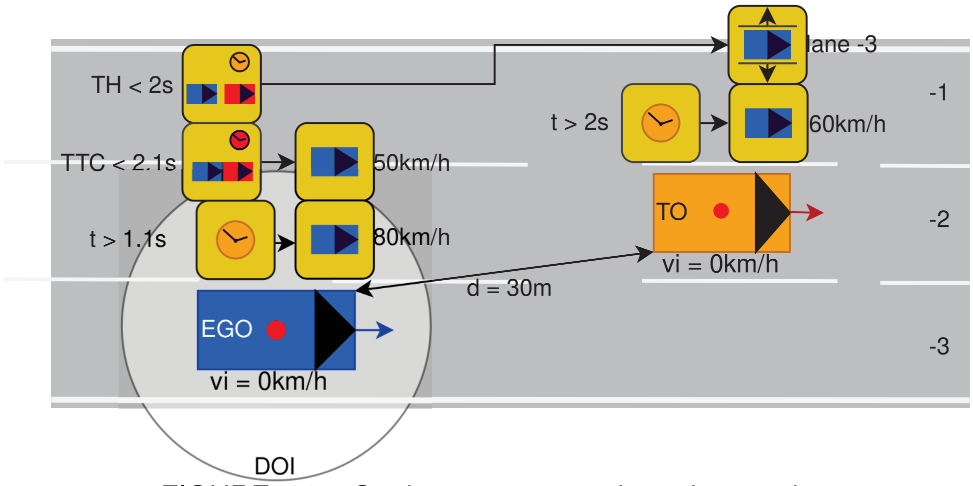


FIGURE 28 – Cut-in maneuver - adapted scenario

Because in the previous simulation the ego vehicle did not achieve the desired speed of 80 km/h on time, the distance between vehicles was increased to 30 meters. The condition for the lane change maneuver was also changed to time-headway (TH), in which the TO changes lane in case TH is less than 2 seconds. For the ego vehicle, if the TTC is less than 2.1 seconds, it will reduce the speed to 50 km/h.

After adapting the scenario, the simulation was executed again, and the results from *FEP Record* were analyzed by the evaluation tool. Table 7 shows the updated results.

As it is possible to notice, with the correct scenario specification, the match between simulations is higher, and the scenario triggers the expected actions in all the tools. The results also show that the evaluation tool is capable of correctly detecting variations in the simulations.

4.2 CCRS SCENARIO ANALYSIS

Next, the results for the Car-to-Car Rear Stationary scenario will be presented. For this use case, two variations were made. First, it was compared the scenario with and without the AEB system. Next, the same scenario was evaluated by modeling the

TABLE 7 – Comparison between adapted cut-in scenario: esmini, CarMaker, and Carla

	Cut-in Scenario			
	esmini	**CarMaker**	**esmini**	**Carla**
Minimum TTC (s)	1.251	1.153	1.251	1.423
Cut-in	Yes	Yes	Yes	Yes
Collision	No	No	No	No
Maneuver matching	72.22%		57.14%	
Trajectory matching	100%		100%	
TTC matching	76.42%		98.40%	
Overall score	79.88%		86.34%	

scenario using actions and conditions similar to the previous use case. The simulation was then compared between esmini, Carla, and CarMaker.

4.2.1 Vehicle Equipped With and Without AEB System for CCRs Analysis

The first comparison is running a CCRs scenario on CarMaker with and without the AEB system. The ego vehicle was set to accelerate from 0km/h to 40km/h in 30m while the TO was set to stand on the same lane with zero speed at a distance of 100m from the ego vehicle, as illustrated by Figure 29.

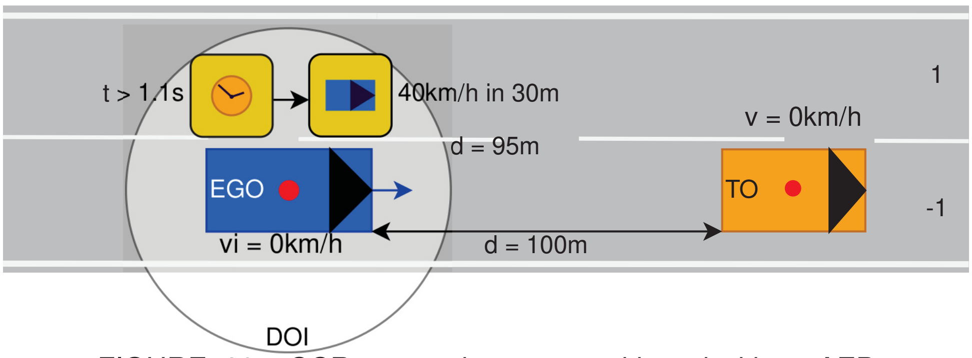


FIGURE 29 – CCRs scenario: system with and without AEB

For the simulation where the vehicle was equipped with an AEB system, if the distance between the ego vehicle and the TO is less than 20 meters, the AEB system is triggered to avoid a collision. The objective is to check if the Evaluation tool is able to analyze a CCRs scenario and detect the difference between both simulations.

Figure 30 shows the criticality analysis for CarMaker running with the AEB system. Notice that at approximately 22 seconds, the ego vehicle breaks to not collide with the traffic object. It's also possible to notice that the Euclidean Distance is approximately 20 meters when the AEB triggers. At this point, the TTC is the minimum, 1.817 seconds.

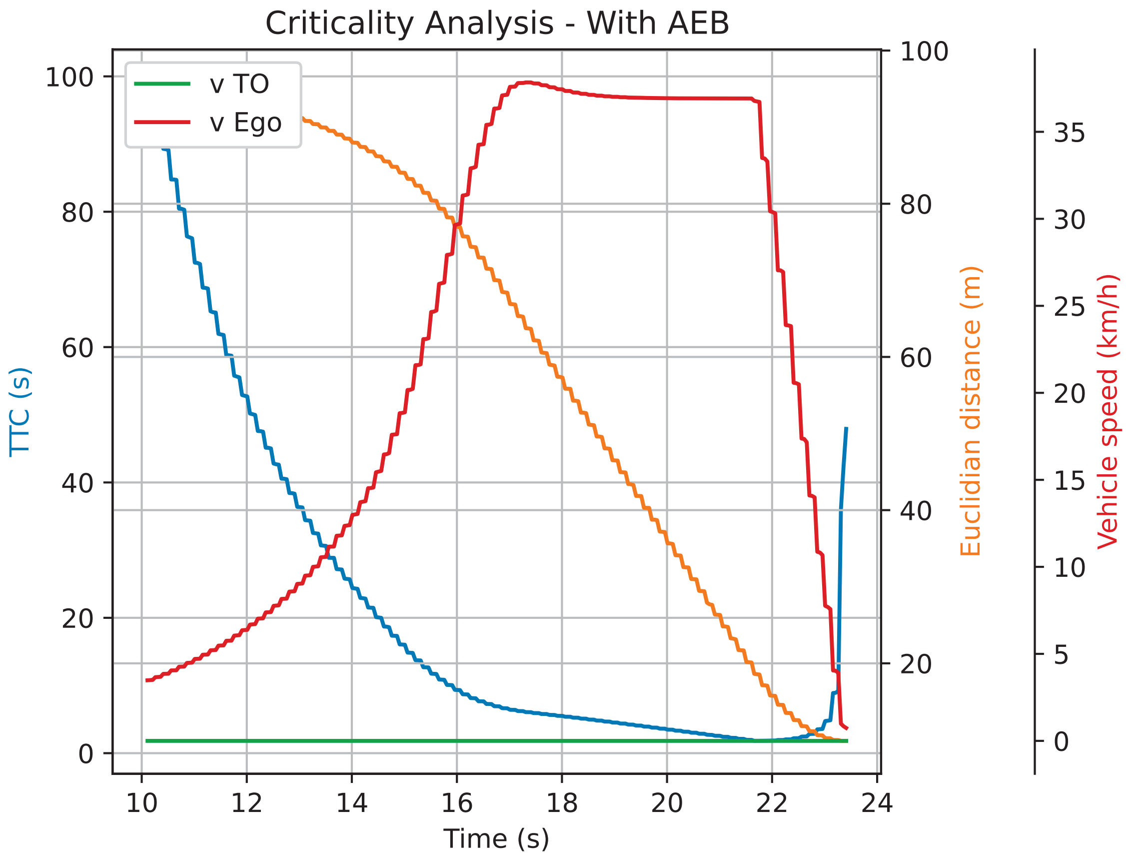


FIGURE 30 – CCRs scenario: CarMaker with AEB

Next, Figure 31 shows the scenario without AEB. In this case, there is no change in the Ego's velocity as it gets closer to the TO. For both cases (with and without AEB), the Euclidean distance between both vehicles and the TTC gets smaller as the ego vehicle approaches the second vehicle. But for the case without an AEB system, after the minimum point of Euclidean distance and TTC, both values start to increase again as the vehicle passes through the traffic object, colliding with it.

