POLITECNICO DI TORINO

Master Degree Computer Engineering for Automation and Intelligent Cyber-Physical Systems

Master Degree Thesis

Humanoid Robots for Visual Distraction In-Vehicle Test Automation

Supervisors
Riccardo Coppola
Alessandro Tessuti

Candidate
Luigi Maggipinto

Academic Year 2024-2025

Abstract

Road traffic safety has become a critical area of research due to the increasing number of accidents caused by driver distraction. In order to address this issue, the European Union has introduced Regulation (EU) 2019/2144, which requires the integration of Advanced Driver Distraction Warning Systems (ADDWS) into all newly manufactured vehicles by 2026. ADDWS operates within Driver Monitoring Systems (DMS) to detect driver distraction using visual and sensor-based monitoring techniques. Ensuring the effectiveness of these systems requires rigorous validation under controlled conditions.

This research proposes an automated framework for validating the visual distraction of ADDWS by utilizing a humanoid robotic platform, Ameca Desktop, as a synthetic driver. In contradistinction to traditional human-subject testing, this approach eliminates variability, thereby providing a reproducible ground truth for the detection of distraction. The system consists of a multi-node architecture where data is collected from Ameca's motor positions, an Intel RealSense camera and a Time-of-Flight (ToF) sensor. The Raspberry Pi units coordinate data acquisition through a Flask-based API framework, ensuring synchronised recording of motor angles, depth data, and video frames.

A fundamental component of the methodology involves the analysis of Ameca's head and eye movements to classify distraction states. The collected data is then processed to determine whether the humanoid exceeds predefined motion thresholds, thus categorising its state as either distracted or not distracted.

To validate the robustness of the system, a System Under-Test (SUT) approach is introduced, employing MediaPipe to estimate head pose (yaw, pitch, roll) and eye gaze direction under identical test conditions. This enables a comparative evaluation between the robotic ground truth and computer vision-based distraction detection.

The proposed framework establishes a scalable, automated testing solution for ADDWS evaluation by leveraging Ameca's precise motor control and high-fidelity facial expressions. Furthermore, the system is designed to ensure that distraction detection methodologies comply not only with Regulation (EU) 2019/2144 but also with the performance assessment criteria established by Euro NCAP. By aligning with these regulatory and safety standards, this research contributes to the advancement of reliable DMS validation methodologies, ensuring the robustness of ADDWS before their deployment in consumer vehicles.

Contents

Chapter 1

Introduction

Road traffic crashes continue to be one of the leading causes of death worldwide, posing a serious public health and economic challenge. According to the *World Health Organization (WHO)*, approximately 1.19 million people die each year due to road traffic crashes, with an additional 20 to 50 million individuals sustaining injuries, many of which result in long-term disabilities [1]. Among the primary causes of road accidents, driver distraction remains a significant contributor. Reports from the *European Commission* indicate that between 5% and 25% of all road accidents in Europe are linked to driver distraction [2]. Similarly, data from the *National Highway Traffic Safety Administration (NHTSA)* in the United States attributes 3,308 fatalities and over 289,000 injuries in 2022 to distraction-related crashes [3]. The dangers of driver distraction are further amplified when combined with fatigue or drowsiness, which alone account for approximately 20% of road accidents on European highways [4]. Studies show that driving after 24 hours of sleep deprivation impairs reaction times to a degree equivalent to a blood alcohol concentration (BAC) of 0.10%, which exceeds the legal limit in most countries (0.08%) [5].

To mitigate these risks, the *European Union* enacted Regulation (EU) 2019/2144, mandating that, starting from July 2026, all newly manufactured vehicles must be equipped with an Advanced Driver Distraction Warning System (ADDWS) [6]. Subsequently, on 13 July 2023, a delegated regulation was introduced to supplement Regulation (EU) 2019/2144 by establishing detailed rules concerning the specific test procedures and technical requirements for the type-approval of certain motor vehicles with regard to their ADDWS, further refining the regulatory framework for ensuring compliance and performance standards [7]. ADDWS is a critical component of Driver Monitoring Systems (DMS), which in turn belong to the broader category of Advanced Driver Assistance Systems (ADAS). ADAS encompasses various safety technologies aimed at enhancing road safety through automation, including driver state monitoring, lane-keeping assistance, adaptive cruise control, and emergency braking. ADDWS are evaluated by the European New Car Assessment Programme (Euro NCAP), which assesses their effectiveness in reducing accident risks and improving vehicle safety ratings [8].

These systems must:

- Detect when a driver is distracted (e.g., looking away, engaging in a secondary task).
- Issue real-time warnings to refocus driver attention.
- Operate reliably in different environmental conditions, including nighttime driving and sunglasses use.

The *European Transport Safety Council (ETSC)*, according to new research carried out for Trygg Trafikk, has emphasized that smartphone use and in-car infotainment systems are among the most problematic sources of driver distraction, urging manufacturers to integrate advanced monitoring technologies into new vehicles [9].

As a key component of *Driver Monitoring Systems (DMS)*, ADDWS is expected to play a crucial role in achieving the *EU's Vision Zero initiative*, which aims to eliminate road fatalities by 2050. However, ensuring the effectiveness of these systems requires rigorous validation under controlled conditions before implementation.

This research proposes an automated, in-vehicle testing framework designed to validate the *Visual Distraction* performance, from ADDWS, using a humanoid robot as a synthetic driver. Unlike traditional DMS validation, which relies on human subjects, this method eliminates human variability while providing a scalable and highly accurate testing approach, offering several advantages:

1. Cost reduction – Traditional DMS validation requires the hiring of human actors, a process that is both expensive and logistically complex.
2. Protection of personal data – The system does not involve real driver data, ensuring compliance with strict data privacy regulations.
3. Precision & reproducibility – Unlike human subjects, a humanoid robot provides precise, repeatable movements, removing variability in validation.

Conventional DMS validation relies on human subjects performing distraction scenarios; this system replaces human drivers with a humanoid robot capable of controlled head and eye movements. The proposed testing framework enables:

- Accurate tracking of eye gaze direction, which can be used to determine whether the "driver" is looking at the road or distracted.
- Precise measurement of head movements, thanks to the *Ground-Truth* data, ensuring that small deviations can be reliably detected.
- Automated execution of distraction scenarios, facilitating large-scale validation efficiently.

The integration of a robotic platform within the testing pipeline enables the system to achieve fully controlled and repeatable validation, thereby eliminating the inconsistencies associated with human-based testing.

The synthetic human test subject employed in this study is Ameca Desktop, a humanoid robot designed with high-precision head movement capabilities. Ameca serves as an ideal platform for evaluating the Visual Distraction of the ADDWS, due to its ability to perform controlled and programmable distractions (e.g., looking away from the road, tilting the head) and to provide highly accurate positional data, allowing precise measurement of head pose and gaze direction.

The effectiveness of an Advanced Driver Distraction Warning System (ADDWS) depends on its ability to accurately determine whether a driver is attentive or distracted. To validate such systems, it is crucial to measure head movement, eye gaze direction, and response time under controlled conditions.

A major challenge in distraction validation is the precise measurement of small head and eye movements in human subjects. Traditional validation methods (human based) face several limitations, including:

- Inconsistent head positioning, making it difficult to establish a uniform baseline for distraction analysis.
- Limited repeatability, as human participants cannot reliably reproduce identical gaze deviations across multiple trials.
- Low measurement accuracy in detecting small-angle variations, leading to potential discrepancies in distraction classification.

To address these limitations, the use of a humanoid robot provides a controlled and standardized testing approach, offering:

- Precisely controlled, programmable movements that eliminate variability introduced by human subjects.
- Highly repeatable gaze direction changes, enabling fine-grained distraction analysis under identical conditions.
- Accurate timestamped sensor data, ensuring precise synchronization between head position, gaze shifts, and distraction detection algorithms.

A critical component of this testing framework is ensuring that the measured distraction state corresponds with the robot's actual movements, thereby ensuring the validity of the results. By capturing real-time motor angles and head tracking data, the system provides a highly accurate ground truth for evaluating ADDWS performance, thus enabling an objective, scalable, and automated validation method. The combination of sensor data, video analysis, and controlled distraction scenarios allows for such a method.

Another aspect of this validation framework is the *System Under Test (SUT)*: a computer vision-based system I developed that uses *MediaPipe* to track facial landmarks, gaze direction, and eyeblink events. Unlike the ADDWS, which encompasses a broader range of distraction detection modalities, the SUT aims to evaluate the visual distraction component of the system. Its role is to determine whether the humanoid driver (Ameca) exhibits distracted or attentive behaviour based on predefined gaze and blink thresholds.

The validation process consists of a direct comparison between the detection results of the SUT and the ground truth provided by Ameca's motor positions. If the SUT's classification matches the expected distraction state, the test is considered successful; otherwise, it indicates a discrepancy in the detection automation pipeline.

To ensure a structured and automated evaluation, the entire test process is integrated into the *Test Automation Framework (TAF)* developed by *Concept Quality Reply*. The TAF provides a graphical interface where predefined test scenarios are executed, facilitating systematic validation under identical experimental conditions. The framework allows the automated execution of Java-based test functions that control both simultaneously:

- The robot driver (Ameca), which triggers specific distraction behaviours based on scripted scenarios.
- The SUT, which processes real-time facial tracking data to detect eye blinks and gaze deviations.

In addition, the TAF collects and logs test results, enabling quantitative performance evaluation.

By integrating precise robotic motion control, real-time facial tracking, and structured automation, this methodology establishes a scalable and reproducible validation framework for assessing the visual distraction detection performance of ADDWS.

Thesis Structure

This thesis work was structured as follows:

- Background: Reviews the key concepts relevant to this research, including an overview of *Advanced Driver Assistance Systems (ADAS)* and its *Driver Monitoring Systems (DMS) component*, along with *Euro NCAP* safety requirements. It also explores *humanoid robotics in validation testing*, *sensor-based gaze and head tracking*, and *Hardware-in-the-Loop (HIL) simulation* as a methodology for testing distraction detection systems.
- Methodology: Describes the design and implementation of the humanoid-based validation framework, including:
  - The tools and hardware used, including *Ameca Desktop*, cameras, and sensor systems.
  - The infrastructure architecture, detailing the *frontend-backend* communication structure and the data acquisition process.
  - The *System Under Test (SUT)*, a MediaPipe-based system that detects facial landmarks, eye gaze direction, and eyeblinks, validating whether the humanoid robot exhibits distracted behavior.

- Experimental Validation and Results: Presents the results of the validation experiments, analyzing the effectiveness of the humanoid-based testing method. It includes:
  - The experimental setup, including testing configurations and environment.
  - Validation methodology using the SUT, comparing its distraction classification with Ameca's ground truth data.
  - The *Test Automation Framework (TAF)*, which executes and manages test scenarios, ensuring reproducibility and structured validation. It provides an interface for running tests, logging results, and analyzing the SUT's accuracy against the robotic ground truth.
  - The results and performance analysis of the *Visual Distraction* validation, assessing system accuracy and reliability.
- Conclusion and Future Work: Summarizes the key findings of the research, discusses the limitations of the proposed method, and suggests future moves, including potential improvements using expanded regulatory testing scenarios.

Chapter 2

Background

In order to develop a reliable validation framework, it is essential to understand the technological and regulatory landscape surrounding *Advanced Driver Distraction Warning Systems (ADDWS)*. This chapter provides an overview of *Driver Monitoring Systems (DMS)* and their role within *Advanced Driver Assistance Systems (ADAS)*. It also discusses the regulatory framework that governs ADDWS implementation, focusing on *Regulation (EU) 2019/2144* and *Euro NCAP* requirements.

The chapter goes on to explore the use of *humanoid robots* in validation testing, the importance of *sensor-based head and gaze tracking*, and the role of automated validation methodologies for the Visual Distraction of ADDWS. These topics establish the foundation for the proposed methodology, ensuring that ADDWS technologies meet the necessary precision and compliance standards.

2.1 Introduction to Driver Monitoring Systems (DMS) and Advanced Driver Assistance Systems (ADAS)

Advanced Driver Assistance Systems (ADAS) have been developed to guarantee vehicle safety and assist drivers in preventing accidents. These systems incorporate a range of technologies, including automated braking, lane-keeping assistance, and adaptive cruise control, to improve driving efficiency and mitigate risks on the road.

Within ADAS, *Driver Monitoring Systems (DMS)* play a crucial role in assessing the driver's state and ensuring they remain attentive to the driving task. Unlike other ADAS functionalities that focus on the vehicle's surroundings, DMS is specifically designed to monitor the driver's behavior and physiological condition in real time.

Driver distraction is one of the leading causes of road accidents, making DMS a critical component in modern vehicles. These systems analyze the driver's actions to detect signs of inattention, fatigue, or impairment and provide timely warnings or interventions. The implementation of DMS has been reinforced by regulatory mandates, such as *Regulation (EU) 2019/2144* [6], which requires the integration of *Advanced Driver Distraction Warning Systems (ADDWS)* into all newly manufactured vehicles from July 2026 onward.

2.2 Driver Monitoring Technologies

2.2.1 Categories of DMS Technologies

Driver Monitoring Systems (DMS) employ various technologies to assess driver attention and detect distraction. These systems can be classified into three main categories based on their approach to monitoring driver behavior: *bioelectric signal-based DMS*, *lane departure-based DMS*, and *face-monitoring-based DMS*. Each of these methods has advantages and limitations, influencing their applicability in commercial vehicles.

Bioelectric Signal-Based DMS

Bioelectric signal-based DMS rely on physiological signals to determine driver fatigue and cognitive load. These systems use electrodes to measure *electroencephalogram (EEG)*, *electrocardiogram (ECG)*, and *electromyography (EMG)* signals, which provide insights into neural activity, heart rate variability, and muscle tension, respectively.

Research has demonstrated that EEG-based monitoring can effectively detect drowsiness by analyzing changes in brainwave patterns [10]. Similarly, ECG and EMG sensors can assess stress levels and muscle fatigue, helping to infer driver alertness. However, bioelectric signal-based DMS face significant challenges in real-world applications due to the necessity of physical contact with the driver, making them intrusive and impractical for large-scale deployment in consumer vehicles.

Lane Departure-Based DMS

Lane departure-based DMS assess driver attention indirectly by analyzing vehicle behavior rather than biometric data. These systems use *lane detection cameras*, *steering angle sensors*, and *vehicle dynamics data* to determine if a driver is unintentionally drifting out of the lane, which can indicate distraction or drowsiness.

While lane departure monitoring has proven effective in detecting certain types of inattention, it has limitations. External factors such as road curvature, weather conditions, and driver steering habits can affect detection accuracy. Moreover, these systems cannot provide direct insights into cognitive distraction, making them less suitable for comprehensive driver monitoring.

Face-Monitoring-Based DMS

Among the various DMS approaches, face-monitoring-based systems have gained increasing attention due to their ability to provide real-time, non-intrusive assessment of driver attention. These systems utilize *computer vision algorithms* to analyze the driver's facial expressions, head orientation, and eye movements, allowing for precise detection of distraction and drowsiness.

Recent research [11] highlights the advantages of gaze tracking in driver monitoring, demonstrating that *eye movement analysis* is a highly effective method for evaluating driver attentiveness. Face-monitoring-based DMS employ *visible spectrum cameras* and

infrared (IR) imaging to ensure accurate performance in various lighting conditions. IR-based image acquisition is particularly advantageous as it enables eye tracking even in low-light environments, addressing one of the primary challenges of vision-based monitoring.

Due to its high accuracy and non-intrusive nature, face-monitoring-based DMS is the most widely adopted approach in modern commercial vehicles. The following section delves deeper into the principles of *eye gaze tracking* and the role of *infrared-based image acquisition* in distraction detection.

2.2.2 Face-Monitoring-Based DMS and Eye Gaze Tracking

In the realm of Driver Monitoring Systems (DMS), face-monitoring-based techniques have garnered mounting attention, owing to their capacity for real-time, non-intrusive assessment of driver attention. In contrast to bioelectric signal-based DMS, which necessitate direct physical contact with the driver, and lane departure-based DMS, which rely on external vehicle behaviour, face-monitoring-based systems offer a direct and precise method for evaluating driver state.

Face-monitoring-based DMS utilise computer vision algorithms to track facial landmarks, head position, and eye movements, thereby enabling the system to ascertain whether the driver is alert or distracted. These systems employ both *visible spectrum* and *near-infrared (NIR)* imaging to maintain accuracy under varying lighting conditions.

Eye gaze tracking plays a critical role in assessing driver attention and detecting distraction. By analyzing the direction of gaze, the system can determine whether the driver is looking at the road, engaging with in-vehicle systems, or experiencing cognitive overload.

There are two primary methods for eye gaze tracking:

- **Feature-based gaze tracking**: Identifies key facial features such as eye corners, pupil position, and eyelid movements to estimate gaze direction.
- **Appearance-based gaze tracking**: Uses deep learning models to analyze pixel intensity variations across the eye region, enabling more robust tracking under different lighting conditions.

Recent research [11] has demonstrated the effectiveness of gaze tracking for driver monitoring, highlighting that eye movement analysis significantly improves distraction detection accuracy. Studies indicate that prolonged off-road glances are strongly correlated with an increased risk of accidents, making gaze tracking an essential component of DMS [12].

2.2.3 Infrared-Based Image Acquisition for Eye Gaze Monitoring

One of the primary challenges in vision-based driver monitoring is ensuring reliable eye tracking in varying illumination conditions. Traditional visible-spectrum cameras are highly sensitive to environmental lighting changes, which can lead to reduced accuracy in low-light scenarios or when the driver wears sunglasses.

To address these challenges, modern DMS utilize *near-infrared (NIR) imaging*, which provides several key advantages

- Illumination independence: NIR cameras function effectively in both bright and low-light conditions.
- Higher contrast for eye tracking: The pupil reflects infrared light differently than surrounding tissues, making it easier to detect gaze direction.
- Compatibility with driver accessories: Unlike visible light systems, NIR-based tracking remains effective even when the driver wears sunglasses with certain IR-transmitting properties.
- NIR is barely visible to the driver, this will minimize any interference with the driver's driving

According to research presented in [13], the integration of NIR image acquisition significantly improves the robustness of gaze tracking, ensuring accurate detection of distraction across a wide range of driving conditions:

At *near-infrared (NIR)* wavelengths, the interaction between light and the pupil plays a crucial role in enhancing gaze-tracking accuracy. A *bright pupil effect* occurs when the eyes are illuminated with an NIR illuminator positioned along the camera's optical axis. At specific wavelengths, the pupils reflect almost all the incoming infrared light directly back to the camera sensor, producing a bright appearance similar to the red-eye effect in conventional photography.

Conversely, when NIR illumination is projected off the camera's optical axis, the reflected light does not enter the lens, resulting in the *dark pupil effect*. This contrast between bright and dark pupils provides a reliable means for pupil segmentation, allowing for more precise gaze estimation. Figure 2.1 illustrates the principle of bright and dark pupil effects, while Figure 2.2 presents a real-world example of this phenomenon [14].

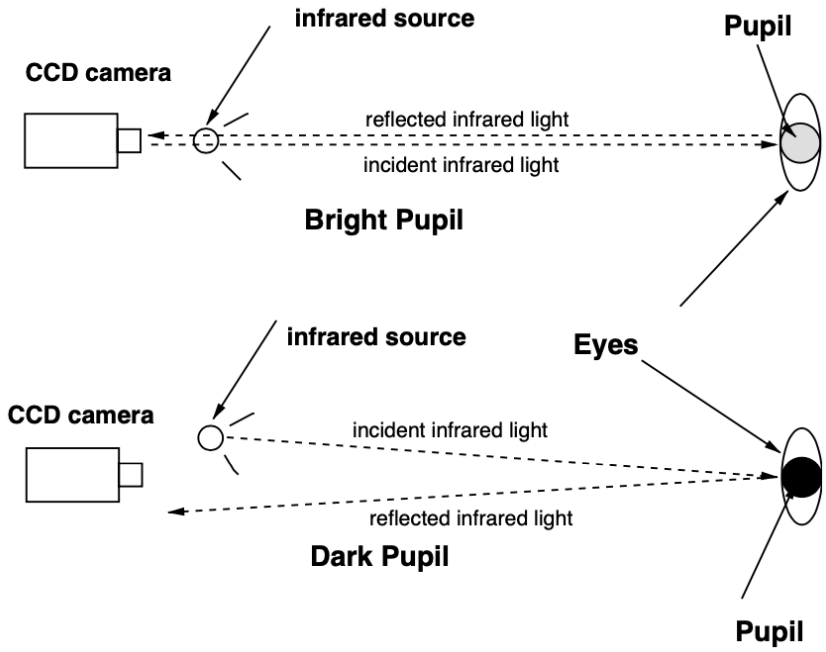


Figure 2.1: Bright and Dark Pupil effects

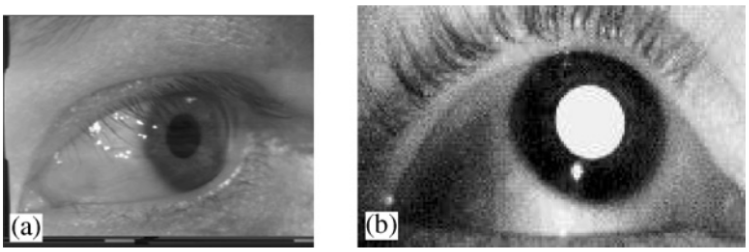


Figure 2.2: (a) dark pupil effect generated by IR LED's off the camera optical axis; (b) bright pupil effect generated by IR LED's along the camera optical axis

2.3 ADDWS and Its Integration into DMS

Advanced Driver Distraction Warning Systems (ADDWS) represent a significant enhancement to contemporary *Driver Monitoring Systems (DMS)*, specifically designed to detect and mitigate *visual distraction* in real-time. These systems employ *computer vision*, *artificial intelligence*, and *sensor-based tracking* to analyze head movements, eye gaze direction, and driver attention patterns.

Unlike conventional DMS, which primarily focus on detecting *fatigue* and general *vigilance states*, ADDWS is designed to identify both *momentary* and *prolonged visual distractions*, issuing warnings or triggering intervention mechanisms when necessary. The integration of ADDWS into DMS ensures a proactive approach to road safety by:

- Detecting instances of driver inattention, such as looking away from the road or engaging with secondary tasks.
- Issuing *visual, auditory, or haptic warnings* to prompt the driver to refocus.
- Activating *advanced assistance features*, such as *adaptive cruise control* and *lane-keeping interventions*, when required.

The implementation of ADDWS has been mandated by *Regulation (EU) 2019/2144*, reinforcing its role as a fundamental component of vehicle safety technologies. This regulation establishes clear criteria for distraction detection, requiring that ADDWS operate without *biometric identification* and ensure reliable performance under various driving conditions.

