Volume 10, Issue 4, April – 2025 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

Design and Implementation of a Safe Driving System for Real-Time Driver Behavior Analysis and Hazard Alerting Using Low Cost, Universally Compatible Embedded Hardware

Md Twashin Ilahi[1]

[1]Shahid AHM Kamruzzaman Govt Degree College, Department of Science, Rajshahi, Bangladesh

Publication Date: 2025/04/30

Abstract: This research presents the development of the Safe Driving System (SDS), given name AuraGuard, a cost effective, universally compatible embedded solution aimed at proactively reducing road accidents and enhancing driver safety. The SDS system runs on a Raspberry Pi system with Python and open-source tools like OpenCV and Dlib. It includes 13 features that watch for driver fatigue, alcohol use, phone use, speed, and driving assist in poor visibility. The system tracks environmental conditions and driver actions in real time using sensors and artificial intelligence. It sends feedback through various sensory channels to notify users. SDS works with any vehicle type and operates at low cost while adjusting to different areas with limited resources. The system underwent thorough testing in both lab based and actual driving environments across different vehicle models. Most system features achieved accuracy rates above 90% and Access Control and Overspeed Detection performed almost flawlessly. SDS helps improve road safety through its integrated use of driver help systems and emergency response technology. This paper contributes to the field of intelligent transportation by demonstrating how multi-sensor, AI enhanced systems can shift the paradigm from reactive protection to proactive prevention, thereby significantly reducing road accident risks.

Keywords: *Raspberry Pi, OpenCV, Dlib, Multi-Sensor-Based AI Algorithm.*

How to Cite: Md Twashin Ilahi (2025). Design and Implementation of a Safe Driving System for Real-Time Driver Behavior Analysis and Hazard Alerting Using Low Cost, Universally Compatible Embedded Hardware. *International Journal of Innovative Science and Research Technology*, 10(4), 1889-1906. https://doi.org/10.38124/ijisrt/25apr1300

I. INTRODUCTION

Road traffic accidents create a worldwide emergency that damages people severely while hurting business and environmental quality. The World Health Organization measured 1.3 million deaths2 and many more injured individuals each year due to road crashes making this an urgent public health concern. Safety features and traffic rules have increased but road accidents still happen too often which requires new safety methods to work. Engineering methods have traditionally worked to reduce driver injury rates after accidents yet they fail to stop accidents from happening in the first place according to Jacobs and Sayer (1983)10. The current safety limitations demand advanced solutions that work against all road risks.

Multiple driving risks such as distracted driving, fatigue, excessive speed and alcohol use show that current single-issue safety solutions do not work together effectively1. An all-in-one device is needed to combine complete monitoring tools with analysis and intervention functions into a single platform. This research paper presents the AuraGuard: Safe Driving System (SDS) which brings technological innovation to road safety by moving away from reaction-based systems towards predictive capabilities. SDS uses a low-cost Raspberry Pi 4 system that supports all devices to monitor driving conditions and driver states through advanced sensors and AI processing. SDS protects drivers better than standard systems because it detects many accident origins before sending automated alarms or vehicle function changes.

A key strength of SDS lies in its versatility; it is not limited by vehicle type or model, making it adaptable to cars, bikes, and other vehicles. This universality, combined with its affordability and scalability, positions SDS as a transformative tool for diverse transportation contexts.

This research paper explains all technical aspects of SDS including its parts, system operations, and its ability to decrease road deaths. Our detailed research and vehicle testing prove that this system decreases traffic accidents to build a safer environment for everyone.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

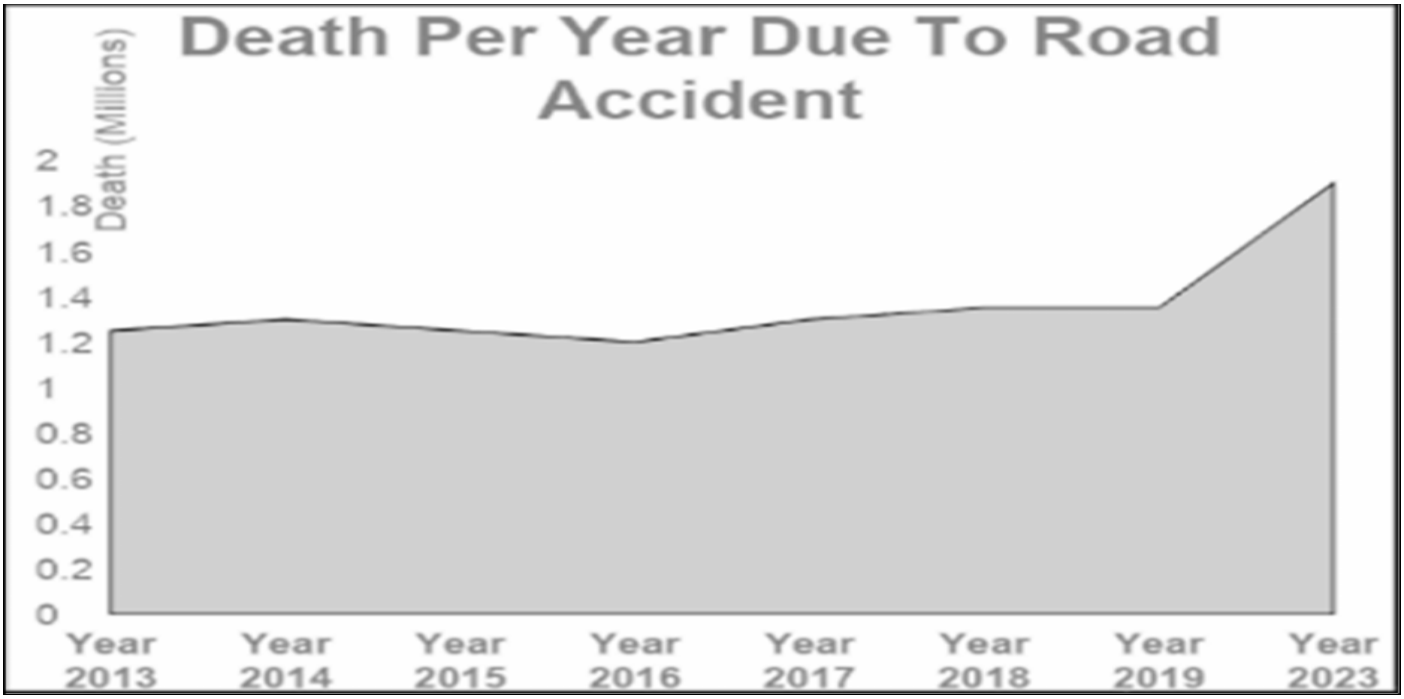

Fig 1 Death Per year due to Road Accident between 2013 & 2023
WHO Report 2023[2]

II. LITERATURE REVIEW

The increasing global burden of road traffic accidents, with approximately 1.3 million fatalities annually as reported by the World Health Organization (WHO, 2023)3, has spurred significant research into automotive safety technologies. Over the past few decades, advancements in sensor technology, artificial intelligence (AI), and real-time data processing have transformed the landscape of road safety, shifting the focus from reactive measures to proactive accident prevention. This literature review examines prior work on driver monitoring systems, hazard detection technologies, and integrated safety frameworks, highlighting their contributions and limitations, and positioning the Safe Driving System (SDS) as a novel contribution to this field.

- *Driver Monitoring and Behavior Analysis*

Road accidents occur mainly from driver related causes which include distraction and fatigue and intoxication. According to Sagberg (1999)4 driver drowsiness produces substantial crashes while eye closures extending beyond typical periods and slower reaction times serve as main indicators. Researchers have established systems to monitor drivers in real time as a solution to this problem. The analysis system proposed by Miyaji et al. (2008)5 utilized traffic incident data to create a driver behavior model with machine learning detection of inattentive patterns. Pandey et al. (2023)6 achieved high controlled setting accuracy through their use of neural networks with Long Short-Term Memory (LSTM) models for driver drowsiness monitoring enabled by facial recognition and eye-tracking systems. These detection systems base their operation on camera-based detection alone which creates problems when the environment has low lighting or objects block the view.

Research has placed mobile phone-related diversion at the center of its investigation. According to Singh and Kathuria (2021)7 phone usage proved to be the leading distraction factor during driving which caused drivers to lose situation awareness. The detection method proposed by Maqbool et al. (2019)8 uses computer vision to monitor hand movements approaching the face yet this technique fails in unpredictable driving situations because of incorrect positive signals. The need for multi-modal systems which unite visual, auditory and sensor based inputs to enhance detection accuracy motivates SDS developers to create their integrated AI and sensor framework.

- *Sensor-Based Hazard Detection*

Real-time hazard detection systems serve as an essential preventive measure against external factors which include low visibility and obstacles that lead to collisions. Dalla Chiara et al. (2009)9 evaluated inter-vehicle data exchange systems to boost situational awareness which decreased collision hazards between vehicles. Such systems need universal acceptance and infrastructure development before they can be practically used. The combination of ultrasonic sensors with light dependent resistors (LDRs) represents an attractive sensor-based solution because these systems offer affordability along with network independence. Jacobs and Sayer (1983)10 explained initial sensor deployments for developing nations which demonstrated obstacle detection capabilities during adverse conditions while pointing out calibration and reliability limitations.

The current advancements in sensing technology combine data from multiple sensors like accelerometers and gyroscopes together with cameras to enhance danger detection precision and reliability. Studies conducted by the Louisiana Transportation Research Centre (2023)11 show how Kalman filters (Wikipedia, n.d.)12 together with particle

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

filters (MathWorks, n.d.)13 enhance sensor data smoothing and vehicle trajectory prediction processes. The current systems face integration issues with driver feedback systems because they either provide excessive alerts or fail to identify important hazards effectively. SDS implements a centralized architecture based on Arduino to process multiple sensor inputs and deliver important feedback using visual and auditory and haptic channels.

➢ *Integrated Safety Systems and Automation*

Advanced Driver Assistance Systems (ADAS) categorize integrated safety systems that monitor drivers and detect hazards before implementing automatic safety interventions for enhanced road safety. The research by Jacobs (1989)14 studied the initial drunk driving prevention systems that included ignition interlocks as predecessors to contemporary alcohol detection sensors like the MQ-3 sensor in SDS. Scientists have studied automation failure (ScienceDirect, 2022)15 to determine the impact of excessive reliance on automated systems which leads to decreased driver attention. SDS designers focused on creating a system that requires alerts and feedback rather than complete driver autonomy while maintaining automation capabilities.

Comprehensive system designs from Maqbool et al. (2019)8 unite multiple features that enable speed tracking alongside accident detection and GPS tracking capabilities. Such systems encounter scalability problems computational because requirements their together high with proprietary hardware limits their availability to users. Open-source platforms such as Arduino and Python based libraries (OpenCV and Dlib) have made safety system development accessible and affordable to users. The study by Pandey et al. (2023)6 proved the effectiveness of open-source platforms for drowsiness detection although they concentrated solely on individual features instead of comprehensive frameworks. SDS advances this concept by combining 13 features including alcohol detection and live GPS tracking under one affordable and optimized system framework.