Table 8 presents the Evaluation Tool results for the criticality analysis of the CCRs with and without the AEB system.

TABLE 8 – CCRs scenario use case: CarMaker with and without AEB

	With AEB	**Without AEB**
Minimum TTC	1.817	0.002
Time for minimum TTC	21.76	23.61
Collision	No	Yes

As expected, the minimum time to collision will be close to zero if no emergency braking system is activated and a collision will happen. With an AEB system, there is no collision between the vehicles, as shown by the evaluation tool results.

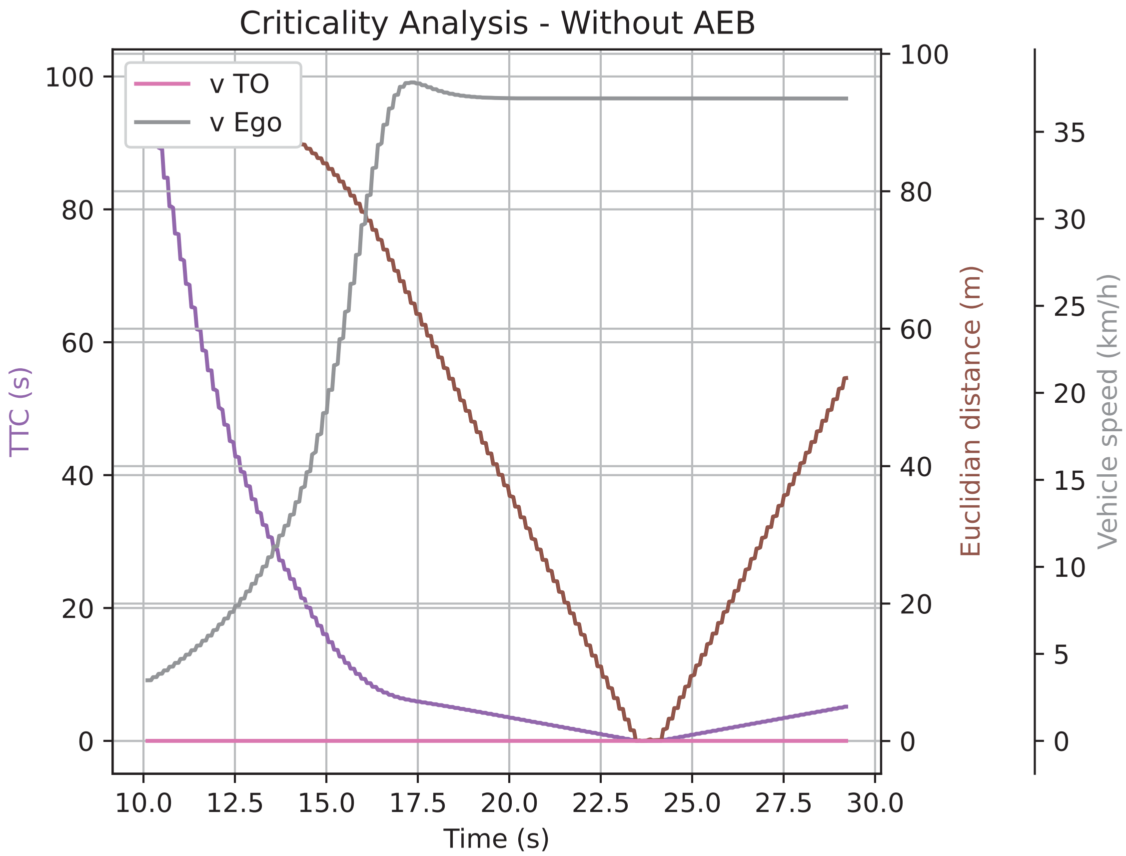


FIGURE 31 – CCRs scenario: CarMaker without AEB

Figure 32 shows the overall score over time for the CCRs scenario with and without the AEB system. Notice that the scenario behaves the same until approximately 23 seconds, in which the AEB system is triggered in the second simulation and not triggered in the first one. After this point, the simulations start to diverge from each other.

4.2.2 CCRs modeled with actions and conditions

Next, the scenario was adapted to use the TTC condition instead of an AEB system. Figure 33 shows the scenario configuration. After 1.1 seconds, the ego vehicle starts accelerating and goes up to 70 km/h. If the TTC between the ego vehicle and the scenario object is less than 1 second, the ego changes lanes to avoid a collision.

The same scenario was run on esmini, Carla, and CarMaker. On esmini, which considers simple vehicle dynamics, the ego vehicle is able to avoid a collision, but on CarMaker and Carla, there is a collision happening, as shown by Figure 34.

Table 9 shows the comparison between esmini, CarMaker, and Carla given by the evaluation tool. As it is possible to observe, the scenario did not trigger the expected actions on Carla and CarMaker. This happened because the vehicle was performing

