POLITECNICO DI TORINO

MASTER's Degree in MECHATRONIC ENGINEERING

MASTER's Degree Thesis

VIRTUAL VALIDATION TOOLCHAIN FOR AUTOMOTIVE IN-CABIN SENSING

Supervisors
Prof. Massimo VIOLANTE
Dottor Alessandro TESSUTI

Candidate
Stefano PERCIVATI

April 2024

Summary

The pursuit of road safety has been a longstanding goal for the European Union, exemplified by its ambitious targets set for 2030 and beyond. Over the years, substantial progress has been made in enhancing vehicle safety, resulting in a notable reduction in road fatalities and major injuries. However, to further advance road safety initiatives, the integration of Advanced Driver Assistance Systems (ADAS) into vehicles has become imperative. A significant milestone in this endeavor occurred with the introduction of mandatory driver drowsiness and distraction warning systems for newly registered cars categorized as M (passenger vehicles) and N (goods vehicles) since July 2022. Notably, the requirements for Advanced Driver Distraction Warning (ADDW) are slated to expand for newly manufactured vehicles starting in 2024 and for all currently registered vehicles by 2026. Collaborative efforts between esteemed institutions like Politecnico di Torino and Concept Quality Reply have led to the development of a validation framework specifically tailored to meet the stringent regulations governing driver and occupant monitoring systems within the European Union. Leveraging the sophisticated capabilities of Unreal Engine, this framework facilitates the testing and refinement of algorithms within a meticulously crafted virtual environment. By doing so, it eliminates the need for extensive real-world testing while providing developers with the flexibility to craft diverse scenarios effortlessly and accurately evaluate algorithm performance against Ground Truth data. Although the initial iteration of the camera validation framework has been successfully completed, numerous opportunities for improvement and expansion abound. Potential enhancements include the augmentation of avatar sets to introduce greater randomization, the incorporation of a broader spectrum of external car scene settings to replicate various driving contexts, and the implementation of different generations of lights to diversify scenario combinations for comprehensive testing. In conclusion, the project epitomizes a significant stride forward in the realm of road safety, driven by innovative technological solutions. By conscientiously addressing identified areas for refinement, the validation framework stands poised to evolve continuously, facilitating the exploration of an extensive array of scenarios and the testing of a broader spectrum of situations. Through ongoing development and refinement,

the validation framework holds the promise of substantially contributing to the realization of the European Union's vision of safer roads for all.

Acknowledgements

I would like to express my deep gratitude to Prof. Massimo Violante, my thesis advisor.

Special thanks go to Reply for providing me with the opportunity to conduct my thesis research. It has been a privilege to collaborate with such competent and passionate professionals, and I am grateful for the welcome and support I received during my time studying at the company.

I warmly thank my manager, Alessandro Tessuti, for his guidance, encouragement, and trust in me. His valuable advice and expertise significantly contributed to the success of this project.

Lastly, I would like to express my gratitude to my esteemed colleagues at Reply, who shared their knowledge, experiences, and resources with me. Their support and collaboration were essential to the completion of this thesis.

To all those who have contributed in any way to the completion of this work, my sincerest thanks. I am grateful for your support and generosity.

Stefano Percivati

Table of Contents

List of Tables

List of Figures

Acronyms

OEM
: Original Equipment Manufacturers

ADDW
: Advanced Driver Distraction Warning

IoT
: Internet of Things

NHTSA
: National Highway Traffic Safety Administration

Euro NCAP
: European New Car Assessment Programme

VATS
: Visual Attention Time Sharing

ESS
: Epworth Sleepiness Scale

SSS
: Stanford Sleepiness Scale

VAS
: Visual Analog Scale

KSS
: Karolinska Sleepiness Scale

SEM

Slow Eye Movements

PERCLOS

Percentage of Eye Closure

AECS

Average Eye Closure Speed

LiDAR

Light Detection And Raging

API

Application Programming Interface

ADAS

Advanced Driver Assistance Systems

DRY

Don't Repeat Yourself

MRT

Multiple resource theory

SEEV

Salience, Effort, Expectation, and Value

SWM

Steering Wheel Movement

SDLP

Standard Deviation of Lateral Position

NIRS

near-infrared spectroscopy

EEG

electroencephalography

EOG
: electrooculography

EMG
: electromyography

ECG
: electrocardiography

SUT
: System Under Test

MIL
: model-in-the-loop

SIL
: software-in-the-loop

PIL
: processor-in-the-loop

HIL
: hardware-in-the-loop

VeHIL
: Vehicle hardware-in-the-loop

AD
: Automated driving

ABS
: Anti Brake-locking System

SAE
: Society of Automotive Engineers

UNECE
: United Nations Economic Commission for Europe

AEBS

Advanced Emergency Braking System

ALS

Advanced Life Support

KMVSS

Korea Motor Vehicle Safety Standard

RGB

Red, Green, Blue

Chapter 1

Introduction

According to the General Regulation on Vehicle Safety, which will come into effect from mid-2024, a fundamental requirement mandates that all newly manufactured vehicle models be equipped with a sophisticated driver distraction detection system. This system represents a crucial step forward for automotive safety, as it focuses on monitoring the driver's eye movements to promptly identify and alert instances of distraction, ultimately aiming to drastically reduce the incidence of road accidents on European Union roads.

At the forefront are automobile manufacturers, commonly known as Original Equipment Manufacturers (OEMs), who have taken on the challenging task of developing and validating these camera-based systems. Leveraging advanced artificial intelligence-driven computer vision algorithms, such systems aim to accurately discern and classify the driver's state during the course of the journey.

To achieve this ambitious goal, a vast treasure trove of data is required, including the driving habits of hundreds of individuals in various conditions. However, OEMs face a series of common challenges closely tied to data acquisition in real-world scenarios, complicated by inherent limitations posed by certain edge cases.

Manual parameterization emerges as a critical element in meeting strict time constraints. However, in contexts such as gaze monitoring, the use of manual input might prove cumbersome. Furthermore, organizing data collection campaigns requires substantial investments in both financial and time resources.

Enter the proposed solution: a sophisticated suite of virtual and photorealistic tools designed to simulate the complex mechanisms of a vehicle's internal sensors. This innovative approach aims to optimize development and validation processes while simultaneously committing to maintaining high safety standards through coverage-guided verification and scenario-based virtual testing.

Central to this paradigm shift is the integration of synthetic data, supplemented by carefully crafted metadata, to facilitate the automatic generation of customized annotations. Once the necessary modules and Application Programming Interfaces

(APIs) are made accessible, the permutation process gains new agility, requiring only a few lines of code to generate thousands of scenario permutations. This grants professionals unlimited control over every aspect of the virtual environment, enabling them to tailor data generation activities in line with specific objectives and requirements.

1.1 The genesis of the thesis

This work stems from the collaboration between Concept Reply and the Polytechnic University of Turin. Concept Reply is a company within the Reply Group specializing in the exploration, creation, and validation of new Internet of Things (IoT) solutions. Concept Reply now has a dedicated Business Unit for this type of service and is recognized by the market as a center of excellence and expertise in testing and quality assurance. The main objective of this initiative is the development of a photorealistic virtual environment capable of simulating Advanced Driver Distraction Warning Systems (ADDW) and being used to validate and verify their proper functioning, with particular attention to compliance with applicable European regulations.

1.2 Reply SpA

Reply is an Italian company that provides IT consulting and support services. Founded in 1996, the company is globally recognized and has expertise in various sectors, including cybersecurity, IT consulting, digitalization, and the Internet of Things (IoT). It is known for collaborating with companies in various industries, including banking, healthcare, public administration, automotive, and telecommunications, to provide cutting-edge technical solutions and services to tackle challenging issues. The company works on projects related to digital transformation and the creation of unique software, data analysis, and other related activities.

Figure 1.1: The Reply headquarters in Turin.

1.3 Document structure

This work is organized as follows:

- **Introduction**: In this chapter, the motivations behind the thesis project are presented, outlining its origins and providing a description of the software used.

- **State of Art**: This chapter delves into the theoretical foundations and studies that have inspired the project's development, focusing on the analysis of driver distraction and drowsiness, as well as the evolution of current regulations in the automotive sector.

- **The Metahuman Plug-in**: This section provides a detailed description of the selected plug-in for the project, which allowed the use of hyper-realistic avatars capable of executing extremely credible animations.

- **Animations of Metahumans**: In this segment, the technical process through which avatars are integrated with facial animations is examined, utilizing both pre-recorded sequences and real-time usage of the Live Link App for iOS.

- **Biometric Parameters**: analysis of the considered biometric data and their implications;

- **Virtual Validation Test Bench**: In this section, the implementation and analysis of Test Benches for ADAS system validation are discussed, exploring two main methodologies for image acquisition and processing to test and optimize the effectiveness of driver assistance systems.

- **Randomization of Features in the Scene**: This chapter details the creation of a complete fuzzing scenario, outlining how such a rigorous and systematic testing approach significantly contributes to the identification and mitigation of potential vulnerabilities in ADAS systems.

- **Conclusion**: In the concluding chapter, a reflection on the journey taken is presented, examining in detail the achieved milestones and outlining perspectives for future implementations aimed at further perfecting the project. This section synthesizes significant contributions to research and development in the field of ADAS systems, highlighting the potential for expansion and optimization of the work done.

1.4 Software and Tools

1.4.1 Unreal Engine

The Unreal Engine game engine, developed by Epic Games, emerges as a robust and versatile platform not only for creating engaging video games but also for developing complex 3D simulations used in various industrial sectors. Among these sectors, the automotive industry stands out, where Unreal Engine has proven to be a formidable tool for validating Advanced Driver Distraction Warning (ADDW) Systems. The adoption of Unreal Engine in this field is motivated by a series of reasons, each of which contributes to its effectiveness and applicability:

- **Photo-realistic Graphics**: Unreal Engine stands out for its ability to generate extremely realistic graphics, essential for recreating virtual environments that faithfully reproduce the intricate details of vehicle interiors. This level of visual realism is crucial for accurately evaluating and validating ADDW systems in simulated scenarios.

- **Advanced Simulations**: Unreal Engine allows programmers to create highly realistic simulations, complete with sophisticated models for driver behavior and distraction detection systems. This capability is essential for exposing ADDW systems to a wide range of complex and realistic scenarios, evaluating their effectiveness and responsiveness in contexts that closely resemble real situations.

- **Real-time Interaction with Metahuman**: thanks to its seamless integration with Metahuman, Unreal Engine introduces a new level of realism in interactive experiences. By enabling real-time synchronization of facial expressions and gestures with incredibly realistic avatars, researchers can assess the performance of ADDW systems in conditions closely mirroring actual driving scenarios.

- **Customization and Variability**: Unreal Engine offers programmers immense flexibility in creating different environments and characters, including Metahuman avatars, to simulate a wide range of driving conditions and scenarios. This versatility is crucial for conducting thorough evaluations of ADDW systems in a variety of real-world situations.

- **Data Logging**: Unreal Engine simplifies the generation of detailed data logs documenting events and interactions within simulations, providing valuable information to researchers for system validation and improvement.

- **Flexibility and Scalability**: thanks to its inherently scalable architecture, Unreal Engine allows programmers to create extremely intricate simulations while maintaining the flexibility to incorporate new features as project requirements evolve.

(a) Epic games logo.

(b) Unreal Engine logo.

Figure 1.2: Application logos.

In particular, with the introduction of Lumen, Unreal Engine 5's dynamic global illumination system, light dynamically adapts to real-time changes in the virtual environment, accurately reflecting the complexity and variability of the real world.

Simultaneously, Nanite radically transforms the way 3D assets are managed and represented in Unreal Engine 5. With the ability to handle millions of polygons with minimal performance impact, Nanite allows programmers to import cinematic-quality models directly into the engine while maintaining an incredibly high level of detail

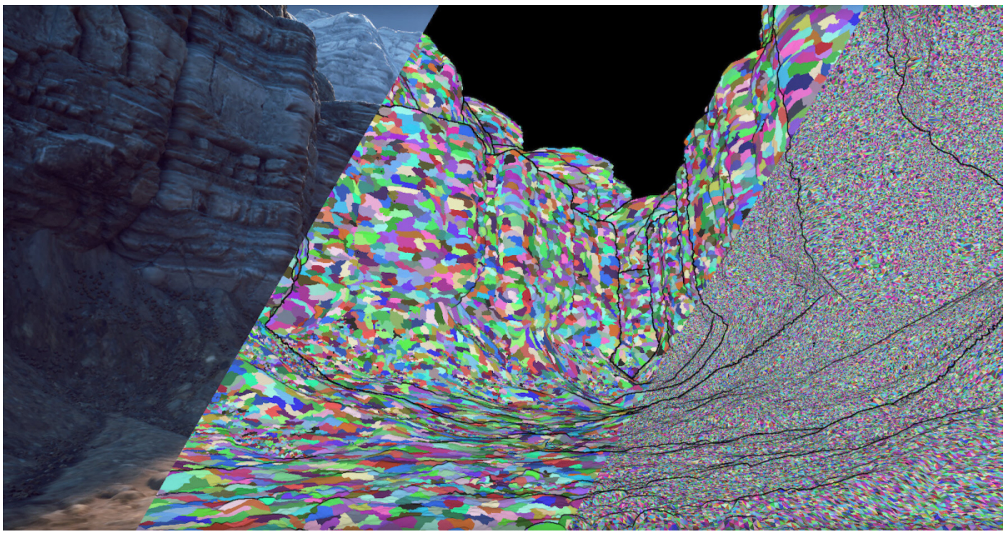


Figure 1.3: Unreal Engine nanite

Scene Generation

The creation of a scene in Unreal Engine 5 follows a sequence of well-defined and detailed phases. The first step involves selecting the most suitable template for the type of experience to be developed, which is a crucial step as the chosen template provides a solid foundation for scene creation.

Subsequently, the organization and arrangement of the elements that will constitute the scene are carried out, including terrains, structures, objects, and lighting systems. The Content Browser provides a wide range of resources and materials that can be used to populate the environment, while the modeling tools integrated into the engine facilitate the customization and detailing of each component.

Materials

In the context of advanced systems like Unreal Engine 5, defining materials and textures is a crucial aspect to achieve increasingly realistic renderings of virtual scenes.

Materials play a fundamental role in the visual appearance, tactile perception, and interaction of light with surfaces and objects within the virtual environment.

Unreal Engine 5 offers integrated access to a platform called Quixel Bridge, which contains an extensive library of predefined materials designed to provide a variety of surfaces and objects. These materials can be customized through the Material Editor, a tool that allows modification of every single material property to create complex and dynamic surfaces.

In addition to applying simple materials, Unreal Engine 5 supports the creation of procedural materials, capable of automatically generating details and variations based on algorithms or specific inputs.

An interesting aspect concerns post-processing materials, which offer an advanced level of visual customization. These materials allow the application of global effects that influence the entire scene and are applied during the rendering phase. Effects like bloom, motion blur, color correction, and depth of field are just some examples of what can be achieved.

Customization of post-processing materials is done through the Material Editor, which provides a wide range of nodes to create complex and dynamic effect chains. It is also possible to modulate these effects over time or in response to specific events within the virtual environment, to get closer and closer to the faithful representation of the real world.

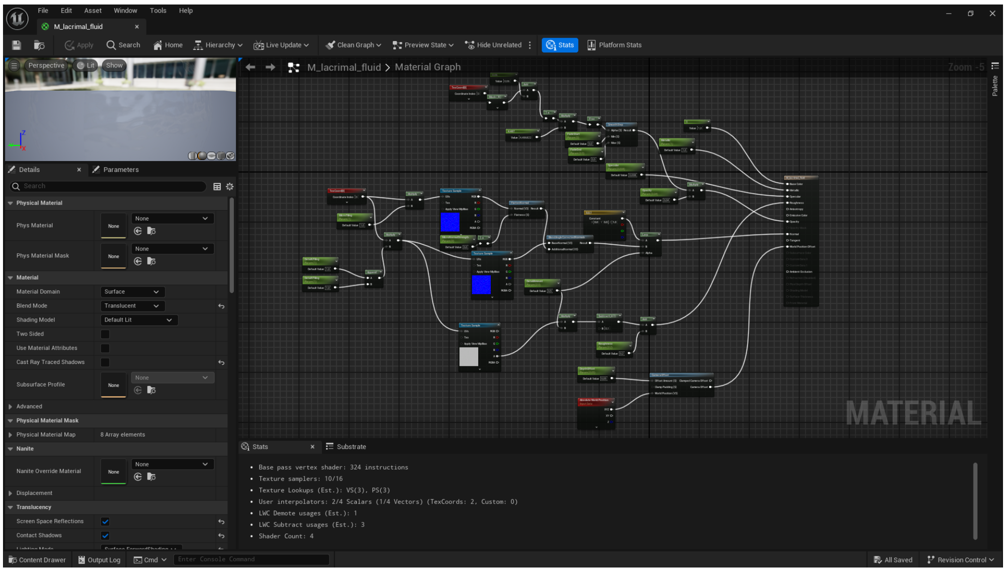


Figure 1.4: Material editor.