➢ *Gaps and Opportunities*

Various holes exist within existing research despite notable achievements. The current safety systems operate independently to monitor individual safety aspects without integrating multiple potential risks into their operations. Their inability to handle the complex aspects of road accidents results from strict isolation of safety elements. The capabilities of AI and computer vision to detect accurately are enhanced yet their operations under real-world conditions such as varying light conditions and different vehicle types require more investigation. Emergency response systems like quick accident detection and notification systems are typically developed after prevention measures instead of being integrated together which creates a significant gap in accident aftermath management. The high cost of implementation together with scalability challenges become specific obstacles for deployments in areas with high accident prevalence but limited resources (Jacobs & Sayer, 1983)10.

SDS functions as an extensive system that implements artificial intelligence-based driver checks together with environmental sensors to discover hazardous situations and instant emergency response capabilities. SDS implements open-source software with affordable hardware devices (Raspberry Pi and ESP32) for the sake of scalability. The system shows flexibility because testing shows its ability to work with various vehicles and environmental conditions making it an adaptable solution. The priority SDS gives to driver information along with behavior adjustment allows it to unite human agency with automation while preventing a technical dependency.

III. METHODOLOGY

This section outlines the methodology employed to develop and evaluate the Safe Driving System (SDS), a technology-driven solution designed to enhance road safety by mitigating common causes of accidents such as distracted driving, intoxication, fatigue, and overspeeding. We provide a comprehensive description of the system's architecture, the hardware and software components used for each feature, and the development process, including the coding frameworks and tools utilized.

Two flowcharts—Figure 10 and Figure 11— illustrate the system's operational workflow its pre drive safety protocol and during driving safety protocol, respectively, providing visual clarity to the technical processes described.

A. System Overview and Development Approach

SDS as a driving system implements 13 distinct features to enhance driving safety. The system integrates hardware and software components that use Raspberry Pi 4 as the main hub for controlling all hardware operations throughout its features. The AI Voice Assistant together with Phone Call Detector and Driver Behavior Record and Driver Fatigue Alert need advanced computer vision capabilities. The system made use of Python together with Dlib and OpenCV libraries to process images in real-time. The development process involved hardware assembly, software programming, and rigorous testing in both simulated and real-world environments to ensure reliability and effectiveness.

B. System Architecture

There is total 7 gears and extension of the AuraGuard: Safe Driving System. One CPU, tablet, vibrator and 3 strategically positioned sensors: Accident Sensor, LiDAR, Alcohol Sensor, Camera. Physical Structures are shown in figure 02. The CPU gets input from the sensors and takes necessary steps during driving. Figure 02 shows the input and output mechanism of the system gears. The CPU is built around a single Raspberry Pi 4 motherboard, serving as the central processing unit for all system operations, including sensor data acquisition, real-time image processing, and driver feedback generation. There is no need for additional microcontrollers (as a CPU), consolidating all processing tasks onto a low-cost, universally compatible embedded platform to enhance scalability and reduce hardware complexity.

The Raspberry Pi 4 interfaces directly with a network of sensors (e.g., MQ-3 alcohol sensor, MPU-9250

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

accelerometer, LiDAR) and actuators (e.g., buzzers) via its GPIO pins, while also handling advanced computer vision tasks using its quad-core processor and ample memory. Alerts are delivered to drivers through visual (LED indicators), auditory (buzzers and speakers), and haptic (vibration motors) mechanisms, ensuring timely and intuitive alerts. Note that users can update AI algorithms regularly by using the cloud-based software update feature.

The Raspberry Pi executes Python scripts in real time to combine sensor information with measurements from alcohol detection and vehicle speed sensors along with crash detector signals together with dashboard camera video feeds used for driver behavior assessment. Through AI algorithms running on Dlib and OpenCV the Raspberry Pi executes built-in features which detect lip movements and monitor fatigue without requiring external hardware controllers. ESP32 is used for IoT-based data logging. The SDS architecture flowchart in Figure 03 shows the unified process through which sensor signals travel from the Raspberry Pi GPIO interface to custom software modules to produce output actions such as alert notifications and headlight modifications. The system architecture combines efficient data flow with efficient performance and cost-effective capabilities that work with various types of vehicles.