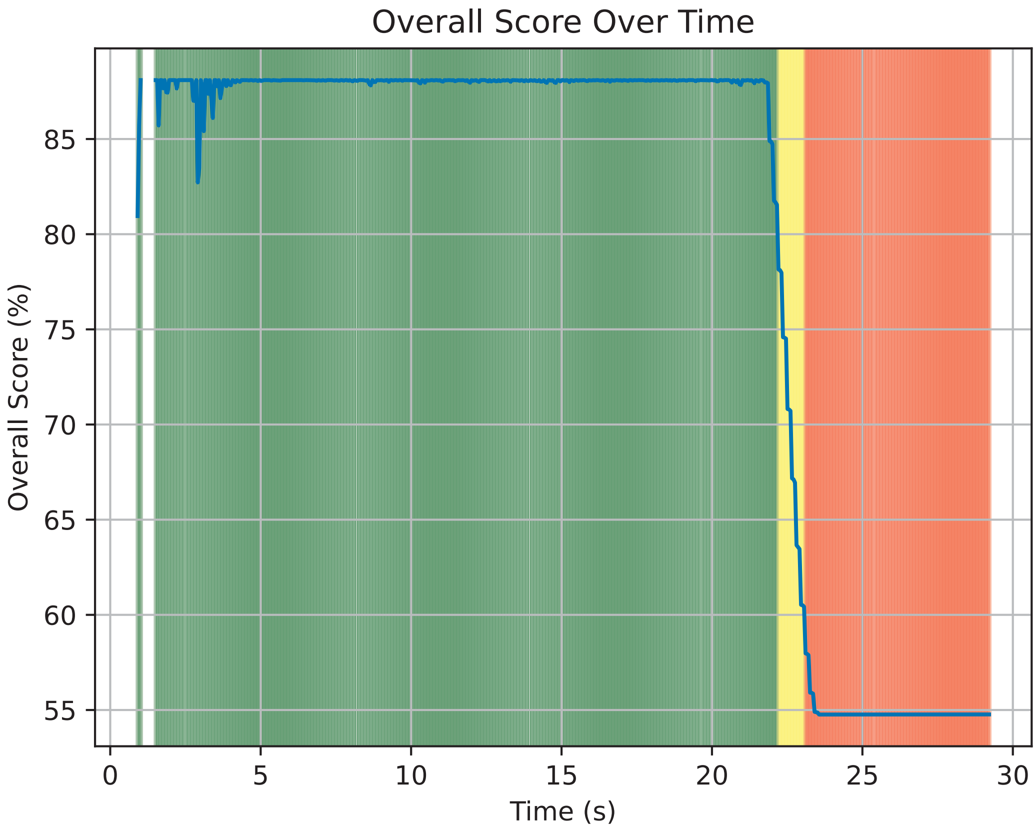


FIGURE 32 – CCRs scenario: Overall score over time

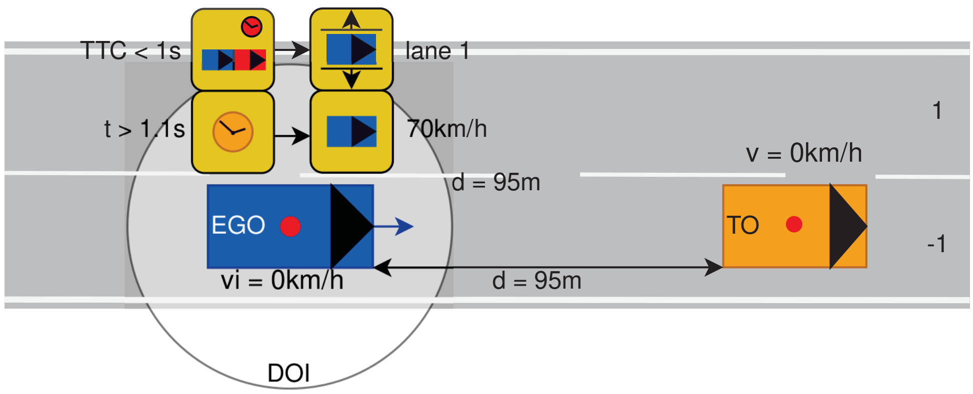
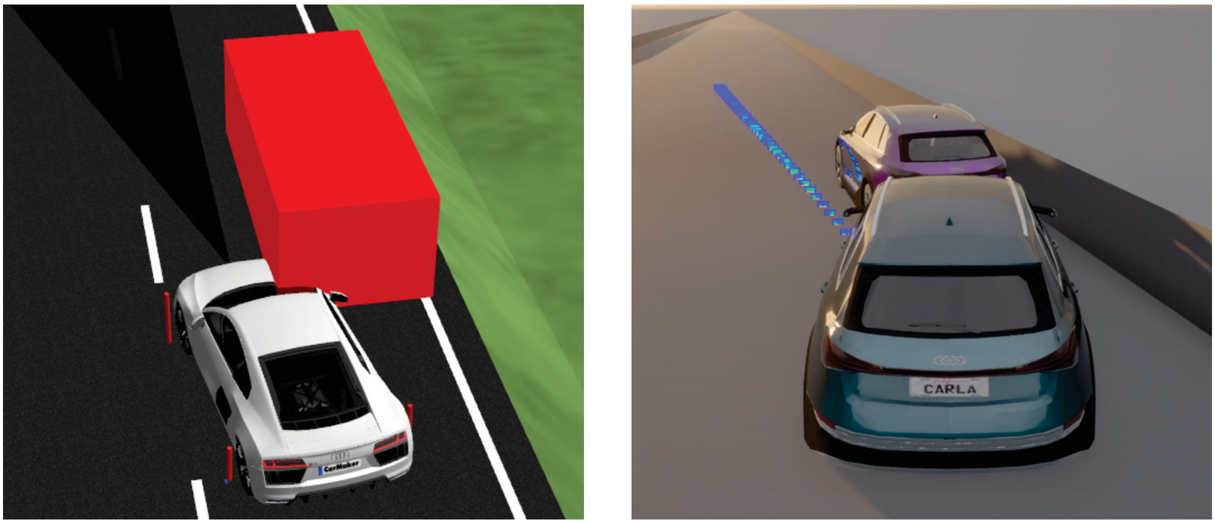


FIGURE 33 – CCRs scenario based on TTC condition

the lane change when it was too close to the TO, leading to a collision between both vehicles.

The scenario was then adapted, and the TTC was changed to 2 seconds instead of 1 second. Table 10 shows the updated results.

Notice an increased matching between the maneuver analysis and TTC for both Carla and CarMaker.



(a) Vehicle collision on CarMaker (b) Vehicle collision on Carla

FIGURE 34 – CCRs scenario: Collision happening on CarMaker and Carla

TABLE 9 – CCRs scenario use case: esmini, Carla and CarMaker

	CCRs Scenario			
	esmini	**CarMaker**	**esmini**	**Carla**
Minimum TTC (s)	0.191	0.043	0.191	0.077
Collision	No	Yes	No	Yes
Maneuver matching	34.48%		55.17%	
Trajectory matching	100%		100%	
TTC matching	22.51%		40.31%	
Overall score	40.40%		55.22%	

4.3 EGO-ONLY SCENARIO ANALYSIS

To evaluate the vehicle dynamics when there is only the ego vehicle in the simulation, three different use cases were defined. First, the ego vehicle was set to drive in a straight line, varying the speed between 0 and 80 km/h. Next, the ego was set to change lanes between -1 and -3 while driving. Lastly, instead of a straight line, it was used a more complex scenario shown in Figure 20. The trajectories and dynamics were then compared between the tools.

4.3.1 Speed Change

Starting with the first use case, the ego was set to vary the speed between 0 km/h and 80 km/h according to figure 35. All the sequences were based on time conditions. The ego vehicle first accelerates to 20 km/h, then decelerates to 10 km/h after 10 seconds. At the 20-second mark, it accelerates to 40 km/h, and by 30 seconds,

TABLE 10 – CCRs scenario use case: esmini, Carla and CarMaker adapted

	CCRs Scenario			
	esmini	**CarMaker**	**esmini**	**Carla**
Minimum TTC (s)	0.224	0.191	0.224	0.149
Collision	No	No	No	No
Maneuver matching	37.93%		48.28%	
Trajectory matching	100%		100%	
TTC matching	85.27%		66.52%	
Overall score	78.75%		69.57%	

it reaches 80 km/h. Lastly, after 40 seconds, the ego decelerates to a standstill at 0 km/h. The scenario was tested on esmini, Carla, and CarMaker, and the speed profile is presented in Figure 36.