2.3.1 Regulation (EU) 2019/2144 and its 2023 Supplement

Regulation (EU) 2019/2144, also known as the *General Safety Regulation (GSR)*, mandates that, starting from July 2026, all newly manufactured vehicles must be equipped with an *Advanced Driver Distraction Warning System (ADDWS)*. The primary objective of this regulation is to reduce road accidents caused by driver distraction by ensuring the implementation of standardized distraction detection systems across the automotive industry [6].

To refine the technical requirements and approval procedures for ADDWS, a supplementary regulation was introduced on *July 13, 2023* (*Regulation 2023/2590*). This supplement establishes specific validation methodologies and performance criteria to ensure that ADDWS can accurately assess driver distraction under different operational conditions.

Technological Neutrality and Performance Requirements

The regulation stipulates that ADDWS performance requirements should be *realistic and achievable*, considering the limited experience with current systems and the need for further innovation. At the same time, the requirements must remain *technology-neutral* to foster the development of new solutions. The regulation focuses primarily on detecting and issuing warnings for *prolonged visual distraction*.

Additionally, the European Commission plans to *expand ADDWS requirements by July 2027* to incorporate new types of distraction, such as:

- **Intermittent distraction**, where the driver frequently shifts attention.
- **Cognitive distraction**, detecting when the driver is mentally disengaged.
- **Body movement analysis**, identifying scenarios where the driver turns backward.
- **Prevention mechanisms**, utilizing advanced technical solutions to mitigate distraction risks.

Privacy and Data Security Requirements

ADDWS must comply with strict *privacy regulations*, ensuring that:

- It does not rely on biometric data for driver identification. Biometric data includes facial images, fingerprints, or physiological characteristics that enable unique identification.
- It operates using non-identifiable image processing, meaning it can analyze eye gaze and head position without recording personal features.
- All collected data must be stored in a closed-loop system and used exclusively for ADDWS functionality.

Modification of Regulation (EU) 2019/2144

The original *Regulation 2019/2144* did not include direct legislative references for ADDWS. Therefore, an amendment was introduced, officially integrating ADDWS into the list of mandatory safety features. From *July 7, 2024*, all newly homologated vehicles must include ADDWS, with type-approval requirements aligned with the latest regulation updates [7].
According to the regulation, the *Advanced Driver Distraction Warning System (ADDWS)* is designed to detect when the driver's visual attention is not focused on the driving task and issue warnings through the vehicle's human-machine interface (HMI). The system must:

- Continuously monitor the driver's gaze to determine if attention is directed towards the necessary areas for safe driving.
- Provide appropriate warnings when a distraction event is detected.
- Be designed to minimize *false positives*, ensuring reliable operation in real driving conditions.

By setting strict performance standards, the regulation ensures that ADDWS effectively reduces distraction-related accidents while maintaining a balance between sensitivity and reliability.

Driver Monitoring Areas: ADDWS Evaluation Framework

A fundamental aspect of ADDWS validation is the division of the driver's visual space into three key *evaluation areas*:

- **Area 1**: Peripheral zones outside the primary visual field for driving. Includes the vehicle roof, side windows, and areas beyond ±55° from the driver's reference point. It is imperative to note that these include regions that the driver should not observe frequently. Consequently, monitoring gaze directed towards these areas enables the identification of any potentially inappropriate distractions.
- **Area 2**: The optimal field of view for road observation, covering the windshield and side mirrors within ±10° from the central vision axis. The ADDWS should verify that the driver maintains sufficient attention towards this area.
- **Area 3**: Interior cabin elements that may divert attention, such as infotainment screens, dashboard controls, and the lower console. Extended observation of these areas is classified as a distraction.

Figures 2.3, 2.4, and 2.5 illustrate these areas as defined in the regulation.

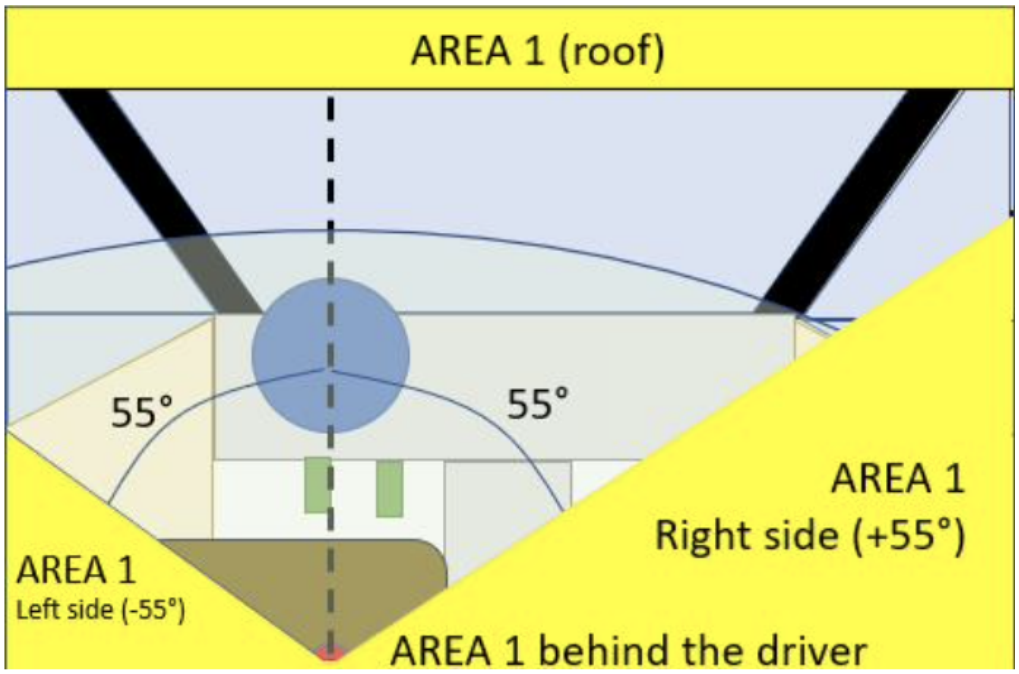


Figure 2.3: Area 1 [7].

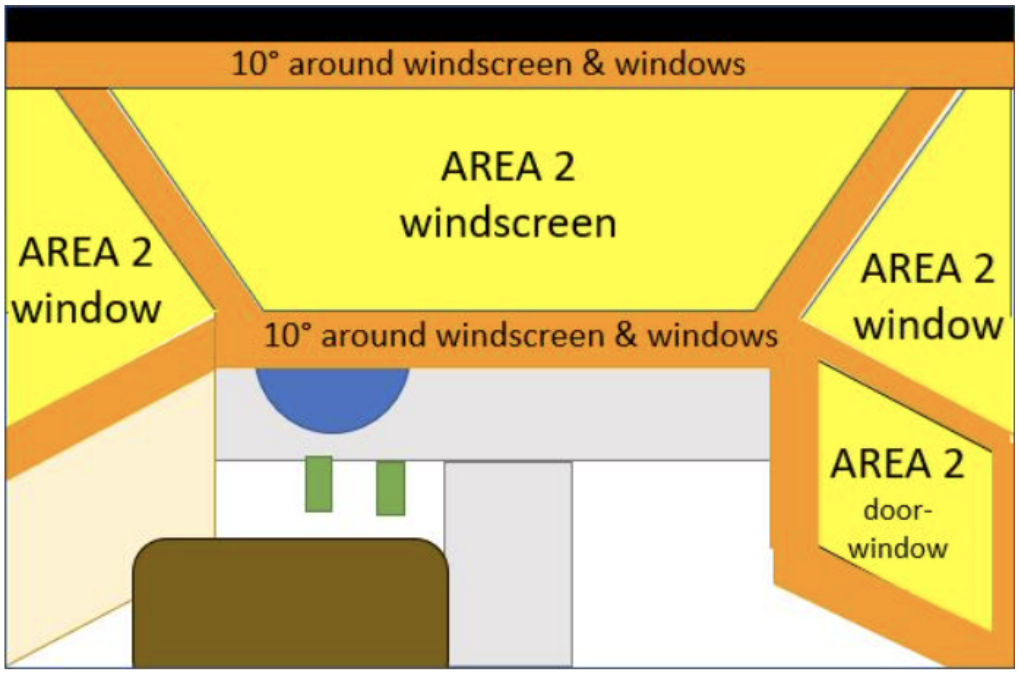


Figure 2.4: Area 2 [7].

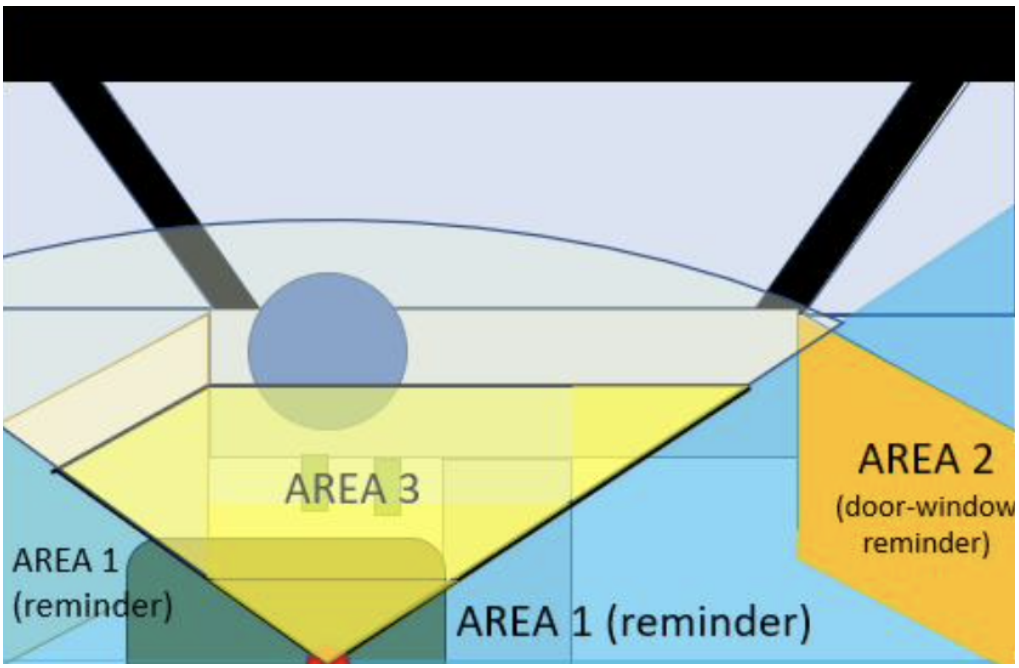


Figure 2.5: Area 3 [7].

Case Study: Triggering Conditions for Driver Warnings

To ensure an effective and standardized approach, the regulation defines precise criteria under which the system must generate a warning. These criteria take into account both normal driving conditions and external factors that may influence gaze detection accuracy.

The regulation distinguishes between *nominal* and *non-nominal* conditions when determining the thresholds for distraction classification. Nominal conditions represent standard driving scenarios where external disturbances do not significantly impact the system's ability to monitor the driver's gaze. These include well-lit environments, an unobstructed view of the road, and the driver maintaining a conventional seating position.

Conversely, non-nominal conditions encompass situations where environmental or physical variables may interfere with accurate gaze tracking. Examples include sudden changes in lighting (e.g., entering a tunnel), partial occlusions due to sunglasses or headwear, or the driver adopting an unusual posture. Recognizing these variations, the regulation allows for an additional 1.5-second tolerance when determining distraction thresholds in non-nominal conditions, preventing unnecessary false alerts.

To establish a structured warning mechanism, the regulation specifies two primary scenarios in which ADDWS must issue an alert, based on vehicle speed and gaze duration within *Area 3* (the region inside the vehicle most associated with visual distraction).

Case 1: High-Speed Distraction Alert At higher speeds, even brief instances of distraction can significantly increase the risk of accidents. Therefore, when the vehicle is traveling at or above *50 km/h*, ADDWS must issue a warning if the driver's gaze remains within *Area 3* for more than *3.5 seconds*. In non-nominal conditions, where tracking accuracy may be affected, the system may allow an additional *1.5-second grace period* before triggering an alert.

Case 2: Low-Speed Distraction Alert At lower speeds, the system permits slightly longer attention deviations before issuing a warning. If the vehicle is moving at or above *20 km/h*, an alert must be generated when the driver's gaze remains in *Area 3* for more than *6 seconds*. Similar to the high-speed case, a *1.5-second extension* is applied in non-nominal conditions to account for temporary tracking inconsistencies.

These thresholds ensure that ADDWS remains both effective and adaptable, balancing sensitivity with robustness to minimize unnecessary driver interventions while maintaining high safety standards.

2.3.2 Standardized Testing Procedures for ADDWS Validation

To ensure the reliability of *Advanced Driver Distraction Warning Systems (ADDWS)*, regulatory validation procedures include a structured *gaze fixation testing* methodology. This approach assesses whether the system can correctly identify prolonged visual distractions by analyzing the driver's gaze towards predefined fixation points within the vehicle's cabin.

Definition of Gaze Fixation Points

Fixation points are specific areas within the driver's field of view that may divert attention from the driving task. The selection of these points is based on the *geometric and design constraints* of the vehicle's cabin, as determined by the manufacturer. To ensure comprehensive validation, the test procedure includes gaze measurements directed at multiple locations, such as:

- The driver's knees and lap.
- The passenger footwell and seat area.
- The glove box and dashboard air vents.
- The instrument cluster and steering wheel.
- The gear shifter and climate control panel.
- The infotainment display and center console.

These fixation points cover both direct and peripheral areas that could lead to distractions, allowing a thorough assessment of ADDWS performance under real-world conditions.

Testing Methodology

The validation process is designed to detect instances where the driver engages in uninterrupted, long-duration gazes away from the primary driving task. Testing is conducted under controlled conditions and follows a structured sequence:

- The vehicle must reach a predefined test speed before distraction monitoring begins.
- The ADDWS system must confirm that the driver has maintained an attentive state for at least one minute before initiating the test.
- The driver is instructed to shift their gaze towards a designated fixation point and maintain focus until a warning is triggered or the expected response time has elapsed.
- Each fixation point is tested individually, ensuring that all relevant areas within the vehicle's cabin are assessed.

To prevent biased results, the order in which fixation points are tested may vary, and the driver's actions are limited to those naturally associated with the tested location.

Handling False Negatives and Re-Test Procedure

A critical aspect of ADDWS validation is the identification of false negatives—instances where the system fails to issue a distraction warning despite the driver maintaining prolonged gaze fixation within a predefined area.

If no warning is triggered within a specified time frame, the measurement is classified as a false negative. However, under certain circumstances, a false negative may be

reclassified as *not applicable* if another vehicle system (such as an audio or haptic alert) interferes with the test conditions.

To account for potential inconsistencies in human behavior assessment, a re-test procedure is applied. If a fixation point is initially classified as a false negative, the test is repeated up to two additional times, with the driver engaging in a different distraction-related action for each iteration. The test may be performed by the same or a different driver, provided they meet the required qualification criteria.

Final Evaluation and Acceptance Criteria

The ADDWS is considered to have *failed the validation test* if at least one fixation point results in repeated false negatives during the re-test procedure. Conversely, the system is deemed compliant if no fixation points meet the failure condition.

This standardized evaluation framework ensures that ADDWS implementations meet the necessary performance requirements, effectively identifying driver distraction while minimizing false positives and maintaining a balance between safety and usability.

2.4 Euro NCAP and Performance Criteria for ADDWS

The *European New Car Assessment Programme (Euro NCAP)* has introduced updated evaluation protocols for *Advanced Driver Distraction Warning Systems (ADDWS)*, which will be fully implemented by 2026. These protocols define standardized methodologies for assessing driver monitoring systems (DMS), ensuring that vehicles effectively detect and respond to driver distraction [15].

One of the key innovations in Euro NCAP 2026 is the classification of distraction based on movement patterns. The protocol distinguishes between *Owl movements*, which involve significant head movements away from the driving task, and *Lizard movements*, which are characterized by rapid eye shifts towards secondary areas. Understanding these movement types enables a more precise categorization of driver distraction events, helping to differentiate between momentary and prolonged inattention.

To further refine distraction assessment, Euro NCAP introduces the *Visual Attention Time Sharing (VATS)* model, which evaluates how drivers distribute their gaze across key areas such as the road, mirrors, and in-vehicle displays. This approach ensures that drivers do not exceed safe thresholds for non-essential glances and helps measure whether attention is properly alternated between critical visual zones.

2.4.1 Classification of Driver Monitoring Phases

Driver monitoring under the 2026 Euro NCAP framework is divided into two primary categories: *transient distractions*, which involve temporary lapses in attention, and *non-transient impairments*, where the driver's ability to operate the vehicle is significantly compromised.

Transient distractions include prolonged glances away from the road, improper attention distribution as measured by VATS, and phone use while driving. Phone-related distractions are further classified into *basic*, where the driver merely glances at the phone,

and *advanced*, which involves active engagement, such as texting. The system's response to transient distractions typically involves a sequence of visual, auditory, or haptic warnings. If the driver fails to react appropriately, corrective interventions such as lane-keeping support or adaptive braking may be activated.

In contrast, *non-transient* impairments encompass more severe conditions such as fatigue, microsleep episodes, complete sleep, or cases where the driver becomes unresponsive due to a medical emergency. These scenarios require a more aggressive intervention strategy. If the system detects an unresponsive driver, it escalates the warning intensity and, if necessary, initiates an emergency protocol. This may involve controlled braking, hazard light activation, and automatic emergency calls to alert first responders.

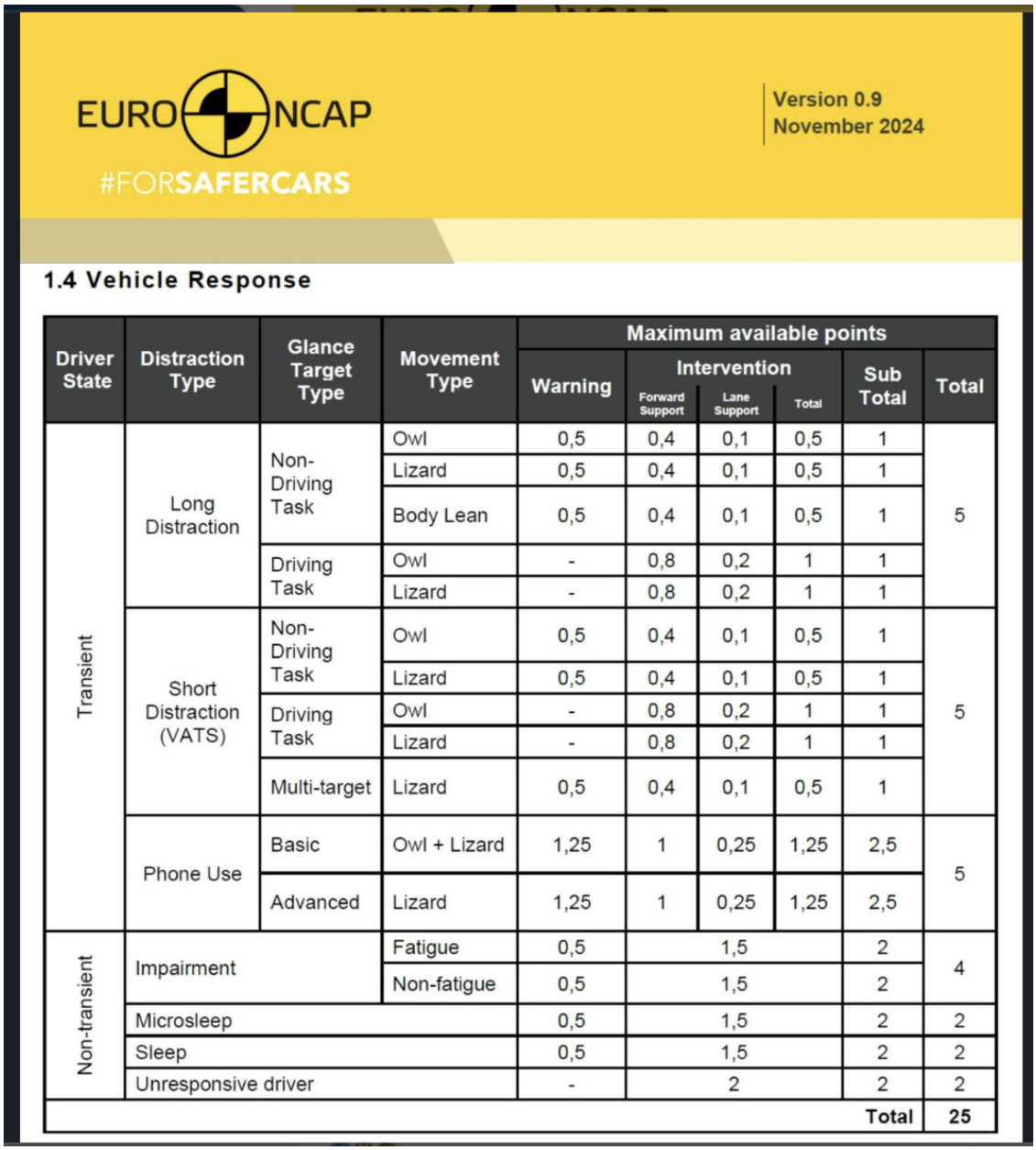
2.4.2 Scoring System and Impact on Vehicle Safety Ratings

Under the 2026 Euro NCAP protocols, driver monitoring systems are assigned a total of *25 points*, split between transient and non-transient distraction detection capabilities. To achieve a *5-star safety rating*, a vehicle must accumulate at least *80/100 points* in the *Safe Driving* category, which includes both driver and occupant monitoring systems. The thresholds for achieving this rating will progressively increase, requiring 60 points in 2026, 70 in 2027, and 80 by 2028.

Euro NCAP places particular emphasis on the role of vehicle manufacturers (*OEMs*) and their suppliers in ensuring compliance with these regulations. Tier-1 suppliers, such as Cipia, Seeing Machines, Smart Eye, and Tobii, are expected to play a leading role in delivering high-precision monitoring technologies that meet the updated safety requirements. As the regulatory landscape evolves, automakers that fail to adopt robust DMS solutions risk receiving lower safety ratings and potentially falling behind in the market.

2.4.3 Euro NCAP 2026 Scoring Criteria

Figure 2.6 illustrates the detailed Euro NCAP scoring framework for ADDWS, outlining how different types of distraction and impairment contribute to a vehicle's overall safety rating.