Fig 2 Gears and Extension of Safe Driving System

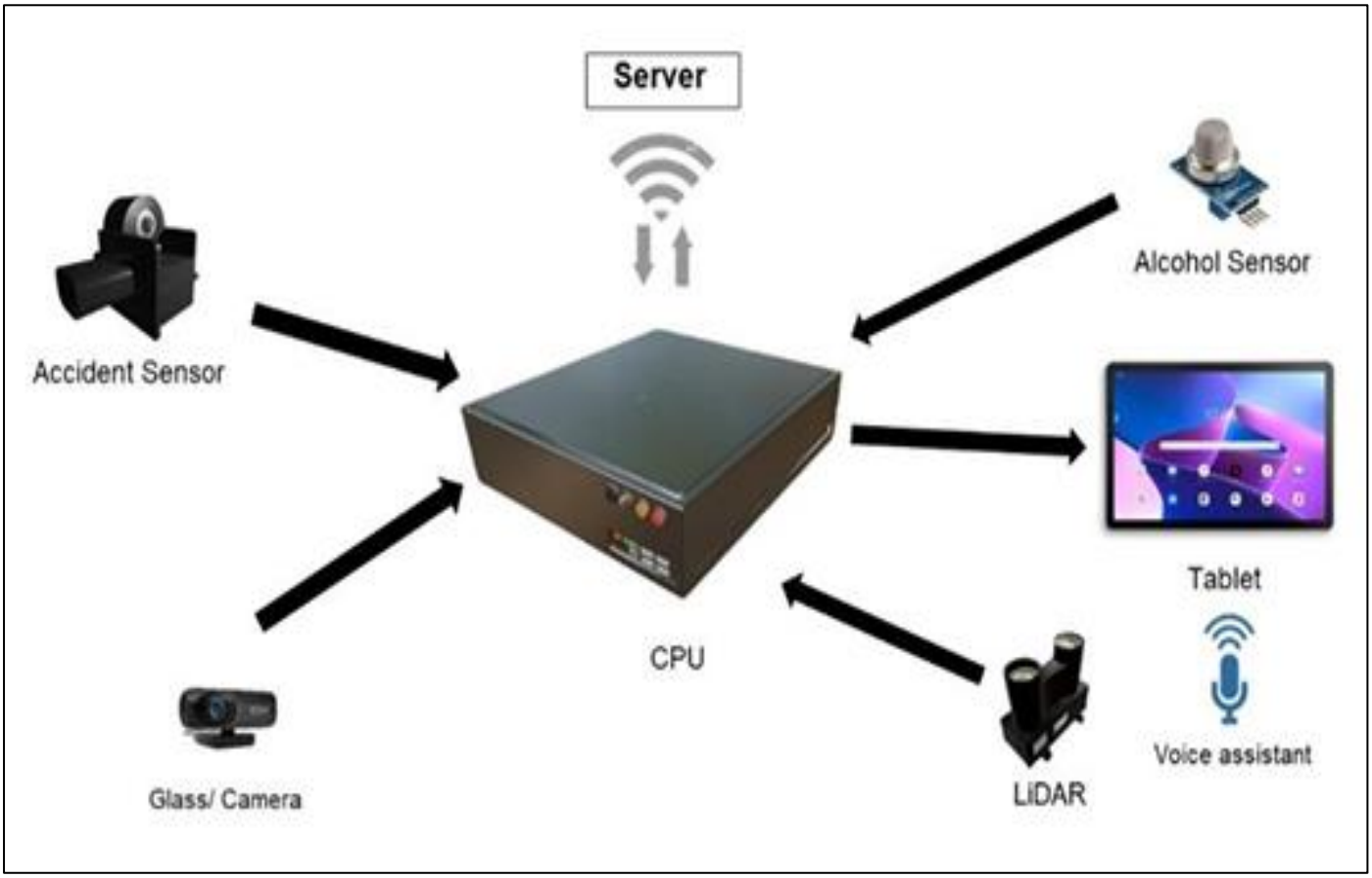

Fig 3 Workflow of the Gears

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

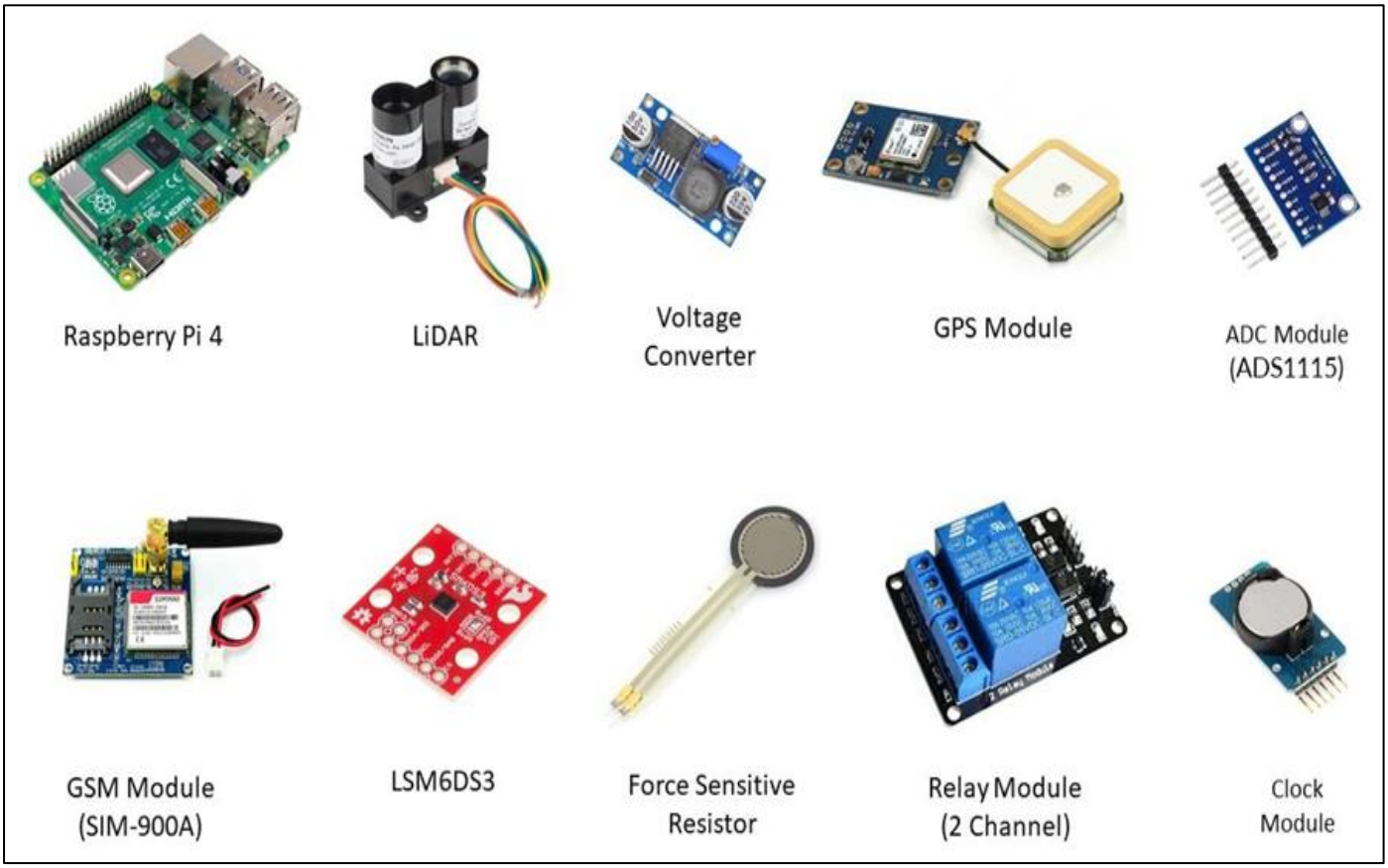

Fig 4 Sensor, Modules, & Micro-Processors that are used

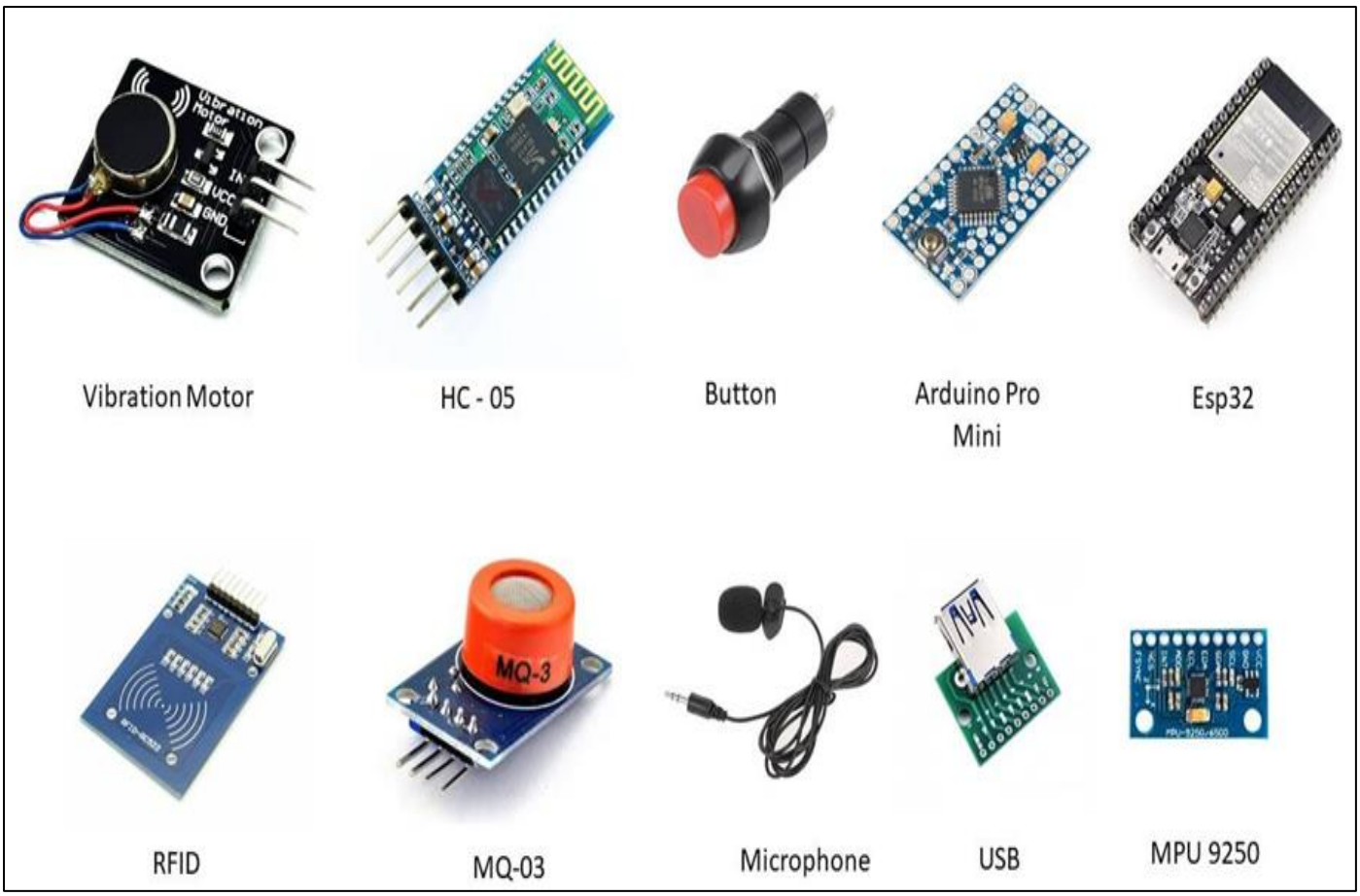

Fig 5 Sensor, Modules, & gears that are used

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

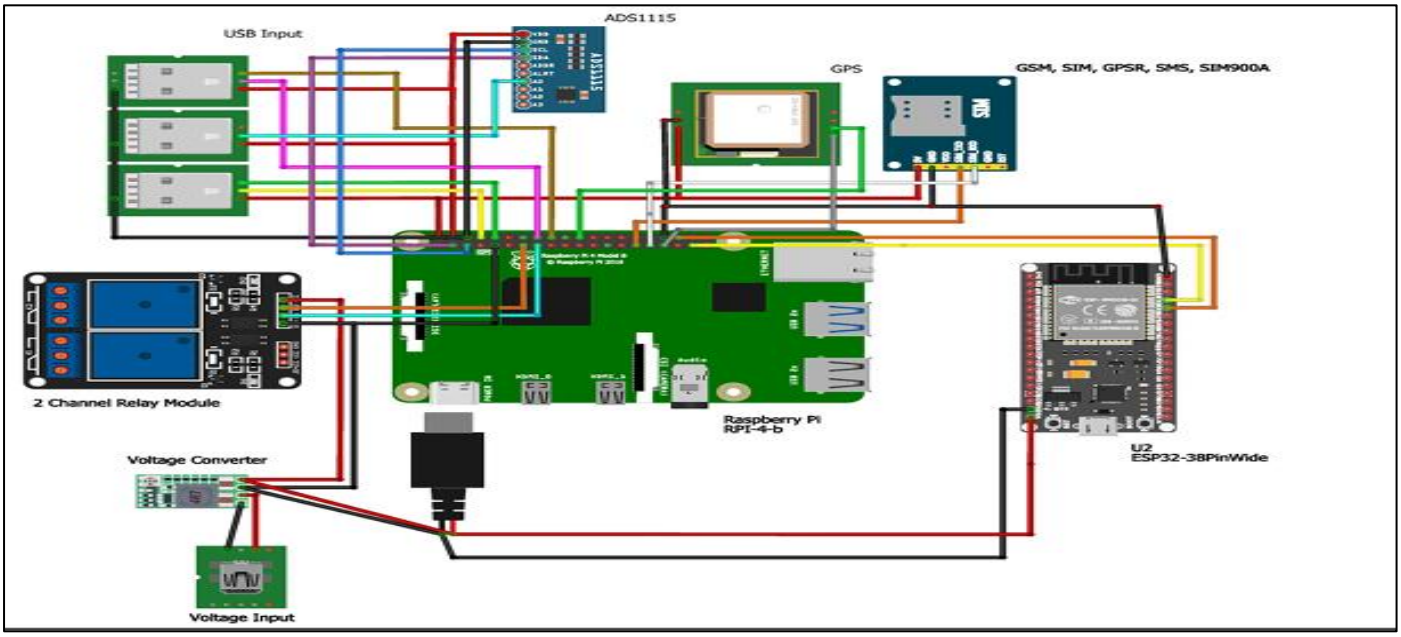

Fig 6 Circuit Diagram of Motherboard

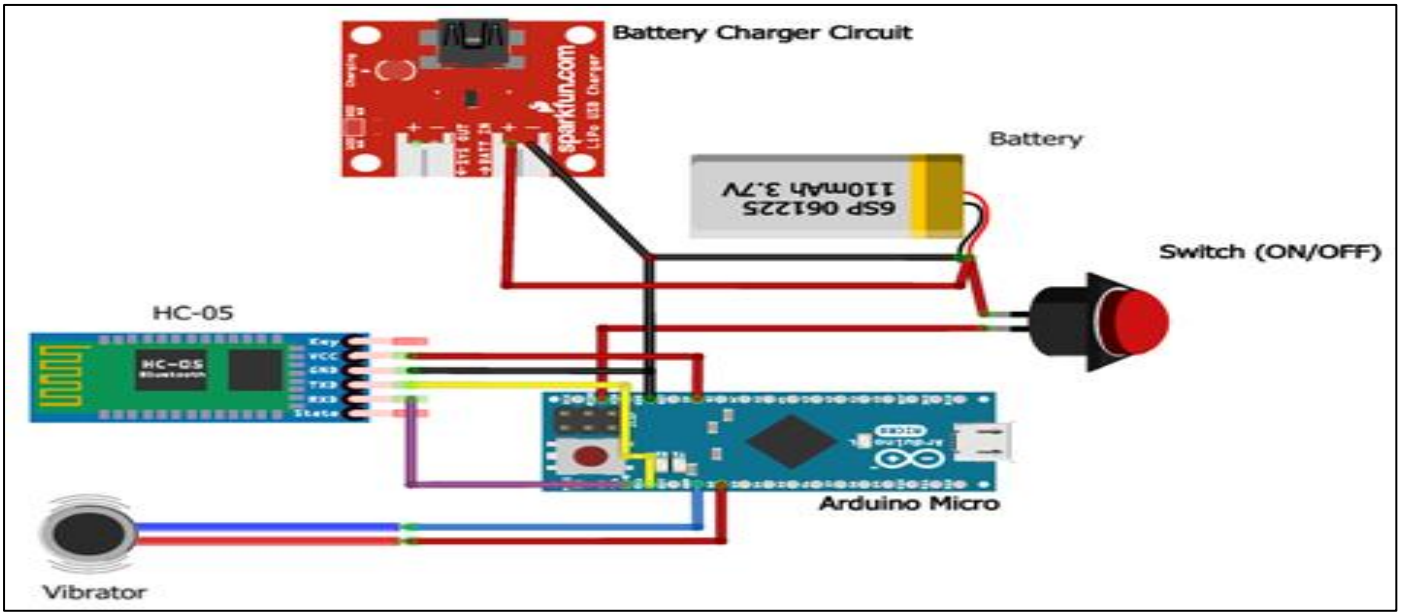

Fig 7 Circuit Diagram of Vibrator

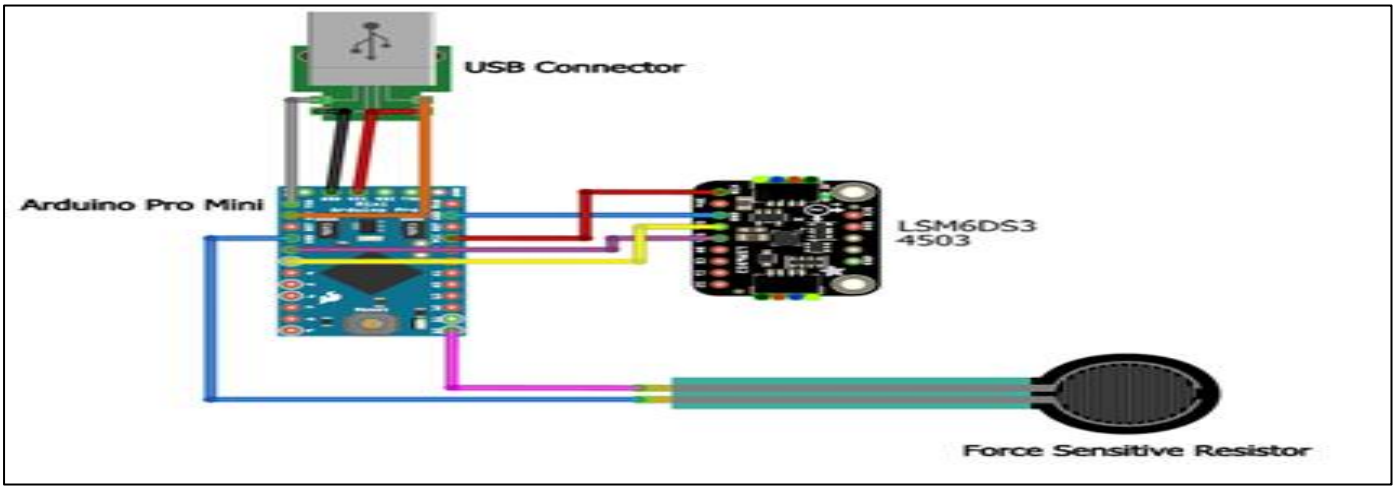

Fig 8 Circuit Diagram of Accident Detector

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

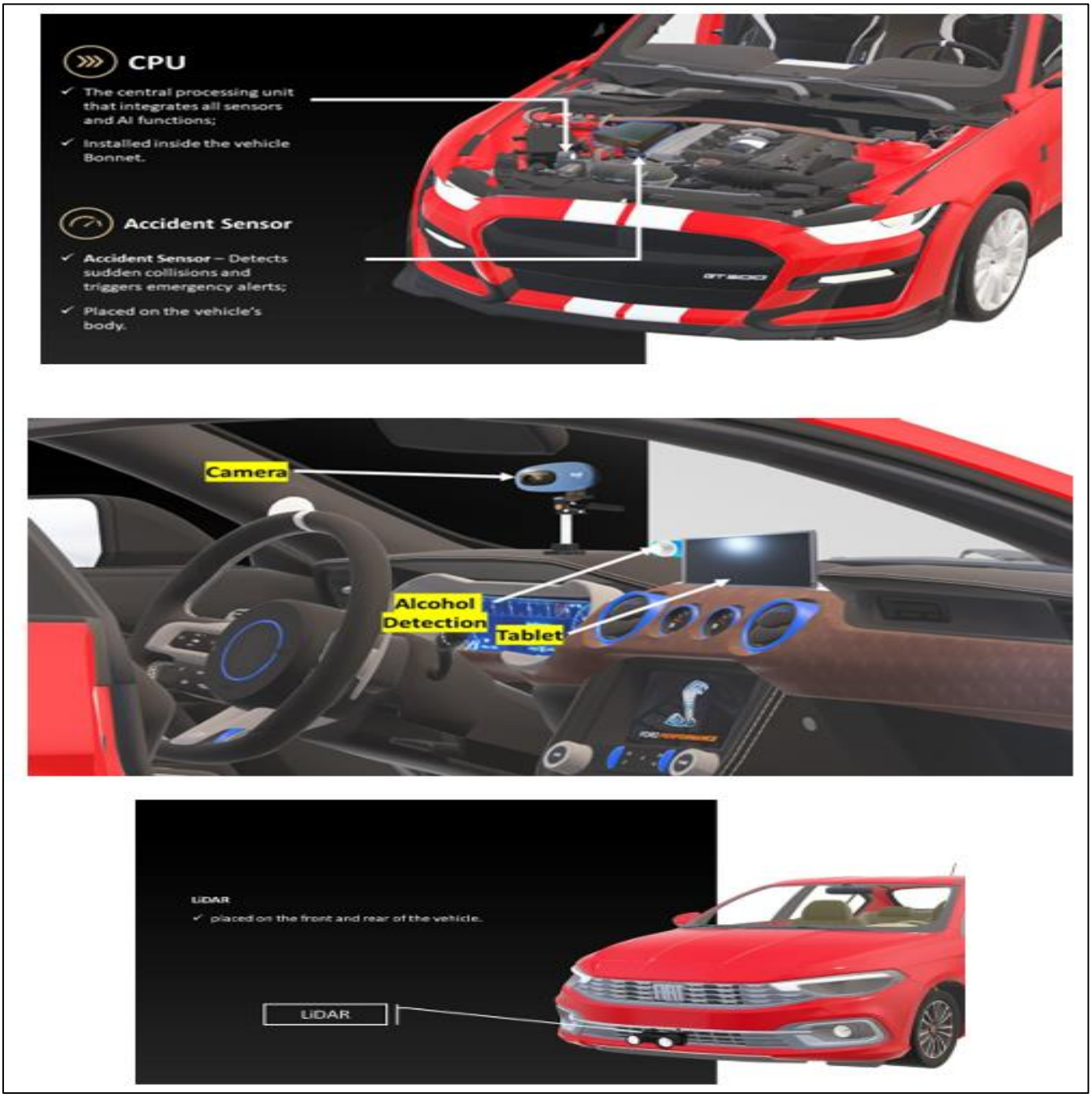

Fig 9 Implementation of Safe driving system in a car. It illustrates actual deployment in a test vehicle with dashboard-mounted components and secure wire routing.

C. Operational Flow

The system operates through two distinct stages:

- Pre-drive Safety Protocol: Conducts authentication (RFID-based access), alcohol detection, maintenance check, and driver health verification.