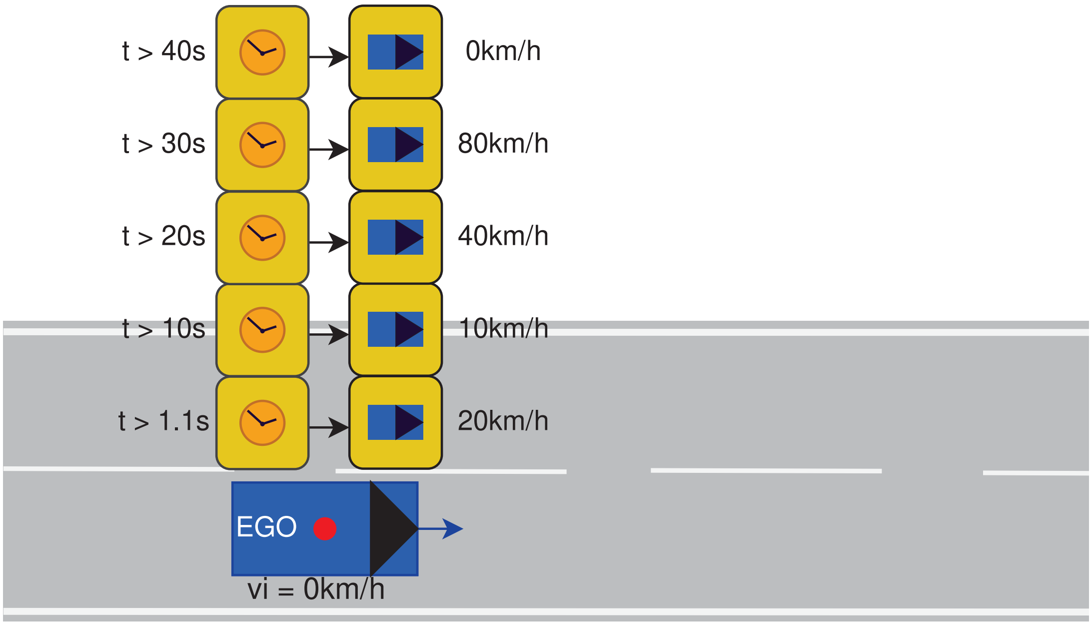


FIGURE 35 – Ego speed variation scenario

By analyzing the data, the main factor that is noticeable is the time shift between the simulations. Table 11 shows the comparison results for the three tools. When comparing esmini with both CarMaker and Carla, there is a 50% and 75% match between maneuvers, respectively. This is because there is a considerable time shift between maneuvers and also because the maneuvers don't follow exactly the same pattern. Specially CarMaker takes longer to accelerate and reach the initial desired speed, causing the mismatch detected by the evaluation tool.

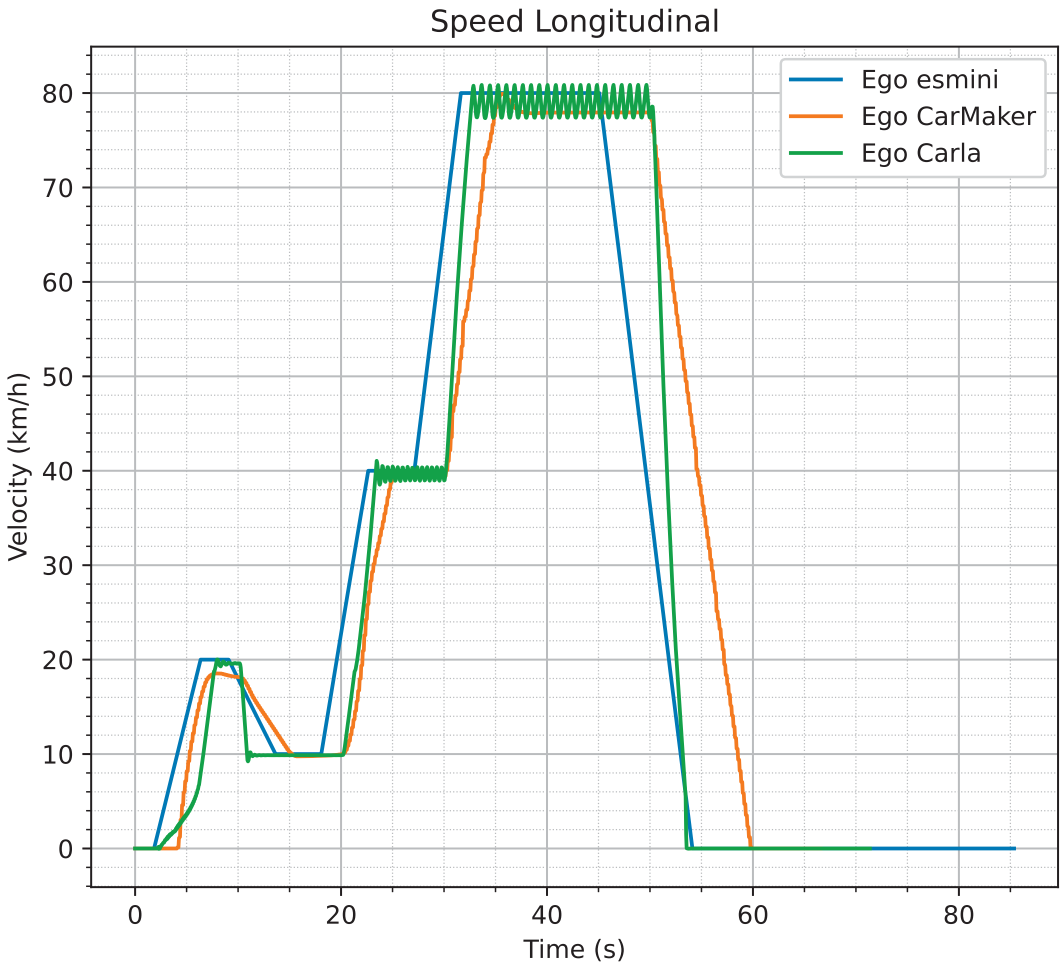


FIGURE 36 – Comparison between esmini, Carla and Carmaker speed profile

TABLE 11 – Comparison between esmini, Carla and CarMaker on speed change scenario

	esmini vs CarMaker	**esmini vs Carla**
Time shift (s)	23.27s	13s
Maneuver Matching	60%	85.71%
Trajectory Matching	99.93%	99.84%
Overall Score	94.19%	89.90%

4.3.2 Lane Change

Next, it was evaluated the difference between simulations in a lane change scenario defined as described in section 3.5.3 and shown in Figure 37. The ego vehicle starts on lane -3 and, after 10 seconds, changes to lane -2. Then, after 20 seconds, it changes to lane -1. Next, at 30 seconds, it performs a double lane change, going to lane -1. Finally, after 40 seconds, it changes again to lane -2, and the simulation ends.

The goal is to analyze the time in which the lane changes are performed and if the action is correctly performed in all tools.

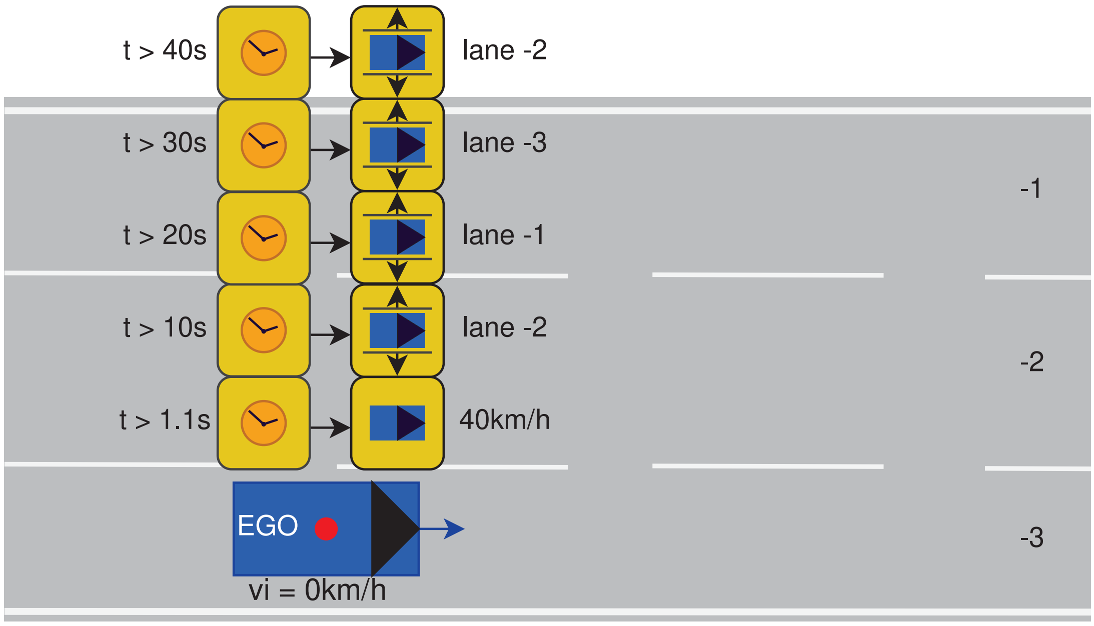


FIGURE 37 – Lane change

Figure 38 presents the lane change action executed in each of the tools. Observe that the CarMaker interface interprets the lanes with an offset, and the lane changes are done right to the left instead of left to right. Also, at the beginning of the simulation, the vehicle starts in lane 0 instead of -3. For both Carla and CarMaker, there is also a time shift between the maneuvers when compared to esmini.