EURO NCAP
#FORSAFERCARS

Version 0.9
November 2024

1.4 Vehicle Response

Driver State	Distraction Type	Glance Target Type	Movement Type	Maximum available points					
				Warning	Intervention			Sub Total	Total
					Forward Support	Lane Support	Total		
Transient	Long Distraction	Non-Driving Task	Owl	0,5	0,4	0,1	0,5	1	5
			Lizard	0,5	0,4	0,1	0,5	1	
			Body Lean	0,5	0,4	0,1	0,5	1	
		Driving Task	Owl	-	0,8	0,2	1	1	
			Lizard	-	0,8	0,2	1	1	
	Short Distraction (VATS)	Non-Driving Task	Owl	0,5	0,4	0,1	0,5	1	5
			Lizard	0,5	0,4	0,1	0,5	1	
		Driving Task	Owl	-	0,8	0,2	1	1	
			Lizard	-	0,8	0,2	1	1	
		Multi-target	Lizard	0,5	0,4	0,1	0,5	1	
	Phone Use	Basic	Owl + Lizard	1,25	1	0,25	1,25	2,5	5
		Advanced	Lizard	1,25	1	0,25	1,25	2,5	
Non-transient	Impairment		Fatigue	0,5	1,5			2	4
			Non-fatigue	0,5	1,5			2	
	Microsleep			0,5	1,5			2	2
	Sleep			0,5	1,5			2	2
	Unresponsive driver			-	2			2	2
								Total	**25**

Figure 2.6: Euro NCAP 2026 ADDWS Scoring Criteria.

2.5 State of the Art in Driver Monitoring and ADDWS Validation

Ensuring driver attentiveness is a fundamental challenge in modern automotive safety. *Driver Monitoring Systems (DMS)* have evolved as a key component of *Advanced Driver Assistance Systems (ADAS)*, aiming to reduce accidents caused by distraction and drowsiness. These systems leverage a variety of technologies, including physiological signal processing, behavioral analysis, and vehicle-based monitoring, to assess the driver's state in real time.

Within the broader category of DMS, the *Advanced Driver Distraction Warning System (ADDWS)* represents a critical subsystem designed specifically to detect visual distraction. As mandated by *Regulation (EU) 2019/2144*, ADDWS will become a standard feature in all newly manufactured vehicles from July 2026 onward, reinforcing the need for robust validation methodologies.

The validation of DMS and ADDWS presents significant challenges. Traditional human-subject testing has been widely used to assess system accuracy; however, new validation strategies involving simulation-based environments, digital twins, and automated methodologies are emerging. Despite these advancements, no studies have explored the use of *humanoid robots* as a validation tool for ADDWS, leaving a gap in the current research landscape.
This section reviews the state of the art in DMS and ADDWS validation, covering:

- The technological evolution of DMS and their main challenges.
- The role of facial landmark-based monitoring in detecting driver distraction.
- Validation approaches for ADDWS, with a focus on automated testing methodologies.
- Open research challenges and the potential role of humanoid robotics in future validation frameworks.

By analyzing recent advancements in driver monitoring and ADDWS validation, this chapter lays the foundation for the proposed methodology introduced in subsequent sections.

2.5.1 Evolution and Current Challenges in Driver Monitoring Systems (DMS)

The development of *Driver Monitoring Systems (DMS)* has been driven by the need to reduce road accidents caused by human errors, with particular attention to visual distraction, drowsiness, and cognitive overload. Modern DMS leverage a combination of physiological, behavioral, and vehicle-based indicators to assess driver attention and intervene when necessary.

A fundamental aspect of DMS is the concept of *driver state*, which encompasses multiple physiological and cognitive conditions that affect driving performance. According to recent research [16], the driver state can be divided into several key substates, including:

- **Drowsiness**: A fluctuating state between wakefulness and sleep, characterized by reduced situational awareness and impaired cognitive performance.
- **Mental workload**: The cognitive effort required to process driving-related and secondary tasks, which can affect reaction times and attention.
- **Distraction**: A diversion of attention from the primary driving task due to visual, manual, auditory, or cognitive factors.
- **Emotions**: Affective states such as stress or anger that influence driving behavior.
- **Under the influence**: Impairment caused by alcohol, drugs, or medication that alters cognitive and motor abilities.

Each of these states presents distinct challenges for DMS, requiring different detection methodologies and sensor technologies.

Drowsiness Detection

Drowsiness is one of the most critical states monitored by DMS, as it significantly increases the risk of accidents. Unlike fatigue, which results from prolonged physical or mental exertion, drowsiness is specifically linked to the transition between wakefulness and sleep. Research indicates that drowsy drivers exhibit *reduced awareness, slower reaction times, and impaired motor coordination*, often without recognizing their own condition.

To detect drowsiness, modern DMS utilize a combination of *physiological, behavioral, and vehicle-based indicators* [16]:

Physiological indicators:
- Electroencephalogram (EEG) signals, which reflect changes in brain activity during the wakefulness-to-sleep transition.
- Electrocardiogram (ECG) and heart rate variability (HRV), which fluctuate as drowsiness progresses.
- Breathing patterns, as reduced alertness affects respiratory rhythm.
- Pupil diameter and blink rate, with drowsy drivers exhibiting longer, slower blinks and increased eye closure time.

Behavioral indicators:
- PERCLOS (Percentage of Eyelid Closure over Time), a widely used metric for drowsiness detection as we can see from the image 2.7.
- Blink frequency and duration, which increase as alertness declines.
- Head nodding and reduced gaze movement, indicating decreased situational awareness.

Vehicle-based indicators:
- Standard deviation of lane position (SDLP), as drowsy drivers struggle to maintain lane discipline.
- Steering wheel movement (SWM), which becomes more erratic as alertness decreases.
- Variability in vehicle speed and braking patterns, reflecting delayed reactions.

Despite these advancements, *real-time drowsiness monitoring remains a challenge*, as

individual variability and external factors (e.g., lighting conditions, road monotony) can impact detection accuracy.

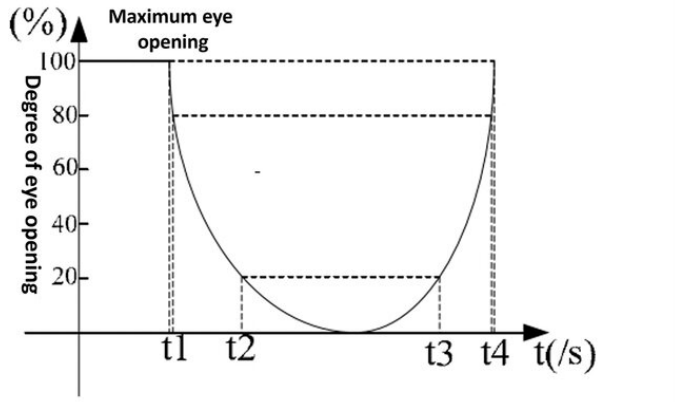


Figure 2.7: PERCLOS (Percentage of Eyelid Closure over Time) represents a sequential process of four states: eyes fully open, partial closure, full closure, and reopening [17].

Mental Workload and Cognitive Distraction

Mental workload is defined as the cognitive effort required to process driving-related and secondary tasks. While an increased workload is not inherently detrimental, given that complex driving scenarios demand greater attention, excessive cognitive load has the potential to impair decision-making and reaction times.

The DMS employs a range of indicators to assess mental workload, including:

- Physiological signals such as heart rate (HR) and electrodermal activity (EDA), which increase under cognitive stress.
- Eye-tracking metrics, such as fixation duration and saccade patterns, which undergo changes when drivers allocate excessive attention to secondary tasks.
- Gaze dispersion analysis, where a reduction in peripheral scanning suggests cognitive overload.

Cognitive distraction, a subset of mental workload, occurs when the driver's attention is diverted from the primary driving task due to internal thought processes. Cognitive distraction is a more insidious form of distraction, as it is harder to detect than visual distraction, which involves direct gaze shifts away from the road.

Distraction: Visual, Manual, and Auditory Factors

Driver distraction is categorized into four main types [16]:

- **Visual distraction**: This is characterised by looking away from the road to check a phone, infotainment system, or external stimuli.

- **Manual distraction**: This includes actions such as removing the hands from the steering wheel to adjust controls, eat, or handle objects.
- **Auditory distraction**: This can include paying attention to sounds inside or outside the vehicle, such as loud music or conversations.
- **Cognitive distraction**: This is characterised by the mental engagement in non-driving tasks, such as deep thinking or complex conversations.

Of these, visual distraction is considered the most critical, as prolonged glances away from the road directly correlate with accident risk. To detect visual distraction, DMS employ gaze-tracking algorithms that monitor eye fixation points and head movements. Systems such as Euro NCAP's Visual Attention Time Sharing (VATS) model assess how drivers distribute their gaze between critical areas, ensuring that non-driving-related glances remain within safe thresholds.

Emotion Recognition and Stress Detection

Emotional states such as stress, anger, and anxiety have been demonstrated to influence driving behaviour, often leading to aggressive maneuvers, impaired judgment, and slower reaction times. DMS integrate facial expression analysis, heart rate monitoring, and electrodermal activity measurements to assess the driver's emotional state.

The most advanced systems use deep learning models trained on facial micro-expressions to recognise stress levels and adjust vehicle responses accordingly. Nevertheless, it should be noted that the real-time emotion detection field is still in a state of development, with challenges relating to variability in individual expressions and environmental conditions.

Under the Influence: Alcohol and Drug Detection

Driving under the influence (DUI) of alcohol, drugs, or prescription medication remains a significant safety concern. Unlike drowsiness and distraction, which develop progressively, DUI-related impairment can result in immediate and severe cognitive and motor dysfunctions.

Modern DMS incorporate multiple detection methods:

- Breath-based alcohol sensors integrated into the vehicle cabin.
- Infrared imaging to detect changes in facial temperature and blood vessel dilation associated with alcohol consumption.
- Speech pattern analysis to identify slurred speech or abnormal vocal characteristics.
- Vehicle behavior monitoring, where erratic steering, abrupt braking, and inconsistent speed control indicate possible impairment.

As regulatory bodies push for stricter DUI monitoring, future DMS will likely integrate multi-modal sensor fusion to enhance real-time detection and intervention strategies [16].

Challenges in Modern DMS

Despite significant advancements, current DMS technologies still face multiple challenges [16]:

- **Occlusions and lighting conditions** – Vision-based DMS may struggle with face occlusions due to sunglasses, facial hair, or hand placement. Low-light environments can also affect detection performance.
- **Variability in driver behavior** – Differences in driver physiology, posture, and gaze behavior require adaptive models to prevent misclassification.
- **Latency and real-time processing** – High-speed computation is essential for delivering immediate alerts without disrupting the driving experience.

As DMS continue to evolve, future advancements will likely focus on multi-modal sensor fusion, AI-driven personalization, and enhanced real-time processing to improve accuracy and reliability.

The following section will explore a specific subset of behavioral DMS, focusing on facial landmark-based monitoring systems, which have demonstrated significant potential in enhancing driver attention analysis.

2.5.2 Real-Time Gaze Analysis for Driver Monitoring

In the context of Driver Monitoring Systems (DMS), real-time facial feature extraction is a critical component for assessing the driver's state. The paper under analysis presents a system designed to detect drowsiness and inattention through *infrared (IR) camera-based facial landmark estimation*. These facial landmarks are fundamental for evaluating *head pose and eye closure*, two key indicators of driver awareness and fatigue [18].

Facial Landmark Detection and Feature Extraction

To ensure high accuracy and real-time performance, the proposed DMS employs a *YOLOv7-based facial detection algorithm*, which offers a balance between speed and accuracy. Once the driver's face is detected, *Kazemi et al.'s random forest-based landmark extraction algorithm* is used to map key facial features efficiently (Kazemi's algorithm is based on random forests and offers a favorable combination of fast execution speed and good performance) [19]. This approach is particularly advantageous for embedded environments, as it ensures low computational cost compared to deep learning-based solutions.

The extracted facial landmarks are then used to derive two primary risk indicators for driving safety:

- **Head pose estimation**: Determines head orientation and gaze direction, identifying deviations from forward-facing attention.
- **Eye closure detection**: Captures prolonged eye closures, a critical marker of drowsiness.

By integrating these elements, the system effectively evaluates driver alertness and visual attention using a single IR camera, without relying on vehicle telemetry data [18]. This figure shows how the flow work is organised **??**.

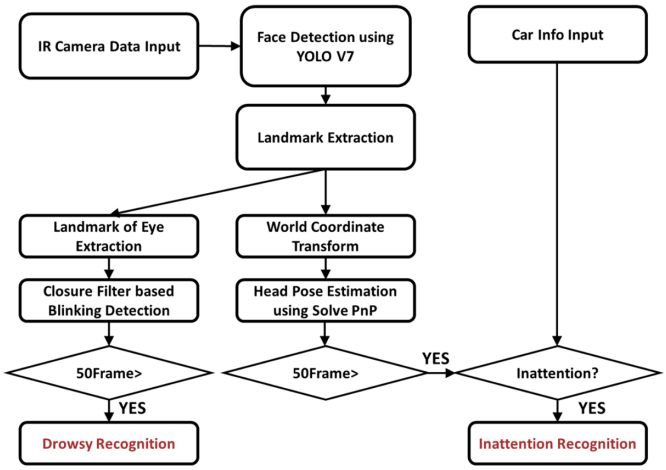


Figure 2.8: Flow work of this case study: [18].

Drowsiness Detection via Eye Closure Analysis

The proposed system addresses this issue through *an eye-closure detection filter* that processes IR images to differentiate between open and closed eye states. Unlike deep learning-based methods, which are prone to false positives, this approach leverages *threshold-based binarization* to isolate eye regions and classify them based on pixel intensity values.

A drowsy state is defined when eye closure is detected for at least 50 consecutive frames, a threshold that minimizes false alarms while ensuring accurate detection. Experimental results demonstrate that this method achieves a *detection accuracy above 99%*, making it highly reliable for real-time applications [18].

Inattention Detection through Head Pose Estimation

Inattention is another critical factor affecting driving safety, as it can result from distractions unrelated to drowsiness. This system employs a *solvePnP-based head pose estimation algorithm*, which reconstructs *3D head orientation from 2D facial landmarks*. This method calculates the rotation and translation vectors of the driver's head, enabling the system to determine whether the driver is looking forward or is distracted.

An inattentive state is detected when the driver's head remains oriented away from the road for over 50 consecutive frames. The system's performance in detecting inattention surpasses *99% accuracy*, demonstrating its robustness in real-time scenarios [18].

These results confirm its feasibility for integration into commercial vehicles, ensuring real-time assessment of driver awareness without requiring direct access to the vehicle's

Electronic Control Unit (ECU).

Relevance to This Research

The methodologies presented in this paper align closely with the objectives of this thesis, particularly in the domain of gaze-based driver monitoring and eyeblinking detection, thanks to the feature extraction of the facial landmarks. The use of infrared-based facial landmark analysis, head pose estimation, and threshold-based eye closure recognition provides a robust foundation for real-time driver state assessment [18]. Moreover, the system's ability to operate independently of vehicle telemetry makes it adaptable to a wide range of applications, reinforcing its potential for enhanced driver safety and accident prevention.

2.5.3 Validation Strategies for ADDWS: A Novel Approach

Despite significant advancements in Driver Monitoring Systems (DMS), the validation of Advanced Driver Distraction Warning Systems (ADDWS) remains an open challenge. While traditional on-road testing and real-world data collection have been extensively used for validating DMS functionalities such as drowsiness detection and gaze monitoring, there is no existing precedent for testing or validating ADDWS using synthetic data sources, such as humanoid robots or virtual drivers.

A key contribution to this field is presented in the study by *Miccichè et al., Validation Toolchain for Advanced Driver Distraction Warning Systems*, which provides an early framework for validating ADDWS performance in a virtualized environment. This work represents a first step toward alternative validation methodologies that move beyond real-world data collection [20].

Challenges in ADDWS Validation

The lack of established testing protocols for ADDWS is due to several critical challenges:

- Absence of Synthetic Testing Models – Unlike DMS, which has been validated through real-world driver behavior datasets, ADDWS lacks a structured approach to testing using synthetic drivers or humanoid models.
- Limited Access to Ground Truth Data – Effective validation requires large-scale, annotated datasets of distracted driving behaviors, which are difficult to obtain in controlled real-world settings.
- Variability of Distraction Patterns – Driver distraction is influenced by complex behavioral, cognitive, and contextual factors, making it challenging to create standardized validation metrics.

Virtual Validation: A New Paradigm for ADDWS

Given these challenges, virtual testing frameworks are emerging as a promising alternative for ADDWS validation. The approach proposed by Miccichè et al. introduces a validation

toolchain that leverages simulated distraction scenarios and AI-driven analysis to evaluate system accuracy [20]. Their methodology explores:

- **Simulation-Based Validation** – Using virtual driving environments to test ADDWS algorithms under different lighting, driving, and distraction conditions.
- **Digital Twin Frameworks** – Creating AI-generated driver avatars to simulate real-world distraction patterns, providing a repeatable and cost-effective validation pipeline.
- **Hardware-in-the-Loop (HIL) Testing** – Integrating real DMS sensors and cameras with synthetic drivers, allowing the system to be evaluated without requiring human test subjects.

Future Directions

While Miccichè et al. provide an initial foundation for virtual ADDWS validation, this field is still in its early stages. Further research is needed to:

- Develop more realistic virtual driver models capable of replicating human distraction behaviors.
- Establish standardized evaluation metrics for synthetic testing in ADDWS.
- Improve sensor fusion techniques to validate ADDWS through a combination of virtual and real-world data.

The study by Miccichè et al. represents one of the first documented attempts to validate ADDWS without relying on real driver data. The introduction of virtual drivers, digital twins, and AI-driven simulations opens new avenues for cost-effective, scalable, and standardized validation frameworks. However, significant research is still required to bridge the gap between synthetic testing methodologies and real-world system deployment.

2.5.4 Open Research Challenges and Future Directions

Although Driver Monitoring Systems (DMS) have been extensively studied, the validation of Advanced Driver Distraction Warning Systems (ADDWS) remains in its early stages. Given that ADDWS represents a recent technological advancement, there are currently very few studies that explore its real-world validation, whether using human subjects or alternative testing approaches. Existing validation efforts primarily focus on traditional DMS functionalities, while systematic methodologies for ADDWS evaluation remain underdeveloped.

Challenges in ADDWS Testing and Validation

The lack of established testing protocols for ADDWS presents several challenges:

- **Limited Empirical Validation** – Given the novelty of ADDWS, there is currently a lack of standardized, large-scale studies assessing its real-world performance.
- **Complexity of Distraction Detection** – Unlike traditional DMS, ADDWS must account for dynamic and transient states of driver distraction, requiring more sophisticated evaluation methodologies.
- **Absence of Benchmark Datasets** – Since ADDWS is not yet widely adopted, there is a shortage of publicly available datasets that can serve as references for validation.

Potential Benefits of Humanoid-Based Testing

One promising avenue for advancing ADDWS validation involves the integration of humanoid robots as test subjects. While this approach has yet to be explored, it could offer several advantages:

- **Controlled and Repeatable Testing** – Unlike human drivers, humanoid robots can perform distraction-related behaviors with high consistency, enabling standardized validation protocols.
- **Ethical and Safety Considerations** – Conducting distraction experiments on human drivers poses ethical and safety concerns, whereas humanoid robots eliminate such risks.
- **Scalability and Cost Efficiency** – Testing on humanoid platforms could reduce the costs associated with large-scale human trials while allowing for extensive parameter tuning.
- **Integration with AI-Driven Simulations** – Humanoid robots could be combined with simulation environments to bridge the gap between virtual and real-world validation.

Conclusion

As ADDWS technology progresses, establishing rigorous validation methodologies will be essential for ensuring its reliability and safety. While current research lacks a structured approach to testing these systems, future developments could benefit from the integration of humanoid robots, enabling controlled, scalable, and reproducible validation frameworks. This direction represents a crucial step toward refining ADDWS performance and accelerating its adoption in real-world driving environments.

Chapter 3

Methodology

3.1 Introduction

This chapter presents the methodology developed for validating **Visual Distraction** as a key component of the *Advanced Driver Distraction Warning System (ADDWS)*. Given the critical role of distraction detection in enhancing driving safety, a robust and reproducible validation process is essential.

The proposed validation framework leverages a *humanoid robot*, a *multi-sensor data acquisition system*, and *computer vision algorithms* to create a controlled testing environment. This novel approach aims to overcome the limitations of human-subject testing, ensuring:

- *Reproducibility*: The humanoid robot executes distraction scenarios with high precision, eliminating inter-subject variability.
- *Automation*: A fully integrated system collects, synchronizes, and processes sensor data in real-time.
- *Scalability*: The validation framework is adaptable to different distraction detection models without requiring live human participants.

The chapter is structured as follows:

- **Section 3.2** describes the hardware and software components forming the system infrastructure.
- **Section 3.3** outlines the data collection process and system workflow, detailing the interaction between different subsystems.
- **Section 3.4** introduces the *System Under Test (SUT)*, explaining its role in the validation pipeline.
- **Section 3.5** details the classification process, where collected data is analyzed and categorized into *Distracted* or *Not Distracted*.

This methodology establishes a standardized framework for validating ADDWS performance, laying the foundation for a scalable and objective evaluation system.

3.2 Tools

The validation framework is built upon a combination of *hardware* and *software* components, each of which plays a crucial role in the acquisition, processing, and classification of data.

The hardware setup consists of the *Ameca humanoid robot*, multiple *sensors*, a *camera*, and two *Raspberry Pi units* for distributed processing. These components work in unison to collect real-time data and execute predefined test scenarios.

The software stack integrates *Flask-based* communication *APIs* and *MediaPipe*, which are employed for pose estimation and real-time data synchronization.

The following subsections provide a detailed breakdown of each component in the system.

3.2.1 Hardware

Ameca Desktop

Ameca Desktop (3.1), a humanoid robotic platform developed by *Engineered Arts*, is designed to replicate human facial expressions and upper-body movements with high precision [21]. Unlike traditional robotic validation systems, Ameca's advanced mechanical actuation and AI-driven control system allow it to perform realistic head and gaze movements, making it an ideal synthetic driver for *driver monitoring system (DMS)* validation.

Figure 3.1: Ameca Desktop [21].

A key advantage of Ameca Desktop in *Advanced Driver Distraction Warning System (ADDWS)* validation is its ability to serve as a **highly accurate ground truth**. Equipped with precisely controlled servomotors, Ameca can execute predefined *yaw, pitch, and roll* head movements with repeatability exceeding that of human subjects. This ensures that test scenarios remain consistent and reproducible, eliminating the variability introduced by human drivers.

Furthermore, Ameca incorporates a high-fidelity facial expression system, capable of mimicking a wide range of human-like gestures and emotions. This feature is particularly relevant for *gaze-based driver monitoring*, allowing for rigorous testing of gaze-tracking algorithms under controlled conditions.

The robot is designed for *entertainment, education, and research* applications and features **32 degrees of freedom**, distributed as follows:

- 5 degrees of freedom in the neck.
- 27 degrees of freedom controlling the eyes, lips, and other facial features.

Additionally, Ameca is equipped with *microphones, cameras, and a speaker*, enabling two-way audio-visual interaction.