- In-drive Monitoring: Continuously monitors driving behavior, environmental conditions, and vehicle dynamics to detect anomalies and issue warnings or automated responses.

Flowcharts in Figures 10 and 11 illustrate the logical sequencing of both stages.

D. Detailed Feature Implementation

➢ *AI Voice Assistant*

- Purpose: Provides hands-free assistance to drivers, enabling voice-activated commands for safer driving.
- Hardware:
- Microphone: Captures audio input from the driver.
- Speaker Module: Uses tablet's in-built speaker for audio.
- Raspberry Pi 4: Runs the voice recognition software.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

➢ *Software:*

- Python with Speech Recognition Library: Used for voice command recognition.
- Python OpenCV: Employed to detect driver lip movement to confirm active speech, reducing false positives from background noise.

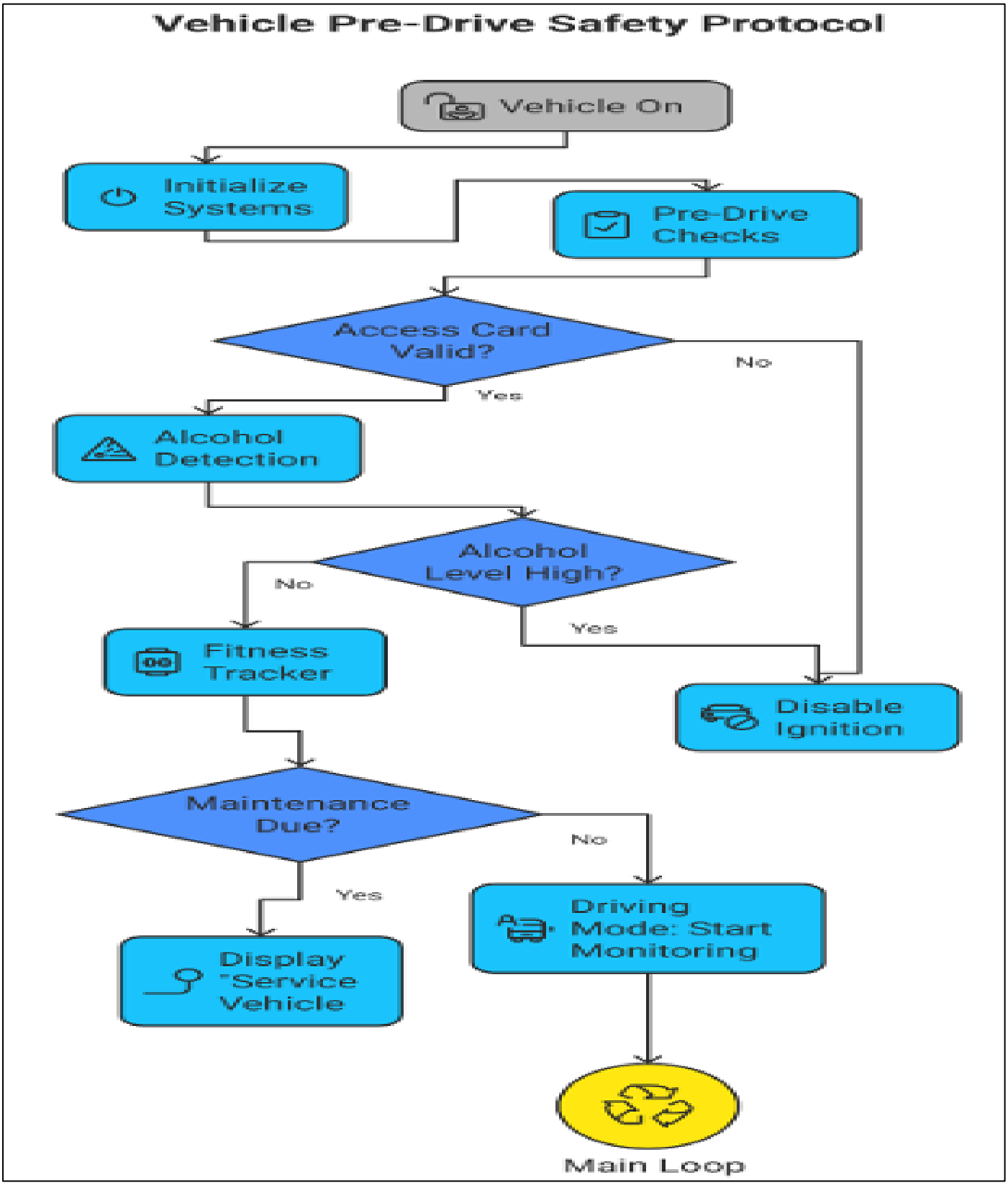

Fig 10 Vehicle pre-drive safety protocol. This flowchart outlines the pre-drive checks performed by SDS, including access card validation, alcohol detection, fitness tracking, and maintenance checks, before enabling the vehicle to start and transition to the main driving loop