Table 12 summarizes the simulation results given by the evaluation tool. When compared to esmini, both Carla and CarMaker show divergence in the maneuver analysis, which is expected, given the mirrored lane changes in CarMaker. As for Carla, when better analyzing the speed profile, it was noticed an oscillatory behavior on the speed profile, and a low-pass filter was implemented in order to filter the signal so the maneuver could be better analyzed.

TABLE 12 – Comparison between esmini, Carla and CarMaker on lane change scenario

	Ego-Only Scenario	
	esmini CarMaker	**esmini Carla**
Time shift between maneuvers	21.14	2.62
Maneuver matching	30.77%	46.15%
Trajectory matching	100%	100%
Overall score	72.31%	78.46%

The filter result is shown in Figure 39. It consists of a Butterworth filter of order 6, with a cut-off frequency of 0.5 Hz and a sample rate frequency of 6 Hz. The filter

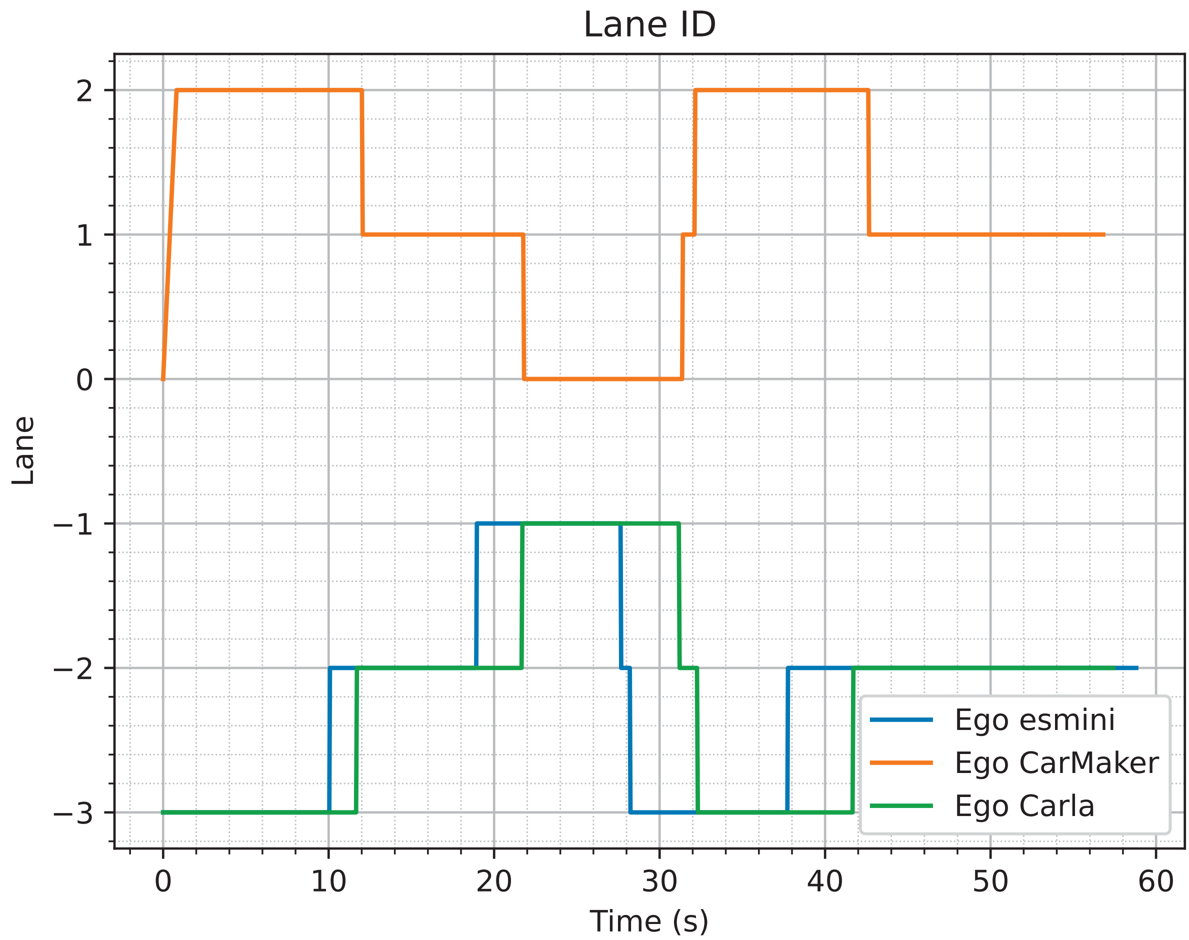


FIGURE 38 – Comparison between esmini, Carla and Carmaker lane profile

was an important step. Without it, the tool would detect each increase and decrease in velocity as different maneuvers.

4.3.3 Simple Trajectory

The final scenario evaluated for the ego-only analysis was a simple trajectory shown in Figure 20. The purpose is to verify if the ego would perform the scenario in all tools considering the vehicle dynamics. For example, if the ego is set to a high speed and there is a turn in the scenario, the vehicle may not be able to perform the maneuver without crashing.

For this scenario, the ego was set to a speed of 20 km/h, and the OpenDRIVE file was edited in order to only allow specific turns. This was necessary so the path chosen by the ego vehicle would be the same in all the tools. Figure 40 shows the simulation result. The green '*X*' symbol indicates where the ego starts (same place for all simulations), and the '*X*' in red indicates where the vehicle stopped.

Because the evaluation tool allows plotting the OpenDRIVE file below the path

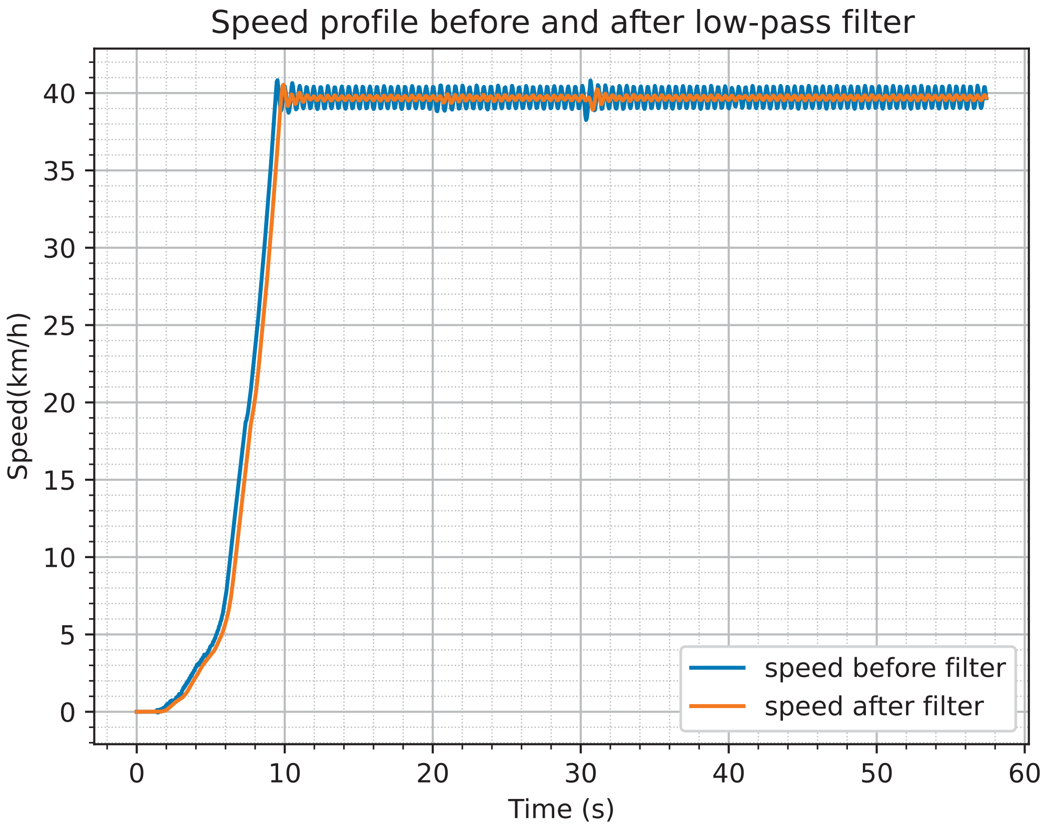


FIGURE 39 – Low-pass filter used for maneuver analysis - Example on Carla

executed by the vehicle, it becomes easier for the user to understand the simulation result and detect if there is any problem within the simulation.