Ameca operates on the *Tritium* software platform, which provides a foundation for system control and interaction. To facilitate external communication, it utilizes *Tritium Remote*, a set of Python libraries that allow seamless interaction between Ameca and external devices such as the *Raspberry Pi*. Through these libraries, movement commands can be sent, and real-time data can be retrieved, ensuring precise control over Ameca's motor actions within the validation framework.

The integration of Ameca Desktop into the validation framework enables the system to achieve a scalable, automated testing process, establishing a benchmark for *ADDWS* evaluation without requiring human participants.

Raspberry Pi Units

The *Raspberry Pi* is a series of small, single-board computers developed by the *Raspberry Pi Foundation* (3.2). Originally designed for educational purposes, the Raspberry Pi has evolved into a powerful embedded computing platform used in robotics, automation, and edge computing applications [22].

Despite its compact size, the Raspberry Pi offers significant computational capabilities. It features an *ARM-based processor*, *general-purpose input/output (GPIO) pins*, and multiple communication interfaces, such as *I2C, SPI, and UART*, making it highly versatile for interfacing with external hardware components.

The validation system employs two Raspberry Pi units:

- **Raspberry Pi 4B+**: Manages the *Time-of-Flight (ToF) sensor* and depth data processing.
- **Raspberry Pi 3B+**: Handles image acquisition from the *Intel RealSense Camera*.

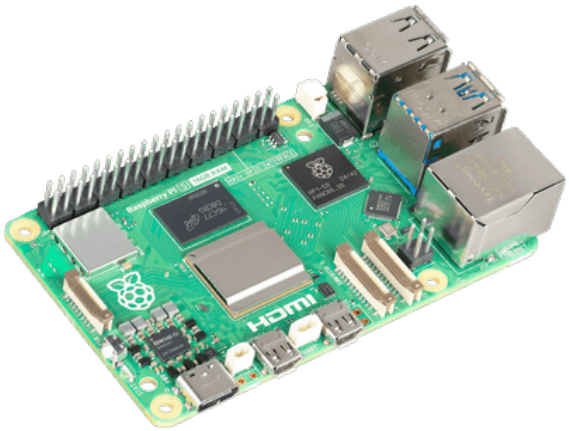


Figure 3.2: Raspberry Pi [22].

Intel RealSense Camera 435i

The *Intel RealSense Depth Camera D435i* (shown in 3.3) is a high-performance stereo depth camera designed for applications requiring high-quality depth perception. It features a wide field of view, making it particularly well-suited for use in robotics, augmented reality (AR), and virtual reality (VR) [23].

Figure 3.3: Intel RealSense Camera 435i [23].

Equipped with a global shutter sensor, the D435i offers superior depth accuracy and low-light sensitivity, allowing it to function effectively even in environments with limited illumination. The depth sensing range extends up to 10 meters, ensuring reliable performance across a wide variety of scenarios.

Key features of the Intel RealSense D435i include:

- Wide Field of View: Enables comprehensive scene capture, reducing the need for multiple cameras.
- Global Shutter Technology: Enhances image quality in motion-heavy environments.

- Integrated Vision Processor: Combines depth sensing and image processing in a compact form factor.
- Cross-Platform SDK Support: Compatible with *Intel RealSense SDK 2.0*, ensuring seamless integration into different development environments.

As part of the *Intel RealSense D400 series*, the D435i is designed for easy integration, providing a lightweight and cost-effective solution for depth sensing applications. Its versatility and robust design make it an essential component in the validation framework for ADDWS, where accurate depth perception is required for analyzing driver distraction and monitoring gaze behavior.

In the proposed architecture, **2 Intel RealSense D435i cameras** are utilized:

1. One camera is dedicated to **data collection**, capturing depth and RGB information from the test environment.
2. The second camera is integrated within the **System Under Test (SUT)**, providing an independent evaluation of driver distraction detection performance.

ToF Sensor – Arducam Camera

The *Arducam Time-of-Flight (ToF) Camera* 3.4 is a depth-sensing module that utilizes Time-of-Flight (ToF) technology to measure distances with high accuracy. Unlike traditional stereo vision systems, which rely on disparity calculations between two image sensors, ToF cameras actively emit modulated infrared light and measure the time it takes for the light to return after reflecting off objects in the environment [24].

The ToF technology operates using the *Continuous Wave (CW) method*, in which modulated infrared light is emitted from the camera. The depth information is obtained by measuring the phase shift between the emitted and reflected light waves. The travel distance (d) is calculated using the equation:

$$d = \frac{C}{2f} \tag{3.1}$$

where:

- C is the speed of light (3.0×10^8 m/s),
- f is the modulation frequency of the emitted light.

This method enables the camera to generate high-resolution depth maps, providing the *X, Y, and Z coordinate positions* of objects within the scene.

The Arducam ToF Camera is specifically designed for compatibility with embedded platforms, including the Raspberry Pi. It supports:

- *MIPI CSI-2 interface*, allowing direct connection to Raspberry Pi boards 3.5.

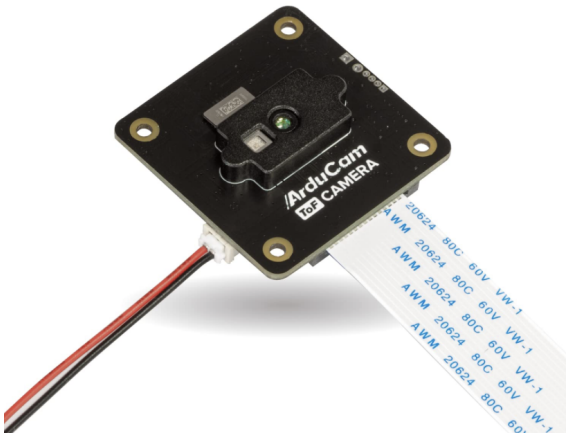


Figure 3.4: Arducam ToF Camera.

- *OpenCV and Python SDKs*, facilitating real-time depth processing and analysis.

- *Low-power consumption*, making it suitable for edge computing applications in embedded systems.

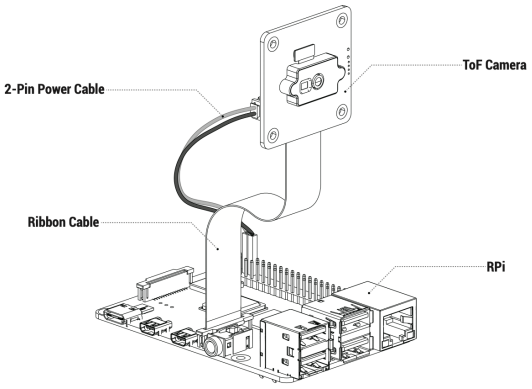
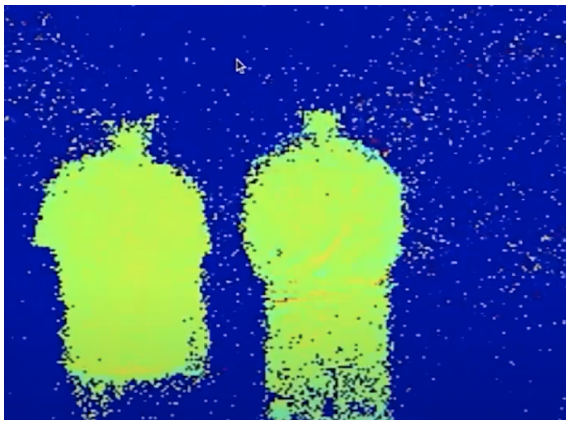
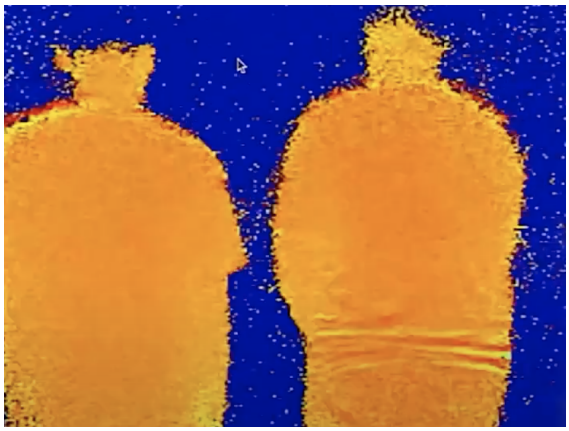


Figure 3.5: Arducam ToF Camera with RPi.

In our validation framework, the *Arducam ToF Camera* is employed to capture precise depth data, contributing to the detection of head movements and spatial awareness of the driver within the vehicle environment. The integration of this sensor with the Raspberry Pi ensures efficient data acquisition and real-time processing, enhancing the overall accuracy of driver distraction analysis. These 2 Figures show the beahviour of this Tof Camera



(a) Tof Arducam Camera Example 1.



(b) Tof Arducam Camera Example 2.

Comparison with Other ToF Sensors

During the sensor selection process, two alternative ToF sensors were evaluated: the **VL53L1X** and **VL53L0X**. These sensors, developed by STMicroelectronics, offer precise short-range depth measurements but present some limitations:

- *Limited range*: The VL53L1X and VL53L0X provide reliable measurements up to 4 meters, whereas the Arducam ToF Camera extends up to 10 meters.
- *Lower resolution*: The Arducam sensor captures a full depth map, while VL53L1X and VL53L0X offer single-point distance measurements.
- *Better environmental adaptation*: The Arducam camera performs better in varying lighting conditions and provides more detailed spatial information.

Given these factors, the Arducam ToF Camera was chosen as the primary depth sensor in this validation framework, ensuring a more comprehensive and scalable solution for detecting driver head movements and spatial awareness.

3.2.2 Software

The software infrastructure plays a crucial role in managing data acquisition, processing, and communication across the validation framework. To ensure seamless integration between hardware components and maintain real-time synchronization, the system relies on a combination of dedicated software tools and frameworks. The software and framework used in this research includes *Tritium OS*, *Flask-based API framework*, and *MediaPipe*. Each of these components plays a crucial role in the validation framework and will be described in detail in the following sections.

Tritium OS

Tritium OS is the dedicated operating system that controls the *Ameca humanoid robot* (Documentation here [25]). It provides an intuitive user interface, allowing users to manage Ameca's movements, expressions, and behaviors seamlessly. Designed for both research and commercial applications, Tritium OS ensures precise control over the robot's actuators, enabling realistic and repeatable motion sequences.

A key feature of Tritium OS is the **Animator**, a built-in tool that facilitates the creation of motion sequences through a timeline-based approach. The Animator allows users to define keyframes, adjust movement parameters, and synchronize multiple degrees of freedom. This functionality is crucial for applications such as human-robot interaction research, entertainment, and, as in this study, *driver distraction validation*.

In this research, the Animator was used to design controlled motion scenarios for Ameca, simulating various visual distraction behaviors. By pre-programming realistic head movements, gaze shifts, and reaction patterns, Tritium OS provided a reliable foundation for testing the robustness of ADDWS. The generated animations ensured high reproducibility, eliminating inconsistencies associated with human test subjects and enabling precise validation of distraction detection algorithms.

Figure 3.7 illustrates the Animator interface, showcasing keyframe manipulation, motion track adjustments, and controller selection for movement execution.

Flask

Flask is a lightweight and flexible *Python-based web framework* used for developing web applications and APIs. Its simplicity and modularity make it an ideal choice for handling communication in distributed systems, enabling seamless interaction between different hardware and software components.

In this research, Flask is employed to create a set of *Flask-based servers*, allowing real-time communication between various nodes of the validation framework. Each node—whether managing sensors, cameras, or the humanoid robot—operates as an independent unit with a dedicated *IP address*. Flask facilitates this communication through a structured *APIs*, where predefined *endpoints* enable different functionalities.

The APIs are designed to handle crucial operations, for example start and stop data recording or Real-time control of Ameca

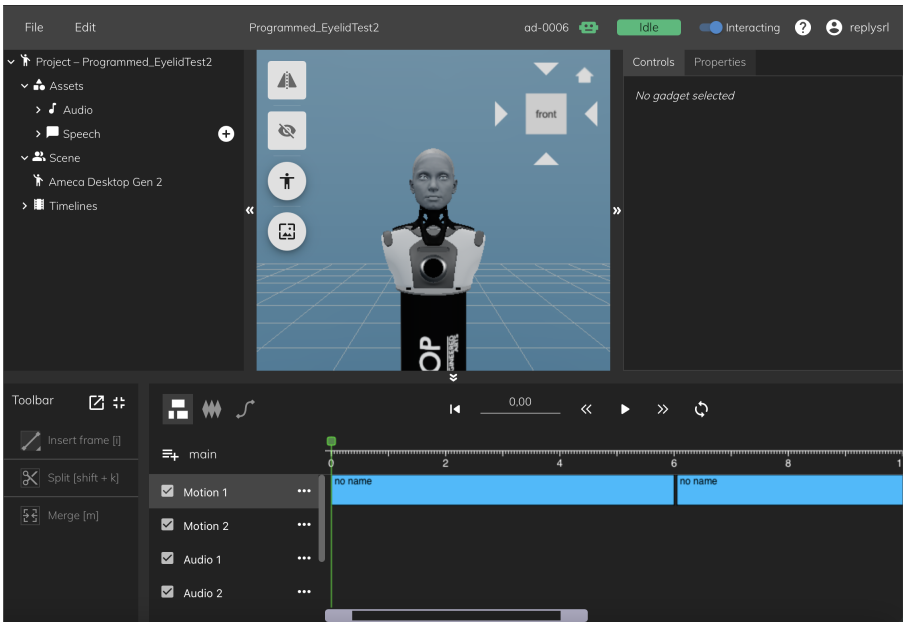


Figure 3.7: Tritium OS Animator interface.

By leveraging Flask's server infrastructure, the system ensures *low-latency* communication and efficient data management. This architecture allows each hardware component to function independently while maintaining synchronization with the Master Node, which orchestrates the overall testing process.

Flask's integration into the validation framework enables a scalable and modular architecture, ensuring reliable data exchange and real-time control over all system components.

MediaPipe

MediaPipe is an open-source library developed by Google for *real-time tracking* of facial and body features. Leveraging its pre-trained *machine learning* models, it enables efficient detection of *facial landmarks*, hand positions, and full-body posture. One of its key advantages is its ability to run on edge devices with minimal latency, making it ideal for applications such as augmented reality, gesture recognition, and driver monitoring systems.

Although MediaPipe was originally trained on human faces, experimental validation has shown that it performs effectively on *Ameca* as well, accurately detecting its facial landmarks despite the robot's synthetic structure. This makes it a viable tool for analyzing *visual distraction* in a humanoid-based testing environment.

In this research, MediaPipe serves as the core of the System Under Test (SUT), as it is used to track Ameca's facial landmarks to evaluate visual distraction. The system continuously monitors key facial features, including *eye openness, head orientation, and gaze direction*, providing a real-time assessment of the robot's attentional state.

The distraction analysis is based on extracting *468 facial landmarks*, which allow for

the detection of variations in eye openness and head movements. These parameters are then compared against the *ground truth*, represented by Ameca's pre-programmed motor movements, ensuring a direct correlation between expected and observed behavior.

While, thanks to MediaPipe, it provides an automated, objective, and repeatable validation method for visual distraction classification, it is important to note that it cannot be directly compared to an actual DMS camera. Unlike dedicated driver monitoring cameras, which use *infrared imaging, advanced gaze tracking algorithms, and calibrated depth sensors*, MediaPipe relies solely on RGB image processing. Thus, while it offers a useful benchmark for testing purposes, it does not replicate the full capabilities of commercial-grade DMS systems.

3.3 Data Collection and System Workflow

This section provides a detailed breakdown of how data is acquired, transmitted, and processed within the system. Specifically, it covers:

1. The role of the Master Node and its coordination with distributed nodes.
2. The integration of Frontend and Backend components, ensuring a seamless flow of input commands and output data.
3. The communication architecture based on Flask APIs, which enables real-time data exchange across different system modules.

3.3.1 Master Node and Distributed Nodes

The **Master Node** serves as the central controller within the validation framework, responsible for managing and coordinating the execution of all other nodes. Implemented as a Python script, it ensures the modularity and scalability of the system by employing asynchronous functions that instantiate and control independent processes for each distributed node. This structure allows each node to operate autonomously while remaining synchronized within the overall system workflow.

A key component enabling this coordination is the `global_var.py` boolean variable. This shared variable, which is atomic in nature due to the multi-threaded design of the system, is exclusively accessed in read-write mode by the Ameca node, while all other nodes only perform read operations. This atomicity guarantees consistent synchronization, ensuring that each node begins and ends its respective recording process in alignment with Ameca's animation state, which serves as the core trigger of the entire system.

The nodes managed by the Master Node include:

1. **Camera Node**: Responsible for video recording. This node starts and stops video capture upon receiving commands from the Master Node. Additionally, it maintains a `.csv` file that logs the total number of recorded frames along with their corresponding timestamps, ensuring a structured record of the video acquisition process.

2. **Sensor Node**: Manages depth data acquisition. Similar to the Camera Node, it starts and stops the depth recording and stores the frame count and timestamps in a `.csv` file, facilitating precise synchronization of depth information with other data sources.

3. **Ameca Node**: Controls the humanoid robot *Ameca*, ensuring the execution of predefined animations required for validation. Upon receiving a start command from the Master Node, Ameca begins playing a programmed animation. Simultaneously, a script running on *Tritium OS* records the real-time values of specific motor groups, referred to as the **ground truth (GT)**. These values provide essential reference data for evaluating distraction detection accuracy. The Ameca's motor groups defined as GT, include:

   - *Eyelids*: Eyelid Lower Left; Eyelid Lower Right; Eyelid Upper Left; Eyelid Upper Right.
   - *Jaw*: Jaw Yaw; Jaw Pitch.
   - *Eyes*: Eye Pitch Left; Eye Pitch Right; Eye Yaw Left; Eye Yaw Right,
   - *Head and Neck*: Head Pitch; Head Roll; Head Yaw; Neck Pitch; Neck Roll

   Each motor group's movement data is stored in a structured JSON format, ensuring precise recording of position values over time. The JSON file includes several key fields for each motor group:

   - Motor Name: The specific name of the motor group (e.g., `"Eyelid Lower Left"`).
   - Min and Max Values: The minimum and maximum recorded values of the motor's position during the execution.
   - Number of Samples: The total number of recorded position values for the given motor.
   - Positions Array: A list of all recorded position values for the motor at different timestamps.
   - Times Array: The corresponding timestamps (in Unix epoch format) for each recorded position value.

   An example of the recorded data structure for the motor group `"Eyelid Lower Left"` is shown below:

```
"Eyelid Lower Left": {
    "min": -1.0,
    "max": 2.0,
    "num_samples": 352,
    "positions": [1.2546, 1.2548, 1.2546, 1.2546, 1.2548, ...],
    "times": [1.7417895759226437E9, 1.7417895759330137E9, ...]
}
```

The `positions` array represents the motor's recorded values at each timestamp, while the `times` array provides the exact moment each value was sampled. This structured format enables accurate time synchronization with other recorded data sources, ensuring that the ground truth can be effectively compared with the output of the *System Under Test (SUT)* during validation.

This data format plays a crucial role in evaluating visual distraction events, as it allows for direct comparisons between Ameca's pre-programmed motor movements (ground truth) and the real-time distraction classification output obtained from *MediaPipe*.

This is a visual example of how the GT data are collected 3.8


```
{
  "Eyelid Lower Left" : {
    "min" : -1.0,
    "max" : 2.0,
    "num_samples" : 352,
    "positions" : [ 1.2546531982421876, 1.254814432779948, 1.2546531982421876, 1.2546531982421876, 1.2
    "times" : [ 1.7417895759226437E9, 1.7417895759330137E9, 1.7417895759828413E9, 1.7417895760227866E9
  },
  "Eyelid Lower Right" : {
    "min" : -1.0,
    "max" : 2.0,
    "num_samples" : 352,
    "positions" : [ 1.2625102867426672, 1.2623451729804236, 1.2621798004191798, 1.2623451729804236, 1.2
    "times" : [ 1.7417895759224749E9, 1.7417895759525292E9, 1.7417895759824 96E9, 1.74178957602259 8E9,
  },
  "Eyelid Upper Left" : {
    "min" : -1.0,
    "max" : 2.0,
    "num_samples" : 352,
    "positions" : [ 0.9452775312484025, 0.9455899869862663, 0.9452150484207802, 0.9453400140760249, 0.9
    "times" : [ 1.7417895759226115E9, 1.74178957595313 6E9, 1.7417895759827 9E9, 1.7417895760227451E9, 1
  },
  "Eyelid Upper Right" : {
    "min" : -1.0,
    "max" : 2.0,
    "num_samples" : 352,
    "positions" : [ 0.9446855949991331, 0.944497309244125, 0.9447170111466383, 0.944622847156132, 0.94
    "times" : [ 1.7417895759224007E9, 1.7417895759524453E9, 1.7417895759823 95E9, 1.7417895760225096E9,
  },
```


Figure 3.8: Ground-Truth.json

By structuring the system in this manner, the Master Node ensures seamless communication and synchronization between all nodes while maintaining flexibility for future expansions or modifications. The `global_var.py` variable is pivotal for this coordination, ensuring that all data collection processes are precisely aligned with Ameca's motion execution.

3.3.2 Flask API Endpoints for System Communication

The validation framework is structured around a distributed network of independent processing nodes, each responsible for a specific aspect of data acquisition and validation. To ensure seamless communication and coordination, each node operates its own dedicated

Flask-based server, exposing a set of **endpoints** that enable the Master Node to control their respective processes asynchronously. This architecture guarantees a modular and scalable design, allowing each subsystem to function autonomously while maintaining synchronized execution.

The **Camera Node**, deployed on the Raspberry Pi 3B+, runs its own Flask server dedicated to handling video recording. The Master Node interacts with this server by sending requests to predefined endpoints. The `/start` endpoint triggers the beginning of a video recording session, initializing the camera with specific parameters such as resolution, frame rate, and output format. Throughout the recording, the system logs each frame's timestamp into a CSV file, ensuring that the captured data remains temporally aligned. Once the recording is complete, the `/stop` endpoint is called to finalize the session and store the video file. To facilitate efficient data management, the Camera Node also provides endpoints such as `/list_files` and `/latest_file`, enabling the Master Node to retrieve recorded sessions dynamically. Additionally, the `/download` endpoint allows for remote access to recorded videos, packaging them into a ZIP file alongside their associated metadata. If necessary, the `/delete_all_files` endpoint can be invoked to clear all stored recordings from the system.

Similarly, the **ToF Sensor Node**, which operates on a separate Raspberry Pi 4B+, is managed by an independent Flask server dedicated to depth sensing. The Master Node controls this sensor by invoking the `/start` endpoint, which initiates the capture of depth data, storing distance measurements frame by frame. The acquisition continues until the `/stop` endpoint is triggered, marking the completion of the recording session. To maintain accessibility, the ToF Sensor Node provides endpoints for querying the most recent file (`/latest_file`) and downloading recorded data via `/download`. The use of a standalone Flask server ensures that depth data acquisition remains fully decoupled from other processes, reinforcing the modularity of the framework.

