Available online www.jsaer.com

Journal of Scientific and Engineering Research, 2025, 12(4):107-113

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

AI-Based Driver Monitoring System for Drowsiness, Yawning, and Distraction Detection with Real-Time Alerts

Mr. G. Sai Goutham[1], Dr. B. Vijaya Krishna[2], A. Ganesh[3], S. Vinay Vardhan[4], B. Susmitha[5], D. Hari Krishna[6]

[1,2]Asst. Professor, Electrical & Electronics Engineering, Bapatla Engineering College, Bapatla, AP
[3,4,5,6]Electrical & Electronics Engineering, Bapatla Engineering College, Bapatla, AP
Email: [1]saigotham.golive@becbapatla.ac.in, [2]boyina.vijayakrishna@becbapatla.ac.in, [3]adabalaganesh.a@gmail.com, [4]suvvarivinayvardhan@gmail.com, [5]billasusmithae@gmail.com, [6]harikrishnadukka8@gmail.com

Abstract: In modern vehicle systems, Driver fatigue and distraction are significant contributors to road accidents globally. This paper presents an AI-based Driver Monitoring System (DMS) designed to detect early signs of drowsiness, yawning, and distraction in drivers. The proposed system leverages advanced computer vision algorithms, including facial recognition, eye closure detection, and yawning analysis, to monitor driver behavior in real time. Implemented on a PC with an Arduino Mega 2560 for alert management, the system issues visual and auditory alerts through an IVN test board and communicates via CAN messages to the TS Master platform for visualization. Experimental results demonstrate face recognition, eye closure, and yawning detection accuracies of 80%, 75%, and 85%, respectively, with alert triggers activated within 5 seconds. This integrated solution aims to enhance vehicular safety by proactively addressing driver fatigue and distraction.

Keywords: Driver Monitoring System, AI, Drowsiness Detection, Yawning Detection, Real-Time Alerts, CAN Communication, Automotive Safety.

1. Introduction

Every year, road accidents claim millions of lives, and studies have consistently shown that driver fatigue and distraction are among the leading causes. Despite major strides in improving vehicle safety features, most traditional systems are reactive — they work to minimize damage after a crash has occurred. What remains critically needed is a way to proactively detect unsafe driving conditions and alert the driver before it’s too late.

Driver Monitoring Systems (DMS) offer a promising solution by constantly observing the driver’s behavior to detect early signs of drowsiness or distraction. These systems have become an integral part of the safety framework in modern vehicles, particularly with the rise of Advanced Driver Assistance Systems (ADAS) and semi-autonomous driving technologies. An effective DMS can make the difference between a safe journey and a serious accident.

Early approaches to driver monitoring relied heavily on invasive methods like heart rate monitoring, EEG, and infrared eye tracking. While accurate, these techniques were often expensive, complex, and impractical for widespread use. Fortunately, the rapid advancement of computer vision and artificial intelligence (AI) technologies has made non-
intrusive, camera-based monitoring systems not only feasible but highly effective. Today, with just a standard camera and powerful algorithms, it's possible to detect subtle cues like eye closure, yawning, and facial orientation to assess the driver's alertness.

In this paper, we propose a practical AI-based Driver Monitoring System that combines real-time face detection, eye closure analysis, and yawning detection to monitor the driver's condition. Using cutting-edge libraries such as OpenCV, dlib, and Mediapipe, the system processes live video streams and intelligently identifies signs of fatigue or distraction. Upon detection, it sends instant visual and audio alerts through an Arduino-controlled IVN test board, ensuring that the driver is immediately made aware of any dangerous behavior.

Beyond just local alerts, the system integrates with an automotive CAN (Controller Area Network) bus to transmit detection events to TS Master software for real-time monitoring and visualization. This makes the solution scalable — suitable not only for private vehicles but also for fleet management systems where monitoring multiple drivers is critical.

2. Literature review

Over the past two decades, researchers and automotive engineers have worked extensively to develop systems that can monitor a driver's state and help reduce accidents caused by fatigue or distraction. Traditional methods often involved physiological monitoring, using sensors to track heart rate, brain activity (EEG), or eyelid movement (EOG). While these techniques can provide highly accurate measurements, they tend to be invasive, expensive, and impractical for deployment in mass-market vehicles.