For example, while both esmini and CarMaker present a similar trajectory for the entire simulation, Carla, on the other hand, had problems performing a turn at the beginning of the simulation, and the vehicle crashed with the road.

Table 13 shows the comparison results for all simulations in the three different tools.

TABLE 13 – Results Ego Only complex scenario

	Ego-Only Complex Scenario			
	esmini	**CarMaker**	**esmini**	**Carla**
Maneuver matching	50%		30.77%	
Trajectory matching	76.87%		90.77%	
Overall score	66.12%		66.77%	

Because the trajectory metric chosen for this work was the cosine similarity, it focuses more on the vector direction instead of magnitude. Thus, Carla presents a

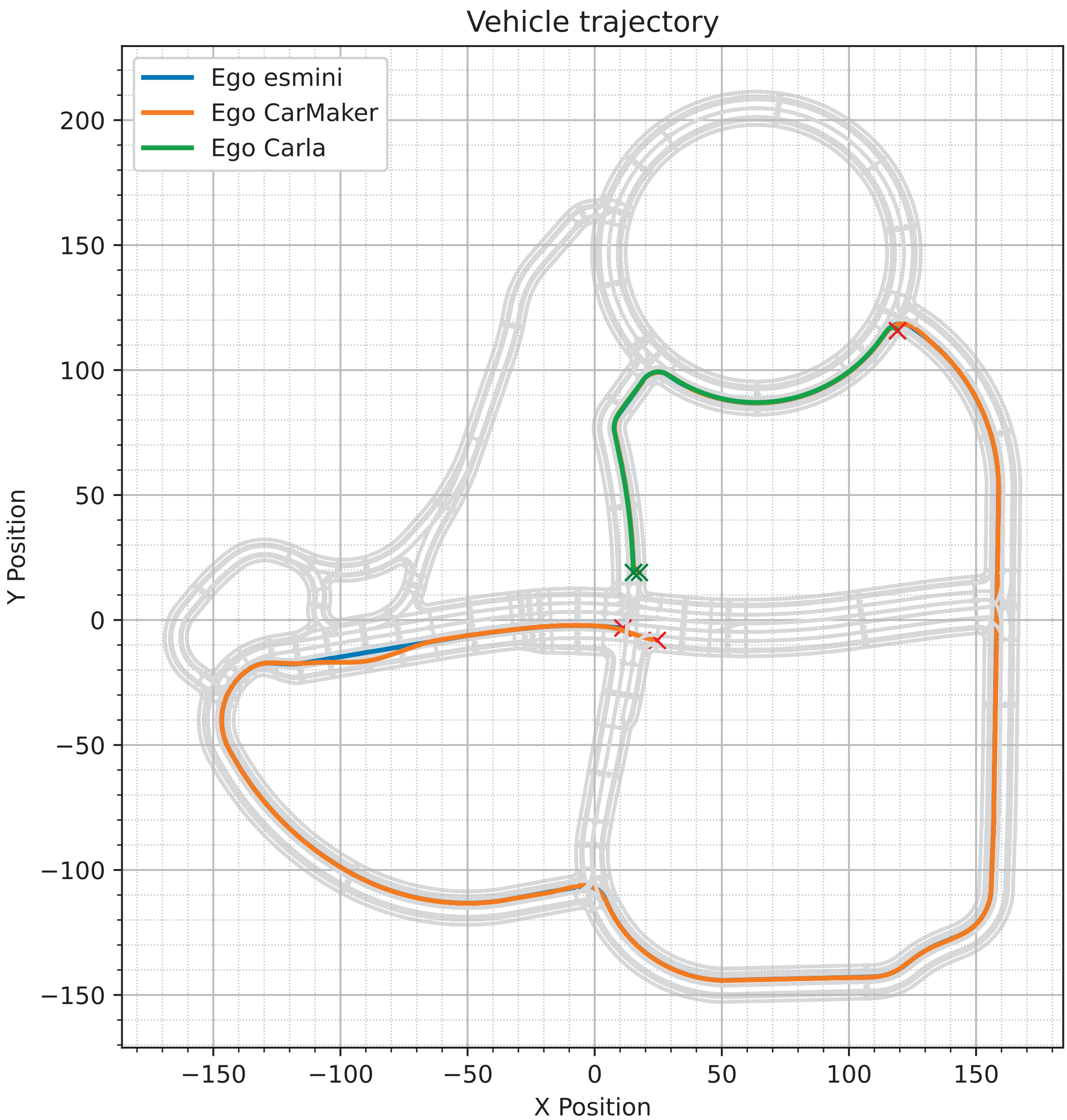


FIGURE 40 – RR scenario

better match than CarMaker even though the vehicle was unable to finish the trajectory.

4.4 CCCSCP SCENARIO ANALYSIS

The final use case to be analyzed is the Car-to-car Crossing Straight Crossing Path. The scenario was defined according to Figure 41. The ego and TO are set to first accelerate to 20 km/h. If the TTC is less than 2 seconds, the ego decelerates to 0 Km/h to avoid a collision.