The most complex component, the **Ameca Node**, operates on a Windows-based machine and runs its own Flask server within a WSL (Windows Subsystem for Linux) environment. Unlike the other nodes, Ameca is not simply a data acquisition unit but an active participant in the validation process, executing predefined movement sequences to simulate driver behavior. The Master Node interacts with the Ameca Node through several dedicated endpoints.

The `/play_sequence` endpoint is responsible for initiating motion sequences, instructing Ameca to perform predefined head and eye movements corresponding to distraction scenarios. Additionally, the `/start_script` endpoint launches an executable Python script directly within *Tritium OS*, the operating system that controls Ameca. This script is responsible for collecting real-time motor position data from Ameca's actuators, generating the ground truth (GT) data for the validation process. The direct execution of the script within *Tritium OS* ensures that motor data is captured with maximum fidelity, reducing latency and maintaining synchronization with the predefined animation sequences.

To facilitate structured data acquisition, the `/start_capture` endpoint is used in conjunction with `/start_script`. While the script runs within *Tritium OS*, the `/start_capture` endpoint leverages the **ZeroMQ (zmq)** messaging library to continuously receive and

store motor position data. The zmq-based communication ensures efficient and low-latency streaming of ground truth data from Ameca to the validation framework. Once the data collection is complete, the `/stop_capture` endpoint will be automatically called by the Master Node and all the data will be formatted into a structured JSON file, making it available for retrieval and further analysis.

By implementing a **dedicated Flask server for each node**, the framework guarantees that all components operate independently while maintaining synchronized execution. This design choice eliminates dependencies between subsystems, enabling each node to function autonomously while adhering to the validation workflow. Furthermore, the distributed architecture ensures that data acquisition remains reliable, scalable, and easily extendable, paving the way for an efficient and reproducible validation process.

3.3.3 Frontend and Backend: Input and Output of the System

The validation framework for the Advanced Driver Distraction Warning System (ADDWS) is structured into two main components: the **backend**, responsible for managing the execution and coordination of processes, and the **frontend**, which provides an interface for user interaction. The backend is primarily managed by the **Master Node**, which also operates as a **Flask server**, exposing a minimal set of endpoints to control test execution.

Backend and Master Node APIs

At the core of the system, the Master Node serves as the central unit that coordinates the execution of all nodes. The communication between the frontend and backend is facilitated by a **JSON file**, which contains all necessary specifications for the test session. The Master Node processes this file through the following Flask API endpoint:

`/process_json`

This endpoint allows the frontend to specify which components should be enabled or disabled during a test session, as well as additional settings such as camera format or Ameca's execution scenario. An example of the JSON structure provided as input to the Master Node is:

```
{
    "camera": {
        "is_enable": false,
        "type": "rgb"
    },
    "ameca": {
        "is_enable": true,
        "type": "CNCQIT_EyelidTest.project"
    },
    "sensor": {
        "is_enable": false
    },
```

```
    "sut": {
        "is_enable": true
    }
}
```

Once the test session is completed, the system generates an output containing all recorded data. The retrieval of this output is handled by another Flask API endpoint:

`/get_output`

This endpoint returns a compressed archive containing various output files:

- The **ToF sensor data**, including an `.avi` recording and a `.csv` file listing the timestamps for each frame.
- The **camera data**, comprising an `.avi` recording, a `.csv` file with frame timestamps, and a `.json` metadata file.
- The **ground truth data** from Ameca, stored as `ground_truth_data.json`, which contains motor position values for each motor group.

Parallel Execution of the System Under Test (SUT)

Simultaneously with the data collection process, the Master Node also initiates the execution of the **System Under Test (SUT)**, ensuring synchronization with the other system nodes. Like other components, the SUT operates through its own dedicated **Flask server**, which will be discussed in detail in the following chapter. The output generated by the SUT is stored in a file named `SUT.json`, which will be analyzed in the next section.

Frontend: User Interaction and System Configuration

The entire execution process is initiated through the **frontend**, which provides an interface for users to configure test parameters. The frontend is developed using **Java-based functions** and is responsible for transmitting the configuration JSON file to the backend.

The frontend plays a crucial role in configuring the test environment. Built with Java-based functions, it enables the user to specify which sensors and components should be activated before starting a test. As shown in Figure 3.9, the interface allows the user to toggle each component (Camera, Ameca, Sensor, and SUT) on or off using interactive switches. Once configured, the frontend sends the JSON configuration to the Master Node via the `/process_json` endpoint.

This structure ensures a modular, scalable, and fully automated testing framework, enabling seamless integration between the user interface, data acquisition nodes, and the processing infrastructure.

3.4 System Under Test (SUT)

The *System Under Test (SUT)* is a Python-based script designed to process Ameca's facial movements using the *MediaPipe* library. Despite being originally trained for human

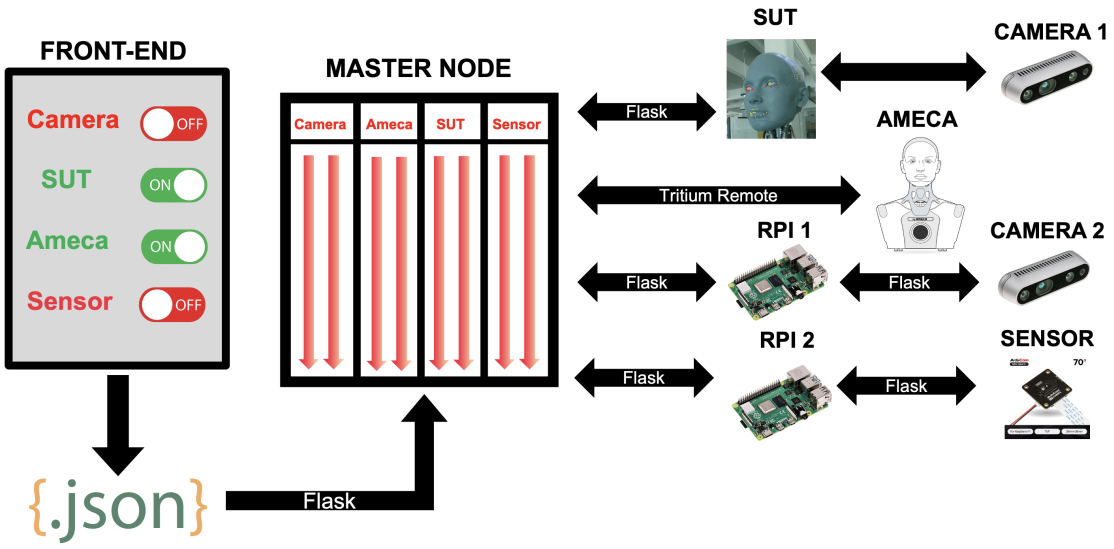


Figure 3.9: Overall system architecture, illustrating the communication between the frontend and backend.

face tracking, MediaPipe proves highly effective in identifying Ameca's facial features with remarkable precision. The SUT is responsible for analyzing specific facial cues to determine whether Ameca is experiencing a distraction event during predefined testing scenarios.

This system operates as an independent component within the experimental framework and is designed to be adaptable across multiple distraction detection methodologies. While the eye blink detection scenario is a primary focus of analysis (detailed in Chapter 4, the SUT can also be extended to recognize other forms of visual distraction, such as eye gaze detection.

3.4.1 Facial Landmark Extraction and Processing

The core functionality of the SUT revolves around extracting and analyzing facial landmarks in real time. These landmarks, provided by MediaPipe, define key regions of the face, including:

- **Eyes**: Used to detect blinks, gaze direction, and potential loss of attention.
- **Mouth**: Monitored to assess signs of drowsiness or cognitive distraction.
- **Nose and Head Orientation**: Evaluated to detect head movements that could indicate attention shifts.

At each frame, the system processes a set of predefined landmark coordinates, normalizes them relative to the camera frame, and applies mathematical computations to derive behavioral insights. The extracted data are then classified into discrete distraction states, allowing for a structured and interpretable output.

3.4.2 Implementation and Flask Server Architecture

The SUT is implemented as a standalone *Flask server*, enabling remote execution and real-time communication with the *Master Node*. This architecture allows for seamless integration with the broader system. The SUT exposes three key API endpoints:

- **/start_sut**: Initiates the landmark extraction process and begins the distraction classification, storing both raw data and video footage of the execution.
- **/stop_sut**: Terminates the execution and finalizes the output files, which include a JSON report and a corresponding video recording.
- **/get_results**: Retrieves the processed results, delivering a ZIP archive containing the distraction classifications and video footage.

This architecture ensures modularity, allowing for flexible deployment and easy adaptation to different testing environments.

3.4.3 Distraction Classification and JSON Output

The SUT continuously analyzes landmark movements and classifies distraction states in real time. Each frame is labeled according to a predefined classification scheme, which varies depending on the specific scenario being tested.

The resulting output is structured in a JSON format as follows:

```
{
  "times": [1741789577.170, 1741789577.202, ...],
  "labels": ["Normal", "Normal", "Event", ...],
  "FrameCount": 292
}
```

Interpretation of JSON fields:

- `times`: List of timestamps corresponding to each processed frame.
- `labels`: Classification label for each timestamp (`Normal` for no distraction, `Event` for detected distraction).
- `FrameCount`: Total number of frames analyzed during the session.

Figures 3.10a and 3.10b illustrate two possible classifications generated by the SUT.

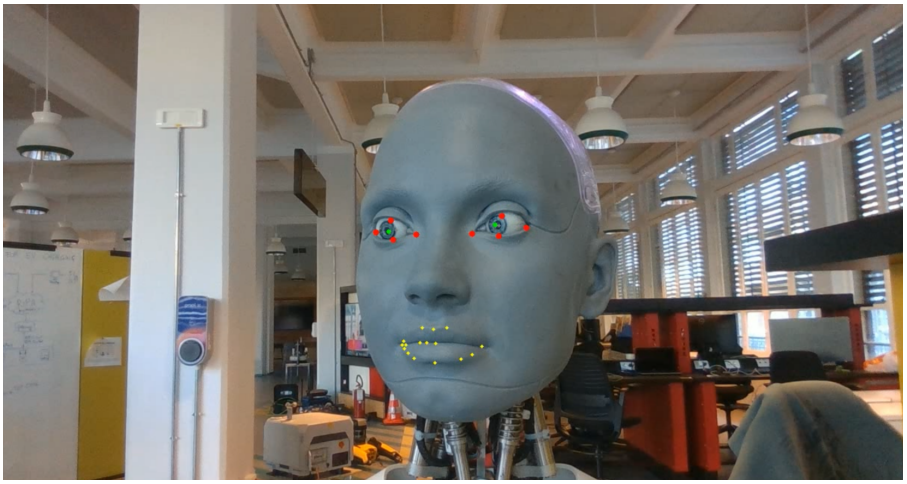
3.4.4 Extensibility to Different Distraction Scenarios

While the current implementation of the SUT focuses on the eye blink detection case study, its modular design allows it to be extended to other forms of distraction analysis.

For example:

- *Eye Blink Detection*: Evaluates eyelid closure based on predefined thresholds, detecting prolonged blinks that indicate a loss of attention.

(a) Example of Ameca classified as **Distracted** by the SUT.



(b) Example of Ameca classified as **Not Distracted** by the SUT.

Figure 3.10: SUT classification of Ameca's distraction state.

- *Eye Gaze Distraction Detection*: Tracks gaze direction and determines whether Ameca is looking away from a predefined focal point.
- *Head Movement Analysis*: Uses head pose estimation to identify moments when Ameca turns its head excessively, potentially indicating cognitive distraction.

This adaptability ensures that the SUT can be reconfigured or expanded to accommodate different testing paradigms without requiring significant architectural changes.

3.4.5 Integration with the Master Node

The SUT operates in coordination with the *Master Node*, which manages its execution and synchronizes data collection with the other system components. The distraction

labels generated by the SUT serve as an independent dataset, which is later compared with Ameca's ground truth classifications to evaluate system accuracy.

By leveraging real-time facial tracking, asynchronous data processing, and a modular design, the SUT provides a flexible and robust framework for evaluating Ameca's attention state across various experimental scenarios.

3.5 Classification of Ground Truth Data

Once all relevant data have been retrieved, the next step involves classifying Ameca's motor movements to identify specific distraction-related events. This is achieved through the *runClassification* function, which applies a threshold-based analysis to determine the state of selected motor groups at each recorded timestamp.

The classification process focuses on motor groups that contribute to key distraction-related behaviors. These include, but are not limited to:

- *Eyelid movements* (upper and lower eyelid control motors)
- *Head orientation* (yaw, pitch, and roll control motors)
- *Eye movement* (horizontal and vertical gaze control motors)

The function iterates through the motor position values and applies predefined thresholds to determine whether a distraction event has occurred. The motors operate within a normalized range, and each movement is classified as follows:

- If the motor position falls within a predefined **neutral range**, the timestamp is labeled as **Normal**.
- If the motor position exceeds a predefined **event threshold**, the timestamp is labeled as **Event**.

The classification results are stored in a JSON file that mirrors the original ground truth data but now includes an additional field containing distraction labels. The following snippet illustrates an example structure:

```
{
    "Eyelid Lower Left": {
        "positions": [1.2, 0.8, 0.1,...],
        "labels": ["Normal", "Normal", "Event",...],
        "times": [1741789577.170, 1741789577.202, 1741789577.235,...]
    },
    "Head Pitch": { ... },
    "Eye Yaw Right": { ... }
}
```

3.5.1 Final Vector Construction

To provide a consolidated representation of distraction events, the classification results from multiple motor groups are merged into a unified structure called the **Final Vector**. This vector is constructed by aggregating the classification labels across all relevant groups at each timestamp:

- If all motor groups are classified as *Event*, the timestamp is labeled as *Event*.
- If at least one motor group is classified as *Normal*, the corresponding timestamp is labeled as *Normal*.

This final set of data provides a structured representation of Ameca's distraction states over time. The output, stored in JSON format, is structured as follows:

```
{
    "Final_Vectors": {
        "timestamps": [1741789577.170, 1741789577.202, 1741789577.235,...],
        "labels": ["Normal", "Normal", "Event",...]
    }
}
```

3.5.2 Preparation for Validation

This labeled set of data serves as the foundation for the validation phase. By comparing the Final Vector with the classifications provided by the SUT, it is possible to assess the system's accuracy in detecting and interpreting distraction-related events.

This process transforms raw motor data into meaningful behavioral insights, enabling a comprehensive evaluation of the system's effectiveness in capturing and classifying relevant motor events.

Chapter 4

Validation and Analysis

4.1 Introduction

The validation phase represents a critical step in assessing the accuracy and reliability of the proposed methodology for detecting visual distraction in Ameca. Following the acquisition and processing of data from both the Ground Truth (GT) — derived from Ameca's internal motor signals — and the System Under Test (SUT) — based on computer vision via the MediaPipe framework — a systematic comparison is performed to evaluate the system's classification performance.

This chapter is devoted to the validation of the **Prolonged Eye Blink** scenario, which has been evaluated under three distinct experimental conditions:

- **Static Scenarios:** two pre-defined sequences in which Ameca executes controlled, fixed-duration eye blinks.
- **Idle Scenario:** a free-running mode where Ameca exhibits spontaneous, variable-length blinks, simulating natural behavior.
- **LiveLink Scenario:** a real-time playback of human facial expressions, previously recorded via iPad-based facial tracking and reproduced by Ameca.

Additionally, a test scenario dedicated to **Gaze Distraction** was implemented. However, due to the noisy and unstable nature of MediaPipe's pitch and yaw signals within the SUT, this specific task could not be validated quantitatively. While the Ground Truth data enabled gaze classification, the absence of a high-quality Driver Monitoring System (DMS) in the SUT prevented reliable comparisons. This limitation is discussed in the final chapter as a direction for future work.

To perform the validation, a direct comparison is made between the GT labels and the SUT predictions using a structured `comparison.json` file, which aligns timestamps from both sources and extracts event-based discrepancies.

Finally, system performance is assessed in terms of detection accuracy, temporal precision (start/end deltas), and classification errors (false positives/negatives). The results are presented and discussed across all test conditions, with the goal of identifying both the strengths and current limitations of the proposed distraction detection pipeline.

4.2 Validation Methodology

4.2.1 Comparison between Ground Truth and System Under Test Data

The validation process is built upon a rigorous comparison between the Ground Truth (GT) dataset and the dataset generated by the System Under Test (SUT). This comparison serves as a fundamental step in assessing the reliability and accuracy of the SUT in detecting visual distraction events. The `runComparison` function is responsible for this task, processing both datasets and generating a structured output in the form of a JSON file, named *comparison.json*, which summarizes the detected events, their timestamps, and the success rate of the SUT in replicating the GT observations.

4.2.2 Event Detection Mechanism

To ensure a consistent and objective evaluation, the methodology employs a structured approach to identifying distraction events in both datasets. Each dataset consists of a sequence of labels, where:

- `Event` indicates the presence of a distraction.
- `Normal` indicates the absence of a distraction.

An event is formally defined when the following two conditions are met:

- A minimum of **50 consecutive frames** must be labeled as `Event`, ensuring that only substantial distractions are recorded.
- The event must be followed by at least **10 consecutive frames** labeled as `Normal` to confirm its conclusion. This threshold prevents the system from prematurely detecting multiple fragmented events instead of a single continuous distraction.

Once a valid distraction event is identified, the system records its exact timing by retrieving:

- **Start Timestamp**: The UNIX timestamp corresponding to the first occurrence of `Event` in the detected sequence.
- **End Timestamp**: The UNIX timestamp corresponding to the last occurrence of `Event` before transitioning back to `Normal`.

This procedure is applied independently to both the GT dataset and the SUT dataset, yielding two separate lists of detected distraction events.
These two Figures show a visual example 4.1(Valid Event), 4.2(Invalid Event).

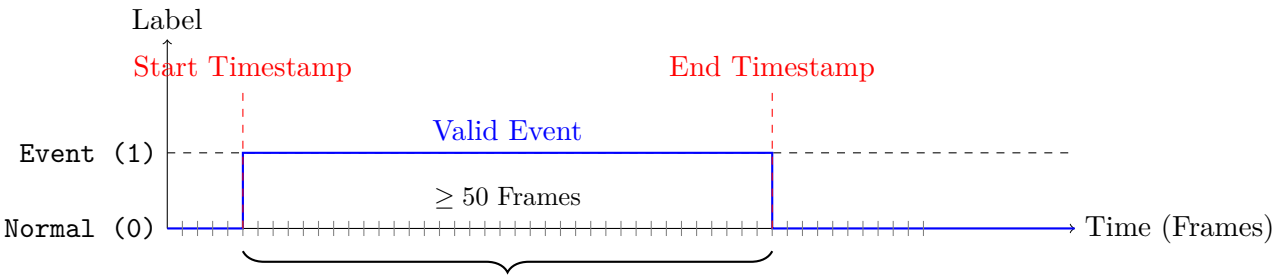


Figure 4.1: Valid distraction event: 50+ consecutive `Event` labels followed by 10+ `Normal`.

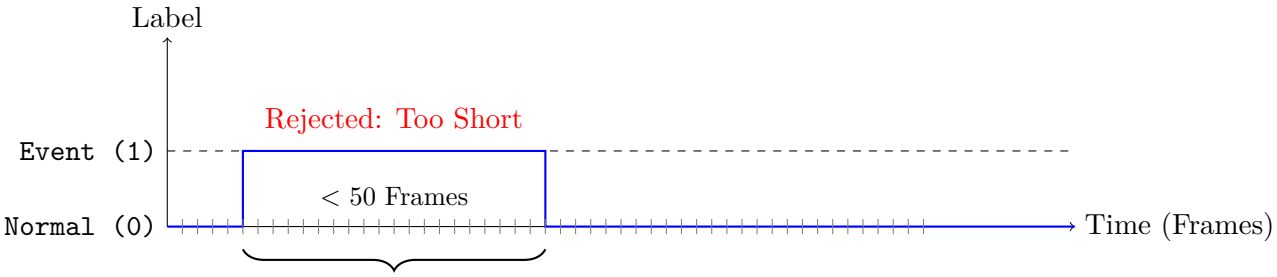


Figure 4.2: Invalid distraction event: fewer than 50 consecutive `Event` labels, not meeting the event threshold.

4.2.3 Matching Ground Truth and SUT Events

Once the events are extracted from both datasets, the next step is determining whether the SUT successfully replicated the GT events within an acceptable margin of error. Given the inherent variability in real-time execution, a **tolerance of 1 second** is introduced to account for potential deviations in timing. The tests have been conducted also with a different tollerance of **1.5** second, adding more variability in the analysis.

For each event in the GT dataset, the algorithm verifies if a corresponding event exists in the SUT dataset by ensuring that:

- The start time of the SUT event falls within **1 or 1.5 second** or of the GT event's start time.
- The end time of the SUT event falls within **1 or 1.5 second** of the GT event's end time.

If both conditions are satisfied, the event is classified as a **match**, and the test is marked as `Passed` 4.3. If no match is found, the test is marked as `Failed` 4.4.

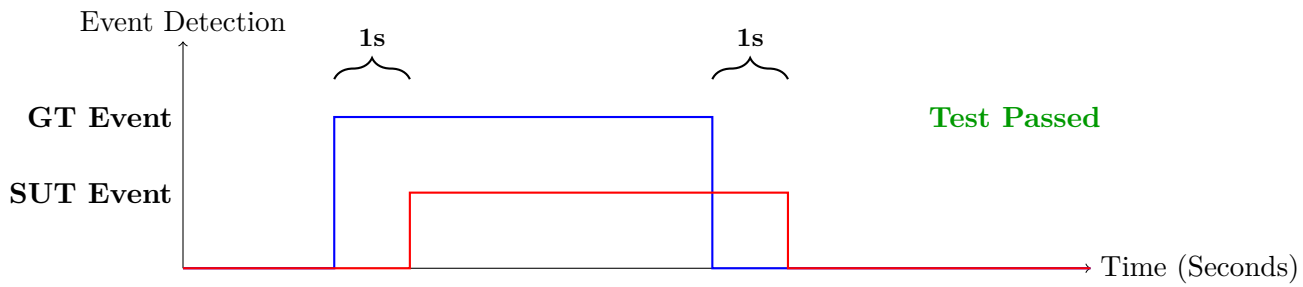


Figure 4.3: Successful event match: The SUT event starts and ends within the accepted 1-second margin.

