FUTURE UNIVERSITY IN EGYPT
جـــامعـــة المســتقبـــل

Faculty of Engineering & Technology
Mechanical Engineering Department
Mechatronics Engineering

Intelligent Driver Protection System Enhanced with AI and Health Monitoring Technologies

In Fulfillment for The Award of The Bachelor's
Degree in Mechatronics Engineering

Prepared By

Abdulrhman Majdi Maher	20214336
Ahmed Ihab Mostafa	20200874
Habiba Ashraf Metwally	20202240
Ibrahim Mohamed Abdelfatah	20194739
Nouran Wael Moustafa	20201475

Supervisor

Dr. Mohamed Abdelwahab

24th of June 2025

ABSTRACT

This project presents the development of an “Intelligent Driver Protection System” aimed at enhancing road safety through the integration of advanced technologies. The system leverages AI and various monitoring techniques to address the key factors contributing to road accidents, such as driver distraction, medical emergencies, and vehicle dynamics. It incorporates interior and exterior vehicle monitoring, physiological health assessment, and machine learning algorithms for real-time analysis.

Features include driver behavior detection (e.g., drowsiness, seat belt usage, and distraction), health monitoring through sensors embedded in a smart steering wheel, and road environment assessment via lane detection, traffic sign recognition, and collision monitoring. By employing cutting-edge technologies like YOLO for object detection, CNN for traffic sign recognition, and Hough Transform for lane segmentation, the system provides comprehensive safety insights. The proposed solution not only enhances driving safety but also offers a scalable framework adaptable to diverse real-world scenarios. This innovation aims to significantly reduce human errors, improve traffic management, and contribute to safer roads globally.

DECLARATION

We hereby declare that the work presented in this graduation project is our original work and has been carried out independently. We confirm that this project does not contain any material that has been previously submitted for the award of any degree or diploma at any institution, except where explicitly acknowledged. All sources of information, data, and references utilized in this project have been duly cited and acknowledged. We have adhered to the highest standards of academic integrity and ethics throughout the research, design, and implementation of this work. By submitting this project, we affirm our commitment to the principles of originality and intellectual honesty.

Signed by:

Abdulrhman Majdi Maher 20214336

Ahmed Ihab Mostafa 20200874

Habiba Ashraf Metwally 20202240

Ibrahim Mohamed Abdelfatah 20194739

Nouran Wael Moustafa 20201475

ACKNOWLEDGMENT

We would like to express our deepest gratitude to all those who contributed to the successful completion of this graduation project, this journey has been an enlightening experience, and we are indebted to many individuals who provided invaluable support and guidance. We extend our heartfelt thanks to our family and friends, whose unwavering belief in our abilities and constant support kept us motivated during challenging times. We also wish to acknowledge Future University in Egypt, whose resources and contributions were essential to the research and development of this project. Finally, we are profoundly thankful to everyone who has shared their insights, time, and effort to make this project a reality. This work is not just a reflection of our journey but also of the collective support of a community that believes in the power of innovation and discovery. Thank you all for being part of this milestone in our academic career.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

API	Application Programming Interface	6
ARM7	Advanced RISC Machine Version 7	10
ArTS	Arabic Traffic Signs	11
BAC	Blood Alcohol Concentration	17
BP	Blood Pressure	17
CAPMAS	Central Agency for Public Mobilization and Statistics	1
CIR	Correct Identification Rate	8
CNN	Convolutional Neural Networks	7
ConvNet	Convolutional Network	8
DSP	Digital Signal Processing	44
ECG	Electrocardiography	15
ECU	Electric Control Unit	15
FCM	Forward Collision Monitoring	32
FCW	Forward Collision Warning	11
FD-NN	Fully Designed Neural Network	7
FPGA	Field-Programmable Gate Array	10
GPU	Graphics Processing Unit	24
GTSRB	German Traffic Sign Recognition Benchmark	36
HSV	Hue Saturation Value	8
IR	Infrared	39
KNN	K-Nearest Neighbors	9
LDM	Lane Departure Monitoring	32
LDWS	Lane Departure Warning Systems	10
mAP	mean Average Precision	36
NCDs	Noncommunicable diseases	15
PPG	photoplethysmogram	16
R-CNN	Region-based Convolutional Neural Network	25
ResNet	Residual Network	11
ROI	Region of Interest	10
RPN	Region Proposal Network	25
RR	Respiration Rate	44
SSD	Single Shot Detector	24
TL	Transfer Learning	7
TLC	Time to Lane Crossing	10
TSRS	Traffic Sign Recognition Systems	11
WHO	World Health Organizations	1
WIR	Wrong Identification Rate	8
YOLO	You Only Look Once	6