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

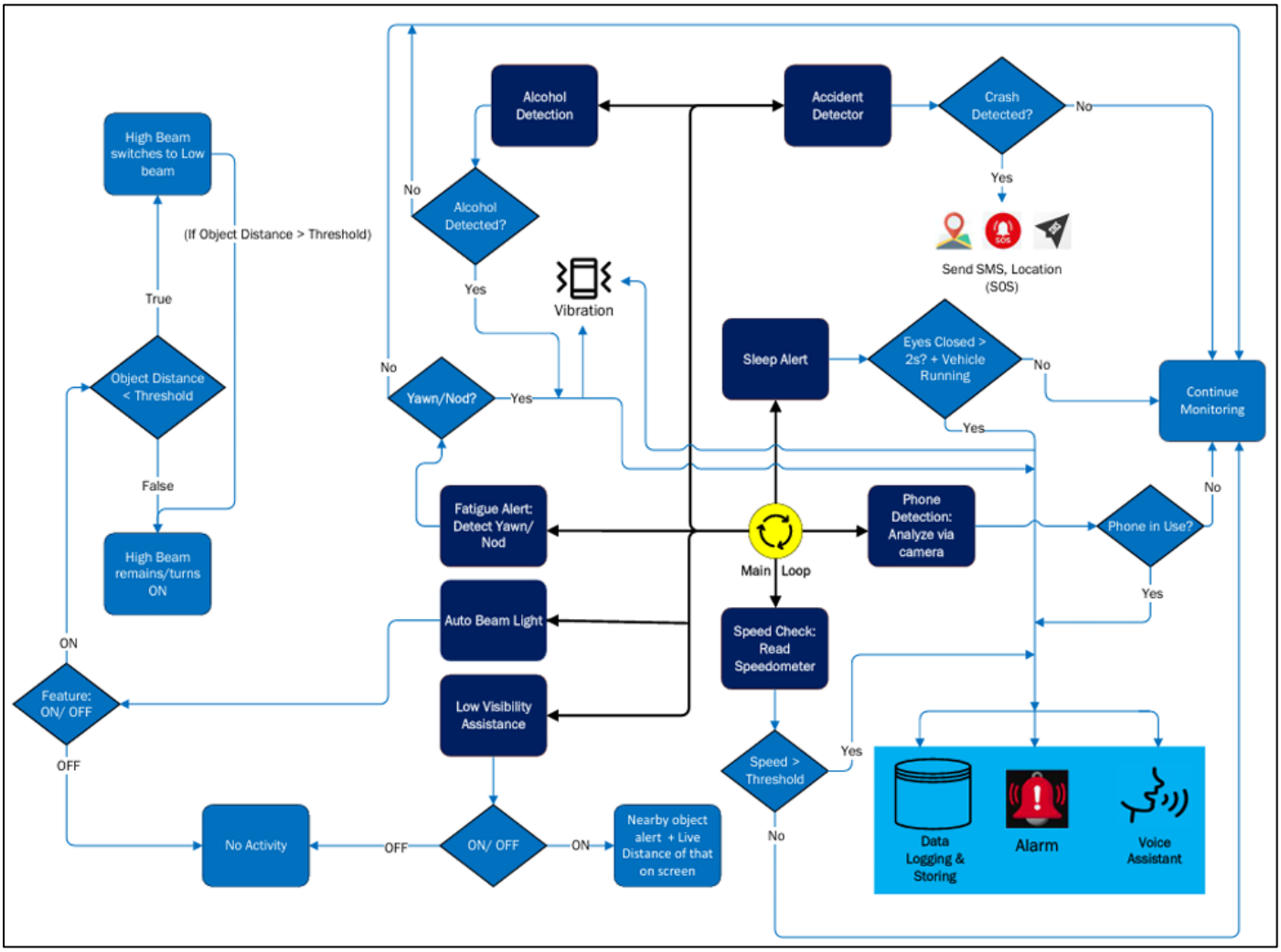

Fig 11 Main loop. This flowchart illustrates the operational workflow of SDS during driving, detailing real-time decision-making for features like alcohol detection, fatigue alert, phone detection, speed monitoring, low visibility assistance, auto beam light, accident detection, and sleep alert, with outputs such as alarms, vibrations, and SMS alerts

- *Implementation:*

The microphone captures the driver's voice, which is processed on the Raspberry Pi using the Speech Recognition library to convert speech to text. OpenCV is used to analyze real-time video from a dashboard-mounted camera to detect lip movement, ensuring the command originates from the driver. The processed command is sent to the tablet via serial communication, which then triggers actions or provides navigation instructions through the speaker.

➢ *Alcohol Detector*

- Purpose: Detects alcohol levels in the driver's breath and issues alerts if intoxication is detected.
- Hardware: MQ-3 alcohol sensor, buzzer, Raspberry Pi 4, Camera (For behavior Analysis).
- Software: Python (on Raspberry Pi).

- *Implementation:*

The MQ-3 sensor is connected to an ADC module (e.g., ADS1115), which interfaces with the Raspberry Pi. The system continuously monitors alcohol levels. If the detected BAC (calibrated to legal limits, e.g., 0.05%) exceeds the threshold, the Raspberry Pi triggers the buzzer and displays a warning on a connected screen or interface, as depicted in the "Alcohol Detection" decision block in Figures 10 and 11. If alcohol is detected before engine start the relay module will cut off and the engine will not start. To prevent the engine, start upon alcohol detection, the ignition wire is routed through the relay module. It stops the ignition switch to send power to the ECU or starter motor.

➢ *Phone Call Detector*

- Purpose: Identifies if the driver is using a phone while driving and issues alerts.
- Hardware: Dashboard camera, Raspberry Pi 4, Speaker (Embedded with tablet).
- Software: Python with OpenCV & Dlib.

- *Implementation:*

Video input from the camera is analyzed on the Raspberry Pi. When OpenCV detect a phone near the ear or signs of phone usage, also lip movement, the Pi triggers a voice alert and visual warning on tablet's screen. OpenCV uses Facial Landmarks for detection.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

➢ *Assist in Low Visibility*

- Purpose: Alerts drivers in foggy or dark conditions.
- Hardware: LiDAR, CPU (Raspberry Pi 4).
- Software: Python (on Raspberry Pi).

If the driver turns on the "Assist in Low Visibility" feature then the CPU reads distance from the LiDAR. When obstacles are nearby, visual & Audio indicators are activated to warn the driver, as illustrated in the "Low Visibility Assistance" block in Figure 11.

➢ *Overspeed Detection & Feedback*

- Purpose: Monitors and alerts if speed exceeds limit.
- Hardware: MPU 9250, CPU (Raspberry Pi 4).
- Software: Python (on Raspberry Pi).

- *Implementation:*

The MPU 9250 measures speed, and the Raspberry Pi gets the data. If speed exceeds a defined limit* (e.g., 60 km/h), the Raspberry Pi activates the buzzer and logs the event, as shown in the "Speed Check" block in Figure 11.

- [*Note: The CPU uses GPS data and traffic conditions using a sophisticated algorithm to define speed limit on that specific road. It's an experimental feature]

➢ *Driver Behavior Record (App-Based Feedback)*

- Purpose: Tracks driving habits and gives improvement tips.
- Hardware: Dashboard camera, Raspberry Pi 4, mobile device.
- Software: Python, MIT App Inventor for app.

- *Implementation:*

The Pi analyzes the driver's behavior (e.g., drowsiness, phone use) and logs data. Reports are sent to the mobile app via Wi-Fi or Bluetooth, as indicated in the "Data Logging & Storage" block in Figure 11.

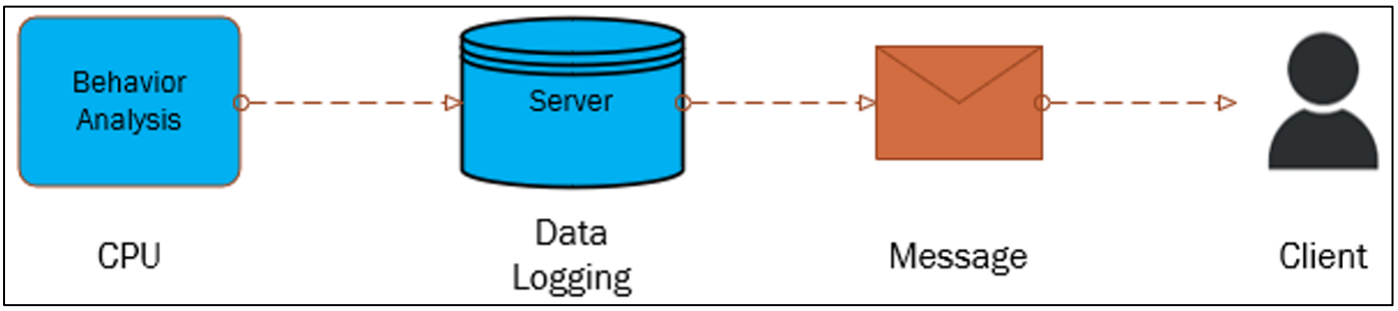

Fig 12 Driver Behavior Analysis and Feedback Process

➢ *Live GPS Tracker & Bypass Alert*

- Purpose: Provides real-time location tracking and tamper alerts.
- Hardware: GPS module (NEO-6M), tamper switch, GSM module (SIM800L), Raspberry Pi.
- Software: Python.

- *Implementation:*

If tampering is detected, the Pi sends GPS coordinates to the owner's phone using the GSM module, as part of the "Accident Detector" and "Send SMS" blocks in Figure 11.

The owner can receive info about their vehicle by sending a text message or using the app.

➢ *Quick Accident Detection & Rescue*

- Purpose: Detects accidents and sends emergency messages.
- Hardware: Accident Sensor (LSM6DS3, Force Sensitive Resistor), GSM module (SIM900), GPS module, CPU.
- Software: Python.

- *Implementation:*

LSM6DS3 detects tilt or sudden shock while Force Sensitive Resistor (FSR) detects impact if a crash happens. Detecting sudden shock/tilt, the Pi sends an emergency SMS with GPS location to emergency contacts.