Programming through Blueprint and C++

Implementing interactivity marks the transformation of the environment from static to dynamic. Unreal Engine 5 offers two main methodologies to achieve this goal: visual programming via Blueprints and traditional programming in C++.

Blueprints represent a visual programming tool based on predefined functions in Unreal Engine. These functions are used and connected to create functional graphs that define events within the simulation. Nodes are visually distinct and organized by category, simplifying the identification of functionalities and understanding the logical flow of the simulation.

Despite its accessibility, Blueprint is extremely powerful and can be used not only for rapid prototyping but also for creating complex logics and sophisticated interactive systems. Its integration with the C++ language allows for extensive customization and modifications, even regarding predefined functions.

However, using Blueprints requires a careful approach to avoid complications, such as performance overhead or difficulty in maintaining the code. It's important to keep Blueprints clean and organized, avoiding overly complex graphs or excessively interconnected nodes that could become difficult to read or modify. Additionally, modularity is essential, with the creation of Blueprints that perform specific functions and can be reused in different parts of the game, promoting a "Don't Repeat Yourself" (DRY) approach to development.

Blueprints also extend to other areas, such as animation and audio, allowing for the creation of dynamic and responsive systems that enhance user immersion and interaction.

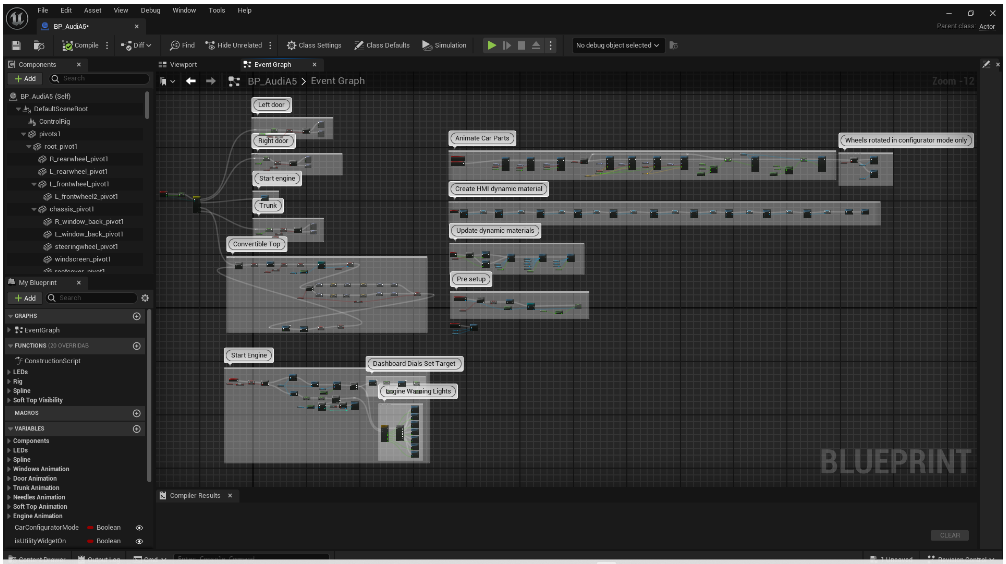


Figure 1.5: Blueprint editor.

Programming in C++ leverages a direct interface with the engine's APIs, allowing programmers to write highly performant code that is tightly integrated with the engine's core. This level of integration offers unprecedented control over every aspect of the engine.

The choice between using Blueprint or C++ in Unreal Engine 5 depends on various factors, including the specific project goals, available time, and the level of experience and preference of the development team. Smaller projects or those with rapid development times may benefit from the visual and intuitive approach of Blueprints, while projects requiring a high degree of optimization, customization, or system complexity will likely find C++ as their preferred tool.

In conclusion, it's evident that this engine positions itself as one of the most advanced and powerful platforms available in the landscape of interactive content development. Its ability to integrate tools like Blueprint and C++ offers unparalleled versatility, allowing teams to customize their workflow based on the specific needs of each project and to make the most of each team member's skills.

The combination of these features, with particular emphasis on realism, was the determining factor in choosing Unreal Engine for the development of the Advanced Driver Assistance Systems (ADAS) validation project, highlighting confidence in the engine's ability to support and enhance our design ambitions.

1.4.2 Metahuman animator plugin

The inception of the "Metahuman" plugin by Epic Games, the same company behind the revolutionary Unreal Engine, marks a significant stride in the realm of character creation and integration within the gaming and entertainment industries. This plugin serves as a testament to Epic Games' commitment to pushing the boundaries of what is achievable in digital character design and animation.

At its core, the MetaHuman plugin stands as a testament to the fusion of cutting-edge technology and artistic vision. Leveraging the power of MetaHuman technology, developers gain access to a vast repository of meticulously crafted characters that exude realism and authenticity. From the subtle nuances of facial expressions to the intricate details of anatomical features, each MetaHuman character is a testament to the precision and artistry embedded within the plugin.

Figure 1.6: Metahuman generator.

One of the most striking features of the MetaHuman plugin lies in its seamless integration with Unreal Engine. Developers are presented with a streamlined workflow that simplifies the process of importing and manipulating MetaHuman characters within the Unreal Engine environment. This integration not only enhances efficiency but also empowers developers to unleash their creativity and bring their visions to life with unparalleled fidelity.

Moreover, the MetaHuman plugin offers a wealth of customization options, allowing developers to tailor characters to suit the unique requirements of their projects. From wardrobe choices to hairstyle variations, the plugin provides a comprehensive toolkit for realizing diverse characters across a myriad of creative endeavors.

Beyond its utility in traditional gaming applications, the MetaHuman plugin holds immense potential across various industries, including film, animation, and virtual production. With its emphasis on realism and immersion, the plugin empowers storytellers to craft compelling narratives and evoke genuine emotional responses from audiences worldwide.

In essence, Epic Games' MetaHuman plugin represents a paradigm shift in character creation and integration, setting new standards for realism, versatility, and artistic expression within the digital landscape. As technology continues to evolve and boundaries continue to blur, the MetaHuman plugin stands as a testament to the endless possibilities that lie at the intersection of art and innovation.

Chapter 2

State of the art

In this phase of the study, an in-depth investigation was conducted into numerous studies, particularly within the automotive industry, where the primary focus was on researching relevant behaviors such as distraction and fatigue during driving. Additionally, existing solutions and technologies on the market were carefully examined, with particular emphasis on identifying options that demonstrate a high level of success in countering such behaviors and increasing road safety.

2.1 Driver distraction

Driver distraction is described as the shifting of attention from activities necessary for safe driving to a concurrent activity. The distracting activity could take numerous forms and originate from both inside and outside the vehicle.

Using naturalistic driving data, researchers have found that drivers spend approximately 23.5% of their driving time engaged in secondary activities, significantly increasing the risk of collision.

The European Commission estimates that driver distraction causes between 10% to 30% of accidents in Europe, while the National Highway Traffic Safety Administration states that driver distraction contributes to 16% of all fatal crashes, 21% of all injury crashes, and 22% of all crashes in the United States[1]. Given the difficulties in identifying relevant variables following a collision, these figures are likely underestimated.

There are four types of distractions, and many drivers face more than one simultaneously:

- **Visual distraction**: Occurs when the driver takes their eyes off the road to engage in an unrelated secondary activity.

- **Auditory distraction**: Noise interferes with the driver's focus on tasks necessary for safe driving.
- **Manual distraction**: The driver removes one or both hands from the steering wheel to perform an action not necessary for safe driving.
- **Cognitive distraction**: The driver's mind is engaged in thoughts unrelated to safe driving.

Distraction and Competing Resources

Figure 2.1 illustrates the connection between the demands of a competing task and the requirements of driving duties. When the combined demand exceeds the capabilities of the driver, it leads to distraction. Drivers can partially manage traffic demand by driving more slowly or maintaining a greater distance from other vehicles, although evidence of such compensatory actions is inconsistent. Unfortunately, driving demands are often unpredictable and go beyond the driver's control; for instance, sudden braking by other vehicles[2].

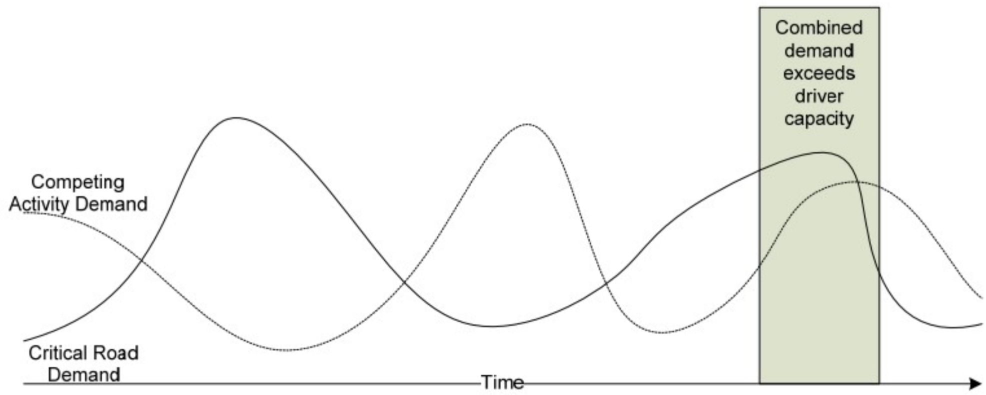


Figure 2.1: The confluence of competing activities and the demands of the roads that lead to distracted driving accidents[2]

Despite drivers being able to adjust their involvement in competing tasks, they often struggle to prioritize driving and adequately adapt to their demands, especially during maneuvers like overtaking[**<empty citation>**].

Additionally, drivers may underestimate the impact of distractions on their performance and mistakenly believe they can drive safely while distracted. Consequently, in challenging driving situations, drivers may neglect to postpone or interrupt competing activities. The operational definition of distraction depicted in Figure 2.1 can be understood in terms of the conflict between driving and information processing resources. Actions such as taking hands off the wheel or shifting position to reach for a phone can impair the driver's ability to respond to driving challenges, while looking away from the road to read a text message hinders hazard detection. Similarly, engaging in cognitive tasks like hands-free phone conversations diminishes the driver's ability to process driving-related information.

Multiple resource theory (MRT) divides the competition between driving and distractions into four dimensions: processing stage, processing code, perceptual modalities, and visual channel. Driving performance suffers when the competing task shares resources with driving, particularly during visually demanding distractions that compete with driving's intensive visual demands. Moreover, the response selection stage acts as a bottleneck for cognitive tasks, potentially delaying responses to driving events while preparing responses to competing activities. The different types of resources suggest the need for varied measurements to detect distraction and its impact on driving safety[2].

Distraction and Attention Dynamics

Salience, Effort, Expectation, and Value (SEEV) represent four factors that influence the direction of attention. Drivers tend to focus their attention on highly visible locations that require low cognitive effort, are likely to provide new information, and are meaningful relative to the individual's goals. Figure 2.2 combines the SEEV elements to create a visual representation of the aspect-sampling process that regulates the dynamics of driver distraction and the alternation of attention between driving and other activities. It is reasonably presumed that safety is more compromised when at least two of these situations coexist.

For example, a situation where a combination of glances away from the road, poor attention to traffic conditions, and a significant unexpected incident occurs. Similarly, if a crucial event does not occur, even a prolonged critical gaze away from the road may have little impact[2]. According to the conceptual framework of compromised driving in Figure 2.2, the four critical variables determining distraction dynamics are[2]:

- **Stimulus saliency**: Characteristics such as size, color, contrast, direction, movement, and brightness significantly influence a stimulus's ability to capture the driver's attention and elicit an appropriate reaction.
- **Visual eccentricity**: This factor is caused by stimuli that affect the retinal periphery rather than the fovea (the central part of the retina responsible for visual processing). Visual performance decreases significantly as one moves toward the retinal periphery. Visual eccentricity corresponds to effort in the SEEV model, as objects with greater eccentricity require more effort to be observed.

- **- Intermittent Vision**: People do not have continuous access to visual information. Blinking and temporary occlusions cause periods of vision loss, during which transient visual responses of low-level feature detection processes are hidden. This jeopardizes bottom-up cognitive capture, event recognition, and reaction. This intermittent factor also reflects frequent sampling of the roadway, which occurs when drivers engage in a concurrent activity that diverts their attention from the road.

- **Cognitive factors**: Attention is governed by two main mechanisms. On the one hand, external events can override attention in an ascending manner, guided by stimuli. Salience and eccentricity characteristics are primarily associated with this process of ascending attention. On the other hand, attention can be directed top-down and oriented by cognitive elements, including knowledge, expectation, and current goals. Additionally, attention is often a combination of ascending and descending processes resulting from competitive interactions.

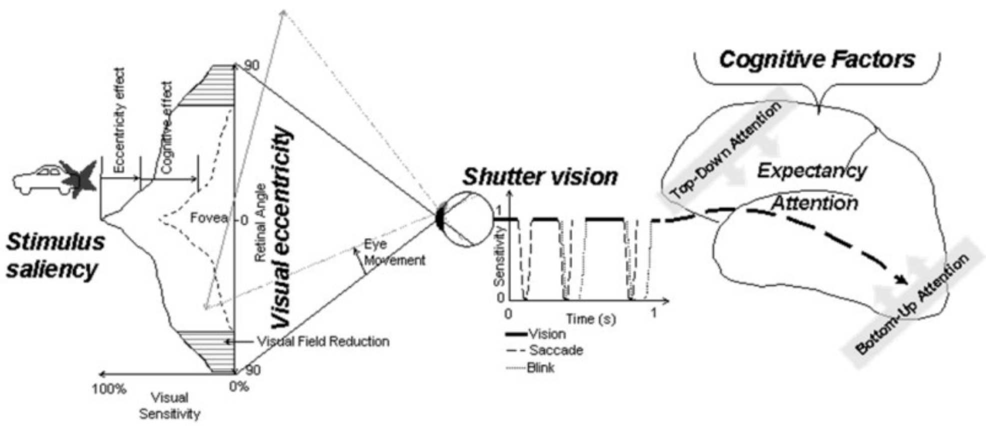


Figure 2.2: Conceptual framework of important factors impacting driving distraction dynamics[2]

2.2 Driver Fatigue

Many road accidents in the modern world are caused by driver lack of concentration and attention, sometimes known as driver fatigue[3]. Long-distance drivers, such as truck and bus drivers, are more prone to experiencing this issue. Driving while sleep-deprived and for long periods of time is extremely dangerous and exhausting. Given the seriousness of driver fatigue, it is estimated that between 70,000 and 80,000 accidents and injuries occur worldwide each year.

Between 1000 and 2000 people die annually due to road accidents caused by driver fatigue. However, many deaths go unreported and do not include confirmation from the driver that fatigue was the main cause of the accident. Due to the high number of innocent deaths caused by this phenomenon, it is essential to identify and alert drivers when they appear to be falling asleep at the wheel in order to prevent collisions. Technological developments in recent years, such as physiological signal detection and monitoring of car and/or driver behavior, have shown promise in detecting dangerously high levels of driver fatigue.