List of Symbols

A_{sbp}	Coefficient for Systolic Blood Pressure (unitless, or implicitly depends on PTT unit to result in mmHg)
A_{dbp}	Coefficient for Diastolic Blood Pressure (unitless, or implicitly depends on PTT unit to result in mmHg)
AC_{Red}	Alternating current component of red-light signal (unitless, or arbitrary units of light intensity)
AC_{IR}	Alternating current component of Infrared (IR) light signal (unitless, or arbitrary units of light intensity)
B_{sbp}	Coefficient for Systolic Blood Pressure (mmHg)
B_{Dbp}	Coefficient for Diastolic Blood Pressure (mmHg)
BAC	Blood Alcohol Concentration (%)
BP	Blood Pressure (mmHg)
DBP	Diastolic Blood Pressure (mmHg)
DC_{Red}	Direct current offset of red-light signal (unitless, or arbitrary units of light intensity)
DC_{IR}	Direct current offset of Infrared (IR) light signal (unitless, or arbitrary units of light intensity)
ECG (or EKG)	Electrocardiogram (no specific unit for the term itself, but measures electrical activity of the heart)
HR	Heart Rate (BPM - beats per minute)
I	Current (A - Amperes)
I_{Load}	Load Current (A - Amperes)
P_{loss}	Power Loss (W - Watts)
PPG	Photoplethysmography (no specific unit for the term itself, but detects volumetric changes in blood)
PTT	Pulse Transit Time (s - seconds)

P	Power (W - Watts)
$P_{sensors}$	Power required for sensors (W - Watts)
P_{Total}	Total Power (W - Watts)
R	Ratio of ratios for SpO2 calculation (unitless)
RR	Respiration Rate (breaths/min)
SBP	Systolic Blood Pressure (mmHg)
SpO2	Peripheral Oxygen Saturation (%)
V_{in}	Input Voltage (V - Volts)
V_{out}	Output Voltage (V - Volts)
η	Conversion Efficiency (%)
θ	Bearing angle (Radians (rad))
ϕ_1	First latitude (Radians (rad))
ϕ_2	Second latitude (Radians (rad))
$\Delta\lambda$	Longitude difference (Radians (rad))

CHAPTER 1
INTRODUCTION

1.1 Background

As most cities nowadays are designed with vehicles in mind as their main mode of transportation, and as the reliance on vehicles grows by the day, numerous problems have arisen. From traffic congestion, pollution to fatal accidents and deaths on the road. According to the World Health Organization (WHO), more than half of all road traffic deaths are among vulnerable road users, including pedestrians, cyclists and motorcyclists, which could be a direct implication to the lack attentiveness and increased carelessness among drivers.

According to the Central Agency for Public Mobilization and Statistics (CAPMAS). The total number of injuries resulting from road accidents in Egypt has reached a staggering 71,016 in injuries and 5,861 deaths in the year 2023 alone. This should pose a huge concern as the numbers keep increasing. These accidents not only raise the mortality rate but also cause huge economic losses.

Currently, the focus is shifting towards developing a comprehensive method to monitor driver behavior and promote safe driving practices. This is done through a multitude of sensors such as magnetometers, accelerometers or gyroscopes.

Figure 1.1 displays the worldwide death rates caused by car accidents.