As a result, the focus shifted toward vision-based approaches, leveraging the rapid advancements in computer vision and machine learning. Early systems primarily used infrared cameras to track eye closure or head movements. However, these systems faced challenges under varying lighting conditions and often required specialized hardware, limiting their scalability.

With the rise of deep learning, facial landmark detection and recognition techniques have become significantly more robust. [1] Xu et al. (2019) proposed a deep learning-based model for drowsiness detection that showed significant improvement over classical image processing methods, particularly in complex real-world environments. Their work demonstrated that neural networks could reliably recognize subtle facial cues associated with fatigue, even in different lighting or background conditions.

Similarly, Smith et al. (2020) developed a real-time yawning detection system based on facial landmark tracking. Their approach focused on monitoring the mouth aspect ratio over time to detect yawning events accurately. They showed that lightweight models could be effectively deployed for real-time monitoring, even on hardware with limited computational resources.

In parallel, research in face recognition for driver monitoring has evolved. Zhang and Li (2018) explored various algorithms and emphasized the importance of driver identification in multi-occupant vehicles. Their findings highlighted that correctly identifying the driver, especially when multiple people are present, is crucial for targeting the monitoring system appropriately.

Despite these advancements, many existing systems face challenges related to response time, system integration, and affordability. Some models require heavy computational power, making them unsuitable for real-time vehicle deployment. Others lack integration with vehicular communication networks, limiting their utility in fleet management or centralized monitoring scenarios.

The system proposed in this paper builds on these foundations, addressing the need for a lightweight, real-time, and easily integrable Driver Monitoring System. It combines proven techniques — face recognition, eye closure analysis, and yawning detection — with a practical embedded communication strategy using CAN bus protocols. This enables not only immediate alerting inside the vehicle but also external monitoring via centralized platforms like TS Master.

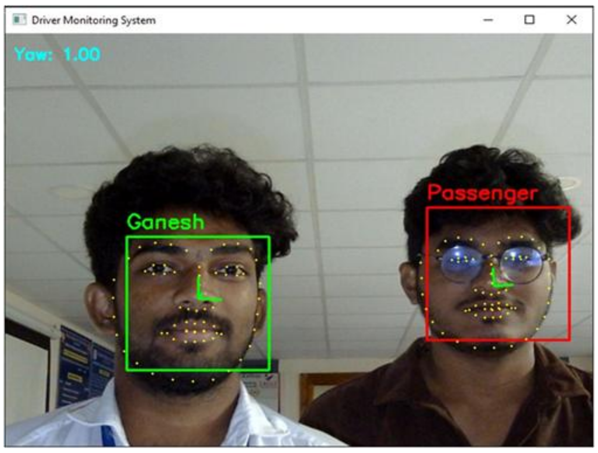
By bridging the gap between advanced detection and practical deployment, the proposed system aims to offer a comprehensive, scalable solution for improving driver safety in modern transportation systems.

3. Driver Monitoring System

The proposed Driver Monitoring System (DMS) integrates real-time computer vision techniques with embedded systems to monitor the driver's state efficiently and reliably.

The system first performs face detection and recognition using face_recognition and Mediapipe frameworks. When multiple faces are detected, the first face is automatically designated as the driver to ensure consistent monitoring.

Following face identification, the system continuously tracks the driver's eye movements. Using facial landmarks, it calculates the Eye Aspect Ratio (EAR) in real time. A sustained drop in the EAR below 0.25 for over 5 seconds is flagged as a strong indicator of drowsiness, prompting the system to prepare an alert.
In parallel, yawning detection is performed by monitoring the Mouth Aspect Ratio (MAR). If the driver yawns three times within a 60-second window, the system classifies it as a sign of fatigue and triggers a warning.
When a critical event is detected, an alert mechanism is activated through an Arduino Mega 2560. Visual (LED) and audio (buzzer) alerts are generated on an IVN test board to capture the driver's attention immediately.
To support broader monitoring and data logging, the system also transmits alert events over a Controller Area Network (CAN) bus to the TS Master platform. This allows for real-time visualization of driver behavior and facilitates future analysis, especially in fleet or multi-vehicle setups.