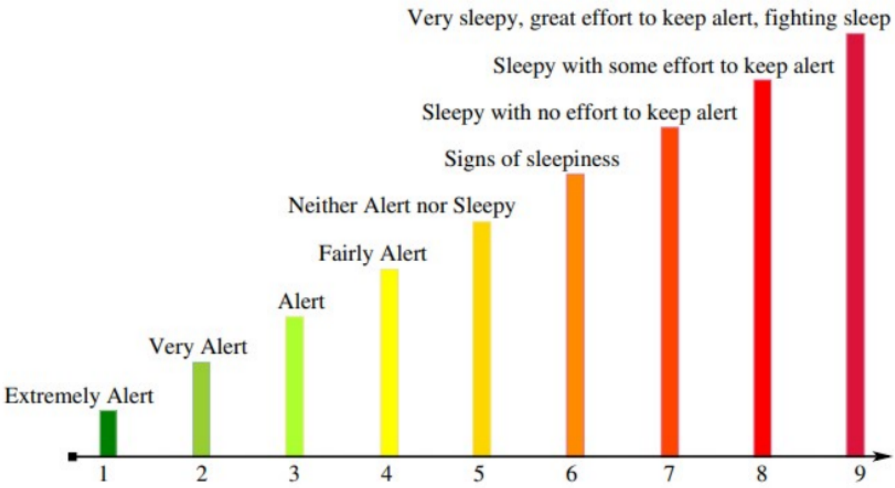


Figure 2.3: Karolinska Sleepiness Scale (KSS)

Using questionnaire responses on a standardized and well-known set of sleep symptoms, it is possible to subjectively assess sleepiness. Some examples of self-reported subjective measures include the Epworth Sleepiness Scale (ESS)[4], the seven-point Stanford Sleepiness Scale (SSS)[5], the Visual Analog Scale (VAS)[6], and a nine-point Karolinska Sleepiness Scale (KSS) that defines linguistic reference for each step of sleepiness (Figure 2.3). However, obtaining a sleepy response in real-time presents a challenge. It is improper and risky to rely solely on the driver's response to gather vigilance or sleepiness indicators because this could compromise driving and underestimate the driver's actual performance.

Therefore, this method is not suitable for application in actual driving situations. Many approaches have been proposed to quantify fatigue, including combining objective metrics related to driver behavior and vehicle performance, or using physiological signal data along with subjective sleep assessments through a series of questionnaires. Researchers have categorized technologies for assessing driver fatigue based on the monitored subject and the monitoring tool used. The commonly used methods for assessing fatigue can be categorized into one or more of the following parameters, or a combination thereof[7]:

- **Driving behavioural measure**: involves monitoring the car and its environment, in addition to examining driving behavior. Various indicators produced by sensors installed in the vehicle are used to assess how focused the driver is.
- **Driver behavioural measure**: this method focuses on driver behavior, analyzing gestures and facial expressions, instead of tracking their driving activities. Some of these movements may be considered indicators of fatigue or drowsiness.
- **Driver physiological signals measure**: this technique relies on examining various physiological signals that are visually associated with fatigue and drowsiness in drivers.

Driving behavioural research

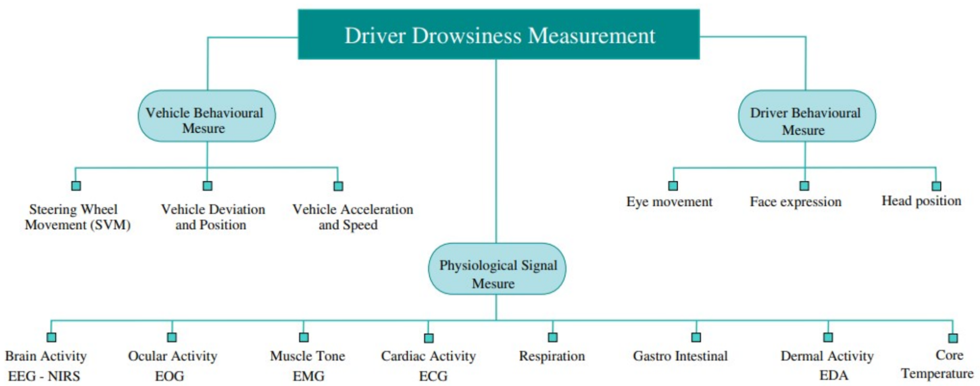


Figure 2.4: An overview of the measurement techniques used to identify driver fatigue

In this research, measurement sensors are used in cars to collect various metric indicators that are used to determine a driver's vigilance or drowsiness based on their driving behavior, as illustrated in Figure 2.4. Three main aspects of the car and its environment are central to this approach[8]:

- **Steering Wheel Movement (SWM)**: Monitoring it can be a useful method for assessing driving behavior. In fact, some unusual steering wheel movements are made by an inattentive driver and may indicate fatigue and drowsiness[8].

This method measures steering wheel behavior using an angular sensor mounted on the steering axis. A variety of metrics, such as steering reversals frequency, steering correction times, and abrupt vehicle movement, can be used to assess inappropriate driving[9].

- **Vehicle Deviation and Position**: The car's position is another sign of driver fatigue and drowsiness. Indeed, this technique relies on metrics such as the car's position relative to the road's center lane, or the SDLP (Standard Deviation of Lateral Position)[10][11], its deviation from the side lane[12], its roll deviation[13], and orientation difference[14].
- **Vehicle Speed and Acceleration**: Research indicates that the driver's level of consciousness and the vehicle's speed are related, and that a drowsy driver tends to accelerate the vehicle more. Measures such as vehicle speed,[15][16] acceleration rate[17], and pedal pressure underpin this approach[13].

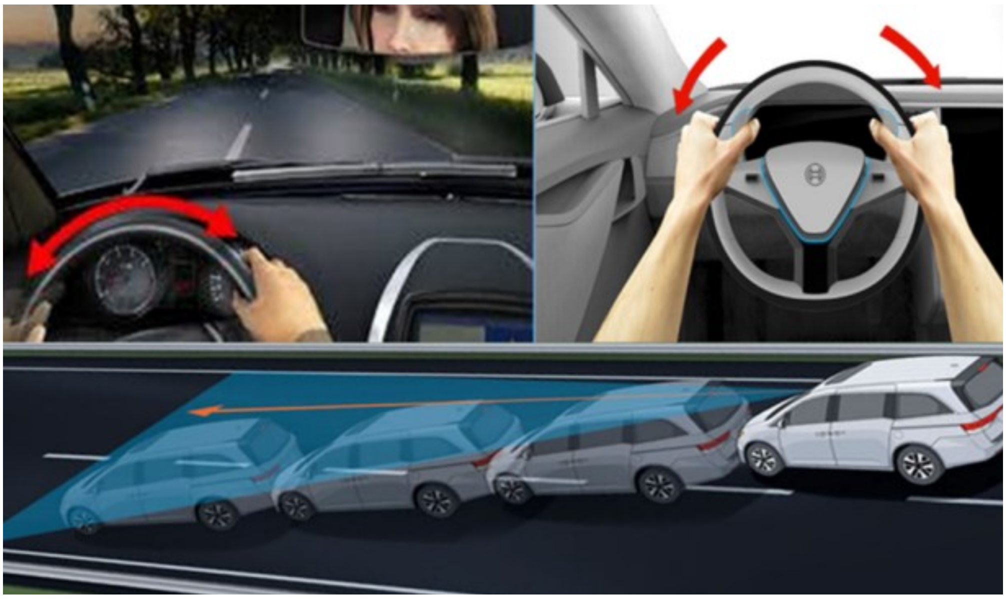


Figure 2.5: An example of how the steering wheel moves and how the lane departs[7]

Driver's behavioural research

As seen in Figure 2.6, research in this field monitors driver actions and collects a variety of indicative metrics that can be used to assess the level of vigilance or drowsiness of the driver. Measuring anomalous driver behaviors allows for the identification of characteristic signs of fatigue and drowsiness. Three main metrics have been at the forefront of research on using driver behavioral monitoring to identify fatigue and drowsiness[7]:

- **Eye movement**: This metric focuses on monitoring the eyes through characteristic eye movements like AECS[18], SEM[19], and eye-closure activities such as PERCLOS[20] [21] statistics and blink rate[22]. Peculiar blinking and eye closure indicate fatigue.

- **Face expression**: Sleepy drivers make facial expressions that can be used to determine how fatigued they are. Facial emotions and activities such as yawning, lowering the jaw, stretching the lips, and raising eyebrows are measured through facial monitoring[23].

- **Head position**: Head position is another indicator of exhaustion and drowsiness. When fatigue reaches a deeper stage, drowsy drivers typically nod their head or lower it[24].

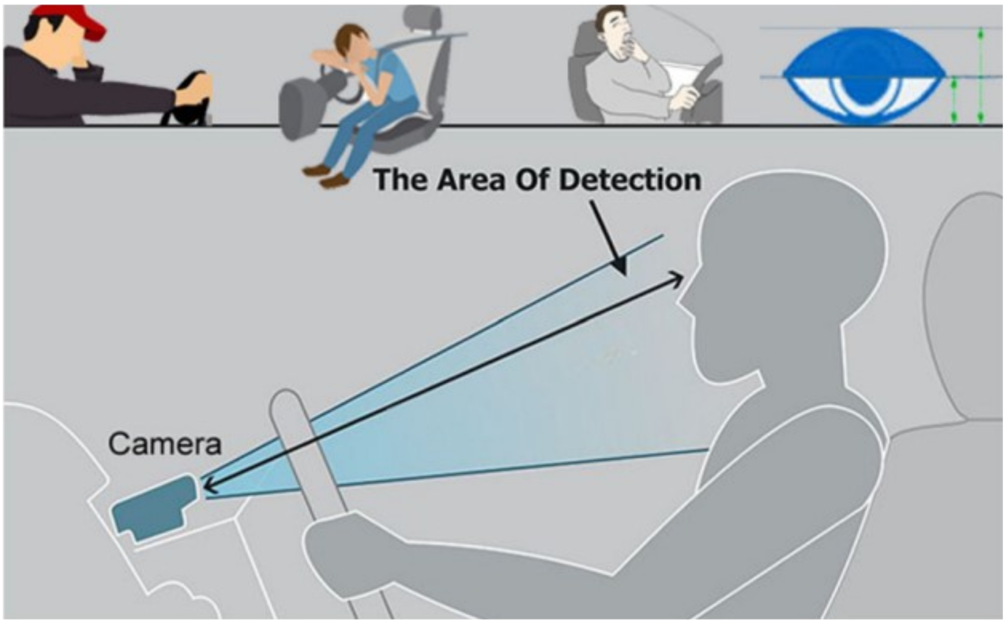


Figure 2.6: An example of a driving behavior measure[7]

Physiological Signals Research

Since physiological signals come from human organs such as the brain, eyes, muscles, and heart (as illustrated in Figure 2.7), they can be used to assess driver attention and enable early identification of fatigue and drowsiness. Physiological signals can be tracked from organs that show a strong correlation with driver fatigue. These include[7]:

- **Brain activity**: which is detectable through near-infrared spectroscopy (NIRS) or electroencephalography (EEG);
- **Ocular activity**: measurement using electrooculography (EOG);
- **Muscle Tone**: using an electromyography (EMG) signal, this can be recorded;
- **Cardiac activity**: blood pressure measurements and electrocardiography (ECG) are used to monitor the patient;
- **Respiration**: evaluating gas levels in the blood, oral and nasal airflow, respiratory effort, and snoring noise during sleep;
- **Gastro intestinal parameters**: obtained by measuring esophageal pH;
- **Electro dermal activity**: skin resistance, conductivity, and skin galvanic response are used to assess skin health;
- **Core temperature**: provides details on the individual's actual circadian phase.

Figure 2.7: An example of the physiological signals measure for drivers[7]

2.3 Verification and validation

Verification and validation are terms used to differentiate control activities based on their scope across different phases of the development process. Verification refers to control actions limited to a single phase. Quality control of tasks completed during a development phase is the objective of checking[25].

Validation is a control process that compares the output of a development phase with the specifications for the final product. Ensuring that the final product possesses the functionality and performance specified at the beginning of the development process is a typical example of validation. Moreover, validation is usually performed on the completed product. However, validation activities can still be carried out during the development phase, focusing on specific areas. One way to validate an architecture is to compare it with the requirements. In this scenario, validation serves as the first line of defense against potential errors in interpreting the requirements[25].

2.3.1 V-shaped model

In the automotive sector, the V-model is often used to connect the management of different phases in the production of safety-critical mechatronic systems. Although the production process goes through multiple iteration cycles rather than strictly following all phases in this order, this diagram uses a top-down design technique and a bottom-up validation strategy.

The compliance of the output with the model specifications is verified during the verification process[26]. An inaccurate specification could lead to a defective product because verification only ensures proper conformity between output and specification. Additionally, it is more challenging to identify the root cause of a defect encountered after the design phase, and there is a possibility of drawing an incorrect conclusion.

Therefore, "validating" the SUT (System Under Test) against its requirements is essential, especially for homologation and certification objectives. Validating a fault-tolerant model is more difficult given its limitations. Indeed, it is a significant task to confirm every potential failure characteristic and replicate every test scenario in which the system operates. From the early stages of building an ADAS (Advanced Driver Assistance Systems) system, simulators are employed to test and model the final outcome. Since virtual tests employ the same conditions every time without incurring hardware damage, they have complete repeatability. Furthermore, the full availability of the hardware model enables complete fault injection, additional debugging, and test monitoring.

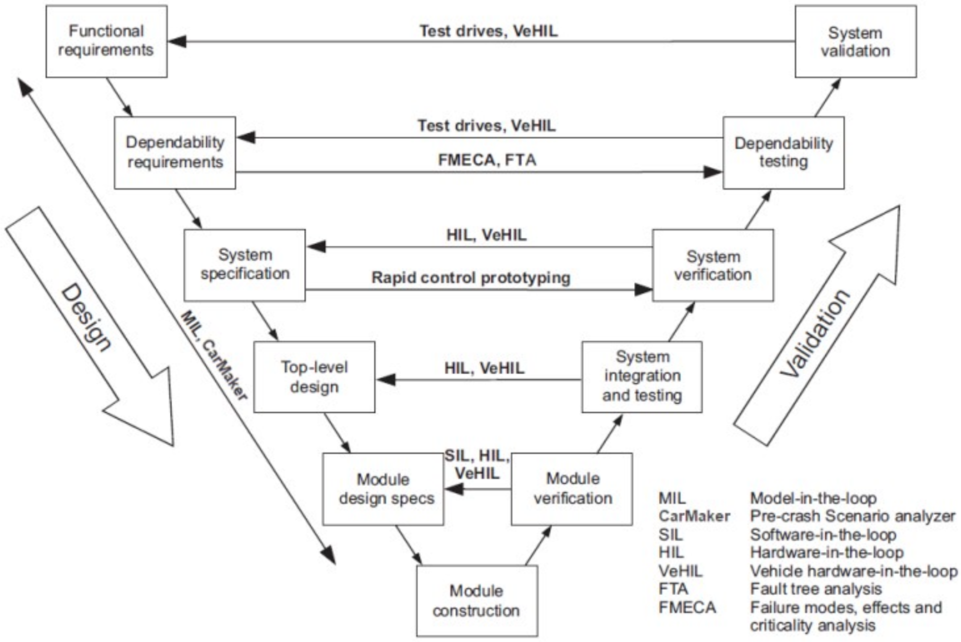


Figure 2.8: V-shaped model.

However, since prototypes can emerge before a physical model or even complete design, models rely on information obtained from previous systems that are comparable to the current process or from development tests of the same. Because the quality of a virtual simulator depends on processing capability, it is also challenging for the simulator to generate accurate results due to incompatible tool interfaces in most of them. Therefore, it is evident that simulation cannot completely replace a physical test, even though virtual tests are widely used in the system development process and physical tests are only used to corroborate hypotheses and calculations made virtually[27].

In-the-loop simulations

To provide rapid, adaptable, and repeatable tests, the automotive industry utilizes multiple "in-loop" simulation approaches to develop and evaluate Advanced Driver Assistance Systems (ADAS) control systems:

- **model-in-the-loop (MIL)**: In this initial step, both the vehicle model (plant) and the controller (system) are simulated on a PC.
- **software-in-the-loop (SIL)**: Actual code is developed and tested based on the processor or FPGA that will be used for the final hardware implementation.

Currently, software models are used to simulate the plant and controller on the PC.

- **processor-in-the-loop (PIL)**: While the plant is still simulated on the PC, the ADAS controller is tested using real-time hardware.
- **hardware-in-the-loop (HIL)**: Since the entire system operates on real-time hardware, it is one of the most crucial steps in ADAS control testing. Parts can be emulated or real; for example, the controller can act as a real ECU, while the car can be simulated using a peripheral that generates the same inputs and outputs as a real one.
- **Vehicle hardware-in-the-loop (VeHIL)**: It is a simulation that combines the advantages of real driving tests with those of virtual simulations.

2.4 Regulations and technical requirements

Every nation has its own set of regulations that govern the specifications and testing of ADAS and automated driving systems. The following concepts, defined in the subsequent subsections, must be understood to grasp the scope of a particular type of document:

- **International Standards**: Documents that outline standards and specifications for goods, services, and systems to ensure their effectiveness, safety, and quality.
- **International Regulations**: Comprehensive guidelines for the application of legislation. Compliance is required.