Table 14 shows the comparison between esmini, Carla, and CarMaker. Notice

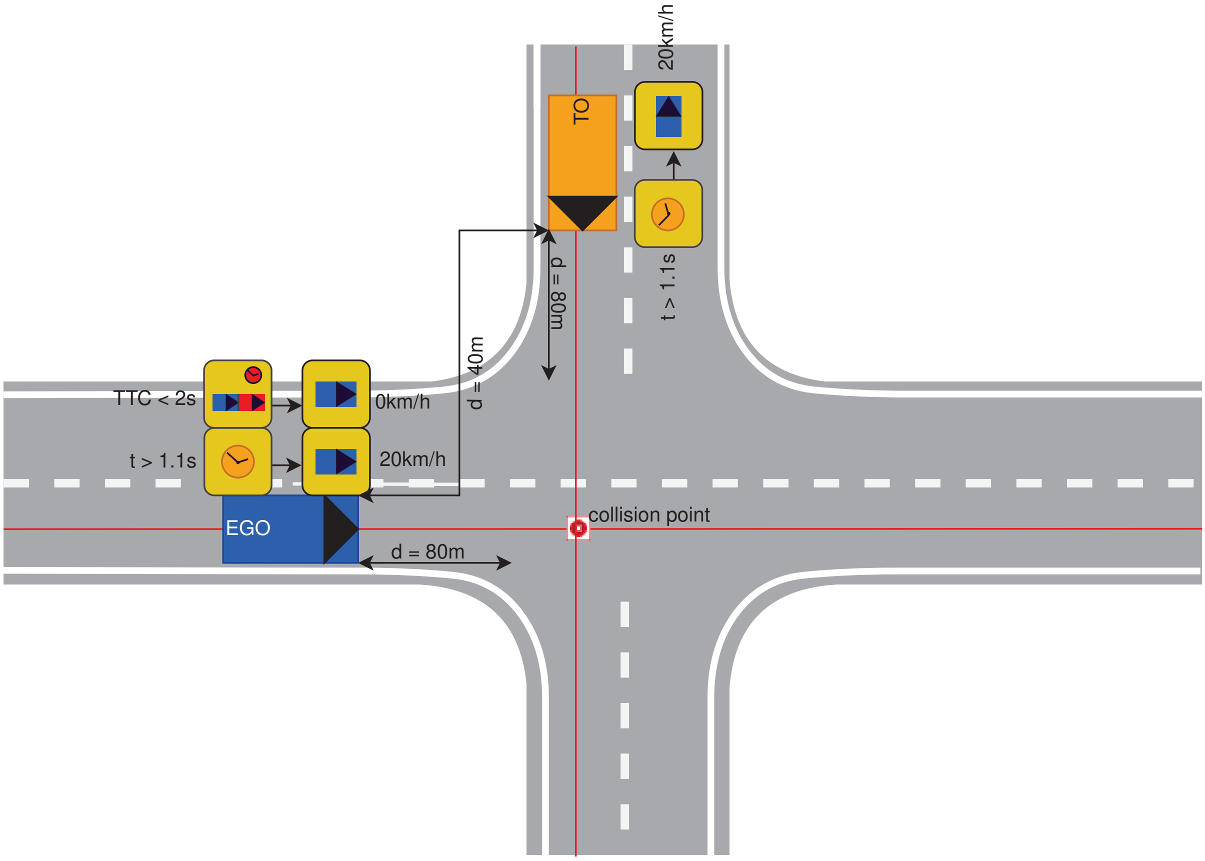


FIGURE 41 – CCCscp scenario

the mismatch between maneuvers. When running the simulation on Carla and CarMaker, the ego vehicle takes around 3 seconds to start accelerating, while the TO accelerates from the start. This leads to the TO being out of the ego's DOI when the ego breaks. If we were to test an emergency brake system, for example, the scenario would not trigger the expected action.

TABLE 14 – Comparison between CCCscp scenario: esmini, CarMaker and Carla

	CCCscp scenario			
	esmini	**CarMaker**	**esmini**	**Carla**
Minimum TTC (s)	1.105	1.271	1.105	0.209
Collision	No	No	No	No
Maneuver matching	59.26%		41.67%	
Trajectory matching	100%		99.94%	
TTC matching	86.94%		82.83%	
Overall score	84.01%		78.02%	

The scenario was then adapted in order to the TO wait 3 seconds before starting to accelerate. Table 15 shows the result for both cases.

TABLE 15 – Comparison between CCCscp scenario: esmini, CarMaker and Carla adapted

	CCCscp scenario adapted			
	esmini	CarMaker	esmini	Carla
Minimum TTC (s)	1.086	1.271	1.086	1.192
Collision	No	No	No	No
Maneuver matching	100%		100%	
Trajectory matching	100%		99.95%	
TTC matching	94.66%		91.11%	
Overall score	96.79%		94.65%	

In both situations, there was a slight improvement in the scenario behavior, and the overall score shows that both Carla and CarMaker performed the desired actions during the simulation.

For all the scenarios presented in this work, results show that the set of similarity metrics is capable of assessing the scenario execution in different configurations, correctly detecting the difference between different scenario executions.

Based on the result of each simulation, the user can have a better understanding of the factors that influence the simulation and how the scenario modeling impacts the desired behavior in different software used to execute the simulation.

5 CONCLUSION

This work presented an evaluation tool able to optimize the analysis and comparison of simulation results from different tools based on a set of similarity metrics.

The metrics defined in this work are deviation analysis, maneuver analysis, and criticality analysis. The first metric, deviation analysis, is based on cosine similarity, and results show that the metric can be used for trajectory analysis if the direction of the vehicle is more relevant than the magnitude of its path. For the maneuver analysis, a new set of object-related maneuvers was defined, and all the actions performed by the ego vehicle while having a second object in its domain of interest were classified. Finally, the criticality analysis counts with a time-to-collision analysis and a collision detection algorithm that simplifies the vehicle dimensions to a three-circle approach.

The tool allows the user to load different simulation files, specify the scenario type, set the weights for each similarity metric, and compare two simulations at a time. An overall score represents the percentage match between simulations and consists of the weighted sum of individual metrics. After calculating and presenting the results, the tool offers the possibility to generate a PDF report containing a detailed overview of the analysis. For each metric, a matching percentage between the expected behavior and the achieved result is calculated.

Four scenario use cases were defined in OpenSCENARIO Editor to validate the tool: cut-in scenario, Car-to-Car Rear Stationary, Car-to-car Crossing Straight Crossing Path, and ego-only scenario. For each use case, the same OpenSCENARIO file was executed on esmini, Carla, and CarMaker, with esmini being the desired behavior and Carla and CarMaker the achieved behavior for comparison. The FEP interface was used as middleware between the tools, recording the data as CSV files to be used as input for the evaluation tool. It is important to mention that the dynamics are an important factor that needs to be taken into account when creating the scenario, and scenarios defined for esmini need to be carefully specified to also work on tools such as Carla and CarMaker.

After calculating the overall score, the tools perform a qualitative analysis of the result. If the final score is more than 80%, it means that the simulation worked with no significant deviation between the simulations and the achieved result is similar to the expected one. If the score is less than 80% but more than 70%, it means that the simulations have a small deviation, but the scenario is still able to perform the expected actions. If the result is between 50% and 70%, the user may need to visually verify the scenario and make sure the simulation is running as intended. If the value is less than

50%, the scenario file needs to be redefined because the expected actions are not being triggered.

Some of the challenges faced during this work were the constant updates for Carla and CarMaker interfaces, which affected the dynamics and scenario reading. Also, not all functions described on OpenSCENARIO Editor are compatible with Carla and CarMaker, making it more challenging to specify a scenario compatible with all tools.

As a future work, it is possible to increase the number of scenarios that can be evaluated by the tool and increase the number of simulations that can be compared each time. Another point is to improve the weights used to calculate the weighted sum of the overall score, as those were defined in an experimental way.

BIBLIOGRAPHY

ARONOV, Boris et al. Fréchet distance for curves, revisited. In: SPRINGER. ALGORITHMS–ESA 2006: 14th Annual European Symposium, Zurich, Switzerland, September 11-13, 2006. Proceedings 14. [S.l.: s.n.], 2006. P. 52–63. Citado na p. 24.

ASAM. **ASAM OpenSCENARIO: User Guide**. [S.l.: s.n.], 2022. https://www.asam.net/index.php?eID=dumpFile&t=f&f=4908&token=ae9d9b44ab9257e817072a653b5d5e98ee0babf8. Accessed: 2023-07-01. Citado na p. 20.

AUTOMOTIVE, IPG. **CarMaker: Virtual development and testing of passenger cars and light-duty vehicles**. [S.l.: s.n.], 2023. https://ipg-automotive.com/fileadmin/data/know-how/media_library/pdf/IPG_Automotive_CarMaker_Product_Sheet_EN.pdf. Accessed: 2023-05-01. Citado nas pp. 28, 29.

BACIC, M. On hardware-in-the-loop simulation. In: PROCEEDINGS of the 44th IEEE Conference on Decision and Control. [S.l.: s.n.], 2005. P. 3194–3198. DOI: 10.1109/CDC.2005.1582653. Citado na p. 21.

BAGSCHIK, Gerrit; MENZEL, Till; MAURER, Markus. Ontology based scene creation for the development of automated vehicles. In: IEEE. 2018 IEEE Intelligent Vehicles Symposium (IV). [S.l.: s.n.], 2018. P. 1813–1820. Citado nas pp. 19, 21.

CHAI, Tianfeng; DRAXLER, Roland R. Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature. **Geoscientific model development**, Copernicus Publications Göttingen, Germany, v. 7, n. 3, p. 1247–1250, 2014. Citado na p. 24.