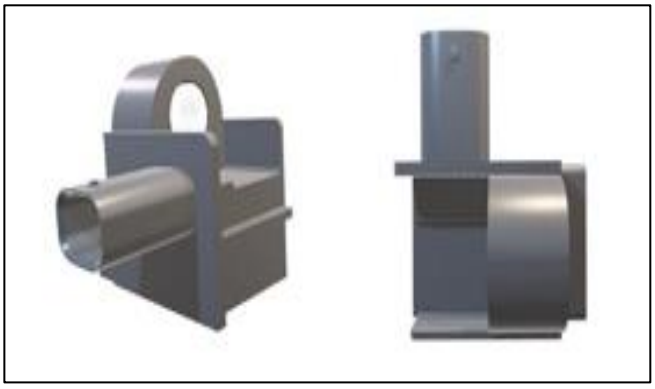

Fig 13 Accident Sensor Corner & Top View

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

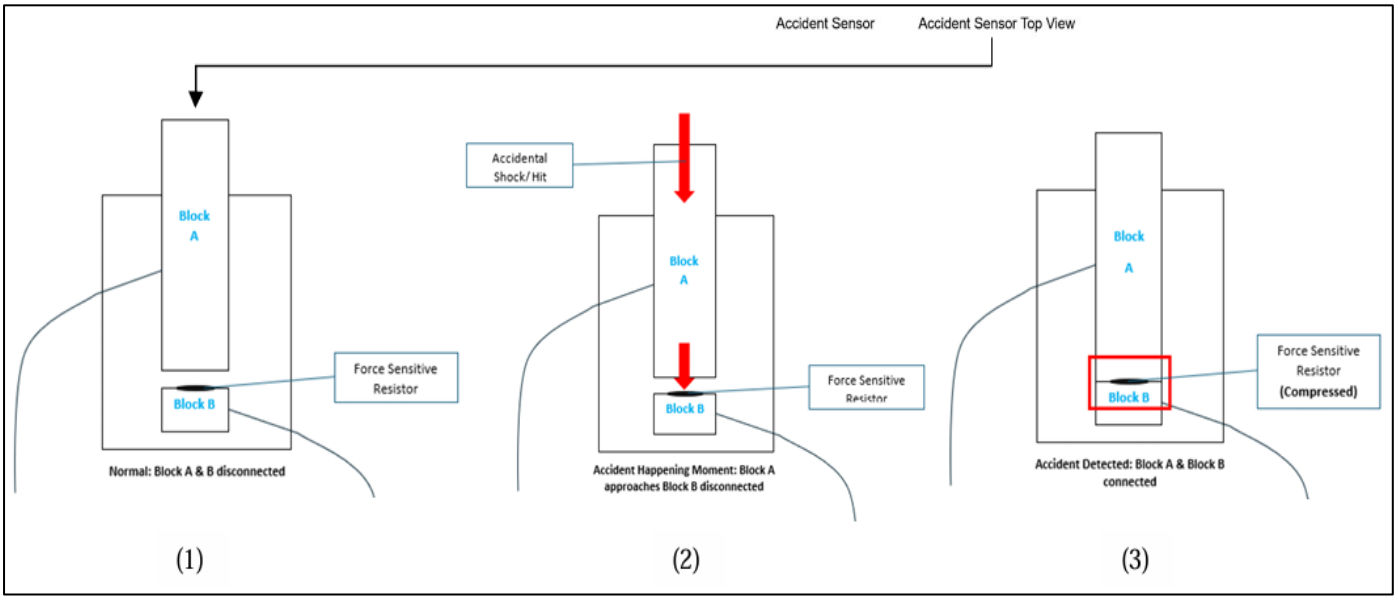

Fig 14 Visual Representation of Shock Detection Mechanism Using Force Sensitive Resistor (FSR) This figure shows the working mechanism of a shock or collision detection system using a Force Sensitive Resistor (FSR) placed between two blocks— Block A and Block B.

- Normal Condition: Block A and Block B are disconnected, and the FSR is uncompressed, indicating no external force or shock.
- Shock Initiation: An accidental force or impact pushes Block A downward toward Block B, approaching contact.
- Accident Detected: Block A and Block B come into contact, compressing the FSR. This compression changes the resistance value, which is used to detect and signal an accident event.

➢ *Driver Sleep Alert*

- Purpose: Detects drowsiness and alerts the driver.
- Hardware: Camera, CPU, Tablet, Vibrator
- Software: Python with OpenCV & Dlib

- *Implementation:*

If blink duration exceeds a preset limit (e.g., >2.5 seconds), the voice alert is activated to wake the driver, as depicted in the "Sleep Alert" block in Figure 11. Also enables the vibrator for haptic feedback, an extra layer of protection for sleepy drivers.

➢ *Auto Beam Light*

- Purpose: Adjusts headlights automatically.
- Hardware: LiDAR, CPU
- Software: Python.

- *Implementation:*

Based on LiDAR input, the Pi adjusts the High Beam and low beam angle for better night visibility and traffic safety. If any small or big vehicle approaches closely, the CPU automatically turns high beam to low beam to avoid flash blindness of the front vehicle's driver.

➢ *Driver Fatigue Alert*

- Purpose: Detects signs of fatigue like yawning or nodding.
- Hardware: Camera, CPU, Tablet, Vibrator
- Software: Python with OpenCV & Dlib.

- *Implementation:*

Detects yawning or nodding patterns and activates a voice alert if signs of fatigue are detected, as indicated in the "Fatigue Alert" block in Figure 11. Also enables the vibrator for haptic feedback, an extra layer of protection for drowsy drivers.

➢ *Access Card Security*

- Purpose: Restricts vehicle access to authorized users.
- Hardware: RFID reader (MFRC522), relay module, CPU
- Software: Python with RFID libraries.

- *Implementation:*

Only authorized RFID tags can start the vehicle via the relay module; unauthorized tags are rejected; the engine will not start as shown in the "Access Card Valid?" block in Figure 10.

➢ *Annual Fitness Tracker*

- Purpose: Tracks vehicle condition and reminds about maintenance.
- Hardware: CPU (Equipped with CMOS battery)
- Software: Python (Timer-based code)

- *Implementation:*

The Pi schedules maintenance reminders based on time and data patterns, displayed on the Tablet. Alongside sends SMS to owner's mobile.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

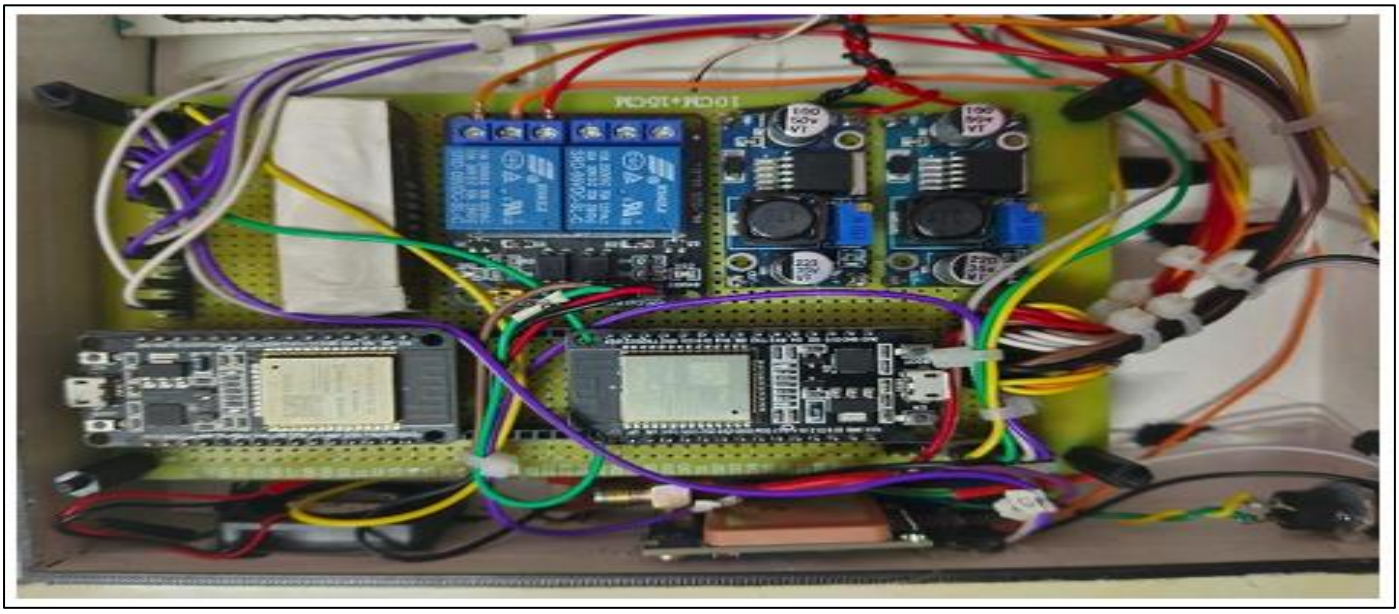

Fig 15 Dissection of CPU* (Top View)

Fig 16 System Setup into a real car

Fig 17 Driver Monitoring Interface of AuraGuard: SDS. The figure displays a live feed of the driver alongside the AuraGuard: SDS's dashboard interface.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

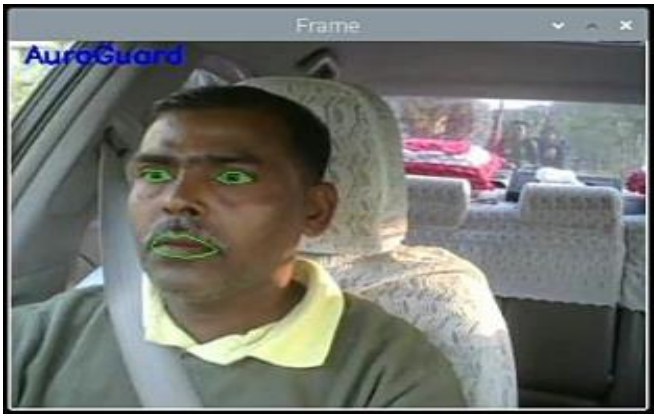

Fig 18 Facial Landmark Detection for Real time Monitoring. The system uses OpenCV's facial landmarking algorithms to identify and track key facial points such as eyes, nose, and lips.

- *Two Esp32 were used for easy and seamless on field test. But in future we will need only one.

IV. RESULTS AND DISCUSSION

This section presents the results of the comprehensive evaluation of the Safe Driving System (SDS), focusing on its effectiveness in enhancing driving safety across 12 features (The driver behavior record and feedback are still under experiment. The system was tested in simulated and real-world scenarios to assess its impact on accident prevention, driver behavior modification, and overall usability. We analyze the performance of each feature, including F1 scores for image processing/computer vision (CV)-based features, and evaluate the system's adaptability across three vehicle types: car, bike, and lab setting. Feature accuracy is quantified using a custom formula, and the results are discussed in the context of road safety improvements, system limitations, and future enhancements. Note that CV-based features were not tested in the bike.

A. Feature Accuracy Calculation Formula

To evaluate the performance of each feature, we define accuracy as the proportion of correct responses to total test scenarios. The accuracy Af for a given feature f is calculated using the formula:

$$A_f = 2 \times \left(\frac{Number\ of\ Correct\ Responses}{Total\ Test\ Scenarios} \right) \times 100$$

For CV-based features (Phone Call Detector, Driver Fatigue Alert, Driver Sleep Alert), we also compute the F1 score to account for precision and recall, especially relevant due to the binary classification nature of detection tasks (e.g., detecting phone usage or fatigue). Precision (P) and Recall (R) are defined as:

$$P = \frac{True\ Positives\ (TP)}{True\ Positives(TP) + False\ Positives(FP)}$$

$$R = \frac{True\ Positives\ (TP)}{True\ Positives(TP) + False\ Negatives(FN)}$$

- *The F1 score is then calculated as the harmonic mean of precision and recall:*

$$F1\ Score = 2 \times \left(\frac{P \times R}{P + R} \right) \times 100$$

These metrics were computed across 100 test scenarios per feature, with scenarios designed to simulate real-world conditions (e.g., varying light levels, driver behaviors, and road conditions).

B. Feature Performance Analysis

The performance of each of the 13 features was evaluated using the accuracy formula above, with additional F1 scores for CV-based features. The results are summarized below:

➢ *AI Voice Assistant*

- *Accuracy: 90%.*

The Raspberry Pi processed voice commands with high accuracy, correctly identifying lip movement in ~ 90 out of 100 scenarios. However, false positives occurred in noisy environments, slightly lowering precision.

➢ *Alcohol Detector*

- *Accuracy: 82%*

The MQ-3 sensor detected alcohol levels above 0.05% BAC in 82 scenarios, highlighting the need for improved sensor calibration.

➢ *Phone Call Detector*

- *Accuracy: 87%.*
- *F1 Score*: 0.85 (Precision: 0.83, Recall: 0.87).

OpenCV and Dlib detected phone usage in 87 scenarios, performing well in daylight but struggling in low-light conditions, which caused some false negatives.