2.4.1 Automated Driving

Automated driving (AD) and Advanced Driver Assistance Systems (ADAS) are the result of a rapid shift in focus from electronically activated activities during routine driving, such as ABS, to the driving task itself. The SAE J3016 standard, developed several decades ago, outlined six different levels of driving automation to identify future technology. It is noteworthy that current technologies are approaching the fifth level of automation as defined by SAE; however, regulations and legal restrictions hinder their usage, primarily due to safety concerns.

2.4.2 Functional safety

The phrase "Functional Safety" gained importance with the release of the ISO 26262 standard in 2011, which marked a significant turning point in the development of AD and ADAS technology[28]. The aforementioned document is a risk-based standard aimed at reducing potential risks caused by malfunctions of automotive mechanical subsystems as well as electrical and electronic systems. To achieve this goal, the standard covers as many use cases as possible at an abstract level and refrains from specifying validation techniques, deferring all specific development and testing information to other more detailed standards such as ISO 26022, ISO 22737, and ISO 15007, among others. International regulations such as UNECE Regulation 152 (for AEBS) and Regulation 157 (for ALS) also govern the use of these specialized systems.

2.4.3 Advanced Driver Assistance Systems

All ADAS systems must meet standards and regulations before they can be formally introduced to the market. Additionally, through Regulation No. 2019/2144 (Table 2.1), the European Commission is also responsible for mandating specific systems, already installed in cars for a specific period, in all new passenger cars registered in order to maximize road safety. EU regulations specifically related to the system under consideration in this document are listed below.

Safety Features	Regulatory Acts	Cat. M	Cat. N
Driver Drowsiness and Attention Warning System	EU 2021/1341	X	X
Advanced Driver Distraction Warning System	EU 2019/2144	Y	Y
Intelligent Speed Assistance	EU 2021/4455	X	Y
Alcohol Interlock Installation Facilitation	EU 2021/1243	X	X
Emergency Stop Signals	UN Regulation No. 53	X	X
Reversing Detection System	UN Regulation No. 158	X	X
Event Data Recorders	EU 2016/679	XZ	XZ

Note:
X: Starting from July 6th 2022
Y: starting from July 7th 2024
Z: Starting from January 7th 2026

Table 2.1: Safety Features and Regulatory Acts[29]

Driver Drowsiness and Attention Warning Systems

By July 6, 2022, according to the European Commission, all new cars must have a ADDW installed. A "system that evaluates driver attention through vehicle system analysis and warns the driver if necessary" is defined as such by Regulation No. 2019/2144[30]. Regulation No. 2021/1341[31], published by the European Commission, sets the technical specifications. It prescribes that the system must:

- Function in all weather conditions, day and night (in compliance with ISO 15008:2017 standard).

- Operate without using biometric information.
- Activate when the speed reaches 70 km/h and within 5 minutes of reaching that speed, as apart from the technical difficulty in detecting fatigue, it is less likely for drivers to fall asleep in urban environments.
- Monitor the driver's level of fatigue and sound an alarm if level 8 is exceeded, based on KSS or another adequately selected and justified metric (such as PERCLOS).
- Maintain a closed-loop process that only retains absolutely necessary data (in line with the data reduction philosophy).

Part 2 of Annex 1 also contains the testing requirements[30]. Specifically, the tests must be conducted with a minimum of ten participants, each of whom must generate at least one true positive and one false negative event within a period of time ranging from five to fifteen minutes, regardless of weather conditions. Lastly, if the test is conducted using a driving simulator, the manufacturer must clearly describe the limitations compared to real-world conditions.

Additionally, acceptance criteria are discussed, with particular attention to two indicators. Finally, the law specifies that the required thresholds for sensitivity and the second criterion must be lower by 5% and 2.5%, respectively, if the system is tested on actual roads.

Advanced Driver Distraction Warning Systems

EU regulations mandating the installation of an Advanced Driver Distraction Warning (ADDW) system in cars starting from July 6, 2024, aim to assist drivers in maintaining focus while driving and alert them if they become distracted. Two particular distractions must be addressed:

- **Type 1**: Described as a single, prolonged gaze for at least two seconds in a non-driving region.
- **Type 2**: Defined as alternating gaze between driving activities and distracting activities.

Experts suggest that due to privacy concerns, the system should primarily monitor the driver's eye movements using a camera and secondary indicators related to the car and surrounding environment. Additionally, the technology is designed to operate in situations where drivers may obscure their face or eyes. Finally, additional sources of disruption, such as lighting, weather, and/or driving circumstances, need to be considered. The following tables provide a summary of many relevant metrics that should be utilized[2].

General indicator	Specific Indicator
Steering wheel	SD of Steering Wheel Angle Steering Wheel Steering entropy High Frequency Component of Steering Wheel
Accelerator	Throttle Hold
Brake	Brake Reaction Time

Table 2.2: Driver Input Data as reported by the NHTSA[2]

General indicator	Specific Indicator
Lane Position	SD Lane Position
Speed	SD Speed
Following time	SD Headway

Table 2.3: NHTSA recommended metrics for vehicle states[2]

Currently, there is limited information available on the testing and validation of these types of systems. The only document with some guidance and integration of Regulation No. 2019/2144[32] has been drafted by the National Highway Traffic Safety Agency (NHTSA)[2].

Due to the lack of essential guidance from EU regulations, which are subject to monthly changes, a validation framework for ADDW systems will be developed as part of the current effort.

General indicator	Specific Indicator
Glance Frequency	Mean/SD Glance Frequency
Glance Duration	Mean/SD Percent Glance Durations Off Road
Percent of gaze on road center	Percent Road Center
Percent of gaze off the road	Mean/SD Percent Off Road
Gaze direction	SD Horizontal (Gaze or Head)
Gaze direction	SD vertical (Gaze or Head)

Table 2.4: NHTSA's eye/head movement metrics[2]

General indicator	Specific Indicator
Pupil	Pupil Size
Blink	Blink Rate
Skin conductance	Galvanic Skin Response

Table 2.5: NHTSA's eye/head movement metrics[2]

Chapter 3

The Metahuman plugin

In the current context of automotive engineering, safety plays a paramount role, especially concerning Advanced Driver Assistance Systems (ADAS). Among the various functionalities of such systems, the accurate detection of driver attention emerges as a crucial element, as it significantly impacts accident prevention and enhances road safety measures. Integrated within the vehicle, ADAS constantly monitor the driver's actions, requiring a comprehensive set of data for their validation. Such data can be effectively generated within a virtual environment, provided it faithfully reflects reality and complies with regulatory requirements.

A prominent virtual environment that stands out for adhering to stringent standards is Unreal Engine, seamlessly integrated with Epic Games' Metahuman plugin. This dynamic set of tools offers an unprecedented level of realism and flexibility in human representation, making it exceptionally suitable for simulating situations and interactions within automotive contexts. The power inherent in this plugin lies in its ability to accurately recreate a wide range of facial expressions, eye movements, and head gestures, thus facilitating the reproduction of different driver distraction scenarios and providing essential data for validation.

Thanks to this innovative tool, it is possible to develop a series of dynamic scenarios that reflect the multiple facets of human behavior during driving. This enables the validation of the ADAS systems in question, making a significant contribution to enhancing automotive safety protocols.

A crucial step in completing the project is the installation and integration of the Metahuman plugin with Unreal Engine, which requires careful verification of system prerequisites. For it to function properly, the computer must meet the technical specifications established by Epic Games, including CPU speed, amount of RAM, available disk space, and, most importantly, the presence of a powerful GPU.

Once system compatibility has been ensured, the next step is to acquire the Metahuman plugin, which is available on the Epic Games Marketplace. The process of obtaining the plugin is relatively straightforward and requires only a few clicks. Subsequently, you proceed with the installation, which may vary in complexity depending on the version of Unreal Engine used. Typically, this involves activating the plugin through the plugin menu in the Unreal Engine editor, making it accessible for both the current project and future developments.

Figure 3.1: Metahuman plugin

After installation, attention shifts to the preliminary configuration, which requires specific customizations of the plugin based on the project's needs. It is essential to verify the presence of all necessary libraries and software to ensure smooth operation.

Subsequently, a thorough system check is conducted to ensure that everything is working correctly. Careful analysis ensures that each component has been installed accurately and is operating at one hundred percent.

3.1 Advanced Technical Aspects

Metahumans represent the pinnacle of human simulation technology, leveraging cutting-edge technologies to create extraordinarily realistic characters. These fundamental technological advancements, essential for the sophisticated rendering of Metahumans, seamlessly integrate to produce an engaging result.

One of the crucial elements lies in subsurface scattering, a fundamental technique in skin rendering, which mimics the intricate interaction of light with skin layers to give Metahuman physiques depth and authenticity.

Equally relevant is the simulation of fabrics, clothing, and accessories. Through the use of advanced physical algorithms, the software ensures realistic movements and seamless integration within the virtual context, thereby enhancing simulation accuracy and getting closer to reality.

Regarding facial animations, these are made possible through meticulous facial rigging and the use of motion capture methodologies, allowing for the reproduction of a wide range of human emotions. This capability is particularly crucial in scenarios where character dynamics and interpersonal exchanges are central.

Completing the picture is the motion capture infrastructure, which captures facial movements in real-time through cameras, further enhancing the naturalness and authenticity of Metahuman animations. Interaction with virtual and environmental elements enables characters to react believably to stimuli and circumstances, significantly enriching the simulation.

These cutting-edge technologies not only increase the visual fidelity and interactive capabilities of avatars but also represent a fundamental element in providing a reliable and realistic platform for exploration and innovation. In ADAS systems, precision and authenticity are indispensable requirements for effectiveness and system evolution.

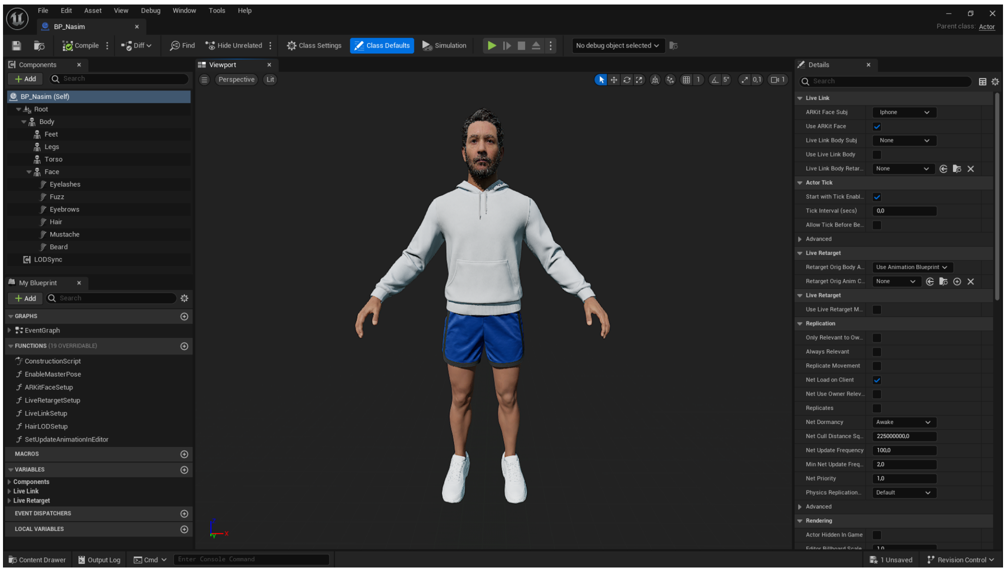


Figure 3.2: Example of an Avatar.

3.1.1 Practical Examples and Case Studies

Among various applications, a particularly significant use case is the study of interactions between individuals and infotainment systems present in automobiles. This innovative approach allows for meticulous examination of how such interactions influence driver attention, reducing road safety risks associated with real-world testing.

Another important application is the implementation of cabin monitoring systems to detect the presence of occupants, especially children, in the absence of other passengers. This is crucial to prevent cases of child abandonment in vehicles, reducing potential threats to passenger safety.

Further applications can be observed in research projects conducted by both automotive companies and academic institutions. In these initiatives, Metahumans are used to assess drivers' responsiveness to a range of distraction signals. Through these simulations, it is possible to carefully examine and document drivers' reactions in controlled environments, avoiding the complexities and dangers associated with on-road testing involving actual subjects.

These examples highlight the crucial role of Metahumans in examining and optimizing the safety and effectiveness of automotive systems. Simulations based on this technology provide a safe and adaptable environment to explore a wide range of scenarios, significantly contributing to automotive engineering and road safety research and development.

3.2 The Diversity of Metahumans and Its Benefits in Simulation

Avatar systems constitute valuable resources due to their ability to accurately capture and replicate a wide range of human attributes and behaviors. These highly versatile avatars enable the creation of scenarios that accurately reflect demographic diversity, incorporating variables such as age, gender, ethnicity, and other distinctive characteristics. This flexibility provides a comprehensive overview of how individuals from different backgrounds interact with automotive systems.

The diversity of Metahumans not only adds a high degree of complexity to simulations but also plays a crucial role in identifying and correcting any inherent biases in detection systems. This careful analysis ensures the optimization of ADAS technologies for all users, regardless of their individual characteristics. By organizing simulations involving diverse driver profiles, it is possible to tailor and improve systems to ensure an intuitive and efficient experience for a wider audience, significantly contributing to road safety. Furthermore, the use of Metahumans to evaluate systems in different cultural contexts offers the opportunity to assess the universality and adaptability of ADAS systems.

In conclusion, Metahumans facilitate more inclusive and representative testing practices, accelerating the refinement of driver assistance systems and promoting the development of safer and more accessible vehicles on a global scale.

Figure 3.3: Diversity of Metahumans.

Chapter 4

Metahuman animation

This chapter undertakes a comprehensive exploration, delving into the necessary steps to enable Unreal Engine to interact with the Live Link Face application. This integration represents a crucial phase, optimizing the use and customization of avatars within the virtual scene and accelerating the development of realistic facial expressions.

Throughout the chapter, essential procedures will be outlined to establish and successfully consolidate the interface between Unreal Engine and the Live Link Face application. Special attention will be devoted to the detailed analysis of the fundamental steps involved in activating and optimizing the required plugins, establishing a reliable connection between Unreal Engine and the iOS device, and accurately transposing facial movement data onto the intricate details of 3D models. Initial Configuration of the Live Link App.

To initiate the implementation of the system and ensure seamless integration between Unreal Engine and the device, the first essential step is activating specific plugins within the software. These plugins extend beyond Live Link alone, encompassing the fundamental Apple ARKit and Apple ARKit Face Support plugins, which lay the groundwork for a solid integration.

Furthermore, it becomes essential to proceed with careful configuration of the Live Link Face app on the iOS device. This configuration primarily focuses on synchronizing the device with Unreal Engine by connecting them to the same Wi-Fi network. Additionally, particular attention is given to configuring the application to ensure accurate transmission of data to the designated IP address of the computer.

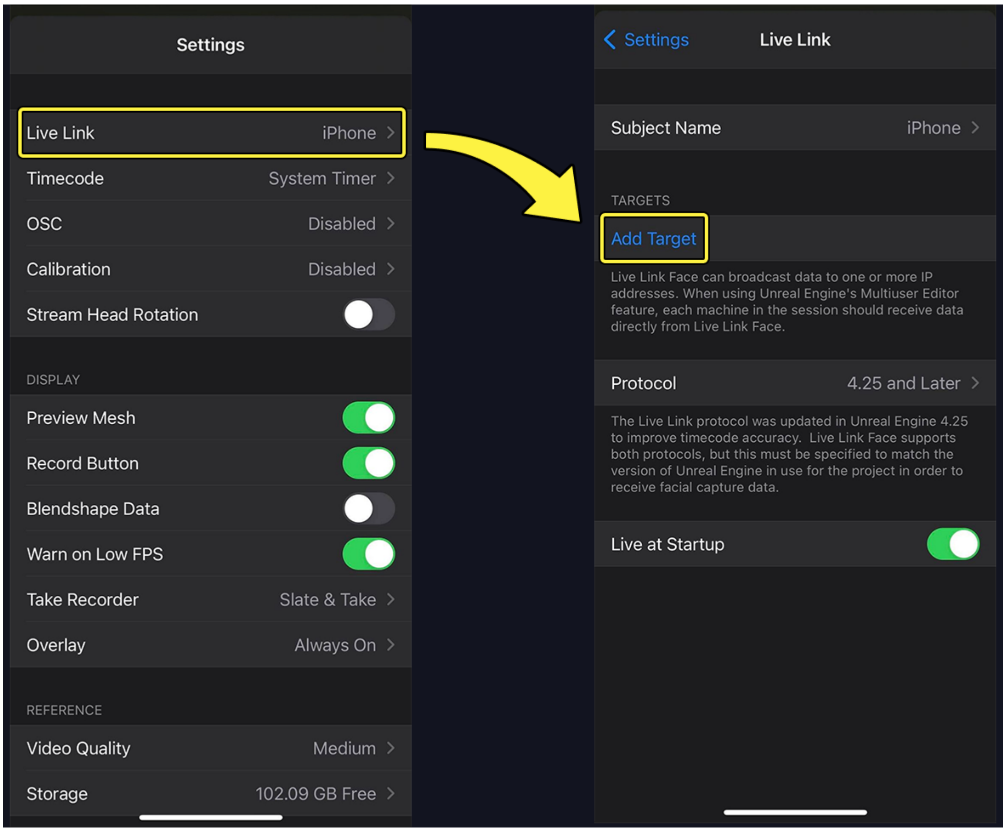


Figure 4.1: The settings of the Live Link Face application

Once this network of connections has been carefully established, Unreal Engine is ready to receive streams of data related to facial animation. From this point onward, a sophisticated mapping mechanism comes into play, integrating this data stream with the virtual character. This results in a transformation from raw facial expression data captured by the application to smooth animations within the virtual environment, accurately replicating the subtle nuances of the real subject.