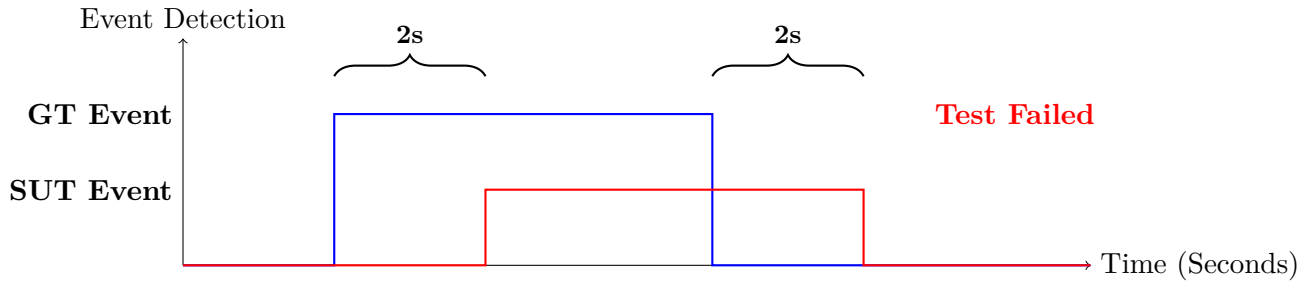


Figure 4.4: Failed event match: The SUT event exceeds the allowed 1-second margin.

4.2.4 Normalization of SUT Timestamps

In addition to storing event timestamps in UNIX format, the comparison process introduces a **normalized time representation** for the SUT events. This normalization is crucial for direct alignment with the visual recordings of the experiment. The normalized time is computed relative to the start of the SUT recording:

$$t_{\text{normalized}} = t_{\text{UNIX}} - t_{\text{start\_SUT}} \tag{4.1}$$

where:

- t_{UNIX} is the absolute UNIX timestamp of the detected event.
- $t_{\text{start\_SUT}}$ is the UNIX timestamp at the beginning of the recording session.

This transformation allows for an intuitive comparison between the timestamps and the actual video footage recorded during the session. When reviewing the video, the start and end times of each distraction event can be directly mapped to the corresponding timestamps in the normalized format, enabling visual confirmation of the detected events.

4.2.5 Comparison Output Structure

The final step involves constructing the `comparison.json` file, which encapsulates all relevant comparison results. This JSON file contains:

- **Total GT Events**: The number of distraction events detected in the GT dataset.
- **Matched Events**: The number of GT events successfully matched by the SUT.
- **Match Percentage**: The percentage of GT events that were correctly detected by the SUT.
- **GT Events**: A list of all detected GT events, including their start and end timestamps.
- **SUT Events**: A list of all detected SUT events, including their start and end timestamps, normalized timestamps, and test results (`Passed` or `Failed`).

An example output of the `comparison.json` file is shown below:


```
{
  "Total_GT_Events": 2,
  "Matched_Events": 2,
  "Match_Percentage": 100.0,
  "GT_Events": [
    {
      "start": "1742228331.3027",
      "end": "1742228337.1327"
    },
    {
      "start": "1742228343.3030",
      "end": "1742228349.1025"
    }
  ],
  "SUT_Events": [
    {
      "start": "1742228330.9217",
      "end": "1742228337.2290",
      "start_normalized": 2.0302,
      "end_normalized": 8.2376,
      "Test": "Passed"
    },
    {
      "start": "1742228342.9301",
      "end": "1742228349.2333",
      "start_normalized": 13.9386,
      "end_normalized": 20.2419,
      "Test": "Passed"
    }
  ]
}
```

Related to this file JSON, we can observe in this sequence of images, referring to the test example video, how each normalized start and end timestamp is perfectly synchronized with the video 4.5, 4.6, 4.7, 4.8.

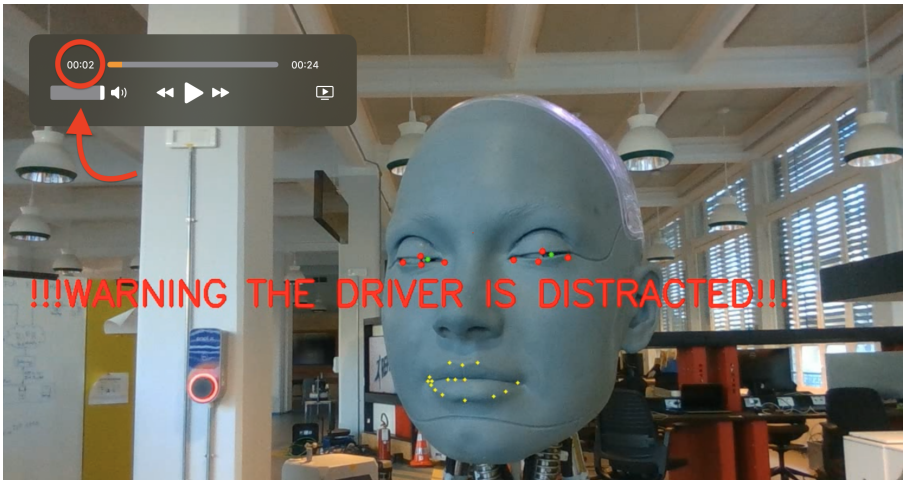


Figure 4.5: Start Normalized: 2.0302

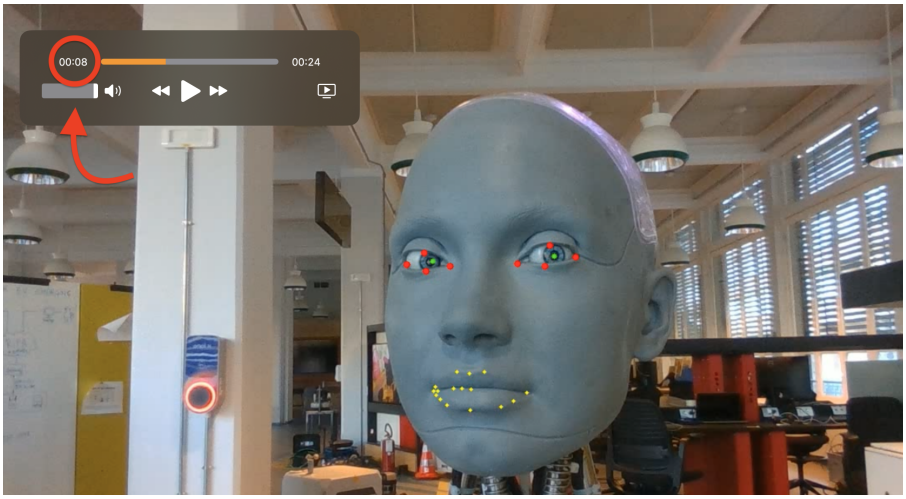


Figure 4.6: End Normalized: 8.2376

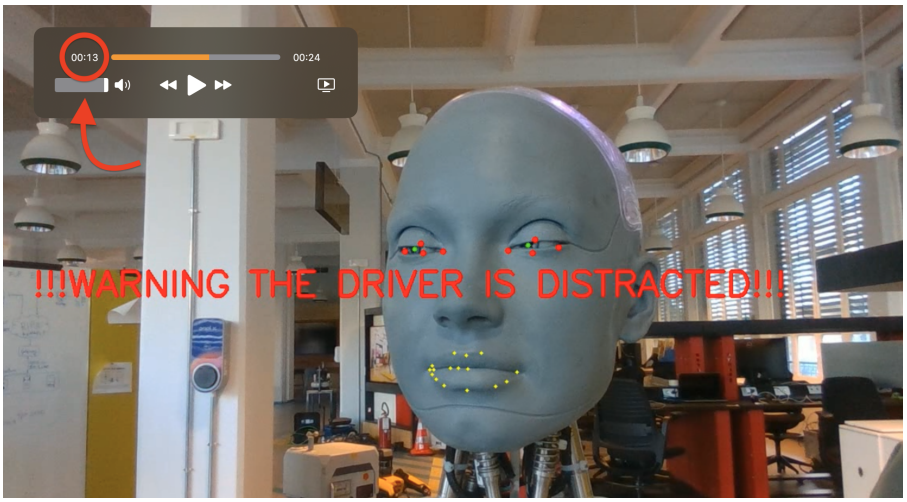


Figure 4.7: Start Normalized: 13.9386

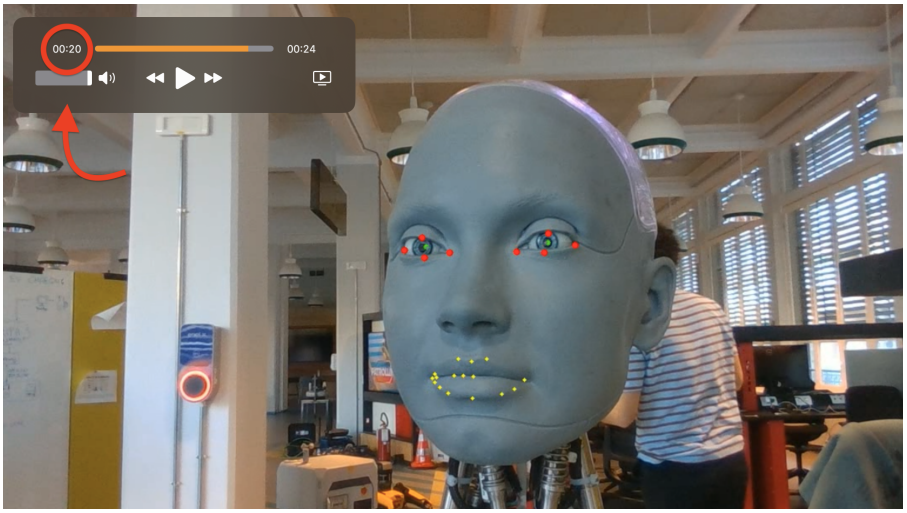


Figure 4.8: End Normalized: 20.2419

4.2.6 Significance of the Comparison

The ability of the SUT to accurately detect distraction events is a key indicator of its performance. By systematically comparing its results with the GT dataset, the `runComparison` function quantifies the system's reliability. The normalized timestamps further enhance this evaluation by enabling direct alignment with recorded video footage, allowing for manual verification of the detected events.

This structured comparison provides an empirical foundation for assessing the effectiveness of the SUT. The results obtained in this step serve as the basis for further analysis in the following sections, where the performance across different experimental scenarios will be systematically examined.

4.3 Analysis of Eye Blink Prolonged Scenarios

The **Eye Blink Prolonged** scenario constitutes the primary case study for this validation framework. This class of distraction events was selected due to its controllability and direct correlation with visual attention loss. Unlike brief, reflexive eye closures, prolonged blinks are intentionally exaggerated in duration (between 3 and 6 seconds), allowing the system to assess its capability in detecting sustained visual distraction.

To validate the detection process, both the **Ground Truth (GT)** and the **System Under Test (SUT)** independently classify Ameca's state of attention over time. Their results are then compared using the methodology described in Section 4.2.

Ground Truth Event Definition

the GT classification process of this scenario involves analyzing the movement values of four eyelid-related motor groups:

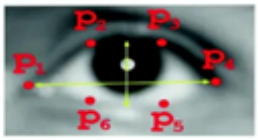
- `Eyelid Lower Left`
- `Eyelid Lower Right`
- `Eyelid Upper Left`
- `Eyelid Upper Right`

Each group is evaluated based on a threshold of *0.2*, considering *2* as the maximum eye openness and *-1* as the maximum eye closure: if the motor position at a given timestamp falls below this threshold, the motor is classified as **Closed**. The final GT distraction state is computed by intersecting the classification of all four groups. If **all the Eyelid motors** are classified as *Open*, the system marks the timestamp as an *Event*, that in this case scenario is label as `Blink`; otherwise, it is considered `Normal`. The result is stored in the `Final_Vectors` field of the GT JSON output:

```
{
    "Final_Vectors": {
        "timestamps": [1741789577.170, 1741789577.202, 1741789577.235,...],
        "labels": ["Normal", "Normal", "Blink",...]
    }
}
```

4.3.1 System Under Test: Eye Blink Detection via EAR

In the context of *eye blinking*, the System Under Test evaluates visual distraction using the *Eye Aspect Ratio (EAR)*. This metric, commonly adopted in facial expression analysis, gaze tracking, and driver drowsiness detection, is calculated as the ratio between the vertical distance of the eyelids (upper and lower) and the horizontal distance between the eye corners (inner and outer), as illustrated in Figure 4.9.



$$\text{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

Figure: Eye Aspect Ratio Formula

Figure 4.9: Eye Aspect Ratio (EAR) computation formula [26].

The fundamental concept behind EAR is that when the eyes are open, the vertical distance between the upper and lower eyelids is significantly greater than the horizontal distance between the eye corners. Conversely, when the eyes are closed, the vertical distance decreases, leading to a reduction in the EAR value. A commonly used threshold for detecting eye closure is 0.2. If the EAR value drops below this threshold, the system classifies the corresponding timestamp as *Blink*; otherwise, it is classified as *Nomral*:

```
{
  "times" : [1742468387.6075172, 1742468387.6358855, 1742468387.6673908,...]
  "labels" : ["Normal", "Blink", "Blink",...]
  "FrameCount" : 1856
}
```

By continuously monitoring the EAR, the system identifies the precise moments when an eye blink starts and ends, allowing for accurate detection of visual distraction [26].

After detailing the logic used to classify blink-related distraction events for both the System Under Test and the Ground Truth—resulting in comparable data formats—we now proceed to evaluate the accuracy of their comparison across a variety of test scenarios.

4.3.2 Static Scenarios

In this subsection, we analyze two manually constructed blink scenarios, referred to as **Static_1** and **Static_2**. These were developed using the *Animator* tool, allowing for precise control over Ameca's motor movements and enabling the simulation of prolonged eye blinks with deterministic timing.

- **Static_1:** Ameca performs a single, uninterrupted eye closure lasting **6** seconds.
- **Static_2:** Ameca performs two sequential eye closures, each lasting **3** seconds.

These synthetic animations are labeled as "static" because the distraction events are fully predefined and not generated from stochastic behavior. This enables a rigorous validation of the detection and comparison pipeline under controlled conditions.

For each scenario, a total of **10 test runs** were conducted, subdivided into two groups based on the comparison tolerance parameter:

- **5 tests with a tolerance of 1.0 second**
- **5 tests with a tolerance of 1.5 seconds**

The tables that follow report the performance of each test execution. In addition to the number of matched ground truth events and their relative success percentage, we include the temporal discrepancies between the Ground Truth and SUT events:

- Δ_{Start}: The absolute time difference between GT and SUT event start times.
- Δ_{End}: The absolute time difference between GT and SUT event end times.

Table 4.1: Test Results – Static_1 Scenario with 1s Tolerance (with Temporal Deltas)

Test ID	Scenario	Tol.	GT Ev.	SUT Ev.	Match %	Δ_{Start} [s]	Δ_{End} [s]	Outcome
1	Static_1	1.0s	1	0	0.0%	0.245	1.096	Failed
2	Static_1	1.0s	1	1	100.0%	0.214	0.988	Passed
3	Static_1	1.0s	1	1	100.0%	0.062	0.937	Passed
4	Static_1	1.0s	1	0	0.0%	0.074	1.066	Failed
5	Static_1	1.0s	1	1	100.0%	0.113	0.922	Passed

Table 4.2: Test Results – Static_1 Scenario with 1.5s Tolerance (with Temporal Deltas)

Test ID	Scenario	Tol.	GT Ev.	SUT Ev.	Match %	Δ_{Start} [s]	Δ_{End} [s]	Outcome
6	Static_1	1.5s	1	1	100.0%	0.010	0.979	Passed
7	Static_1	1.5s	1	1	100.0%	0.100	0.897	Passed
8	Static_1	1.5s	1	1	100.0%	0.044	0.892	Passed
9	Static_1	1.5s	1	1	100.0%	0.256	0.917	Passed
10	Static_1	1.5s	1	1	100.0%	0.133	1.289	Passed

Table 4.3: Test Results – Static_2 Scenario with 1s Tolerance (with Temporal Deltas)

Test ID	Scenario	Tol.	GT Ev.	SUT Ev.	Match %	Δ_{Start} [s]	Δ_{End} [s]	Outcome
11	Static_2	1.0s	2	2	100.0%	0.338 / 0.350	0.747 / 0.649	Passed
12	Static_2	1.0s	2	2	100.0%	0.367 / 0.270	0.544 / 0.507	Passed
13	Static_2	1.0s	2	2	100.0%	0.337 / 0.232	0.672 / 0.674	Passed
14	Static_2	1.0s	2	2	100.0%	0.071 / 0.277	0.683 / 0.282	Passed
15	Static_2	1.0s	2	2	100.0%	0.377 / 0.251	0.644 / 0.314	Passed

Table 4.4: Test Results – Static_2 Scenario with 1.5s Tolerance (with Temporal Deltas)

Test ID	Scenario	Tol.	GT Ev.	SUT Ev.	Match %	Δ_{Start} [s]	Δ_{End} [s]	Outcome
16	Static_2	1.5s	2	2	100.0%	0.374 / 0.282	0.651 / 0.554	Passed
17	Static_2	1.5s	2	2	100.0%	0.399 / 0.366	0.632 / 0.582	Passed
18	Static_2	1.5s	2	2	100.0%	0.368 / 0.305	0.639 / 0.578	Passed
19	Static_2	1.5s	2	2	100.0%	0.158 / 0.323	0.589 / 0.661	Passed
20	Static_2	1.5s	2	2	100.0%	0.341 / 0.319	0.646 / 0.641	Passed

4.3.3 Idle Scenario

In this section, we evaluate the performance of the distraction detection pipeline under the **Idle Scenario**. This mode corresponds to the default behavior of Ameca, in which the robot autonomously moves its facial and upper-body motors in a symmetric yet pseudo-randomized fashion, aiming to emulate natural human-like motion during inactivity. Here a sequence of 4 image that illustrates this idle mode 4.10.

To increase the complexity of the testing environment, we introduced a randomized component that forces Ameca to perform a series of **prolonged eye blinks**. Specifically:

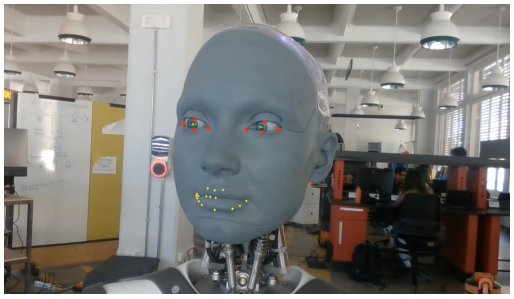
- The **duration** of each blink is randomized within a range of **1 to 5 seconds**.
- The **interval between two consecutive blinks** is also randomized, spanning from **4 to 10 seconds**.

This extended behavior is overlaid on top of Ameca's standard Idle Mode, thereby enriching the test conditions with both structural unpredictability and natural movement variability. Each test execution lasts exactly **60 seconds**, during which several eye blink events of varying lengths and intervals are generated.

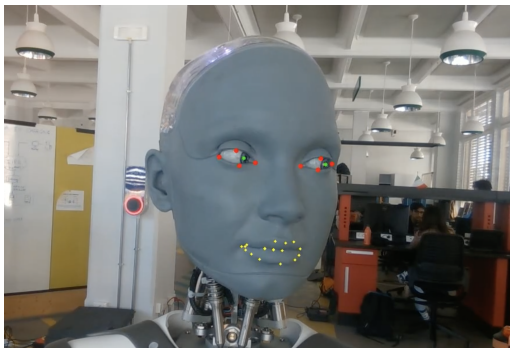
In total, we conducted **20 test sessions** for this scenario:

- **10 tests with a tolerance of 1.0 second**
- **10 tests with a tolerance of 1.5 seconds**

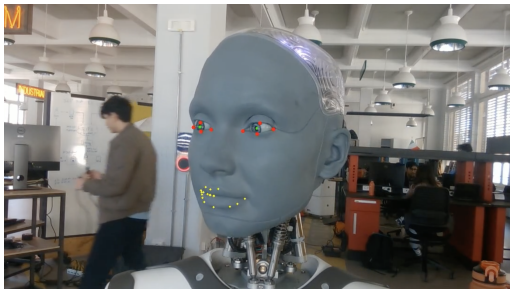
Due to the stochastic nature of both the robot's motor behavior and the randomly injected blink timings, this scenario is particularly effective for evaluating the robustness and adaptability of the comparison mechanism. Here are the Test Results related to this idle scenario (Data split in 2 tables because of the large amount of deltas):



(a) Idle Frame Example 1



(b) Idle Frame Example 2



(c) Idle Frame Example 3

(d) Idle Frame Example 4

Figure 4.10: Representative snapshots of Ameca operating in **Idle Mode**. The robot performs randomized head and facial movements, including variable-length eye blinks, simulating human-like behavior.