CHEN, He et al. Generating Autonomous Driving Test Scenarios based on OpenSCENARIO. In: 2022 9th International Conference on Dependable Systems and Their Applications (DSA). [S.l.: s.n.], 2022. P. 650–658. DOI: 10.1109/DSA56465.2022.00093. Citado na p. 14.

CUSTOMTKINTER. [S.l.: s.n.]. https://customtkinter.tomschimansky.com/. Accessed: 2023-07-25. Citado na p. 34.

DOSOVITSKIY, Alexey et al. CARLA: An open urban driving simulator. In: PMLR. CONFERENCE on robot learning. [S.l.: s.n.], 2017. P. 1–16. Citado na p. 29.

E.V, ASAM. **ASAM OpenDRIVE**. [S.l.: s.n.], 2023. https://www.asam.net/standards/detail/opendrive/. Accessed: 2023-05-01. Citado na p. 21.

E.V., Dekra. High Risk of Accidents Caused by People's Behaviour. Springer, 2021. Citado na p. 14.

EITER, Thomas; MANNILA, Heikki. Computing discrete Fréchet distance. Technical Report CD-TR 94/64, Christian Doppler Laboratory for Expert . . ., 1994. Citado na p. 25.

ESMINI. **Environment Simulator Minimalistic (esmini)**. [S.l.]: GitHub, 2023. https://github.com/esmini/esmini. Accessed: 2023-04-28. Citado na p. 27.

EURO, NCAP. Test protocol-aeb car-to-car systems. **Brussels, Belgium: Eur. New Car Assess. Programme (Euro NCAP)**, 2022. Citado na p. 38.

EV, ASAM. **ASAM OpenSCENARIO**. [S.l.: s.n.], 2023. https://www.asam.net/standards/detail/openscenario/. Accessed: 2023-04-30. Citado nas pp. 19, 20.

FATHY, Hosam K et al. Review of hardware-in-the-loop simulation and its prospects in the automotive area. In: SPIE. MODELING and simulation for military applications. [S.l.: s.n.], 2006. v. 6228, p. 117–136. Citado na p. 21.

GÜHMANN, Clemens; RIESE, Jens; VON RÜDEN, Klaus. Simulation and Testing for Vehicle Technology. In: SPRINGER. 7TH Conference, Berlin. [S.l.: s.n.], 2016. Citado na p. 31.

HAN, Jiawei; PEI, Jian; TONG, Hanghang. **Data mining: concepts and techniques**. [S.l.]: Morgan kaufmann, 2022. Citado na p. 25.

HARTJEN, Lukas et al. Classification of driving maneuvers in urban traffic for parametrization of test scenarios. In: 9. Tagung Automatisiertes Fahren. [S.l.: s.n.], 2019. Citado na p. 25.

HAYWARD, John C. Near miss determination through use of a scale of danger. Pennsylvania State University University Park, 1972. Citado na p. 22.

HUBER, Bernd et al. Evaluation of virtual traffic situations for testing automated driving functions based on multidimensional criticality analysis. In: IEEE. 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC). [S.l.: s.n.], 2020. P. 1–7. Citado nas pp. 22, 23.

INC., The MathWorks. **RoadRunner: Design 3D scenes for automated driving simulation**. [S.l.]: The MathWorks Inc., 2023. Disponível em: <https://www.mathworks.com/products/roadrunner.html>. Citado na p. 28.

STANDARDIZATION, International Organization for. **Road vehicles - Functional safety**. v. 2. [S.l.], dez. 2018. Citado na p. 26.

ISO/PAS, 21448:2019. Road vehicles - Safety of the intended functionality. In: ISO. Citado na p. 18.

KLÜCK, Florian Steffen et al. Quality assurance methodologies for automated driving, 2018. Citado na p. 14.

MCKINNEY, Wes et al. Data structures for statistical computing in python. In: AUSTIN, TX, 1. PROCEEDINGS of the 9th Python in Science Conference. [S.l.: s.n.], 2010. v. 445, p. 51–56. Citado na p. 34.

MENZEL, Till; BAGSCHIK, Gerrit; MAURER, Markus. Scenarios for development, test and validation of automated vehicles. In: IEEE. 2018 IEEE Intelligent Vehicles Symposium (IV). [S.l.: s.n.], 2018. P. 1821–1827. Citado nas pp. 17, 18.

OPENSCENARIO PLAYER, esmini. **esmini user guide**. [S.l.]: GitHub, 2023. https://esmini.github.io/. Accessed: 2023-04-28. Citado na p. 27.

PEGASUS. **V-Model and Process Analysis**. [S.l.: s.n.]. https://www.pegasusprojekt.de/files/tmpl/Pegasus-Abschlussveranstaltung/03_V-Model_and_Process_Analysis.pdf. Accessed: 2023-08-12. Citado nas pp. 26, 27.

PETERSSON, L.; FLETCHER, L.; ZELINSKY, A. A framework for driver-in-the-loop driver assistance systems. In: PROCEEDINGS. 2005 IEEE Intelligent Transportation Systems, 2005. [S.l.: s.n.], 2005. P. 771–776. DOI: 10.1109/ITSC.2005.1520146. Citado na p. 21.

RIEDMAIER, Stefan et al. Survey on Scenario-Based Safety Assessment of Automated Vehicles. **IEEE Access**, v. 8, p. 87456–87477, 2020. DOI: 10.1109/ACCESS.2020.2993730. Citado nas pp. 14, 17.

SCHÜTT, Barbara et al. A taxonomy for quality in simulation-based development and testing of automated driving systems. **IEEE Access**, IEEE, v. 10, p. 18631–18644, 2022. Citado na p. 17.

SOUSA, Roniel S De; BOUKERCHE, Azzedine; LOUREIRO, Antonio AF. Vehicle trajectory similarity: models, methods, and applications. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 53, n. 5, p. 1–32, 2020. Citado nas pp. 23, 24.

STADLER, Christoph; GERMAN, Reinhard; DJANATLIEV, Anatoli. Towards a Multivariate Scoring Metric for the Evaluation of Simulation Scenarios in Automated Driving. In: IEEE. 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC). [S.l.: s.n.], 2022. P. 2508–2513. Citado nas pp. 22, 26.

STANDARDIZATION, International Organization for. Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary. In: ISO. Citado na p. 20.

STANDARDIZATION OF AUTOMATION, Association for; SYSTEMS, Measuring. ASAM SIM: Guide. Standardization for Highly Automated Driving. ASAM e.V. Citado na p. 19.

STEVIĆ, Stevan et al. Development of ADAS perception applications in ROS and"Software-In-the-Loop"validation with CARLA simulator. **Telfor Journal**, v. 12, n. 1, p. 40–45, 2020. Citado na p. 14.

ULBRICH, Simon et al. Defining and substantiating the terms scene, situation, and scenario for automated driving. In: IEEE. 2015 IEEE 18th international conference on intelligent transportation systems. [S.l.: s.n.], 2015. P. 982–988. Citado na p. 18.

WACHENFELD, Walther; WINNER, Hermann. The release of autonomous vehicles. **Autonomous Driving: Technical, Legal and Social Aspects**, Springer, p. 425–449, 2016. Citado na p. 17.

WAGNER, Sebastian Karl. **Efficient scenario-based assessment of automated driving systems through virtual testing**. 2021. Tese (Doutorado) – Technische Universität München. Citado nas pp. 21–23, 26, 27.

WATZENIG, Daniel; HORN, Martin. **Automated driving: safer and more efficient future driving**. [S.l.]: Springer, 2016. Citado na p. 14.

XIE, Guoqi et al. **Functional Safety for Embedded Systems**. [S.l.]: CRC Press, 2023. Citado na p. 26.

YURTSEVER, Ekim et al. A survey of autonomous driving: Common practices and emerging technologies. **IEEE access**, IEEE, v. 8, p. 58443–58469, 2020. Citado na p. 17.