However, this transition from raw data to realistic animation is anything but simple. It depends on the complexities of facial rigging and precision in expression mapping, requiring meticulous attention to detail.

The transfer of facial data between the mobile device and the software occurs through two main modes: the first is real-time, while the second occurs using prerecorded video.

In real-time mode, every movement and expression of a real individual is instantly replicated on the avatar's face. To implement this application, expertise in facial rigging technologies and complex Application Programming Interfaces (APIs) is essential.

The system consists of an iOS smartphone configured in "Live Link (ARKit)" mode, which uses its TrueDepth camera as input to capture the expressions of the real person. Subsequently, the data captured by the device is transmitted to the workstation and then to the Unreal Engine software, where it is immediately converted and replicated on the virtual avatar.

For accurate and well-replicated avatar representation, specific parameters need to be configured within the metahuman blueprint. Particularly, it's important to set the animation acquisition source, specifying the connected physical device.

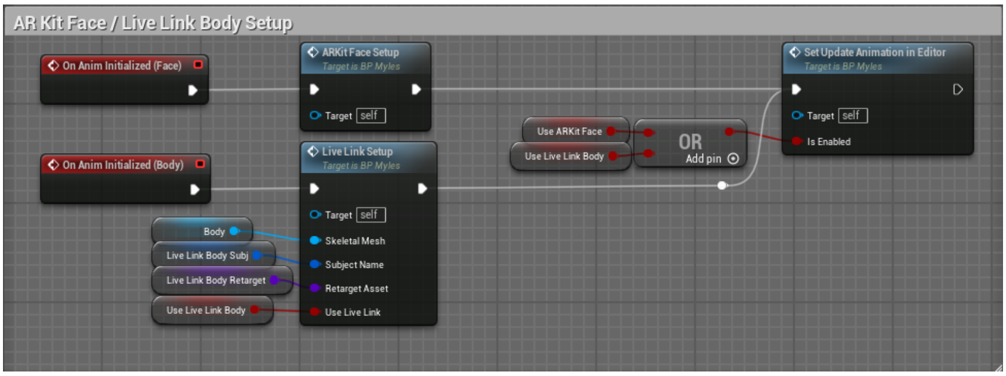


Figure 4.2: Blueprint of metahuman for real-time connection

The second mode adopts a different technique, based on prerecorded videos. This approach differs from the previous mode because the real person's animation and the avatar's animation do not occur simultaneously; rather, the animation of the physical person is recorded first and then applied to the avatar's face.

The first step to using this mode involves configuring the application on the mobile device, setting it to "Metahuman Animator" mode. This allows recording a video saved directly within the application itself.

Subsequently, it's necessary to create and record two sets of videos: one for calibration and the other for the main animations to replicate. To import the videos into Unreal Engine, it's essential to add the "Capture manager" to the project, which contains all the settings to recognize the device as the video source. A crucial parameter is the IP address of the mobile device on which the videos were generated. Once set correctly, you can locate the mobile device through the software and import the generated videos.

For accurate and well-replicated avatar representation, specific parameters need to be configured within the metahuman blueprint. Particularly, it's important to set the animation acquisition source, specifying the connected physical device.

The second mode adopts a different technique, based on prerecorded videos. This approach differs from the previous mode because the real person's animation and the avatar's animation do not occur simultaneously; rather, the animation of the physical person is recorded first and then applied to the avatar's face.

The first step to using this mode involves configuring the application on the mobile device, setting it to "Metahuman Animator" mode. This allows recording a video saved directly within the application itself.

Subsequently, it's necessary to create and record two sets of videos: one for calibration and the other for the main animations to replicate. To import the videos into Unreal Engine, it's essential to add the "Capture manager" to the project, which contains all the settings to recognize the device as the video source. A crucial parameter is the IP address of the mobile device on which the videos were generated. Once set correctly, you can locate the mobile device through the software and import the generated videos.

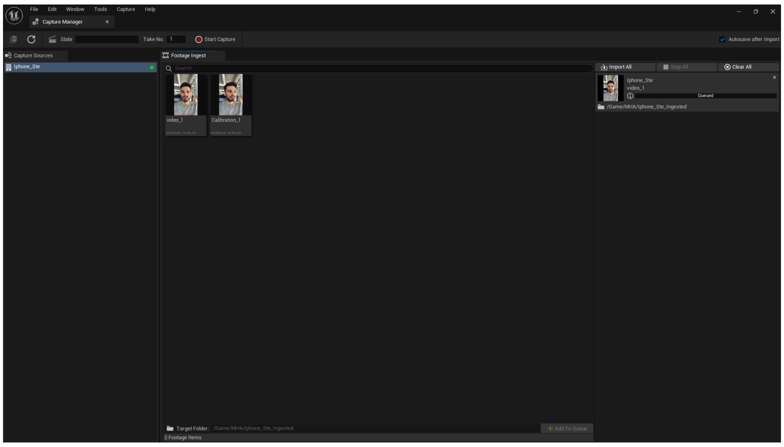


Figure 4.3: Pre-recorded videos

The first video used is for calibration purposes, creating the "Metahuman identity". During this phase, four frames are selected from the video: one with the person facing forward, two frames showing the right and left sides of the face, and the last one displaying the person's teeth. These frames are processed by the software to create the "Skeletal mesh", which is crucial for the next phase.

Next, the second video is used in conjunction with the two files generated previously. Each frame of the person's video is processed to transform it into a virtual animation to apply to avatars. To achieve this, the "Skeletal mesh" is utilized to process each point of the person's face and create the complete animation.

Once the animation is exported, a "Sequencer" can be created to reuse it and associate it with a specific avatar. During this phase, it's necessary to modify a parameter of the newly generated animation to allow the avatar to faithfully replicate the animation without locking the neck position. This parameter is the "Blend", where the head parameter needs to be set from zero to one. Once this step is completed, the avatar will be able to faithfully replicate the animation generated from the video.

4.1 Animation Customization

Customizing and adjusting specific animation parameters are crucial for generating videos and sequences useful for testing and validating in-cabin monitoring systems.

In particular, two different methodologies can be adopted: generating animation from scratch, i.e., without any pre-set parameters, or modifying data generated from a prerecorded video sequence.

In the first case, starting from an avatar without animation, one proceeds frame by frame, manually setting all facial parameters. This approach requires considerable time for creating the complete simulation and may not accurately replicate real facial behaviors. However, precise facial parameter data for testing purposes are obtained.

In the second case, starting from a base of parameters generated from a prerecorded video, facilitates the generation of a more realistic video sequence since the movements stem from a real person and in less time. However, the data would only be slightly modified from those of the original video, thus significant alterations to the data sequence would not be possible, only minor adjustments for testing purposes.

For example, if one wishes to replicate a person's startled reaction while driving, creating data from scratch might be complex and the final result might not accurately reflect real-life behaviors. Instead, starting from a video created by a real actor would certainly yield more realistic behavioral sequences.

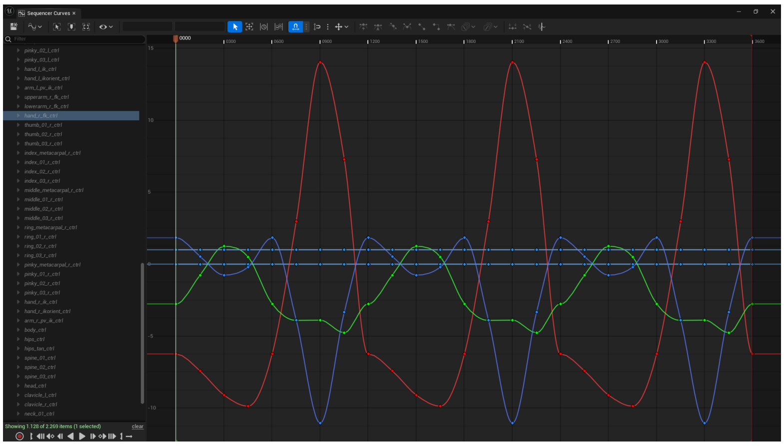


Figure 4.4: Example of graphics generated for avatar animation

4.2 Environment

A crucial aspect that plays a fundamental role in realizing this project is the consistent and synergistic integration of the avatar with the surrounding environment, with particular emphasis on the automotive model. In our specific context, within a car cabin, it is essential to ensure an intrinsic harmony between the avatar and the car design, respecting aesthetic, functional, and technical criteria.

The integration process begins with the meticulous selection of the car model, which must meet specific requirements to ensure a smooth and organic interface. Each car, with its peculiar characteristics, significantly influences the interaction between the avatar and the surrounding context. Elements such as cabin size and dashboard complexity directly impact the visual and technological experience of the interaction.

Figure 4.5: Virtual car model

The choice of a model equipped with modular components has proven crucial, allowing targeted modifications aimed at ensuring greater realism and a variety of options. For example, customizing the car's interior seats and steering wheel is crucial for adapting the avatar's dimensions, ensuring smooth interaction and perfect synchronization between the avatar itself and the steering wheel.

The selected car model, carefully identified through meticulous research in the Unreal Engine marketplace, is the renowned "Car Configurator" plugin, which offers a fully customizable virtual replica of an Audi car to meet the specific needs of our project.

Once the car replica was acquired, attention shifted to integrating the avatar into the model, with particular focus on the arrangement of seats, steering wheel, and all the internal equipment of the car, aiming to make it as natural as possible. This integration involves optimal sizing of the avatars, defining their positions, and managing associated animations, carefully considering available spaces.

Effective integration of the avatar in this context not only enhances immersion and interactivity for the end-user but also opens up new innovative horizons in the fields of driving simulation and automotive infotainment systems.

4.2.1 Advanced Integration and Parametric Spawn of Virtual Characters

The integration of virtual characters into an automotive environment, as we have already explored, is of crucial importance. However, it is equally essential to consider the implementation of dynamic and parametric spawning for such virtual actors.

To further delineate this concept, parametric spawning of actors within the Unreal Engine scene allows for the dynamic generation of specific entities and events in response to specific behaviors, actions, or conditions within the simulation. This not only enriches the user experience but also increases overall flexibility and interactivity of the simulation.

In our specific case, it is vital to realize targeted spawning of virtual characters to ensure accurate interaction with the steering wheel, pedals, and other elements of the car's cockpit.

The first step to implement parametric spawning is the creation of a specific Blueprint for each virtual actor, serving as a template for the avatar. Within this Blueprint, not only the aesthetic aspects of the virtual character are defined, but also all the elements necessary for animation and interaction with the environment. A series of parameters can be configured for the actor, such as the position, orientation, and scale of the avatar. This type of Blueprint is fully integrated into the Metahuman plugin, thus offering an immediate and sophisticated solution to meticulously map every part of the avatar's body.

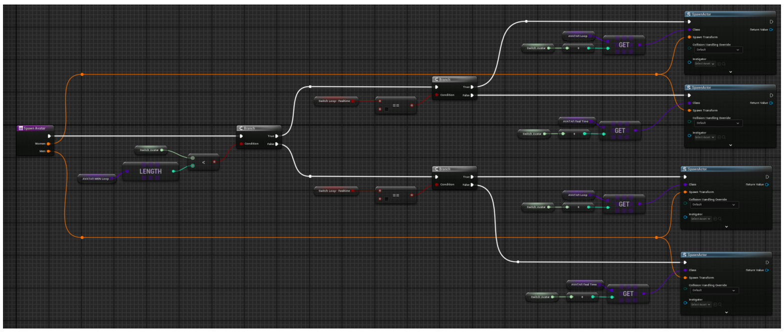


Figure 4.6: Blueprint for parametric avatar spawning

This integration allows for precise linkage of each anatomical element of the virtual character, ensuring movements and interactions managed with a remarkable level of detail and precision. Thanks to this advanced synergy between Blueprint and Metahuman, extremely realistic animations can be created, significantly enhancing the immersion and interactivity of simulations.

After defining the Blueprint of the virtual character, the next step involves creating the logic for spawning. This can be accomplished through another Blueprint specifically designed for this task, which could be a level Blueprint or a spawn manager.

The basic logic relies on selecting the actor to spawn and the desired position. This entails implementing a series of nodes, the main one being "Spawn Actor from Class," which allows selecting the type of actor to spawn. Subsequently, various elements can be used to determine and index the actor's position, such as "Get Actor Location" and "Set Actor Transform," defining the specific position and orientation of the actor.

A critical aspect of parametric spawning is its ability to adapt in real-time to events, such as environmental changes or user actions. The spawn Blueprint must be designed to be responsive and flexible, adjusting the generation of virtual characters to evolving circumstances. For example, if the user adjusts the interior environment of the cockpit during the simulation, virtual characters must adapt to such changes in real-time.

In our project, we have adopted an actor spawn approach at the simulation's start, using parametric logics that determine their position based on the car's position. This method ensures that each virtual character is optimally positioned relative to the vehicle's internal elements, ensuring natural and consistent interaction. In subsequent development stages, we will further explore the introduction of a randomization element for virtual characters, to demonstrate how parametric spawning facilitates not only diversity but also scene variation quickly and effectively.

This technique of parametric spawning proves fundamental to our goal of making the simulation more versatile and responsive, allowing rapid scene customization. The adoption of this method goes beyond spawning virtual characters, extending to numerous scenic components such as cameras, lighting, and vehicles, emphasizing the importance of a flexible and dynamic spawning system.

4.2.2 Detailed Animation of the Avatar's Body and Steering Wheel

After effectively implementing the avatar spawn technique in the scene, we directed our focus towards creating detailed animation for both the avatar's body and the steering wheel. This approach was adopted not only to ensure overall consistency with the simulation but also to enhance the level of visual realism, making the experience more faithful to reality.

While animation might not be considered crucial for our ultimate goal, which primarily focuses on simulating driver visual and behavioral distractions, it still plays an important role in showcasing the system's capabilities. Carefully designed and structured animation can reproduce not only the driver's facial expressions but also gestures and habits such as smartphone usage, smoking while driving, or distraction caused by the car's infotainment system.

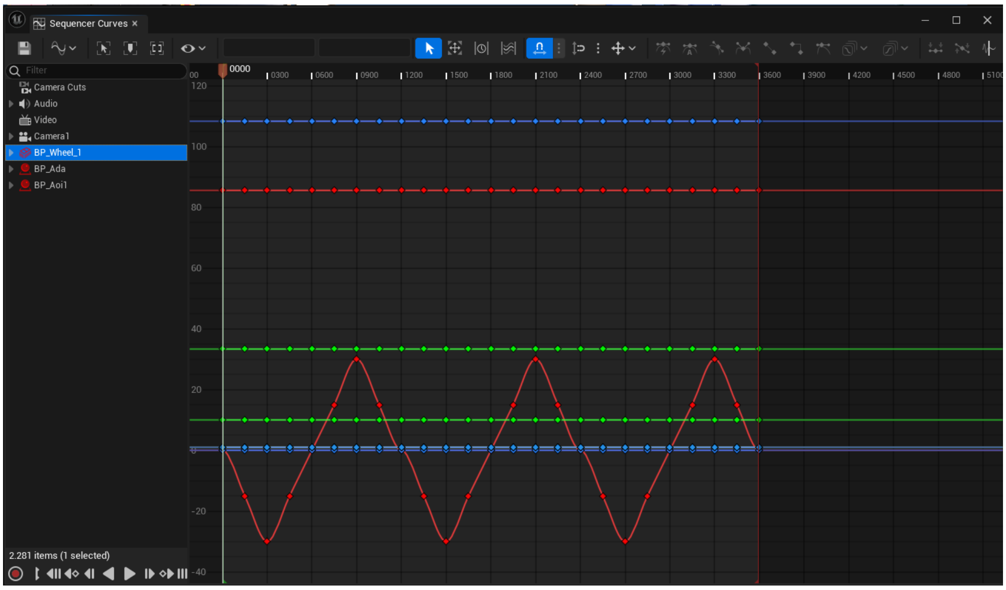


Figure 4.7: Wheel animation.