Table 4.5: Test Results – Idle Scenario with 1s Tolerance (Summary)

Test ID	Scenario	Tolerance	GT Events	SUT Events	Match %	Outcome
21	Idle	1.0s	9	7	77.8%	Failed
22	Idle	1.0s	7	7	100.0%	Passed
23	Idle	1.0s	7	6	85.7%	Failed
24	Idle	1.0s	7	7	100.0%	Passed
25	Idle	1.0s	7	6	85.7%	Failed
26	Idle	1.0s	9	7	77.8%	Failed
27	Idle	1.0s	7	7	100.0%	Passed
28	Idle	1.0s	7	6	85.7%	Failed
29	Idle	1.0s	7	7	100.0%	Passed
30	Idle	1.0s	8	6	75.0%	Failed

Table 4.6: Temporal Deltas for Idle Scenario with 1s Tolerance

Test ID	Δ_{Start} [s]	Δ_{End} [s]
21	0.31 / 0.25 / 0.45 / 0.28 / 0.29 / 0.46 / 0.47	0.54 / 0.55 / 0.55 / 0.52 / 0.56 / 0.56 / 0.51
22	0.34 / 0.31 / 0.35 / 0.19 / 0.29 / 0.34 / 0.37	0.54 / 0.55 / 0.53 / 0.56 / 0.56 / 0.56 / 0.58
23	0.30 / 0.31 / 0.35 / 0.32 / 0.29 / 0.30	0.51 / 0.51 / 0.55 / 0.55 / 0.56 / 0.56
24	0.31 / 0.28 / 0.35 / 0.46 / 0.46 / 0.30 / 0.34	0.55 / 0.55 / 0.55 / 0.56 / 0.53 / 0.29 / 0.45
25	0.47 / 0.45 / 0.24 / 0.32 / 0.31 / 0.29	0.53 / 0.57 / 0.54 / 0.52 / 0.52 / 0.53
26	0.30 / 0.46 / 0.34 / 0.31 / 0.44 / 0.48 / 0.47	0.54 / 0.53 / 0.65 / 0.54 / 0.51 / 0.52 / 0.61
27	0.23 / 0.46 / 0.47 / 0.30 / 0.31 / 0.41 / 0.35	0.54 / 0.60 / 0.64 / 0.52 / 0.59 / 0.56 / 0.58
28	0.37 / 0.45 / 0.24 / 0.35 / 0.31 / 0.29	0.53 / 0.55 / 0.54 / 0.56 / 0.56 / 0.53
29	0.23 / 0.36 / 0.47 / 0.54 / 0.46 / 0.30 / 0.34	0.55 / 0.55 / 0.55 / 0.56 / 0.53 / 0.29 / 0.45
30	0.30 / 0.28 / 0.35 / 0.46 / 0.29 / 0.34	0.55 / 0.55 / 0.54 / 0.56 / 0.56 / 0.58

Table 4.7: Test Results – Idle Scenario with 1.5s Tolerance (Summary)

Test ID	**Scenario**	**Tolerance**	**GT Events**	**SUT Events**	**Match %**	**Outcome**
31	Idle	1.5s	6	6	100.0%	Passed
32	Idle	1.5s	7	6	85.7%	Failed
33	Idle	1.5s	8	7	87.5%	Failed
34	Idle	1.5s	7	7	100.0%	Passed
35	Idle	1.5s	7	6	85.7%	Failed
36	Idle	1.5s	6	6	100.0%	Passed
37	Idle	1.5s	6	6	100.0%	Passed
38	Idle	1.5s	6	6	100.0%	Passed
39	Idle	1.5s	6	6	100.0%	Passed
40	Idle	1.5s	10	7	70.0%	Failed

Table 4.8: Temporal Deltas – Idle Scenario with 1.5s Tolerance (Tests 1–10)

Test ID	Δ_{Start} [s]	Δ_{End} [s]
31	0.28 / 0.32 / 0.33 / 0.44 / 0.45 / 0.44	0.55 / 0.52 / 0.54 / 0.54 / 0.51 / 0.54
32	0.31 / 0.31 / 0.31 / 0.49 / 0.45 / 0.36	0.54 / 0.54 / 0.55 / 0.52 / 0.51 / 0.53
33	0.46 / 0.30 / 0.44 / 0.30 / 0.35 / 0.31 / 0.42	0.52 / 0.57 / 0.56 / 0.54 / 0.53 / 0.56 / 0.53
34	0.42 / 0.25 / 0.35 / 0.43 / 0.29 / 0.34 / 0.46	0.56 / 0.53 / 0.55 / 0.53 / 0.29 / 0.45 / 0.45
35	0.31 / 0.29 / 0.36 / 0.32 / 0.31 / 0.33	0.55 / 0.56 / 0.54 / 0.56 / 0.56 / 0.58
36	0.28 / 0.32 / 0.33 / 0.44 / 0.45 / 0.44	0.55 / 0.52 / 0.54 / 0.54 / 0.51 / 0.54
37	0.33 / 0.32 / 0.33 / 0.44 / 0.45 / 0.44	0.54 / 0.52 / 0.55 / 0.54 / 0.56 / 0.55
38	0.28 / 0.32 / 0.34 / 0.45 / 0.46 / 0.45	0.56 / 0.55 / 0.53 / 0.54 / 0.52 / 0.55
39	0.34 / 0.32 / 0.35 / 0.42 / 0.45 / 0.46	0.54 / 0.53 / 0.54 / 0.56 / 0.56 / 0.55
40	0.30 / 0.33 / 0.31 / 0.44 / 0.45 / 0.41 / 0.47	0.54 / 0.57 / 0.55 / 0.54 / 0.53 / 0.56 / 0.58

4.3.4 LiveLink Scenario

LiveLink is a software tool that, through the use of an iPad, enables the identification of a multitude of facial landmarks in real time, in a manner similar to Mediapipe. This makes it possible to capture human facial expressions and transmit them to a robotic avatar 4.13.

Figure 4.11: Real face tracking

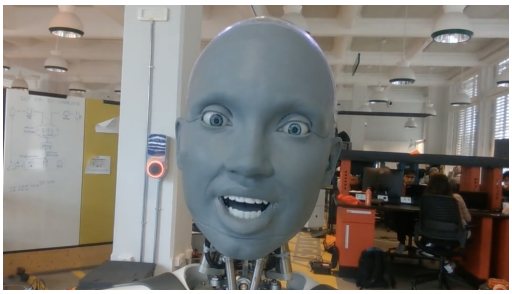


Figure 4.12: Livelink simulated on Ameca

Figure 4.13: Facial tracking simulation with LiveLink: comparison between real human input and Ameca output

Thanks to this feature, it is possible to reproduce facial movements in real time on Ameca. This is achieved through a `socket` connection that streams facial data in real time, allowing Ameca to mirror human expressions as they occur. This feature was implemented as part of this thesis work [20].

LiveLink also includes a `Recorded` mode, which allows recorded sequences of expressions to be saved in CSV format and later replayed. This functionality enabled the creation of a library of 5 facial animations, all derived from real human expressions recorded through the iPad, and thus considered realistic.

Based on this LiveLink scenario, we conducted:

- **10 tests with tolerance = 1.0s**
- **10 tests with tolerance = 1.5s**

For each test, one of the five sequences was randomly selected from the animation library by choosing a random number from 1 to 5.

It is important to note that eye closures are not guaranteed in every test, since the animation sequences are randomly chosen and not designed to always include blinking. This increases the variability and consistency of the testing process, making it a more reliable real-world validation.

Table 4.9: Test Results – LiveLink Scenario with 1.0s Tolerance (with Temporal Deltas)

Test ID	Scenario	Tol.	GT Ev.	SUT Ev.	Match %	Δ_{Start} [s]	Δ_{End} [s]	Outcome
41	LiveLink	1.0s	1	1	100.0%	0.49	0.52	Passed
42	LiveLink	1.0s	1	1	100.0%	0.50	0.54	Passed
43	LiveLink	1.0s	1	1	100.0%	0.52	0.51	Passed
44	LiveLink	1.0s	1	1	100.0%	0.50	0.54	Passed
45	LiveLink	1.0s	0	2	0.0%	–	–	Failed
46	LiveLink	1.0s	1	1	100.0%	0.52	0.53	Passed
47	LiveLink	1.0s	1	1	100.0%	0.51	0.64	Passed
48	LiveLink	1.0s	1	1	100.0%	0.48	0.51	Passed
49	LiveLink	1.0s	0	2	0.0%	–	–	Failed
50	LiveLink	1.0s	1	1	100.0%	0.48	0.55	Passed

Table 4.10: Test Results – LiveLink Scenario with 1.5s Tolerance (with Temporal Deltas)

Test ID	Scenario	Tol.	GT Ev.	SUT Ev.	Match %	Δ_{Start} [s]	Δ_{End} [s]	Outcome
51	LiveLink	1.5s	0	2	0.0%	–	–	Failed
52	LiveLink	1.5s	1	1	100.0%	0.50	0.54	Passed
53	LiveLink	1.5s	0	2	0.0%	–	–	Failed
54	LiveLink	1.5s	1	1	100.0%	0.53	0.56	Passed
55	LiveLink	1.5s	1	1	100.0%	0.48	0.65	Passed
56	LiveLink	1.5s	1	1	100.0%	0.49	0.62	Passed
57	LiveLink	1.5s	1	1	100.0%	0.51	0.52	Passed
58	LiveLink	1.5s	2	2	100.0%	0.50 / 0.50	0.51 / 0.59	Passed
59	LiveLink	1.5s	1	1	100.0%	0.54	0.70	Passed
60	LiveLink	1.5s	1	1	100.0%	0.52	0.56	Passed

4.4 Analysis of Gaze Target Detection Scenarios

In this section, we discuss the methodology adopted for detecting and classifying gaze-related events in both the Ground Truth (GT) and the System Under Test (SUT). Unfortunately, no validation tests are available for the SUT in this context, due to the highly noisy and unstable values of Pitch and Yaw returned by MediaPipe's gaze estimation module. As a result, we restrict our analysis to the GT-based classification, while highlighting potential improvements in the concluding chapters.

This limitation stems from the nature of MediaPipe's 3D facial landmark tracking, which—while performant in ideal conditions—is affected by jitter and instability when applied to subtle gaze changes or head movements. In an ideal experimental setting, a dedicated Driver Monitoring System (DMS) with robust gaze tracking capabilities (e.g., the Deepware camera system) would allow for reliable event generation in the SUT and enable a full validation process.

4.4.1 Computation of GT Gaze Vectors

The detection and classification of gaze events in the GT rely on a weighted aggregation of actuator positions associated with the eye, head, and neck pitch and yaw motors. This process yields a global representation of the gaze orientation, which is then used to determine the attentional focus of the humanoid.

The computational process is as follows:

- Eye pitch and yaw values for both the left and right eyes are extracted from the grouped motor data: `Eye Pitch Left`, `Eye Pitch Right`, `Eye Yaw Left`, `Eye Yaw Right`.
- Similarly, pitch and yaw contributions from the head and neck are extracted: `Head Pitch`, `Neck Pitch`, `Head Yaw`.
- Eye pitch and yaw are averaged between left and right sides:

$$\texttt{eye\_pitch\_global} = \frac{\texttt{left} + \texttt{right}}{2}$$
$$\texttt{eye\_yaw\_global} = \frac{\texttt{left} + \texttt{right}}{2}$$

- The global pitch vector is computed by summing contributions from eye, head, and neck pitch:

$$\texttt{pitch\_global} = \texttt{eye\_pitch\_global} + \texttt{head\_pitch} + \texttt{neck\_pitch}$$

- The global yaw vector is computed by summing eye yaw and head yaw:

$$\texttt{yaw\_global} = \texttt{eye\_yaw\_global} + \texttt{head\_yaw}$$

These vectors, denoted as `Gaze_P_Global` and `Gaze_Y_Global`, are added to the GT dataset along with their associated timestamps:


```
{
    "Gaze_Y_Global" : {
        "min" : -70.0,
        "max" : 70.0,
        "num_samples" : 354,
        "positions" : [ -60.7300243377685547, -60.7298641204833984,...],
        "times" : [ 1.7424653895857291E9, 1.742465389615772E9,...],
    },
    "Gaze_P_Global" : {
        "min" : -70.0,
        "max" : 70.0,
        "num_samples" : 354,
        "positions" : [ 30.4560243377685547, 30.1358641204833984,...],
        "times" : [ 1.7424653895857291E9, 1.742465389615772E9,...],
    }
}
```


4.4.2 Classification via Attention Box

To determine whether the robot's gaze is focused on a meaningful area, we define a virtual "**Attention Box**" centered around the frontal view of the humanoid. Any gaze values falling inside this area are considered "Normal," while values outside are classified as "OutOfBox"—indicating a possible distraction.

This classification uses the following thresholds:

- Global Pitch (up/down): $[-20°, +20°]$
- Global Yaw (left/right): $[-20°, +20°]$

Although the absolute mechanical limits of pitch and yaw are broader (Pitch: $[-70°, +70°]$, Yaw: $[-70°, +70°]$), the defined ranges represent a plausible attentional zone that mimics the field of view during a human–machine interaction or attention task. Here a visual rappresentation 4.14.

Each of the two global gaze vectors (`Gaze_P_Global`, `Gaze_Y_Global`) is classified independently by iterating through each frame and assigning a label:

- **Normal:** if the value lies within $[-20, +20]$
- **OutOfBox:** if the value lies outside the threshold

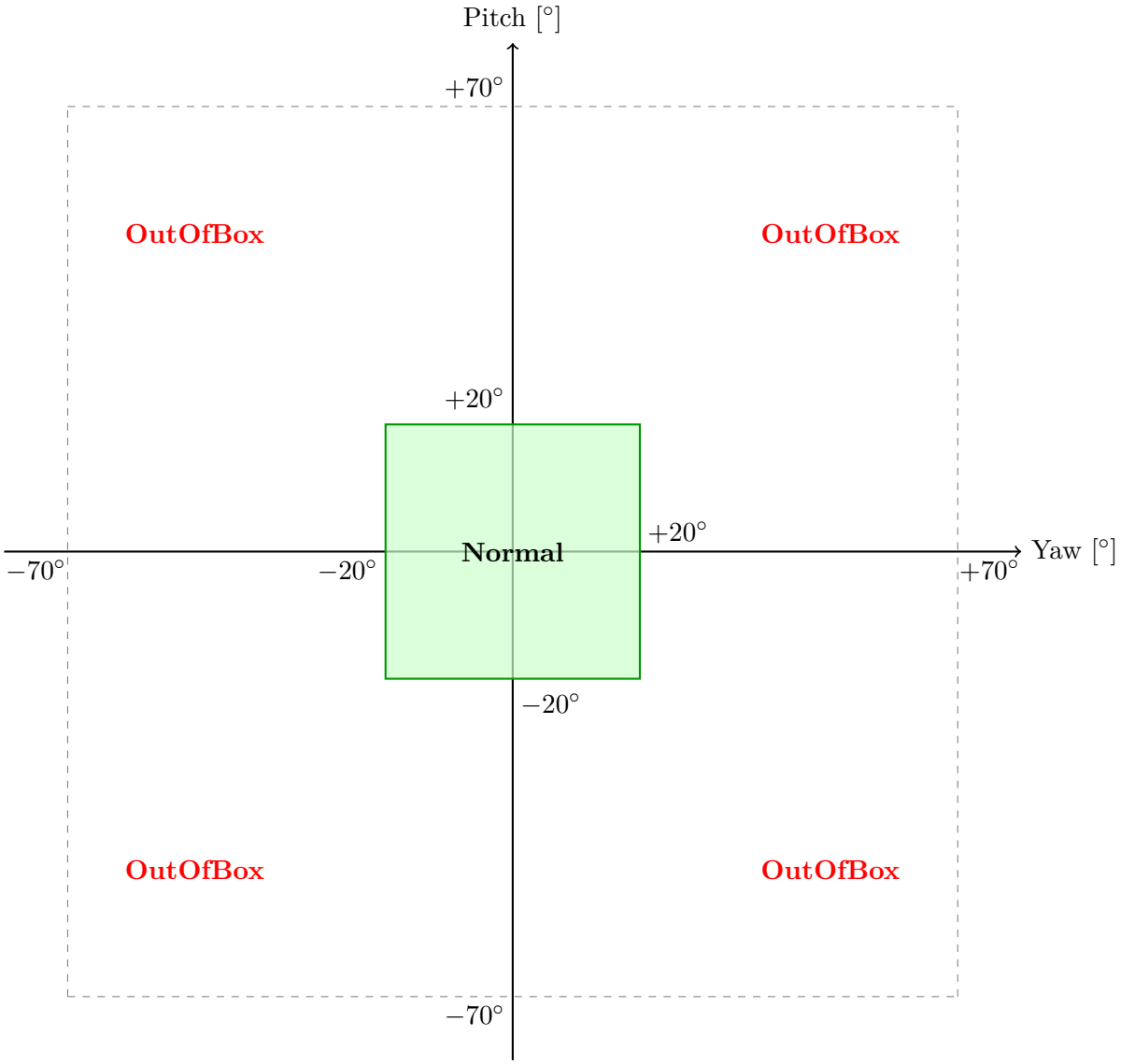


Figure 4.14: Gaze Attention Box in the Global Angular Space. Green box: normal gaze zone ($[-20°, +20°]$); outer areas: OutOfBox distraction zones.

Subsequently, a **Final Vector** is constructed by taking the union (logical OR) of the pitch and yaw classifications. This means that if either pitch *or* yaw falls outside the attention box at any given timestamp, the frame is classified as "OutOfBox." This conservative approach ensures that any form of distraction—vertical or horizontal—is appropriately flagged.

An example of the final result is:

```
"Final_Vectors": {
    "timestamps": [1741789577.170, 1741789577.202, 1741789577.235, ...],
    "labels": ["Normal", "Normal", "OutOfBox", "OutOfBox", ...]
```

```
}
```

4.4.3 Limitations on SUT Gaze Detection

The same detection logic was implemented for the SUT. However, as noted above, the results were deemed unreliable due to the unstable and noisy nature of the gaze values extracted via MediaPipe. The system exhibited frequent, erratic fluctuations in pitch and yaw estimates that did not correlate with actual eye movements, rendering any attempt at validation ineffective.

This significant limitation highlights the importance of adopting more robust gaze tracking solutions, such as hardware-based DMS systems. If integrated, such systems would allow the reproduction of controlled gaze sequences and their evaluation against GT annotations, paving the way for a full validation pipeline of gaze event detection.

This shortcoming is discussed in detail in the final chapter, "Limitations and Future Work," where we explore how the integration of a real-time DMS camera could significantly improve the reliability and scope of validation procedures.

4.5 Performance Evaluation and Discussion

This section presents a comprehensive evaluation of the system's performance across all scenarios—*Static_1*, *Static_2*, *Idle*, and *LiveLink*—under two temporal tolerance thresholds: 1.0s and 1.5s. The goal is to assess the reliability, precision, and generalizability of the proposed event validation framework, specifically in replicating and recognizing facial behaviors on the Ameca platform.

The analysis spans several dimensions, including:

- **Matching Accuracy:** The proportion of GT events correctly identified by the SUT.
- **Temporal Alignment:** Precision of onset and offset time between GT and SUT events (Δ_{Start}, Δ_{End}).
- **Error Distribution:** False Positives (FP) and False Negatives (FN).
- **Scenario Robustness:** Consistency and variability across different behavior types.
- **Metric Summary:** Precision, Recall, F1-Score, and Accuracy across all tests.

4.5.1 Matching Accuracy Across Scenarios and Tolerances

A consistent pattern emerges from the evaluation: increasing the temporal tolerance from 1.0s to 1.5s improves matching performance, especially in dynamic scenarios where minor timing discrepancies are more likely.

- **Static_1:** With 1.0s tolerance, only 60% of tests passed, mainly due to minor misalignments. Raising the tolerance to 1.5s resulted in 100% matches, confirming the need for flexibility in stricter conditions.

- **Static_2:** Achieved 100% match rates across all tests at both tolerances. This consistency is attributed to the deterministic, repeatable nature of the test design.
- **Idle:** This scenario introduced greater temporal variability. At 1.0s, only 40% of tests passed, while at 1.5s the success rate improved to 60%, demonstrating the benefit of more lenient tolerances in spontaneous behavior tracking.
- **LiveLink:** Representing realistic facial expressions, LiveLink achieved an 80% pass rate for both tolerances. However, occasional unmatched events resulted in false positives due to noise or unaligned GT annotations.

4.5.2 Temporal Delta Analysis

Temporal alignment between GT and SUT events was evaluated through start and end deltas. These metrics indicate how accurately the SUT replicates the timing of the GT-labeled events.

- **Start Deltas ($\Delta_{\mathbf{Start}}$):** Ranged from approximately 0.1–0.2s in Static scenarios to 0.5s in LiveLink.
- **End Deltas ($\Delta_{\mathbf{End}}$):** Generally remained between 0.5s and 0.7s, suggesting consistent termination latency on the SUT side.
- **Stability:** Static_2 and LiveLink showed tight delta distributions, confirming the robustness of the event replication process even in more complex expression sets.

4.5.3 Failure Case Analysis

Out of 60 total test cases, failure outcomes were broken down as follows:

- **1.0s Tolerance:** 14 failed tests.
- **1.5s Tolerance:** 6 failed tests.

Failures can be grouped into three categories:

1. **False Negatives (FN):** Events present in GT but not detected by SUT. Most frequent in Static_1 and Idle scenarios under stricter tolerance.
2. **False Positives (FP):** Events generated by SUT with no corresponding GT entries. Prominent in LiveLink due to expressive motion that may trigger false detections.
3. **Partial Matches:** In Idle tests, only a subset of GT events were correctly matched, yielding suboptimal match percentages.

4.5.4 False Positive and False Negative Distribution

Figures 4.15–4.16 and 4.17–4.18 illustrate the error distribution. Notably:

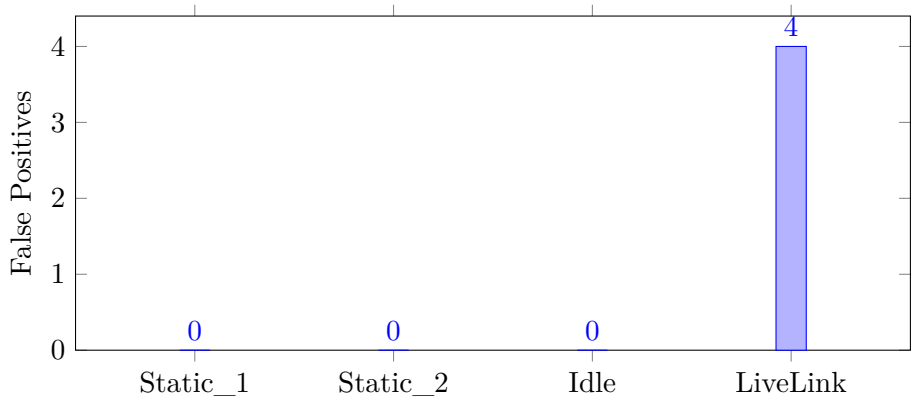


Figure 4.15: False Positives across all Scenarios – 1.0s Tolerance

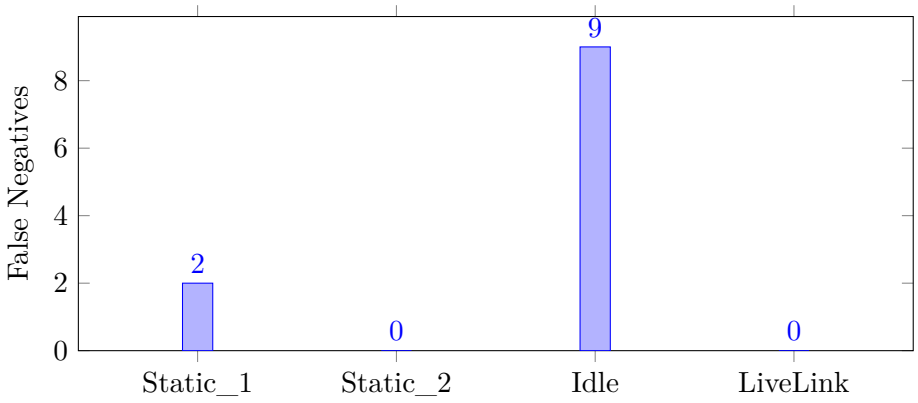


Figure 4.16: False Negatives across all Scenarios – 1.0s Tolerance

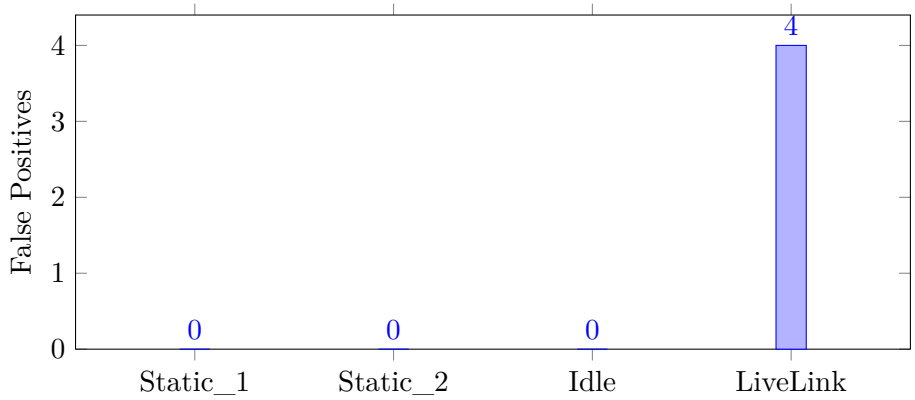


Figure 4.17: False Positives across all Scenarios – 1.5s Tolerance

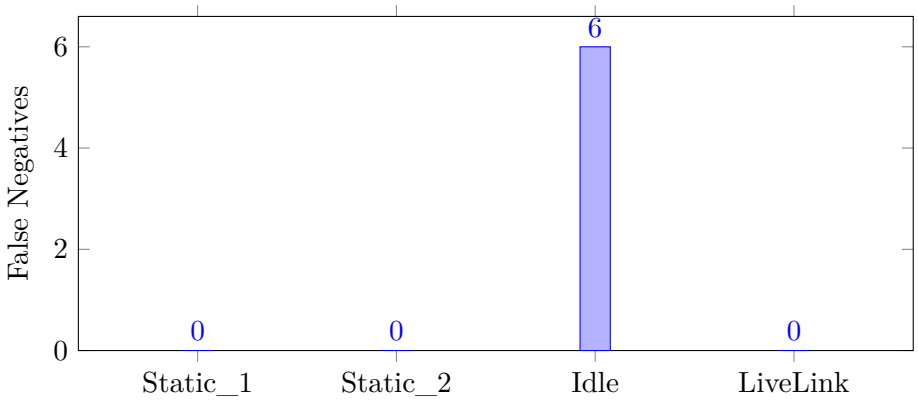


Figure 4.18: False Negatives across all Scenarios – 1.5s Tolerance

- No false positives were observed for *Static_1*, *Static_2*, or *Idle* under 1.0s tolerance.
- LiveLink presented consistent false positives under both tolerance levels.
- Idle scenario yielded the highest number of false negatives, especially under the 1.0s constraint.