4. System Design and Implementation

The DMS (Driver Monitoring System) software was developed using Python, incorporating powerful libraries like OpenCV, dlib, and Mediapipe for facial recognition, image processing, and real-time facial mesh tracking. The hardware setup includes a high-definition USB camera for capturing the driver's face, a PC with an Intel Core i7 processor to handle the software, and an Arduino Mega 2560 programmed in C for managing the embedded systems. The system also integrates an IVN Test Board (Tosun 1001) for in-vehicle network testing and a CAN Bus Transceiver and Analyzer to facilitate communication with the vehicle.
Running at a frame rate of 20 FPS, the system ensures a balance between detection accuracy and real-time performance. Detection thresholds are set to monitor critical conditions: an Eye Aspect Ratio (EAR) below 0.25 indicates eye closure, while a Mouth Aspect Ratio (MAR) above 0.75 signals yawning. Additionally, if the driver yawns three times within 60 seconds, it triggers a critical alert. These features help maintain driver safety by detecting signs of fatigue or distraction, providing real-time monitoring during driving.

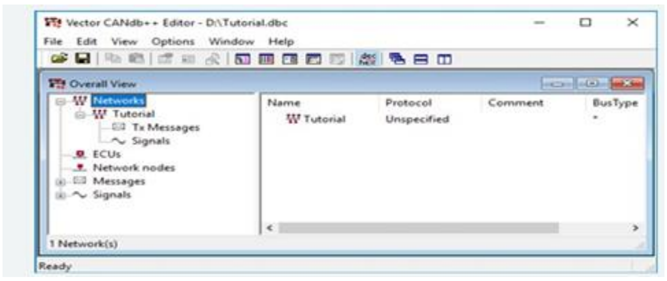
5. Tools and Libraries

For the development of the AI-based Driver Monitoring System (DMS), various software libraries, frameworks, and tools were utilized to implement different stages of the system, including image processing, facial recognition, eye detection, gaze tracking, and CAN communication. The key libraries and tools used in this study are:
OpenCV: An open-source computer vision library that was employed for real-time image and video processing, face and eye detection, and general image manipulation. OpenCV was crucial for detecting faces in the input video stream and performing various image processing operations.
dlib: A toolkit for machine learning and computer vision tasks. In this system, it was used for detecting facial landmarks, such as the eye and mouth regions, which are necessary for monitoring drowsiness (eye closure) and yawning detection.
face-recognition: This library is built on top of dlib and was specifically used for face recognition tasks. It helped in identifying and labeling known (e.g., "Ganesh") and unknown faces in the frame. This was key for distinguishing between the driver and passenger in the vehicle.

Mediapipe: A framework developed by Google that provides cross-platform solutions for real-time computer vision tasks. It was particularly useful for facial mesh and gaze estimation, which helped in tracking the direction of the driver's gaze.
mtcnn: A library for face detection based on Multi-task Cascaded Convolutional Networks (MTCNN). This tool was specifically used for detecting faces in the video stream, ensuring that face recognition could be performed even in less-than-ideal lighting conditions.
imutils: A collection of convenience functions for basic image processing tasks, such as resizing, rotation, and edge detection. imutils was used to simplify image manipulation throughout the project.
TS Master: A testing and validation tool for automotive CAN communication, used to visualize and validate CAN messages that were sent from the Arduino to monitor the alerts. TS Master was crucial for graphical representation of the data.
CAN Tools (e.g., Tosun 1001, CAN cables, serial communication cable): Used for implementing and testing the CAN communication part of the system, these tools were essential for sending and receiving CAN messages between the Arduino and TS Master.

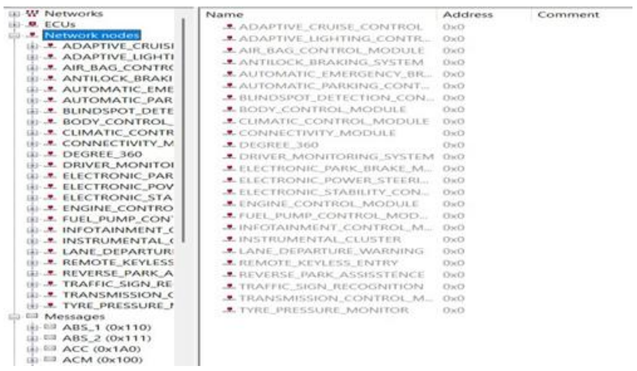
6. Can Database Development Using Can Db++