The process of creating these animations relies on the use of Unreal Engine's Level Sequencer, a powerful feature that allows for the generation of video sequences. The Level Sequencer enables the modification of actors' positions and orientations frame by frame, facilitating the creation of smooth and realistic animations.

We focused on synchronously animating both the car's steering wheel and the avatar's body. For the steering wheel, we developed an animation that recreates the smooth and consistent left and right steering movement, aiming to make it as natural as possible.

Animating the avatar's body proved to be more complex, as any modification to a single body component affects the structure of the entire avatar. After a series of iterations and careful adjustments, we managed to achieve a satisfactory solution for our project, even though it was not at the forefront of our main focus.

In conclusion, the combined use of the Level Sequencer and Blueprint programming proved crucial in creating a standard and reusable animation in various situations. This approach allowed us to avoid the need for separate animations for each avatar and scenario, ensuring a more efficient and streamlined development process.

4.2.3 External Environment and Data Collection

In the conclusive analysis of the project, a brief period was dedicated to studying elements external to the car cabin within a virtual environment. Although this aspect is not directly related to recognizing driver distraction behaviors, integrating data from outward-facing cameras with those from internal monitoring cameras proved essential to validate the collected information and identify false positives generated by the internal cameras. Exclusive analysis of internal camera data does not definitively determine what the driver is focusing on or looking at. For instance, the driver being oriented towards one side instead of concentrating on the road doesn't necessarily imply a lack of attention to driving; it could be due to the presence of another vehicle or obstacle in the direction the driver is facing, indicating careful attention to the driving situation instead.

To further deepen the research, an animated urban setting was developed, populated with avatars and vehicles, aiming to create a wide range of replicable scenarios.

Regarding the cameras used for image acquisition, four identical ones were positioned: one at the back of the car, one at the front, and two on the sides (one on the right and one on the left), to achieve near-complete coverage of the area surrounding the vehicle, with an angle of view close to 360°. The positioning of the cameras was illustrated in Figure 4.8.

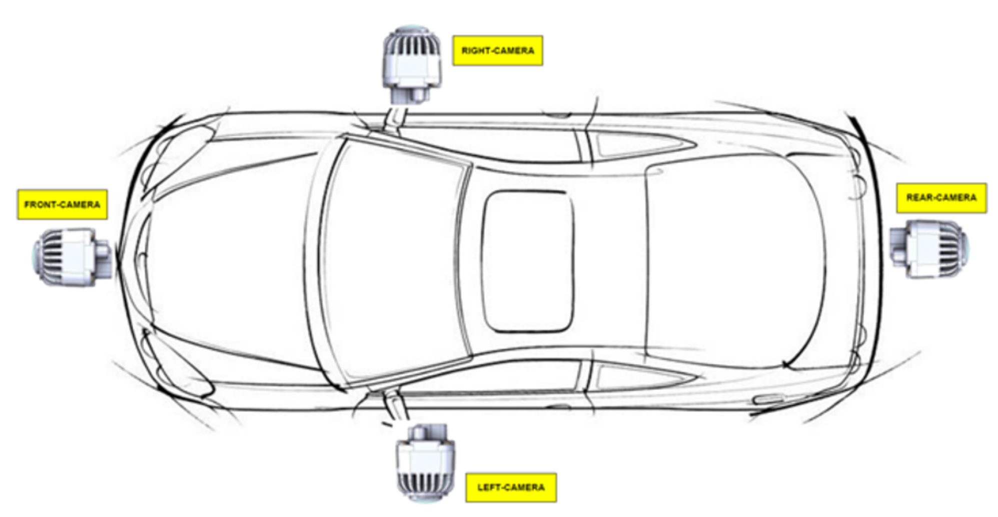
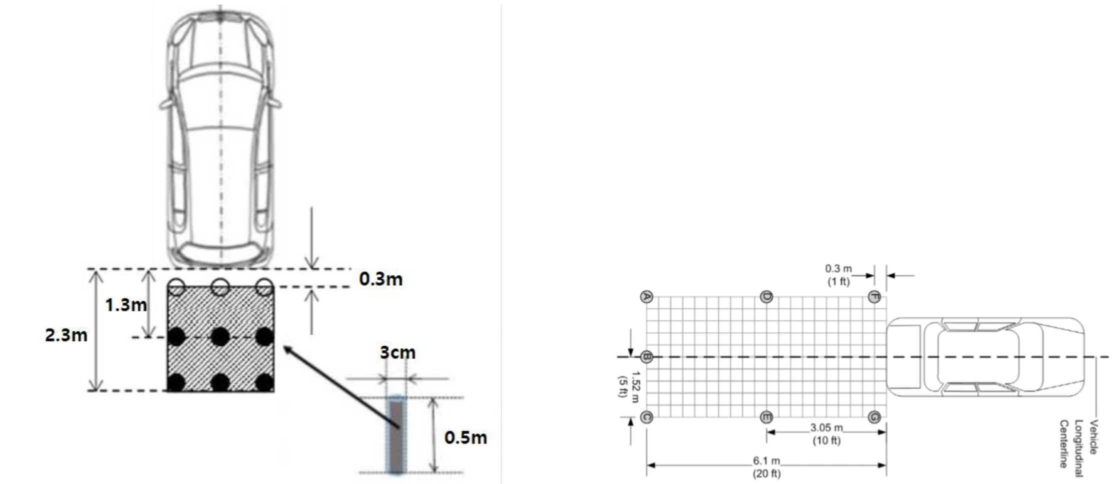


Figure 4.8: Position of the external cameras

The choice of camera type and position was guided by two tests: one following NHTSA specifications and one following KMVSS specifications. Both tests involve placing a camera on the car's trunk to capture images of the rear environment.

In both tests, an area behind the car was set up with colored poles, each with specific dimensions and proportions. The cameras needed to recognize specific portions of these poles. For the NHTSA test, the area had to be rectangular with seven poles, while for the KMVSS test, it had to be square with nine poles. The precise dimensions and positions for both tests were illustrated in Figure 4.9.



(a) KMVSS test.

(b) NHTSA test.

Figure 4.9: Environment camera test

After replicating the scene for the tests, trials were conducted using different cameras placed in various positions, aiming to capture a large number of frames to determine the optimal position and camera.

An additional test conducted was to simulate situations where the cameras were obstructed by external events such as rain, smoke, objects, or the camera's own glass breaking.

Chapter 5

Biometrics parameter

In our engineering project, a crucial aspect concerns the monitoring of the driver's cabin, with a specific focus on collecting, analyzing, and utilizing relevant biometric parameters, primarily concentrated on the face and eyes. These parameters play a fundamental role in interpreting the driver's behavioral dynamics and conditions during driving. Analyzing such data allows for customizing the driving experience based on the user's reactions and needs, significantly contributing to improving road safety.

In this context, the use of Unreal Engine proves essential: this versatile and powerful tool enables the precise capture of a wide range of driver biometric parameters, converting every facial and eye movement into analyzable data.

Data collection occurs through two distinct modes, depending on the method of generation and transmission from the live link face application to the software. In the case of simulations based on prerecorded videos, all data is accessible, analyzable, and modifiable before the simulation starts. In contrast, during real-time simulation data acquisition, they are generated and captured in real-time based on the actor's movements positioned in front of the camera.

Regardless of the acquisition method, within Unreal Engine, a sequence of parameters is obtained for each frame of the simulation, divided for each tracked facial component. This ensures accurate and detailed generation of parameter sequences ready for analysis.

5.1 Deep Analysis of Biometric Parameters

During the simulation, sixty-one biometric parameters are collected and analyzed by the application and sent to the software. These parameters provide a wide range of fundamental data for studying driver distraction and attention during driving. In particular, the combined control of different parameters and recurrent behavioral patterns during driving proves to be even more powerful for analysis.

Below are some examples of the main analyzable parameters generated during the simulation in a virtual environment:

1. Left eye blink (EyeBlinkLeft)
2. Right eye blink (EyeBlinkRight)
3. Eye movements downward, upward, inward, and outward of the left visual field (EyeLookDownLeft, EyeLookUpLeft, EyeLookInLeft, EyeLookOutLeft)
4. Eye movements downward, upward, inward, and outward of the right visual field (EyeLookDownRight, EyeLookUpRight, EyeLookInRight, EyeLookOutRight)
5. Jaw opening (JawOpen)
6. Yaw head rotation (HeadYaw)
7. Vertical head tilt (HeadPitch)
8. Lateral head tilt (HeadRoll)

These parameters have been selected for analysis in accordance with current regulations, ensuring accurate and detailed monitoring of the driver during virtual driving simulation.

In particular, the first two parameters, EyeBlinkLeft and EyeBlinkRight, quantify the degree of closure of the corresponding eyes during the simulation. Their value varies from zero, indicating a fully open eye, to one, denoting a completely closed eye. These parameters are of vital importance as they provide an immediate indication of the driver's vigilance level. A high percentage of eye closure suggests a possible lack of driver concentration, with the specific hypothesis of potential drowsiness.

The next two groups of parameters represent two identical sets, but respectively referred to the right eye and the left eye. These parameters monitor the position of the pupil within each eye and are expressed in values ranging from zero to one, indicating a specific direction of distraction. For example, EyeLookDownLeft records the gaze directed downwards, EyeLookUpLeft the direction upwards, EyeLookInLeft the direction towards the nose, and EyeLookOutLeft the opposite direction. The same reasoning applies to the right eye. These parameters are equally crucial as they provide a precise indication of where the gaze is focused and for how long. This allows for a rapid assessment of whether attention is focused on the road and surrounding obstacles, or if it is directed elsewhere. Furthermore, they enable the evaluation of situations bordering on regulatory limits, both in terms of angles of deviation from the normal gaze position, and in terms of the duration of distraction.

Another crucial parameter for assessing the driver's state is mouth opening, represented by the parameter JawOpen. This parameter indicates the degree of mouth opening, assuming a value of zero when the mouth is fully closed and a value of one when it is fully open. The JawOpen parameter provides valuable information about the driver's behavior, helping to determine if they are speaking or yawning and with what frequency. This is essential for understanding their level of tiredness and attention while driving.

Finally, the last three parameters are of utmost importance as they provide data on the complete orientation of the head. HeadYaw, HeadPitch, and HeadRoll record the rotation of the head relative to the yaw, pitch, and roll axes, expressing its relative position. These parameters can take values in the range from minus one to one. When the head is in the normal position, the values are set to zero; however, depending on the direction of rotation, they increase or decrease to one or minus one.

HeadYaw controls rotation around the vertical axis passing through the center of the head, indicating whether the driver's attention is directed to the right or left. HeadPitch monitors rotation around the axis passing through the shoulders, detecting whether the driver's attention is directed downwards or upwards, useful for evaluating distractions due to mobile device usage or signs of drowsiness. Finally, HeadRoll controls rotation relative to the axis perpendicular to the face, indicating whether the head is tilted to one of the shoulders, a crucial parameter for detecting signs of extreme fatigue or drowsiness.

These represent the main and most crucial data considered during the simulation, but it is important to note that the set of usable data could be much broader. The more data that are collected and analyzed, the more precise and comprehensive the analysis of the driver's psychophysical state becomes. The wide range of available data contributes to a thorough understanding of the driver's conditions during driving, allowing for the adoption of targeted preventive and corrective measures to ensure road safety.

5.1.1 Data Acquisition

In the context of this research, a primary focus is on the correct acquisition and storage of facial parameters outside the Unreal Engine software environment. This operation is essential to ensure the integrity and accessibility of the collected data. Therefore, the emphasis has been placed on the extraction and preservation of a specific set of previously analyzed data.

To this end, a code has been developed using blueprints to enable the continuous recording of parameters generated during simulations. This code is capable of extracting a TXT file for each simulation and recording the values of relevant parameters within it. It is important to note that the acquisition process is highly customizable, allowing the user to select which parameters to include in the storage and how frequently to perform this operation. This flexibility provides the option to choose between a denser and more detailed data collection or a more sporadic one with more significant variations, depending on the specific needs of the research.

Head Roll	Head Pitch	Head Yaw	Month Open	Blink Left	Blink Right
0,01892001	0,2010548	0,04675015	0,01701747	0,13511226	0,13506639
-0,02813737	0,1944818	0,02292297	0,02248207	0,76403111	0,76351881
0,324864	0,22841181	0,14440303	0,03002209	0,74945223	0,74892551
0,49643391	0,42767331	0,51699609	0,06345342	0,00002197	0,00002175
0,01877123	0,21473636	-0,18907604	0,02960617	0,79883492	0,80175984
-0,06595927	0,25299519	-0,44132796	0,02972192	0,10854218	0,12317633
-0,13609073	0,29414755	-0,77941805	0,05317822	0,15635955	0,22100109
0,00353778	0,2129374	-0,27996686	0,03473404	0,78100711	0,77857214
0,02428342	0,22796966	-0,02374261	0,04587141	0,78126544	0,44446105
0,01187345	0,23986189	-0,02589477	0,03204394	0,10537855	0,10511746
0,01988265	0,21763802	-0,00674247	0,05635563	0,19983009	0,83509481
-0,00091583	0,21306558	-0,02286476	0,56213069	0,00026364	0,00027919
-0,0040002	0,22967869	-0,14570652	0,28720394	0,00008356	0,00008848
-0,06314762	0,21156633	-0,44019872	0,13762911	0,3825514	0,38162449
-0,07285941	0,24226014	-0,45831645	0,04424928	0,05112623	0,05100989
0,00202064	0,19167465	0,32508785	0,03281929	0,00275631	0,00247508
0,0833212	0,20922624	0,74185109	0,05834779	0,21648219	0,21576239
-0,08815973	0,02211794	0,49985707	0,02954158	0,27900565	0,26500335
-0,04592923	-0,12446313	0,01541862	0,0264399	0,00684302	0,00682775
-0,06063072	0,08133638	-0,30973879	0,02348854	0,09651974	0,11855058
-0,14767903	0,21134718	-0,61313188	0,03097793	0,05781417	0,07137082
-0,19462609	0,31443363	-0,7363857	0,04879508	0,13326414	0,17016791
-0,15124668	0,33577609	-0,61844939	0,05016378	0,09293227	0,09970241
-0,07388483	0,33335462	-0,28931686	0,05941242	0,15295088	0,15586704
0,0613637	0,33857498	0,29256648	0,05501249	0,08032713	0,09737213
0,09106616	0,27338865	0,65931809	0,06186603	0,17572911	0,17497832
-0,15651025	-0,16028772	0,66878879	0,10391679	0,44473019	0,44446105
-0,15270986	-0,27245539	0,1696455	0,05189388	0,50361377	0,51554781
-0,08893833	-0,17042132	-0,36964753	0,03882285	0,00314861	0,00277059
-0,02288471	-0,06527988	-0,29465467	0,02584998	0,19853872	0,19851415
-0,02521471	0,38983631	0,00043883	0,05035721	0,00647722	0,80301577
-0,00707899	0,56388152	-0,01343003	0,07266673	0,00690429	0,72007155
-0,01254194	-0,22983666	-0,04102144	0,03280425	0,89436173	0,89434075
-0,00155265	-0,26485524	-0,03188777	0,05209985	0,91365069	0,91848058
-0,02592377	0,48312625	-0,00795445	0,04824206	0,08239024	0,08174307
-0,0041848	0,75282574	0,0235647	0,05200456	0,57087696	0,56757212
0,15848526	0,66598862	0,27854794	0,05092975	0,29431874	0,25932801
0,80617929	0,56617504	0,14993535	0,04106379	0,11956068	0,08118044

Table 5.1: Example of biometrics parameter

5.2 Eye gazing feature

Among all the parameters considered and extracted during the simulation, one of the most complex to evaluate and understand is undoubtedly Eye Gazing.

Eye Gazing implies the ability to precisely determine where the driver's gaze is focused; that is, to understand exactly where the person behind the wheel is looking and where they are directing their attention. This is of paramount importance for safety while driving.