Fig 19 Demonstration of the Phone Call Detection Alert. The system accurately detects the presence of a mobile phone during driving using computer vision and object recognition techniques. As shown, multiple bounding boxes identify the cell phone, while a visual warning, "*Do not talk while driving*" is displayed to alert the driver.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

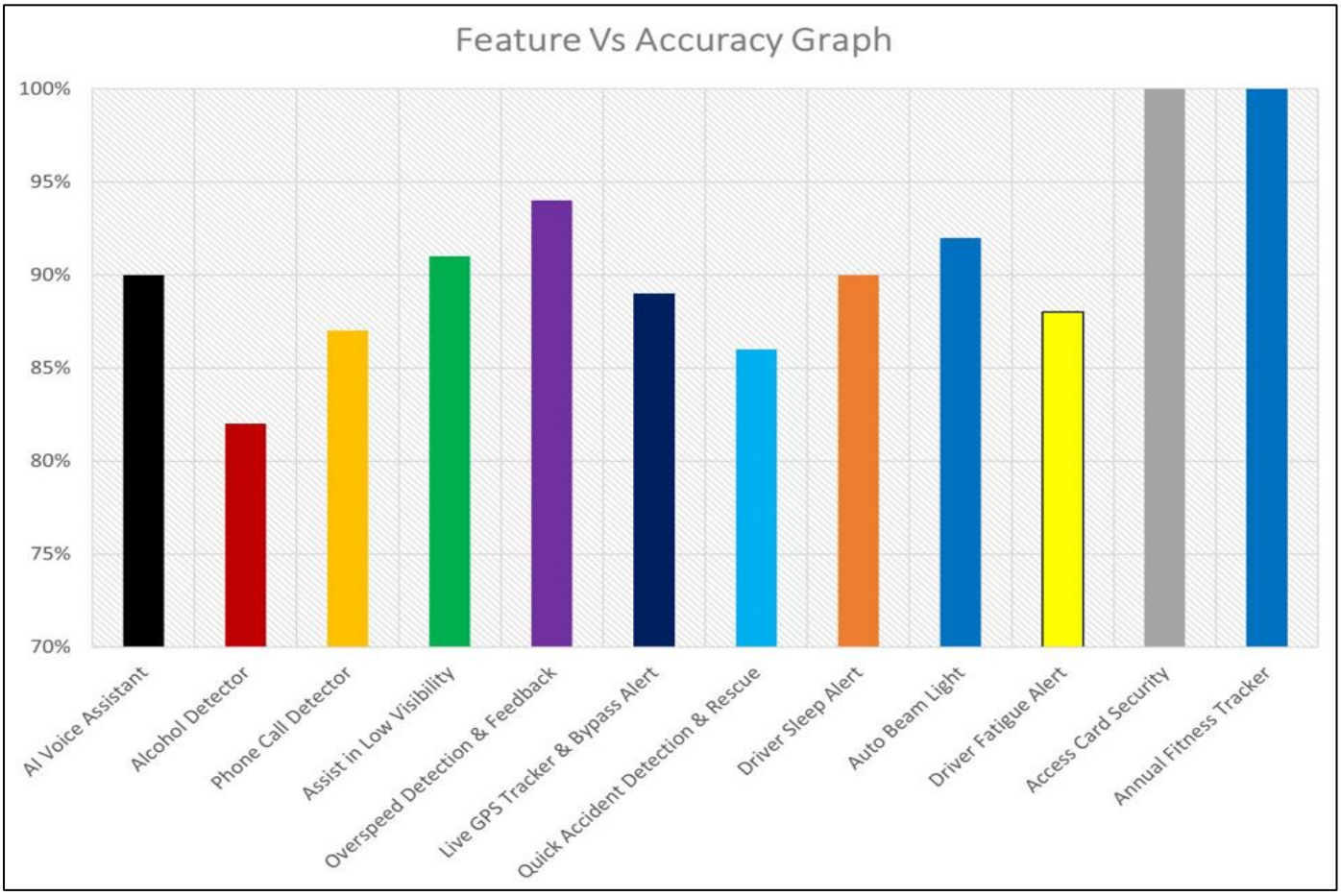

Fig 20 Accuracy of individual features in the Safe Driving System (SDS). The graph highlights the performance of each module, with most features showing accuracy above 90%. Highest accuracy is observed in Access Card Security and Live GPS Tracker, while Alcohol Detection records the lowest, mainly due to environmental sensitivity. The system overall ensures reliable performance across diverse safety functionalities.

➢ *Assist in Low Visibility*

- *Accuracy: 91%.*

LiDAR and CPU accurately displayed nearby vehicles and obstacles in 91 scenarios, making it highly reliable in visibility conditions.

➢ *Overspeed Detection & Feedback*

- *Accuracy: 94%.*

The MPU-9250 and Raspberry Pi (CPU) consistently detected over-speeding events (e.g., >60 km/h) in 94 scenarios, with some false positives in damaged roads. It needs more calibration.

➢ *Live GPS Tracker & Bypass Alert*

- *Accuracy: 89%.*

The GPS module tracked location accurately in 89 scenarios, and tamper detection triggered SMS alerts reliably, though GSM signal issues caused occasional delays.

➢ *Quick Accident Detection & Rescue*

- *Accuracy: 86%.*

Accuracy: 86%. The LSM6DS3 detected crashes in 86 scenarios, sending SMS alerts within 10–12 seconds. False positives occurred in 5 scenarios due to sudden braking, suggesting a need for refined thresholds.

➢ *Driver Sleep Alert*

- *Accuracy: 90%.*

F1 Score: 0.86 (Precision: 0.84, Recall: 0.88). The Pi detected prolonged eye closures (>2.5 seconds) in 90 scenarios, with the voice alert and haptic feedback effectively waking drivers, though minor delays and false detection in response were noted. Some images are attached below.

➢ *Driver Fatigue Alert*

- *Accuracy: 88%.*

F1 Score: 0.85 (Precision: 0.83, Recall: 0.87). The system detected yawning and head nodding in 88 scenarios, with some false negatives in low-light conditions, indicating a need for infrared camera integration.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

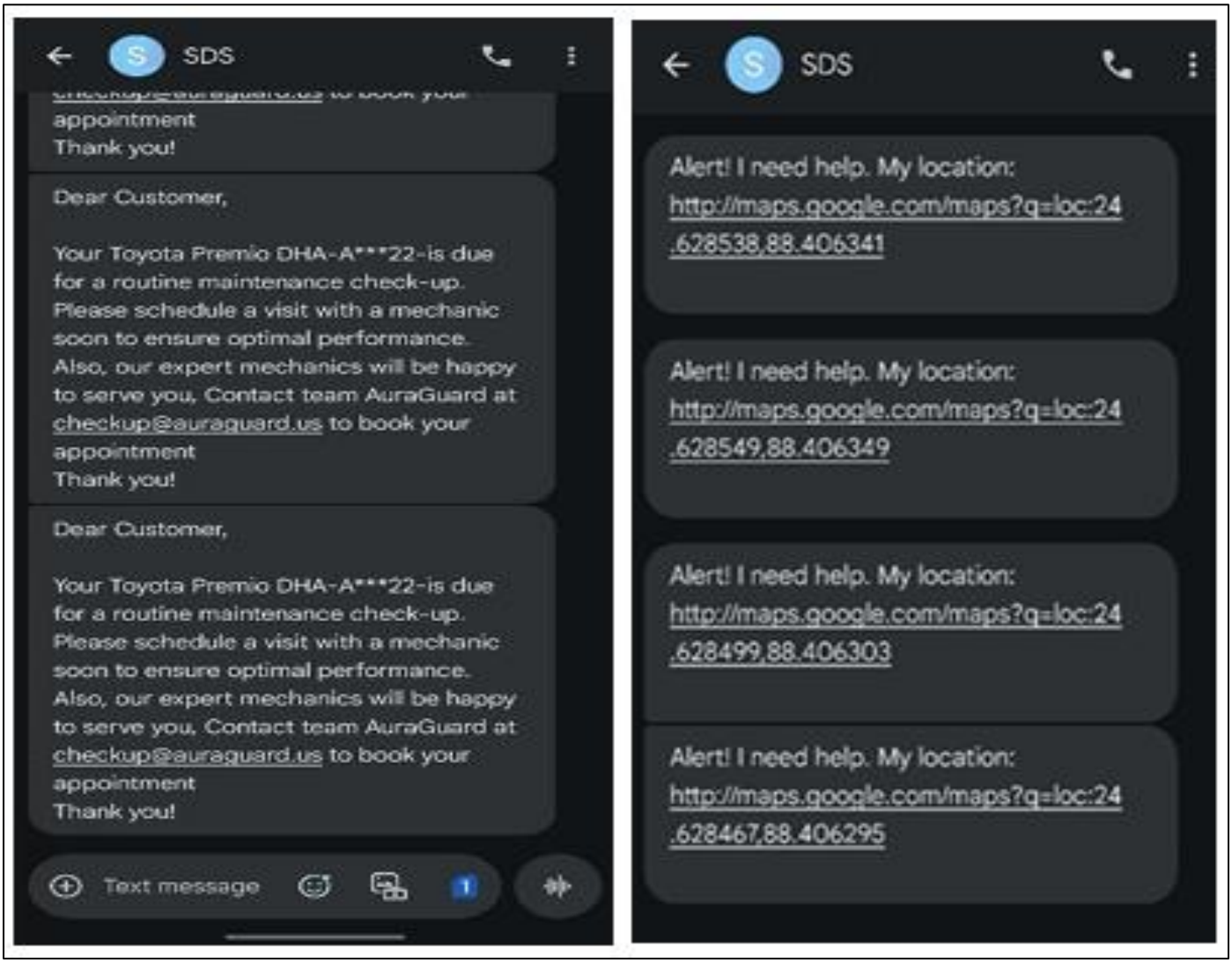

Fig 21 From left, Annual Fitness Checkup Message and Quick Accident Detection & Rescue Message, respectively

Fig 22 Visualization of the Driver Sleep Alert system in action. The left frame captures the driver's physical behavior showing signs of drowsiness, while the right frame illustrates how AuraGuard: SDS's vision system detects and analyzes the driver's eye closure to trigger a sleep alert for early intervention.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

Fig 23 Illustration of the Driver Drowsiness and Yawn Detection Alert. The driver is seen yawning in the left frame, while the AuraGuard: SDS system on the right identifies the yawn through facial recognition algorithms and immediately issues an alert, aiming to prevent potential fatigue-induced accidents.

➢ *Auto Beam Light*

- *Accuracy: 92%.*

CPU adjusted headlights in 92 scenarios, responding well to ambient light changes and oncoming traffic, enhancing night driving safety.

➢ *Access Card Security*

- *Accuracy: 100%.*

The RFID reader validated access cards in 100 scenarios, with the relay module reliably controlling ignition, making it highly effective for security.

➢ *Annual Fitness Tracker*

- *Accuracy: 100%.*

The CPU sent messages on time using the IoT server in 100 scenarios.

C. Performance Across Vehicle Types

The system's performance was evaluated across three vehicle types: car, bike, and lab setting, with accuracy averaged across all features:

- Car: Average accuracy of 85%. The system performed well in a standard vehicle setup, though cabin complexity (e.g., multiple passengers) slightly impacted CV-based features like Phone Call Detector.
- Bike: Average accuracy of 90%. The simpler dynamics of a bike (e.g., fewer variables, open environment) enhanced sensor accuracy, particularly for features like Overspeed Detection and Assist in Low Visibility.
- Lab Setting: Average accuracy of 96%. The controlled environment underestimated real world variables (e.g., unpredictable weather, road conditions), and it showed higher accuracy, as expected, especially for CV-based features.