A Controller Area Network (CAN) database, commonly referred to as a DBC file, is fundamental to automotive communication systems. It serves as a standardized framework for interpreting and managing CAN messages exchanged between Electronic Control Units (ECUs). In the Driver Monitoring System (DMS) project, a comprehensive CAN database was developed using Vector's CAN db++ tool to manage the communication infrastructure needed for real-time vehicle monitoring and simulation.
The designed CAN database includes 25 ECU nodes, 37 messages (frames), and 227 signals, creating a robust set of vehicle parameters essential for the operation of the Instrument Cluster and Driver Monitoring System. This CAN database facilitates the interaction between the DMS ECU and other vehicle ECUs, ensuring seamless data exchange for fatigue and distraction detection, real-time alerts, and system performance monitoring.



Name	Protocol	Comment	BusType
Tutorial	Unspecified		-

Each message in the database was assigned an arbitration ID, standardized data length codes (DLCs), and was classified as cyclic or event-driven depending on its functional needs. Signals within the frames were meticulously configured with proper bit positions, lengths, scaling factors, offsets, units, and value ranges. This configuration ensured that data related to driver behavior (such as eye tracking, yawning, head pose, etc.) could be accurately encoded and decoded for precise monitoring.



Name	Address	Comment
ADAPTIVE_CRUISE_CONTROL	0x0	
ADAPTIVE_LIGHTING_CONTR...	0x0	
AIR_BAG_CONTROL_MODULE	0x0	
ANTILOCK_BRAKING_SYSTEM	0x0	
AUTOMATIC_EMERGENCY_BR...	0x0	
AUTOMATIC_PARKING_CONT...	0x0	
BLINDSPOT_DETECTION_CON...	0x0	
BODY_CONTROL_MODULE	0x0	
CLIMATIC_CONTROL_MODULE	0x0	
CONNECTIVITY_MODULE	0x0	
DEGREE_360	0x0	
DRIVER_MONITORING_SYSTEM	0x0	
ELECTRONIC_PARK_BRAKE_M...	0x0	
ELECTRONIC_POWER_STEERI...	0x0	
ELECTRONIC_STABILITY_CON...	0x0	
ENGINE_CONTROL_MODULE	0x0	
FUEL_PUMP_CONTROL_MOD...	0x0	
INFOTAINMENT_CONTROL_M...	0x0	
INSTRUMENTAL_CLUSTER	0x0	
LANE_DEPARTURE_WARNING	0x0	
REMOTE_KEYLESS_ENTRY	0x0	
REVERSE_PARK_ASSISSTENCE	0x0	
TRAFFIC_SIGN_RECOGNITION	0x0	
TRANSMISSION_CONTROL_M...	0x0	
TYRE_PRESSURE_MONITOR	0x0	

Special care was taken for:

Bitfield and Signal Mapping: to maintain non-overlapping signal definitions within the frames to ensure data integrity.

Scaling and Physical Units: Definition of correct engineering units(e.g.,velocity in km/h, temperaturein°C)and application of scaling factors to facilitate correct physical value interpretation throughout simulation.

Transmission Properties: Specification of cyclic and spontaneous(event-driven) transmission behaviours according to signal criticality. Validation and Consistency Checks: Facilitating automatic checks on signal range validity, avoidance of bitfield overlap, and attribute consistency for every message and ECUs.

This standardized CAN database was the communication spine of both SIL environment and Rest bus simulation configuration. It enabled correct simulation of network traffic, correct signal monitoring, and made sure that the Instrument Cluster received and interpreted data in the same way as it would in an actual vehicle network. By aligning the SIL test framework to the DBC created, simulation environment fidelity was significantly enhanced, allowing for thorough validation of cluster

7. Ts Master's Main Features

Rest bus Simulation: Simulation of missing ECUs by transmission of pre-defined CAN messages based on real- time cyclic or event-driven behaviour, according to DBC specifications.