This aspect can be inferred as a parameter from the generated data related to the relative position of each pupil within each eye. However, such an approach would not be exhaustive as Eye Gazing is the result of the summation of the relative positions of various components such as the eyes and the head.

Even if it were possible to obtain a precise value obtained from the interaction of multiple positions, this would only have a numerical value, useful for validation according to current regulations but not sufficient from a graphical standpoint. For this reason, I have focused my project on the graphical representation of this parameter.

The ultimate goal of this implementation is to make visually evident and easily recognizable, in a virtual environment, where the avatar is looking and where they are directing their attention. The first step was to clearly understand and structure the desired visual rendering. We chose to associate a beam of blue light directly with the avatar's pupil, illuminating the surrounding environment and indicating the direction of the gaze.

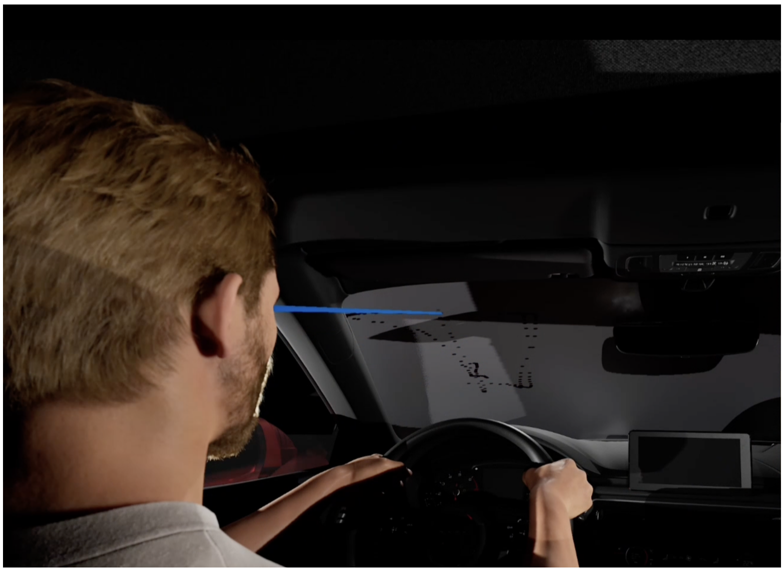


Figure 5.1: Eye-gazing

The technical process to implement this functionality proceeded as follows:

The first step involved working on the attachment of light beams to the avatar's eyes, achieved by placing two sockets inside the pupils of the avatar's eyes.

Sockets are virtual anchoring points within three-dimensional models. These points allow for dynamically positioning objects or components within the model, enabling the creation of interactive environments and realistic animations. They offer flexibility and precision in object placement, simplifying the creation of complex and interactive scenarios.

Using these sockets, the light beams, which are small cylinders, were programmatically generated inside the avatar's eyes and made responsive to the movements of these components. This implementation made Eye Gazing more understandable and appreciable during the simulation.

At this point, a cylindrical component attached to the avatar's pupil was obtained, interacting with the surrounding environment during the simulation, particularly with the car's windshield. It was then thought to leverage this interaction to record and make visible all the specific points where the gaze was focused.

To achieve this, a system was implemented where each interaction between the light beam and the car's windshield generated a trace on it to indicate the point of contact between the two. Technically, this was achieved by recording the coordinates (x, y, z) of each interaction point between the eye and the windshield and simultaneously changing the color of the windshield to highlight the trace left by the gaze passage.

The final rendering is exhaustive in understanding Eye Gazing at every moment of the simulation and pinpointing where the avatar is directing its attention. This is crucial for both system validation and data generation. With the ability to faithfully replicate a real car inside the simulation and having such projections, it becomes easy to understand the exact point where the avatar is focusing its attention.

For example, if we wanted to test recognition of distraction caused by using the car's infotainment system, it would be necessary to precisely position the avatar's attention at that specific point. This would be very complicated to manage in terms of parameters, as it would require accurately choosing the relative position values of the pupil based on the surrounding environment, without a clear understanding of the relationship between the two reference systems. However, thanks to the graphical representation provided by the light beams, it would be sufficient to use these lasers as reference points and then adjust the avatar's position accordingly. Only then would it be necessary to obtain and analyze the parameters.

In summary, the graphical representation of light beams not only simplifies the understanding of Eye Gazing during the simulation but also facilitates the generation of precise and realistic data, crucial for testing and validating automotive safety systems.

Chapter 6

Validation test bench

In this chapter, we focus on the detailed description of the different types of test benches for ADDW (Advanced Driver Distraction Warning) systems applicable to the generated system. Fundamentally, there are two approaches to fully leverage the project's capabilities: the first involves collecting frames generated by the system and directly sending them to the control unit for image processing to recognize distraction. The second approach, on the other hand, uses capturing images produced by the camera through a screen displaying the frames.

The selection between these two methodologies depends on the nature of the system being tested. In the first case, it is assumed to have direct access to the camera's CPU for image processing, allowing detachment of the camera block and direct insertion of images into the CPU. In the second case, however, we are faced with a situation where the camera block cannot be disconnected from the CPU; therefore, acquisition must occur obligatorily through the camera's lenses itself.

In the first technique, once the frames and the final video are generated, there are no further complications as the images are processed directly by the CPU, obtaining data as if they had been captured directly from the camera block's lenses.

On the contrary, in the second technique, there are several considerations to take into account. After the generation of frames and the video set, the images must be reacquired from the camera block before they can be processed by the system for distraction recognition. This process, if not properly controlled, can lead to distortions.

The main complication of the second method lies in the fact that the camera system is designed to capture and record elements in 3D, with specific depths and a lighting effect that interacts with objects in front of the lenses. However, by positioning a screen in front of the camera, the captured and processed images are already in 2D, with predetermined depth effects and a lighting source emanating from the images themselves.

To assess the feasibility and distortion level of the acquired images, it was necessary to conduct in-depth studies. Once again, we leveraged the capabilities of Unreal Engine for this purpose.

In the first type of image usage, it is crucial to precisely and carefully set all the different camera configurations and position it correctly inside the car during frame generation. This is necessary because the output produced by Unreal must perfectly match what the real camera would record when capturing images in the real world.

The adjustable parameters for the camera within the software are numerous, including:

- **Filmback**: The Filmback property contains a list of real-world preset Camera Bodies to choose from. These presets affect the Sensor Width and Height properties in order to emulate the selected camera body. The camera's base field of view and aspect ratio are also affected by these settings.
- **Sensor Width/Height**: The Sensor Width and Height properties emulate the dimensions of your film or Digital Sensor in millimeters. These values automatically change whenever a new Filmback preset is selected.
- **Sensor aspect ratio**: This property displays your current aspect ratio computed from the selected sensor's dimensions. This value cannot be changed manually as it is intended for read-only reference.
- **Lens settings**: The Lens Settings property contains a list of real-world preset Camera Lenses to choose from. These presets affect the Focal Length, FStop, and Diaphragm properties to emulate the selected lens.
- **Min/Max focal length**: The minimum and maximum Focal Length range for the camera. Setting these will affect the value range of the camera's Current Focal Range property, which is used to emulate Zoom Lenses. Setting both of these properties to the same value will emulate Prime Lenses, which are not designed to be zoomable.
- **Min/Max FStop**: The minimum and maximum Aperture range for this camera.
- **Diaphragm Blade Count**: This controls the Diaphragm (blade) count on the camera lens. The number of diaphragms on a lens correlates to the shape of the Bokeh effect. Lower numbers cause bokeh to appear more square, and higher numbers cause it to appear more rounded.
- **Focus method**: Focus Method controls the depth of field and focus settings. It contains the following options:

  - Do Not Override, which disables the camera-influencing depth of field. This still allows depth of field from other sources, such as from a post-process volume.
  - Manual, which enables depth of field and controls the focus, relative to the camera position, using the Manual Focus Distance property.
  - Tracking, which enables depth of field and controls the focus by tracking Actors using the Tracking Focus Settings properties.
  - Disable, which disables depth of field entirely and does not allow any depth-of-field influence from other sources.

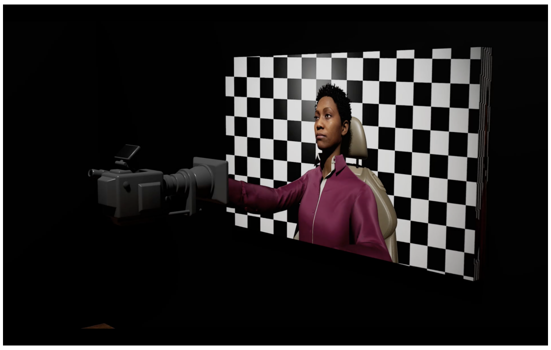
- **Manual focus distance**: - : Controls the focus distance relative to the camera's position in centimeters. This is a type-aware field, so you can input 5m and it will convert the value to 500cm. There is also an eye dropper to enable selecting an object directly from your viewport and snapping your camera's focus to it. To use it, click the button, and then left-click the object in your viewport. This property only appears if Focus Method is set to Manual.
- **Actor to track**: The Actor used as the focus point for the camera's depth of field. This is enabled only if Focus Method is set to Tracking.
- **Relative offset**: A positional offset to apply to the tracked Actor relative to its local space. This is enabled only if Focus Method is set to Tracking and is useful for fine-tuning the focus point on tracked Actors.
- **Draw debug tracking focus**: Enables a debug locator cube to show where the depth of field is focused if Focus Method is set to Tracking. This is useful if you are using it in conjunction with Relative Offset.
- **Draw debug focus plane**: Enables a camera-oriented transparent plane for previewing your focus point.
- **Debug focus planen color**: Specifies the color of the Debug Focus Plane.
- **Smooth focus change**: Enables focus distance changes to be automatically interpolated over time, instead of instantaneously.
- **Focus Smoothing Interp Speed**: The speed of focus-change interpolation if Smooth Focus Changes is enabled. Lower numbers are slower, and higher numbers are faster.
- **Focus Offset**: Offsets the focus point relative to the position of the tracked Actor, if Tracking focus is being used. The focus point will instead use the Manual Focus Distance, if Manual focus is being enabled. Positive numbers

increase the distance from the camera, and negative numbers decrease the distance.

- **Current Focal Length**: The focal length property of the camera, which is limited based on the focal length range defined by the Min/Max Focal Length properties in Lens Settings.
- **Current Aperture**: The aperture or FStop property of the camera, which is limited based on the FStop range defined by the Min/Max FStop properties in Lens Settings.
- **Current Focus Distance**: A read-only property that displays the final focus distance output based on either the Manual or Tracking focus, including any additional offsets applied to them.
- **Current Horizontal FOV**: A read-only property that displays the final horizontal field of view based on a combination of the camera's Focal Length and Sensor Dimensions.
- **Constrain Aspect Ratio**: Enables the drawing of black bars which preview the Sensor Dimension and its base aspect ratio.
- **Use Pawn Control Rotation**: If the camera is being used as a component on a Pawn, then enabling this will cause the Pawn's rotation to influence the camera component.
- **Post Process Blend Weight**: Blends the influence of the camera's Post Process layer between fully on and off. A weight of 1 will cause it to be fully enabled, and 0 will cause it to fall through to other layers.
- **Lock to Hmd**: Locks the camera's position and orientation to a connected head-mounted display, when a VR headset is connected for XR Development.
- **Use Field of View for LOD**: When enabled, the camera's field of view will influence what level of detail the object should display, instead of relying on camera distance only. This fixes issues caused by low-quality objects that are rendered from far away cameras, and that are zoomed in on the subject.

In the second mode of image usage, we have developed a virtual test bench within the virtual environment to evaluate the feasibility and sustainability of image acquisition from a screen or panel. This test bench consists of a scene divided into two distinct areas. In the first area, an avatar sitting at the wheel of a car performs facial movements, which are captured by a frame-generating camera. These frames are subsequently used in the second area of the scene, where another

(a) Initial acquisition setup.



(b) Second acquisition setup.

Figure 6.1: Virtual validation setup.

camera, identical to the first one, captures a panel displaying the previous frames, generating further frames.

This virtual setup faithfully replicates what happens in the real world, namely the acquisition of an image previously recorded by a camera. It's important to note the presence of a checkerboard background, which provides a geometric reference to assess any distortions introduced by this double acquisition.

From a technical standpoint, it was crucial to properly position the second camera relative to the panel on which the frames are displayed and to set the proportions of the panel itself in relation to the dimensions of the generated frames. After examining the sets of generated frames, we found that there were no geometric differences or distortions, but only variations in light and depth effects.

However, in the real world, there are measures to minimize these differences: for example, using curved or hemispherical screens to project images can reduce depth distortions, while placing the camera inside a black box, where the only light source is the panel itself, can drastically reduce light interference.

In conclusion, although directly inserting images into the CPU for image processing ensures greater fidelity to the real world, acquiring frames from a screen remains a valid and sustainable solution in cases where this is not possible.

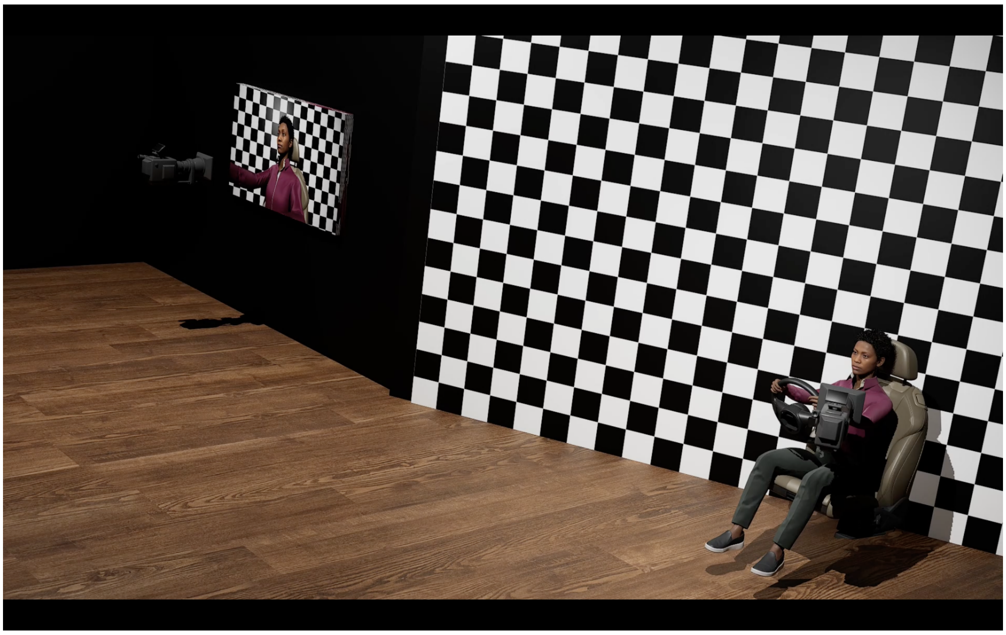


Figure 6.2: Entire scene of the virtual test bench

Chapter 7

Random scene generation

In order to ensure the solidity and structure of the scene database used for validating in-cabin monitoring cameras, it has been crucial to develop and implement a randomization system for them.

The main objective is to create a system, both automatic and manual, that allows for the modification of the actors present in the scene and their characteristics. This system operates by manipulating the fundamental parameters of the scene, including avatars, vehicles, sets of lights, and cameras used for frame generation, thus ensuring a wide variety of randomized combinations.

Figure 7.1: Final outcome of randomization.

7.1 Avatar Randomization

The avatar, representing the key figure of the real person driving, constitutes the central and most significant element of the scene; therefore, particular attention has been devoted to its randomization.

To achieve this, an approach has been adopted which involves collecting all avatars usable for testing and subsequently creating an array containing their respective directories. In this way, with each randomization, an avatar is randomly selected from the array and placed in the scene at its correct position. Subsequently, specific animation for body and facial movements is applied to this avatar.

From a technical standpoint, the randomization process operates as follows: at the beginning of each scene and with each subsequent randomization, the software generates a random number ranging from zero to the total number of selectable avatars. A precautionary check is then implemented to avoid the repeated selection of the same avatar in subsequent randomizations, thus ensuring a complete variation of the scene. The value of the selected array is then stored for use in the next iteration. Finally, the avatar, accompanied by the appropriate animation, is physically integrated into the scene.

7.2 Car Randomization

Similarly, attention has been paid to the randomization of the car. The adopted process involves varying the graphical rendering of this component by changing the color of specific elements of the interior and exterior bodywork for each iteration.

To implement this process, the elements to be modified were identified, and the RGB values of the main material associated with each of them were adjusted in each iteration. It was chosen to allow complete randomization of the three RGB values from zero to two hundred fifty-five, thus enabling a wide range of colors. This approach allowed us to explore different chromatic combinations without restrictions.