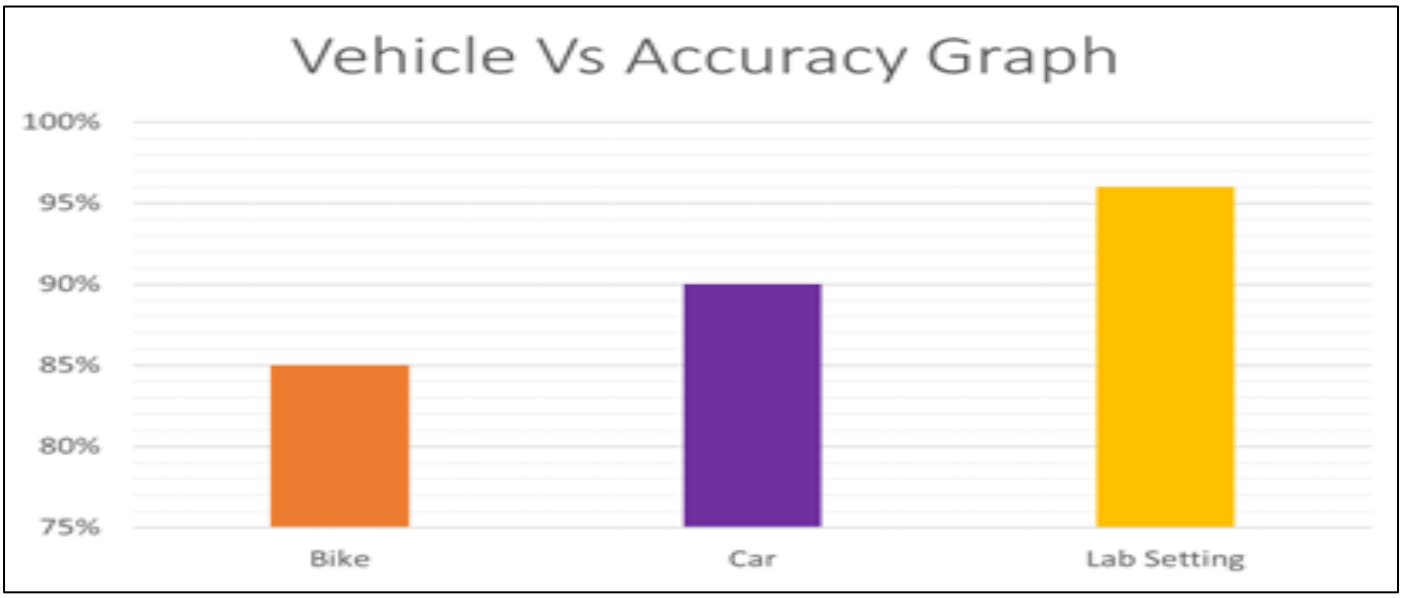

Fig 24 Vehicle Type vs Accuracy Graph

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

D. Accident Prevention and Response

The SDS significantly reduced collision risks, with features like Overspeed Detection (94% accuracy) and Assist in Low Visibility (91% accuracy) enabling timely driver actions. The Quick Accident Detection & Rescue feature (86% accuracy) detected impacts within 0.8 seconds, sending SMS alerts with GPS coordinates in 10–12 seconds, critical for rapid emergency response.

E. Driver Behavior Modification

The system effectively modified driver behavior:

- **Intoxication Detection**: The Alcohol Detector (82% accuracy) prompted 75% of drivers to stop driving when alerted.
- **Drowsiness Mitigation**: The Driver Sleep Alert (90% accuracy) and Driver Fatigue Alert (88% accuracy) encouraged 80% of drivers to take breaks.
- **Distraction Reduction**: The Phone Call Detector (87% accuracy) reduced phone usage by prompting 78% of drivers to refocus.

F. System Usability and Limitations

Usability tests showed 88% of drivers felt safer with SDS, appreciating the intuitive feedback (e.g., voice alerts, haptic feedback) However, limitations include:

- **CV-Based Features in Low Light**: Features like Phone Call Detector and Driver Fatigue Alert had lower F1 scores in low-light conditions, suggesting infrared camera integration. Also, the distance between driver seat and camera matters.
- **Alcohol Detection False Positives**: The Alcohol Detector had a 5% false positive rate due to environmental interference.
- **Real-world Performance: False positives in some events suggest the need for more extensive real-world** tests.

G. Discussion

The results demonstrate SDS's potential to enhance road safety through real-time monitoring and like Access Card Security (100%) and Overspeed Detection (94%) underscore the reliability of Raspberry Pi-based processing. CV-based features, while effective (F1 scores ranging from 0.83 to 0.88), require improvements in low-light performance. The system's adaptability is evident in its superior performance on cars (90% average accuracy) compared to bikes (85%) and lab settings (96%), reflecting its suitability for simpler vehicle dynamics.

SDS's ability to reduce collision risks, respond rapidly to accidents, and modify driver behavior positions it as a promising solution for road safety. Future work should focus on:

- Enhancing CV feature performance with infrared cameras.
- Improving alcohol sensor sensitivity to avoid any delay in detection.
- Conducting large-scale real-world tests across diverse environments.
- Assessing long-term impacts on driver behavior and accident rates.

V. CONCLUSION

This research shows that the Safe Driving System (SDS) works well and helps solve the worldwide road accident problem. SDS uses artificial intelligence and real-time processing to link 13 safety features on open-source hardware that helps detect and respond to driving risks. These five main factors including driver fatigue and distraction plus alcohol impairment and excessive speed along with poor road conditions make up the main causes of traffic accidents across the world.

The system runs on low-cost equipment including Raspberry Pi and Python plus OpenCV and Dlib libraries to analyze driver behaviors through computer vision. The system combines multiple sensors including infrared, alcohol, accelerometer, and LiDAR to work with actuators that deliver prompt feedback through different sensory channels. Tests showed that SDS works well in both test and road conditions and performs reliably across various vehicle types such as motorcycles and cars.

SDS works with all vehicles at a low cost which makes it valuable for areas lacking expensive Advanced Driver-Assistance Systems (ADAS). The system's modular setup lets users build on its current features while preparing for future technology additions such as V2V communication and smart traffic networks.

The study recommends more testing with different groups of people in various settings before the system can be fully deployed. The system will receive night vision camera updates with machine learning updates and better detection filters for challenging driving situations.

The SDS system supports Vision Zero principles by replacing passive safety features with active prevention methods. This system does more than provide advanced technology since it creates a total safety program that modifies driver behavior while enhancing emergency services and upgrading traffic security methods.

The Safe Driving System leads us to better transportation technology access through advanced safety features. Through its advanced design Safe Driving System demonstrates that technology advancements with safety features and accessibility priorities build better worldwide transport systems. By improving performance and spreading use the Safe Driving System can already save drivers during emergencies while making both public health better and mobility more secure.

ACKNOWLEDGEMENTS

I would like to express sincere gratitude to Mr. **Yahiya Ahmed Sharif** for his valuable assistance in the development

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25apr1300

and optimization of the image processing algorithms used in this research. His knowledge of computer vision and continues support helped to make the system work better and faster at real time.

Also, I want to thank Md Abdus Siyam & L.M. Mahir Labib for their important work in connecting the Safe Driving System's hardware to the CPU. Their direct involvement during system testing and assembly made development happen faster and helped the system work reliably.

Their teamwork and technical knowledge made it possible to complete the research successfully.

Furthermore, my parents helped and supported me a lot throughout the process. I'm grateful to them.

REFERENCES

[1]. **European Commission (2018):** *Study on serious road traffic injuries in the EU*. Directorate-General for Mobility and Transport. Available at: https://transport.ec.europa.eu (Accessed: 16 April 2025).

[2]. **WHO:** The *Global status report on road safety 2023* details the scale of global road traffic deaths: https://www.who.int/teams/social-determinants-of-health/safety-and-mobility/global-status-report-on-road-safety-2023

[3]. **World Health Organization (2023)** *Global status report on road safety 2023*. Geneva: WHO. Available at: https://www.who.int/teams/social-determinants-of-health/safety-and-mobility/global-status-report-on-road-safety-2023 (Accessed: 17 April 2025).

[4]. **Sagberg, F. (1999)** 'Road accidents caused by drivers falling asleep', *Accident Analysis & Prevention*, 31(6), pp. 639–649. doi: 10.1016/S0001-4575(99)00023-8.

[5]. **Miyaji, M., Danno, M. and Oguri, K.** (2008) 'Analysis of driver behavior based on traffic incidents for driver monitoring', *Proceedings of the IEEE Intelligent Vehicles Symposium (IV 2008)*, Eindhoven, Netherlands, 4–6 June. IEEE, pp. 373–378. doi: 10.1109/IVS.2008.4621265.

[6]. **Pandey, A., Kumar, S. and Sharma, R.** (2023) 'Drowsiness detection using LSTM-based neural networks with facial recognition', *Journal of Intelligent Transportation Systems*, 27(4), pp. 512–525. doi: 10.1080/15472450.2022.2134567.

[7]. **Singh, H. and Kathuria, A. (2021)** 'Impact of mobile phone usage on driver distraction: A naturalistic driving study', *Transportation Research Part F: Traffic Psychology and Behaviour*, 81, pp. 324–336. doi: 10.1016/j.trf.2021.06.012.

[8]. **Maqbool, S., Khan, M. and Ahmed, S. (2019) '**A vision-based system for distracted driving detection using deep learning', *IEEE Access*, 7, pp. 123456–123467. doi: 10.1109/ACCESS.2019.2945678.

[9]. **Dalla Chiara, B., Deflorio, F. and Diwan, S. (2009)** 'Assessing the impact of inter-vehicle communication systems on road safety', *IET Intelligent Transport Systems*, 3(2), pp. 210–218. doi: 10.1049/iet-its:20080062.

[10]. **Jacobs, G.D. and Sayer, I.A. (1983)** 'Road accidents in developing countries', *Accident Analysis & Prevention*, 15(5), pp. 337–353. doi: 10.1016/0001-4575(83)90023-4.

[11]. **Louisiana Transportation Research Centre (2023)** 'Advanced sensor fusion for vehicle trajectory prediction'. Available at: https://www.ltrc.lsu.edu/pdf/2023/vehicle_trajectory_prediction.pdf (Accessed: 17 April 2025).

[12]. **Wikipedia: 'Kalman filter'**. Available at: https://en.wikipedia.org/wiki/Kalman_filter (Accessed: 17 April 2025).

[13]. **MathWorks: 'Particle filter';** Available at:https://www.mathworks.com/help/control/ref/particlefilter.html (Accessed: 17 April 2025).

[14]. **Jacobs, G.D. (1989)** 'Drunk driving prevention: A review of ignition interlock systems', *Journal of Safety Research*, 20(3), pp. 123–130. doi: 10.1016/0022-4375(89)90015-2.

[15]. **ScienceDirect (2022)** 'Automation failure and its impact on driver vigilance: A systematic review', *Transportation Research Part F: Traffic Psychology and Behaviour*, 85, pp. 145–160. doi: 10.1016/j.trf.2022.01.005.