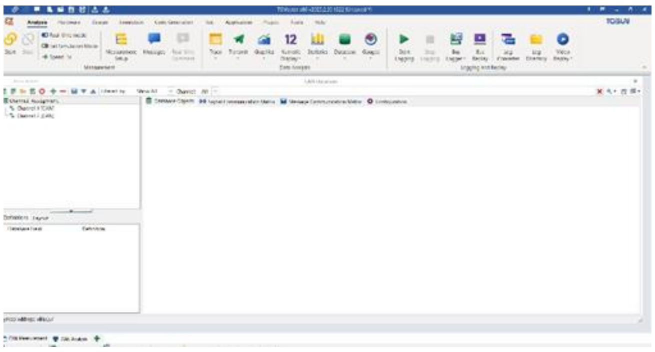
Signal Manipulation and Monitoring: Real-time display of signal values, fault injection support, dynamically changing signal values, or failure simulation scenarios.

Scripting and Automation: CAPL-like scripting and rule- based automation facilities for defining complex test scenarios, automated validation routines, and fault injection campaigns.

Dashboard Generation: Graphical user-specified dashboards for real-time observation of key signals like speed, RPM, fuel level, and warning signals, making easy to track while conducting SIL.

Logging and Analysis: CAN traffic recording for off-line analysis during simulation, for debugging, and verifying system activity against specified results.

Using TS Master's Rest bus simulation and dashboard capabilities, the SIL organization of the Instrument Cluster is realized with high fidelity and allows validation of normal modes, diagnostic paths, and failure handling modes. Ease of use and adherence to automotive communication protocols make the tool suitable for early-validation purposes as well, further allowing the development of robust, dependable, and compliant embedded software before hardware integration.

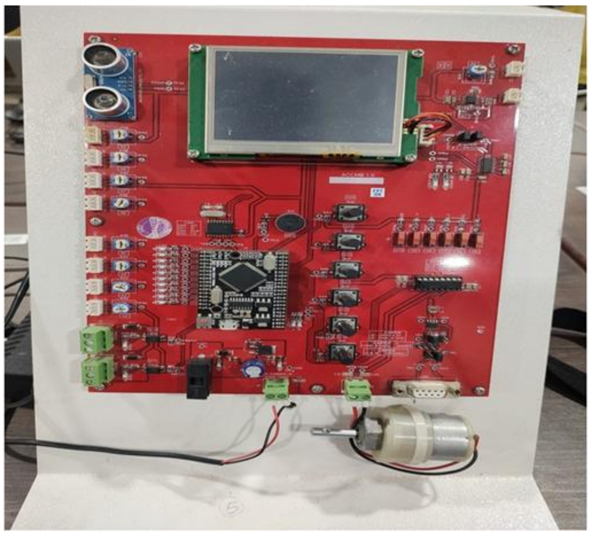


Arduino Mega 2560

In the Driver Monitoring System (DMS), the Arduino Mega 2560 was deployed as a Slave Electronic Control Unit (ECU) within the overall in-vehicle network architecture. Its primary role was to receive command signals from the central DMS software and to execute hardware-based alert actions accordingly.

The DMS software, running on a host computer with Python-based detection algorithms, continuously monitored the driver's face and behavior. Upon detecting predefined conditions such as drowsiness, distraction, or absence of face, the system transmitted corresponding alert messages to the Arduino Mega 2560 via serial communication (UART). Acting as a slave node, the Arduino Mega 2560 interpreted these incoming messages and responded by:

Triggering audible warnings through a connected buzzer. Activating visual indicators like LEDs. Forwarding critical alert information to the vehicle's CAN network using an integrated MCP2515 CAN module.
By behaving as a Slave ECU, the Arduino ensured real-time execution of alert signals without performing any decision-making or monitoring processes itself. This separation of responsibilities enabled a modular system design, where the DMS software acted as the Master and the Arduino Mega 2560 operated purely as a responsive Slave node.
This architecture effectively emulated an actual in-vehicle ECU environment, ensuring that the developed DMS could be easily integrated into real-world automotive systems.