The choice not to limit the randomization of colors was motivated not only by the desire to maintain chromatic variety but also by the objective of verifying whether specific color combinations, combined with the avatar and lighting effects, could influence the camera's ability to detect facial movements.

7.3 Light Randomization

Subsequently, the focus shifted to the randomization of lighting effects present in the scene. Despite appearing as a purely aesthetic aspect, it is crucial to consider that the arrangement of lights can significantly impact the camera's ability to detect facial movements of the avatar or even accurately identify its presence.

A technical approach was adopted, involving the definition of an area surrounding the car and the avatar where light sources would be subsequently placed, allowing the software to select parameters to apply to these elements. These generated parameters include a triplet of values (x, y, z) for the position of the light source and another triplet of values for its orientation.

Once the parameters were determined, the software proceeded to insert the light sources into the scene at the indicated positions. This process was iterated multiple times for each randomization, allowing the insertion of multiple light sources into the scene. Furthermore, the ability to select and modify the number of light sources to be integrated into the scene was ensured.

7.4 Camera Randomization

Another aspect for which randomization has been implemented concerns the position of the camera used for capturing and generating frames.

This type of randomization was introduced with the aim of simulating and evaluating the performance of a specific camera positioned in various areas of the cabin. It is particularly advantageous when the final position of the camera has not yet been determined but aims to identify the optimal placement to monitor all driver distraction actions.

To achieve this goal, several positions within the cabin where the camera could be placed were defined and discretized, which were then inserted into a dedicated array. Subsequently, the same approach used for avatar randomization was adopted, ensuring that a different position for the camera was selected in each iteration.

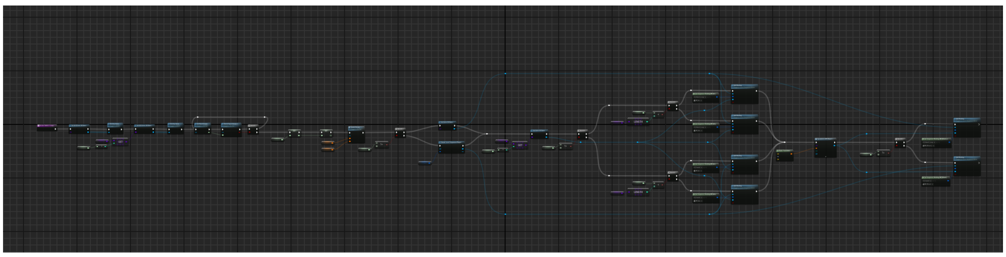


Figure 7.2: Blueprint of randomization

7.5 Types of Randomizations

Randomization has been implemented through two distinct modes: automatic and manual. Although sharing the same basic principle, they differ in the input methods that trigger the randomization process.

In the first case, randomization occurs automatically at the start of the simulation and subsequently at regular intervals, specifically at the conclusion of each animation sequence. This approach allows for generating a series of videos with identical animations but with the scene constantly varied.

In the second case, randomization is manually controlled through human-machine interaction. A user interface has been developed that allows for randomizing the entire scene, only the avatar, only the car, or only the camera using specific buttons. After selection, the simulation can be initiated by pressing the "Start" button.

Figure 7.3: User Interface for randomization

This interface not only presents the simulation elements but also offers the flexibility of selection, thus adapting the simulation to specific testing objectives or user preferences.

Furthermore, the possibility of activating randomization during the simulation has been introduced by associating specific keyboard keys with randomization events. For example, the "Q" key activates complete scene randomization, while the "W", "E", "R", and "T" keys respectively activate camera, lighting effects, avatar, and car randomization.

Finally, the user interface includes real-time visualization of key avatar parameters during the simulation, such as head yaw, head pitch, head roll, eye blinking and mouth opening. These parameters, normally displayed in white, change color to red and are accompanied by an alert signal when they exceed predefined thresholds. This immediate feedback system is crucial for quickly identifying anomalous behaviors or critical issues, allowing users to intervene promptly or record events for in-depth analysis.

7.6 Output Frames

Regarding the final output, the primary goal is to obtain a sequence of frames with the highest possible resolution. This is achieved by configuring the desired quality level directly in the rendering settings.

Once the rendering process is complete, a final output is obtained in the form of EXR files, one for each frame processed by the software. These files can then be converted into the most suitable video format for the final use, such as MP4. This procedure facilitates sharing and analysis of the obtained results.

Chapter 8

Conclusions and forward implementation

8.1 Conclusion

In the concluding chapter of this discourse, there is reflection on the ambitious objectives outlined at the beginning of the journey and an analysis of how these goals have not only been achieved but also surpassed. Furthermore, the potential of the system developed in the field of testing and validation in the automotive sector, with particular attention to ADAS systems, is examined, thus opening up new perspectives.

This project represents a significant advancement in the realm of innovation. The final result consists of obtaining an accurate replica of an in-cabin simulation in the automotive context, characterized by a surprising level of realism that faithfully recreates human expressions and behaviors.

Through the use of Unreal Engine 5.3 and its Metahuman plugin, an unprecedented level of detail and flexibility has been achieved. This has made it possible to create and obtain extremely realistic avatars capable of accurately replicating human facial expressions and reflecting complete human diversity. This approach has allowed testing of the systems under examination through a wide range of different scenarios and situations.

The accuracy of the generated and extracted metadata has been carefully verified and evaluated for both real-time data generation and pre-recorded video data. The results demonstrate an extremely high level of precision, offering a detailed and comprehensive representation over time and thus providing a solid foundation of data for the creation of future ground truth. It is crucial to emphasize that such data is generated continuously and uninterrupted, ensuring access to all facial parameters moment by moment.

Another strength of this project lies in the ease of use of the applications, the ability to customize each parameter individually, and the careful implementation of reasoning, characterized by clarity and coherence. These combined elements transform the tool into a modular platform suitable for testing and validation not only of ADAS systems in the automotive sector but also for a wide range of other fields of interest.

The results achieved through the project have received positive feedback not only in academic circles but also in the industry, attracting the attention of several renowned automotive companies. The approval and enthusiasm shown by such clients underscore the success of the project, confirming its relevance and impact in the field of simulation for ADAS system validation.

In conclusion, the project has achieved two main objectives: it has pushed the limits of simulation technology in the ADAS sector and has demonstrated the practical usefulness of such innovations in the automotive industry. The synergy between theory and practice, research and industrial application, represents the cornerstone of this success, marking a significant progress in our understanding and optimization of ADAS systems through advanced testing.

Looking ahead to the future, the work done opens up various opportunities for further study and development. The infrastructure developed, the methodologies adopted, and the results obtained constitute a solid foundation on which to build, explore new ideas, and tackle emerging challenges in the automotive safety and simulation sector.

8.2 Possible Future Improvements

Considering that this project represents a fundamental starting point towards the creation of a validation system in virtual field, the prospects for improvement and potential future implementations are diverse and extensive.

The main and most significant improvement concerns the translation of the code used for all processes and phases of the project. Currently, such implementations are carried out using Unreal Engine blueprints, which, although powerful, have technical limitations. Therefore, the main improvement would consist of translating all processes into native C++ code. C++ offers more detailed and granular control, especially regarding memory management and system resources. This would allow for a more performant and less computationally burdensome project on the hardware.

This improvement would bring numerous benefits, including a significant reduction in the loading times of the entire system and the ability to generate increasingly complex and detailed simulations. Furthermore, C++ code is highly customizable, allowing for further optimization of the system based on the specific needs of various projects.

In addition to code conversion, there are several areas where the project could be further expanded and enriched. For instance, integrating artificial intelligence (AI) and machine learning (ML) algorithms could significantly enhance the analytical capabilities of the system. This would enable extracting and interpreting data collected during simulations in more sophisticated and comprehensive ways.

With AI and ML, it would be possible to train the system to recognize and understand complex patterns in data, identify anomalies or critical situations, and suggest improvements or optimizations for the ADAS systems under examination. For example, ML models could be employed to predict driver behavior in complex traffic situations, thus allowing for testing and optimizing ADAS systems under realistic conditions.

Moreover, AI could be used to create autonomous virtual agents that interact with the ADAS system similarly to real drivers, enabling the evaluation of system performance in a variety of driving scenarios and conditions.

In summary, the integration of AI and ML algorithms would open up new possibilities to enhance the effectiveness and accuracy of the simulation system, allowing for richer and more meaningful results for ADAS system validation.

An area that could benefit from further improvement is the expansion and enhancement of the Metahumans and vehicle database. Currently, the project has an acceptable variety of avatars and cars, but by further expanding this library, it would be possible to cover a wide range of human and driving scenarios. This improvement would not only increase the versatility of the system but also its ability to provide valid test results in a variety of contexts, making them more representative of different real-world situations.

Another aspect that could be improved concerns the simulation of variable environmental conditions. Currently, the system is able to replicate lighting scenarios and some urban or highway contexts. However, by expanding the range of weather conditions, times of day, and traffic situations, the test bench could offer an even more comprehensive and detailed testing environment. This improvement could help enhance the robustness of the tested ADAS systems, allowing for a more accurate assessment of performance in a wide range of realistic scenarios.

Another perspective for development could involve adopting a modular approach in the design of the system. This would allow for greater customization and flexibility, enabling users to select or modify specific modules based on their testing needs without the need to reconfigure the entire system. Such a modular approach could simplify the upgrading and integration of new features into the existing system, allowing for greater adaptability to the specific needs of users and applications.

In conclusion, although this project has already achieved significant milestones, the insights for future implementations presented here are just a part of the vast landscape of potential enhancements. The path toward innovation is continuously evolving, and with the potential improvements and expansions proposed, the virtual test bench can continue to evolve, tackling new challenges and contributing even more substantially to the field of ADAS system validation.

Bibliography

[1] GROW.I.2. «Report on Advanced Driver Distraction Warning systems». In: (June 2021), pp. 1–2 (cit. on p. 12).

[2] N.H.T.S.A. «Distraction Detection and Mitigation Through Driver Feedback.» In: *Appendix A: Theoretical and Empirical Considerations for Distraction Detection And Mitigation* (May 2013) (cit. on pp. 13–15, 25–27).

[3] Rohith Chinthalachervu. «Driver Drowsiness Detection and Monitoring System using Machine Learning». In: 2022 (cit. on p. 15).

[4] M.W. Johns. «A new method for measuring daytime sleepiness: the Epworth sleepiness scale.» In: (1991), pp. 540–545 (cit. on p. 16).

[5] Hoddes, E.Zarcone, V.Smythe, H.Phillips, R.Dement, and W.C. «Quantification of sleepiness: a new approach. Psychophysiology». In: (1973), pp. 431–436 (cit. on p. 16).

[6] T.H. Monk. «A visual analogue scale technique to measure global vigor and affect.» In: (1989), pp. 89–99 (cit. on p. 16).

[7] M. Doudou, A. Bouabdallah, and V. Cherfaoui. «Driver drowsiness measurement technologies: Current research, market solutions, and challenges». In: *International Journal of Intelligent Transportation Systems Research* 18 (Sept. 2019) (cit. on pp. 17–20).

[8] J. Fukuda, E. Akutsu, and K. Aoki. «An estimation of driver's drowsiness level using interval of steering adjustment for lane keeping.» In: (1995), pp. 197–199 (cit. on p. 17).

[9] Y. Dong, Z. Hu, K. Uchimura, and N. Murayama. «Driver inattention monitoring system for intelligent vehicles: a review.» In: *IEEE,Trans. Intell. Transp. Syst.* (2011), pp. 596–614 (cit. on p. 18).

[10] M. Ingre, T. Akerstedt, B. Peters, A. Anund, and G. Kecklund. «Subjective sleepiness, simulated driving performance and blink duration: examining individual differences. J. Sleep Res.» In: *J. Sleep Res.* 15 (2006), pp. 47–53 (cit. on p. 18).

[11] Y. Peng, L.N. Boyle, and S.L. Hallmark. «Driver's lane keeping ability with eyes off road: insights from a naturalistic study.» In: *Accid. Anal. Prev.* 50 (2013), pp. 628–634 (cit. on p. 18).

[12] P.M. Forsman, B.J. Vila, R.A. Short, C.G. Mott, and H.P.V. Dongen. «Efficient driver drowsiness detection at moderate levels of drowsiness.» In: *Accid. Anal. Prev.* 50 (2013), pp. 341–350 (cit. on p. 18).

[13] S. Furugori, N. Yoshizawa, C. Iname, and Y. Miura. «Estimation of driver fatigue by pressure distribution on seat in long term driving.» In: *Rev. Automot. Eng.* 26 (2005), pp. 053–058 (cit. on p. 18).

[14] D.M. Morris, J.J. Pilcher, and F.S.S. III. «Lane heading difference an innovative model for drowsy driving detection using retrospective analysis around curves.» In: *Accid. Anal. Prev.* 80 (2015), pp. 117–124 (cit. on p. 18).

[15] J.T. Arnedt, G.J.S. Wilde, P.W. Munt, and A.W. MacLean. «How do prolonged wakefulness and alcohol compare in the decrements they produce on a simulated driving task?» In: *Accid. Anal. Prev.* 33 (2001), pp. 337–344 (cit. on p. 18).

[16] S.H. Fairclough and R. Graham. «Impairment of driving performance caused by sleep deprivation or alcohol: a comparative study.» In: *Hum. Factors* 41 (1999), pp. 118–128 (cit. on p. 18).

[17] Z. Chen, C. Wu, M. Zhong, N. Lyu, and Z. Huang. «Identification of common features of vehicle motion under drowsy distracted driving: a case study in Wuhan China.» In: *Accid. Anal. Prev.* 81 (2015), pp. 251–259 (cit. on p. 18).

[18] Q. Ji, Z. Zhu, and P. Lan. «Real-time nonintrusive monitoring and prediction of driver fatigue.» In: *IEEE Trans. Veh. Technol.* 53 (2004), pp. 1052–1068 (cit. on p. 19).

[19] H.S. Shin, S.J. Jung, J.J. Kim, and W.Y Chung. «Real time car driver's condition monitoring system.» In: *IEEE Sensors* (2010), pp. 951–954 (cit. on p. 19).

[20] L. Wang, X. Wu, B. Ba, and W. Dong. «A vision-based method to detect perclos features.» In: *Comput. Eng. Sci.* 6 (2006), pp. 1–17 (cit. on p. 19).

[21] Q. Xia, Y. Song, and X. Zhu. «The research development on driving fatigue based on perclos.» In: *Tech. Autom. Appl.* 6 (2008), pp. 1–13 (cit. on p. 19).

[22] C. Papadelis, Z. Chen, C. Kourtidou-Papadeli, P.D. Bamidis, I. Chouvarda, E. Bekiaris, and N. Maglaveras. «Monitoring sleepiness with on-board electrophysiological recordings for preventing sleep-deprived traffic accidents.» In: *Clin. Neurophysiol.* 118 (2007), pp. 1906–1922 (cit. on p. 19).

[23] B.C. Yin, X. Fan, and Y.F. Sun. «Multiscale dynamic features based driver fatigue detection.» In: *Int. J. Pattern Recogn. Artif. Intell.* 23 (2009), pp. 575–589 (cit. on p. 19).

[24] T. Brandt, R. Stemmer, and A. Rakotonirainy. «Affordable visual driver monitoring system for fatigue and monotony.» In: *IEEE International Conference on Systems, Man and Cybernetics* 7 (2004), pp. 6451–6456 (cit. on p. 19).

[25] Laura Semini Giovanni A. Cignoni Carlo Montangero. «Il controll del software: verifica e validazione». In: (2009) (cit. on p. 21).

[26] Olaf Gietelink. «Design and Validation of Advanced Driver Assistance Systems.» In: *Netherlands TRAIL Research School* (2007) (cit. on p. 21).

[27] *Can simulation tools replace physical tests completely?* 2015. URL: `https://www.designworldonline.com/can-simulation-tools-replace-physical-tests-completely/` (cit. on p. 22).

[28] *ISO26262 Standard*. 2018 (cit. on p. 24).

[29] E. F. Yacometti I. «Driving Style Estimation for Driver State Monitoring: Truthful Data Acquisition via Driving Simulator». In: (2023) (cit. on p. 24).

[30] *EU Regulation No. 2019/2144*. 2019 (cit. on pp. 24, 25).

[31] *EU Regulation No. 2021/1341*. 2021 (cit. on p. 24).

[32] *Delegated regulation - C(2023)4523*. July 2023 (cit. on p. 26).