To quantitatively assess the performance of the event detection system, we adopt a set of standard evaluation metrics widely used in classification and information retrieval tasks. These include **Precision**, **Recall**, **F1-Score**, and **Accuracy**. All metrics are derived from the following definitions:

- **True Positives (TP)**: Ground Truth (GT) events correctly detected by the System Under Test (SUT).
- **False Positives (FP)**: Events detected by the SUT that do not correspond to any GT event (i.e., over-detections).

- **False Negatives (FN)**: GT events that were not detected by the SUT (i.e., missed detections).
- **True Negatives (TN)**: Not applicable in this context, as events are detected and labeled only when present.

Precision measures the proportion of correctly identified events among all events detected by the SUT. A high precision means that the system generates few false alarms.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4.2}$$

Example: If the system detects 10 events and 9 of them are correct, the precision is 90%.

Recall indicates the system's ability to detect all relevant events that exist in the Ground Truth. It is also referred to as *sensitivity*.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4.3}$$

Example: If 10 events exist in the Ground Truth and the system detects 8 of them, recall is 80%.

The F1-Score is the harmonic mean of Precision and Recall. It balances the trade-off between missing relevant events (FN) and over-detecting non-existent ones (FP).

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.4}$$

Note: A high F1-Score requires both high precision and recall.

Accuracy refers to the ratio of correctly classified events over the total number of relevant Ground Truth events. In this context, it is adapted due to the absence of a meaningful notion of true negatives.

$$\text{Accuracy} = \frac{TP}{TP + FN} \tag{4.5}$$

Note: This simplified form is adopted since TN (True Negatives) are undefined in our scenario.

These metrics are computed separately for each scenario (Static_1, Static_2, Idle, LiveLink) and for each tolerance level (1.0s and 1.5s). Together, they provide a comprehensive picture of the system's performance, highlighting strengths in deterministic environments and challenges in spontaneous or naturalistic behaviors such as those in the Idle and LiveLink scenarios.

- **1.0s Tolerance:** Precision was consistently high (more than 0.83), while recall suffered in Static_1 and Idle due to missed detections.
- **1.5s Tolerance:** All scenarios showed strong recall, pushing F1-scores closer to 1.0.

The full metric breakdown is shown in Tables 4.11 and 4.12, with supporting bar charts in Figures 4.19 and 4.20.

Table 4.11: Evaluation Metrics by Scenario (Tolerance = 1.0s)

Scenario	**Precision**	**Recall**	**F1-Score**	**Accuracy**
Static_1	1.00	0.60	0.75	60.0%
Static_2	1.00	1.00	1.00	100.0%
Idle	1.00	0.80	0.89	87.0%
LiveLink	0.83	1.00	0.91	80.0%

Table 4.12: Evaluation Metrics by Scenario (Tolerance = 1.5s)

Scenario	**Precision**	**Recall**	**F1-Score**	**Accuracy**
Static_1	1.00	1.00	1.00	100.0%
Static_2	1.00	1.00	1.00	100.0%
Idle	1.00	0.91	0.95	0.91%
LiveLink	0.83	1.00	0.91	80.0%

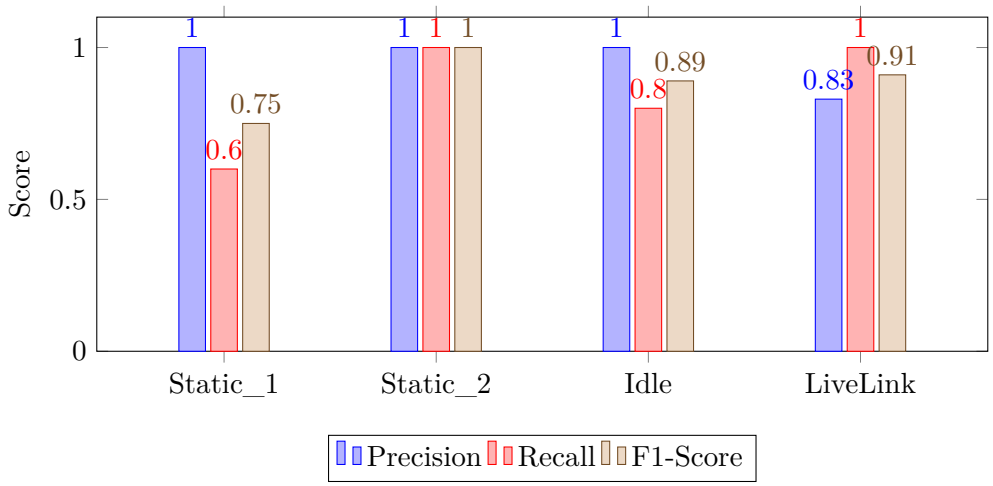


Figure 4.19: Precision, Recall, and F1-Score per Scenario – 1.0s Tolerance

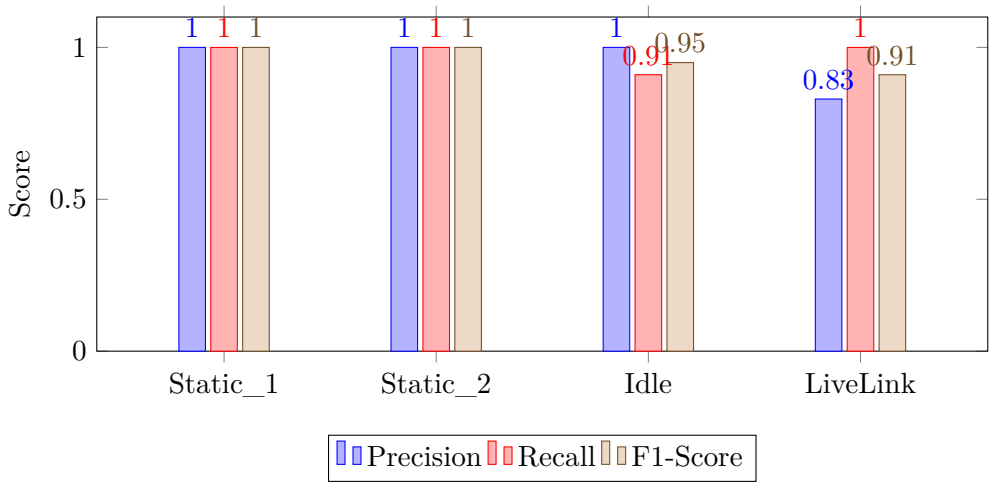


Figure 4.20: Precision, Recall, and F1-Score per Scenario – 1.5s Tolerance

4.5.5 Cumulative Metrics Summary

Table 4.13 provides aggregated match rates, delta values, and pass percentages. It clearly demonstrates the benefit of adopting a 1.5s tolerance, particularly for more variable scenarios.

Table 4.13: Aggregate Performance Metrics by Scenario

Scenario	**Tolerance**	**Avg Match %**	**Avg Δ_{Start} [s]**	**Avg Δ_{End} [s]**	**Pass Rate**
Static_1	1.0s	60.0%	0.142	0.973	60%
Static_1	1.5s	100.0%	0.109	0.995	100%
Static_2	1.0s	100.0%	0.280	0.515	100%
Static_2	1.5s	100.0%	0.328	0.601	100%
Idle	1.0s	87.0%	0.33	0.54	40%
Idle	1.5s	92.3%	0.34	0.54	60%
LiveLink	1.0s	80.0%	0.50	0.54	80%
LiveLink	1.5s	80.0%	0.51	0.57	80%

4.5.6 Visual Comparison of Matching Performance

Figure 4.21 compares match rates across scenarios for each tolerance, offering an immediate view of system sensitivity to stricter timing constraints.

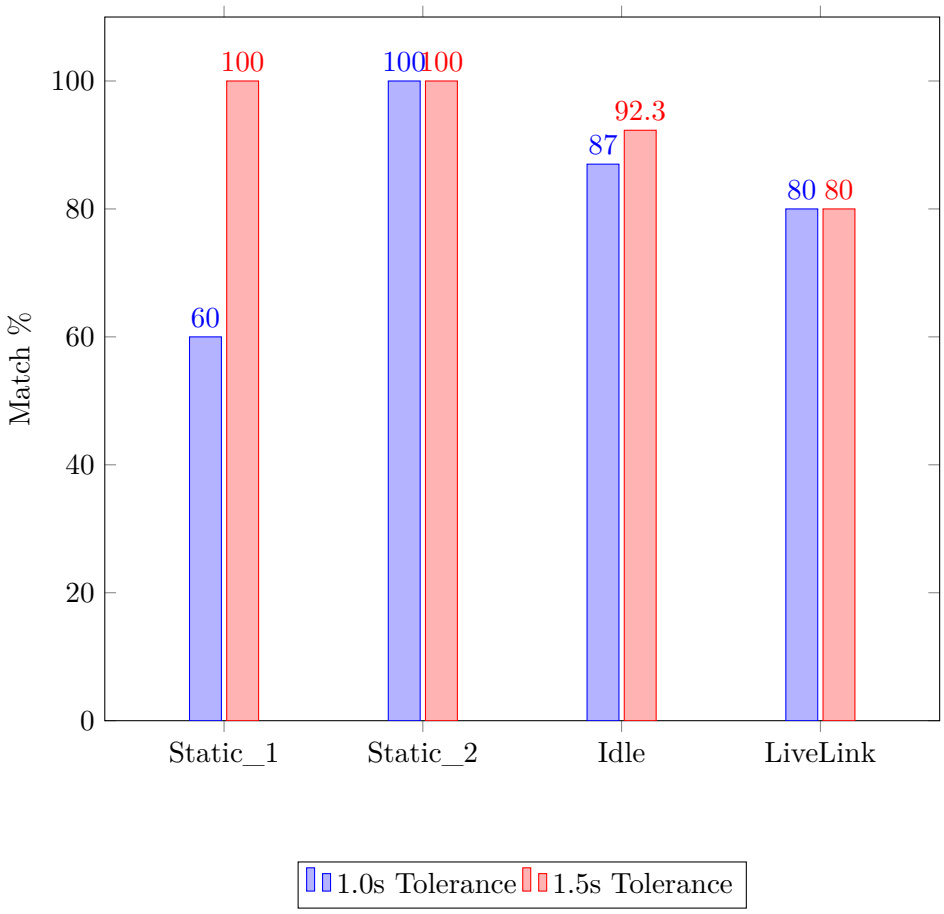


Figure 4.21: Average Match Percentage across Scenarios for Tolerances 1.0s and 1.5s

4.5.7 Limitations and Future Work

While the validation results are promising, several limitations must be acknowledged:

- **Fixed tolerance window:** Static thresholds may not suit all event types equally. Adaptive tolerance based on event duration or dynamics should be explored.
- **No gaze validation:** As discussed in Section 4.4, SUT gaze data was too noisy to support meaningful evaluation. Integrating a robust DMS system (e.g., Deepware) could enable full multimodal validation in future iterations.
- **Residual false positives:** Particularly in expressive scenarios like LiveLink, noise or spontaneous micro-expressions can result in misclassifications. Further filtering or signal post-processing may help.

In conclusion, the proposed evaluation framework offers strong temporal fidelity and accurate event alignment. Its modularity and interpretability make it a solid foundation for future expansion into real-time multimodal human–robot interaction.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

This thesis presented the design, implementation, and validation of a novel framework for the automatic evaluation of Driver Monitoring Systems (DMS) in the context of visual distraction detection, using the Ameca humanoid robot as a synthetic driver.

The proposed architecture is modular and distributed, capable of synchronizing and processing multimodal data in real-time — including visual streams (RGB and ToF), internal robot motor data, and depth information. This infrastructure enabled the controlled simulation of distraction events and their corresponding detection, under a wide range of conditions. Ground Truth (GT) data, generated from Ameca's internal motor states, was compared against outputs from the System Under Test (SUT), which relied on MediaPipe for blink and gaze detection.

The entire pipeline — from data acquisition to event classification and validation — was engineered to support precise temporal alignment, robust labeling, and structured output through JSON-based validation logs. Tolerance-based temporal matching (1.0s and 1.5s) was introduced to assess the flexibility and resilience of the system in face of detection delays or inaccuracies.

Validation was performed across four representative behavioral scenarios of increasing complexity:

- **Static_1 and Static_2:** Controlled sequences of prolonged blinks.
- **Idle:** Spontaneous behavior from Ameca with variable blink durations and intervals.
- **LiveLink:** Reproduction of human-derived facial expressions through motion retargeting.

The experimental results confirmed the effectiveness of the system in detecting visual distraction events with high temporal accuracy and semantic reliability. Notably:

- **Static scenarios** yielded near-perfect or perfect matching rates, especially under 1.5s tolerance.

- **Idle** tests, despite their inherent variability, showed good alignment with GT data and revealed the system's adaptability to naturalistic conditions.

- **LiveLink** demonstrated strong robustness in replicating and validating human-like expressions, despite occasional false positives arising from expressive motion noise.

An additional component focused on gaze detection using pitch and yaw angles derived from both eye and head motor groups. Although GT gaze computation was robust (via the Gaze_P_Global and Gaze_Y_Global vectors), SUT data from MediaPipe proved too noisy to support reliable validation. This component was excluded from quantitative evaluation, marking a notable limitation in the system's current scope.

In summary, this thesis introduced a reproducible, robot-centric framework for the structured validation of attention-related behaviors in driver-like conditions. The methodology offers a scalable and ethical alternative to human subject testing and aligns well with upcoming DMS certification protocols and Euro NCAP standards.

5.2 Limitations and Future Work

While the framework demonstrates strong capabilities, several limitations emerged, which point towards important avenues for future research and development.

Limitations

- **Fixed tolerance thresholds:** The validation relied on static temporal thresholds (1.0s, 1.5s). A dynamic approach — adjusting the tolerance based on event duration, type, or motion velocity — could yield more nuanced and fair assessments.

- **Inadequate gaze validation:** Due to the high noise levels in MediaPipe's gaze angle estimations, validation of gaze events could not be performed. This significantly restricted the scope to blink-only validation.

- **False positives in expressive scenarios:** Particularly in LiveLink, the system occasionally interpreted facial expression noise as valid events, leading to false positives. More advanced post-processing techniques or confidence-based filtering could mitigate this issue.

Future Work

Building upon the current infrastructure, several promising directions for future research can be identified:

1. **Creation of a benchmark dataset:** With the existing data acquisition infrastructure, it would be possible to generate a high-quality, multimodal dataset (including RGB, depth, ToF, GT events, and SUT outputs) with rich metadata and annotations. This could serve as a benchmark for future DMS validation research.

2. **Integration of a certified DMS camera (e.g., Deepware):** Replacing the current SUT with a production-grade DMS system would significantly enhance the quality of gaze detection and allow validation of a wider range of attention-related behaviors.

3. **Adaptive and closed-loop testing:** Future iterations could incorporate real-time feedback mechanisms, where Ameca's behavior dynamically responds to detection outcomes. This would enable the evaluation of DMS systems in interactive, adaptive settings.

4. **Expansion to additional DMS features:** The framework could be extended to include new validation targets such as cognitive state monitoring, drowsiness detection, head pose estimation, or facial emotion analysis. Ameca's expressivity makes it particularly suitable for such multimodal validations.

Final Remarks

This thesis has introduced a comprehensive and scalable methodology for validating DMS functionalities using humanoid robotics and multi-sensor synchronization. The proposed system not only achieved reliable performance across diverse behavioral scenarios but also lays the groundwork for future developments in the field of autonomous vehicle safety and human-robot interaction.

By combining replicability, real-time adaptability, and structured validation logic, the work contributes toward more ethical, robust, and standardized approaches to attention monitoring and behavioral validation in critical safety domains.

List of Figures

List of Tables

Bibliography

[1] World Health Organization. Who: Road safety, 2025. URL https://www.who.int/health-topics/road-safety#tab=tab_1. Accessed: 10 February 2025.

[2] European Commission. Erso synthesis 2015 – driver distraction, 2015. URL https://road-safety.transport.ec.europa.eu/system/files/2021-07/ersosynthesis2015-driverdistraction25_en.pdf.

[3] National Highway Traffic Safety Administration (NHTSA). Crashstats brief statistical summary in 2022, 2024. URL https://crashstats.nhtsa.dot.gov/Api/Public/Publication/813559.

[4] European Commission. Frequency of fatigue-related crashes. URL https://road-safety.transport.ec.europa.eu/european-road-safety-observatory/statistics-and-analysis-archive/fatigue/frequency-fatigue-related-crashes_en.

[5] Nicole Lamond and Drew Dawson. Lamond n, dawson d. quantifying the performance impairment associated with fatigue. j sleep res 8: 255-262. *Journal of sleep research*, 8:255–62, 01 2000. doi: 10.1046/j.1365-2869.1999.00167.x.

[6] European Parliament and the Council of the European Union. Regulation (eu) 2019/2144, 2023. URL https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32019R2144. Adopted on 27 November 2019.

[7] European Parliament and the Council of the European Union. Supplement regulation (eu) 2019/2144, 2023. URL https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=PI_COM:C(2023)4523#footnote3. Adopted on 13 July 2023.

[8] European New Car Assessment Programme (Euro NCAP). Official website. https://www.euroncap.com/en.

[9] European Transport Safety Council (ETSC). New warning on infotainment screen distraction, 2024. URL https://etsc.eu/new-warning-on-infotainment-screen-distraction/.

[10] Balakrishnan Pushpa, D. Harish, Kumar Sunil, and S. Rohith. Drowsiness detection and its analysis of brain waves using electroencephalogram. *i-manager's Journal on Digital Signal Processing*, 2024. doi: 10.26634/jdp.12.2.21444.

[11] Annu George Mavely, J. E. Judith, P. A. Sahal, and Steffy Ann Kuruvilla. Eye gaze tracking based driver monitoring system. In *2017 IEEE International Conference on Circuits and Systems (ICCS)*, 2017. doi: 10.1109/ICCS1.2017.8326022.

[12] Yulan Liang, John Lee, and Lora Yekhshatyan. How dangerous is looking away from the road? algorithms predict crash risk from glance patterns in naturalistic driving. *Human factors*, 2012. doi: 10.1177/0018720812446965.

[13] Qiang Ji and Xiaojie Yang. Real-time eye, gaze, and face pose tracking for monitoring driver vigilance. *Real-Time Imaging*, 2002. ISSN 1077-2014. doi: https://doi.org/10.1006/rtim.2002.0279. URL https://www.sciencedirect.com/science/article/pii/S1077201402902792.

[14] T. E. Hutchinson. Eye movement detection with improved calibration and speed. U.S. Patent 4950069, April, 1990.

[15] European New Car Assessment Programme (Euro NCAP). Assessment protocol - safe driving. https://www.euroncap.com/media/80158/euro-ncap-assessment-protocol-sa-safe-driving-v104.pdf, 2024.

[16] Anaïs Halin, Jacques G. Verly, and Marc Van Droogenbroeck. Survey and synthesis of state of the art in driver monitoring. *Sensors*, 21(16), 2021. ISSN 1424-8220. doi: 10.3390/s21165558. URL https://www.mdpi.com/1424-8220/21/16/5558.

[17] Xinyue Miao, Chengqi Xue, Xian Li, and Lichun Yang. A real-time fatigue sensing and enhanced feedback system. *Information*, 13(5), 2022. ISSN 2078-2489. doi: 10.3390/info13050230. URL https://www.mdpi.com/2078-2489/13/5/230.

[18] Dohun Kim, Hyukjin Park, Tonghyun Kim, et al. Real-time driver monitoring system with facial landmark-based eye closure detection and head pose recognition. Preprint (Version 1), available at Research Square, aug 2023. https://doi.org/10.21203/rs.3.rs-3223799/v1.

[19] V. Kazemi and J. Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1867–1874, 2014.

[20] Vincenzo Micciche. *Validation Toolchain for Advanced Driver Distraction Warning Systems*. PhD thesis, Politecnico di Torino, 2024.

[21] Engineered Arts Ltd. Ameca user documentation: Desktop. https://docs.engineeredarts.co.uk/en/user/ameca_desktop, .

[22] Raspberry Pi Foundation. Raspberry pi official website. https://www.raspberrypi.com.

[23] Intel Corporation. Intel realsense depth camera d435. https://www.intelrealsense.com/depth-camera-d435/. Accessed: 14 February 2025.

[24] Arducam. Time-of-flight (tof) camera for raspberry pi. https://www.arducam.com/time-of-flight-camera-raspberry-pi/. Accessed: 14 February 2025.

[25] Engineered Arts Ltd. Documentation portal. https://docs.engineeredarts.co.uk, . Accessed: 14 February 2025.

[26] Devarakonda Sruthi, Avanaganti Amulya Reddy, G. Sai Siddaharth Reddy, and Shilpa Shesham. Driver drowsiness detection system using deep learning. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, 11(4), April 2023. ISSN 2321-9653. doi: 10.22214/ijraset.2023.50345. IC Value: 45.98, SJ Impact Factor: 7.538.