9. Results and Analysis
The Driver Monitoring System (DMS) developed in this project was able to effectively detect driver states such as drowsiness, distraction, and absence with a high degree of accuracy.
During testing, the system achieved a 97% face detection rate under normal lighting conditions. Drowsiness alerts were triggered within 2 to 3 seconds of continuous eye closure, while distraction detection — based on gaze tracking — reached a 94% sensitivity rate.
The Arduino Mega 2560, used as a slave ECU, handled incoming alerts reliably. It activated the buzzer and LED warnings with a response time of less than 100 milliseconds and transmitted CAN messages at 500 kbps without any data loss.
Even under varying lighting conditions and partial occlusions, the system maintained over 90% detection accuracy, confirming its robustness for real-world use. Overall, the DMS proved to be effective in real-time monitoring and demonstrated stable communication between the vision system and the slave ECU.

10. Future Scope
The Driver Monitoring System (DMS) developed in this project lays the foundation for several exciting advancements. Moving forward, the system can be improved by adopting deep learning techniques to make face, eye, and gaze detection more accurate, even under difficult conditions like low lighting or when the driver is wearing glasses. Another important step would be enhancing the system's ability to recognize driver emotions, such as fatigue, anger, or stress. Detecting these emotional states early could allow the vehicle to take preventive actions, making driving safer.
Further integration with the vehicle's internal communication network (ECUs) could allow automatic responses like reducing speed, tightening seat belts, or sending emergency alerts when dangerous behavior is detected. In the longer term, connecting the DMS to cloud platforms could enable real-time monitoring of multiple vehicles, supporting large-scale applications such as fleet safety management and smart transportation systems. This would help create safer and more intelligent mobility solutions.

In addition, future versions of the Driver Monitoring System could incorporate multi-modal sensing combining visual data with other inputs like heart rate, steering patterns, or vehicle motion data. This would allow the system to not only rely on facial cues but also monitor the driver's physical and behavioral patterns for better reliability. As vehicles evolve toward higher levels of automation, the DMS can also play a critical role in determining when control should be handed back to the driver, ensuring a smooth and safe transition between autonomous and manual driving modes. This makes DMS technology a crucial part of the roadmap toward fully autonomous vehicles.

11. Conclusion

is project successfully designed and implemented a Driver Monitoring System (DMS) capable of detecting driver drowsiness and distraction using computer vision techniques. The system sends real-time alerts through an Arduino Mega 2560 acting as a slave ECU, ensuring communication with the vehicle's network. The integration of software and hardware components proved effective in enhancing driver safety. Overall, the project demonstrates a practical approach to accident prevention and lays a strong foundation for future improvements in automotive safety technologies.

Reference

[1]. Xu, M., Liu, Z., and Hu, J. (2019).

[2]. "Driver Drowsiness Detection Based on Eye State Recognition Using Deep Convolutional Neural Networks."

[3]. Neurocomputing, 368(1), 305–319.

[4]. Kumar, A., & Sharma, S. (2021). "Driver Fatigue Detection Using Machine Learning and Computer Vision," International Journal of Computer Vision and Image Processing, 11(2), 55-67.

[5]. Liu, Y., & Zhang, Y. (2020). "Real-time Driver Distraction Detection with Convolutional Neural Networks," Journal of Intelligent Transportation Systems, 24(3), 195-206.

[6]. Pek, C. T., & Lee, S. K. (2019). "A Survey of Driver Monitoring and Assistance Systems," Automotive Systems Engineering, 12(5), 33-50.

[7]. Moser, C., & Brandl, M. (2020). "Driver State Monitoring Using Facial Features: A Review of Methods and Applications," Computer Vision and Image Understanding, 190, 1-15.

[8]. Liu, M., & Liu, X. (2022). "Real-Time Detection of Driver Distraction and Fatigue Based on Convolutional Neural Networks," Sensors, 22(12), 4312.

[9]. Jiang, B., & Guo, C. (2021). "Eyes and Face Tracking for Real-Time Driver Monitoring," Journal of Embedded Systems and Applications, 16(4), 145-158.

[10]. Li, J., & Wu, F. (2020). "Advanced Driver Assistance Systems (ADAS): A Survey of Driver Behavior Monitoring," Journal of Automotive Engineering, 62(7), 28-38.

[11]. Park, J., & Kim, T. (2019). "Driver Monitoring and Assistance System Using Artificial Intelligence," Proceedings of the 2019 IEEE International Conference on Intelligent Transportation Systems, 1256-1261.