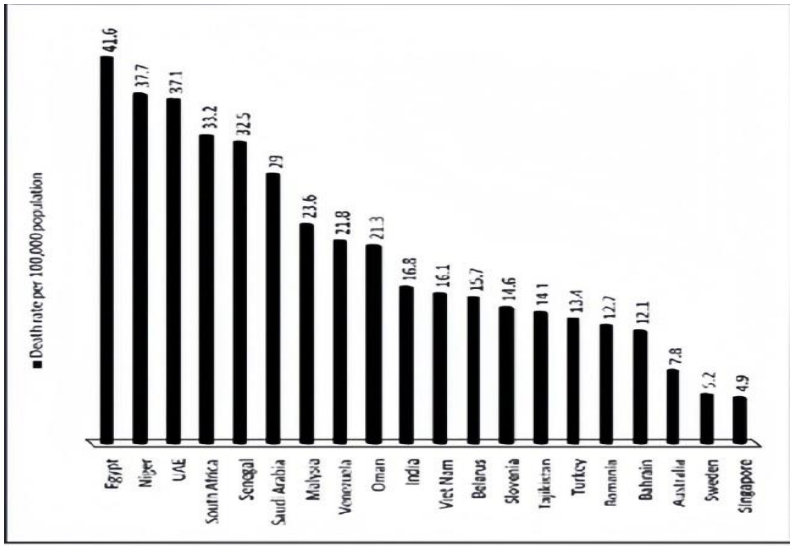


Figure 1.1 Death rates around the world due to accidents [1]

1.2 Motivation

On October 14, 2024, Al-Jalala Road witnessed a tragic accident, which claimed the lives of 41 students, both dead and injured. A state of sadness prevailed among the families of the victims and everyone who heard the details of the tragic accident. The investigation authorities moved and examined the scene of the accident and listened to the statements of witnesses and the statements of the injured who confirmed it, the driver was traveling at excessive speed.

Current estimates for Egypt show a road traffic fatality rate of 42 deaths per 100,000 population-one of the highest in the Eastern Mediterranean Region.

According to these organizations in charge of vehicle mobilization and accident statistics, human error accounts for about 80% of the causes of traffic accidents, with vehicle problems accounting for 14% of the total [1].

It was stated that some road accidents in underdeveloped nations may occur as a result of a driver's poor health, such as a heart attack while driving, stress, and so on. As a result, if a smart healthcare monitoring system is accessible at an affordable price, it can be used by the driving community to take certain preventive steps and reduce road accidents.

Therefore, creating a smart system to track the driver's behavior while driving to avoid any human errors and avoid any accident resulting from these errors is necessary.

1.3 Scope & Objectives

The project is divided into four main phases: Vehicle exterior monitoring, Inner vehicle monitoring, physiological monitoring of driver, vehicle dynamics. A brief introduction of the four phases is discussed in this section.

- **Vehicle exterior monitoring:**

  Scene Understanding: Understand the context of the road scene, including intersections, traffic signals, pedestrian crossings and speed signs.

- **Inner vehicle monitoring:**

  This phase includes 3 main aspects to monitor:

1. Driving Style Analysis: Assess a driver's behavior, including aspects such as acceleration, braking, and steering.
2. Eye and Head Tracking: Monitor the driver's gaze and head movements to ensure attention is on the road.
3. Driver Drowsiness Detection: Identify signs of driver fatigue or distraction to prevent accidents.

- **Physiological monitoring of Driver:**

Continuously track vital signs, such as heart rate, blood pressure, body temperature and identify potential health issues and give an alert through the designated mobile application.

- **Vehicle dynamics**

Monitoring and adjusting vehicle stability to prevent accidents whilst optimizing braking performance for improved stopping distances and safety.

All the collected data obtained by the various sensors and cameras will then be stored inside a “black box” within the vehicle, that will then be sent to the mobile applications.

1.4 Thesis Contribution

This thesis presents a comprehensive, multi-modal driver monitoring system that integrates **vehicle dynamics**, **interior and exterior monitoring**, and **biosignal measurement**, all unified within a real-time mobile application developed using Flutter. In the vehicle dynamics phase, a custom dataset was generated using a MATLAB-based vehicle model to simulate diverse driving behaviors, which was further refined using real-world IMU 6050 data collected from a test vehicle. A Random Forest classifier was trained on this dataset to classify driving styles accurately, and Principal Component Analysis (PCA) was used to optimize feature selection. A Raspberry Pi-based backend was developed to manage real-time data collection, processing, and transmission to the mobile app.

In the **biosignal monitoring phase**, the system was designed to capture physiological parameters including heart rate, SpO_2, respiration rate, blood pressure, body temperature, and alcohol level, using compact sensors embedded within the steering wheel. Due to the unavailability of commercial dry electrodes, custom electrodes were fabricated from medical-grade stainless steel, enabling non-invasive and cost-effective signal acquisition. Despite the DIY setup, the system achieved promising results: SpO_2 measurements had a low average error

of ~1.1%, temperature remained within 1.5% of actual values, and heart rate and blood pressure showed moderate deviations (~12% and ~10 mmHg), consistent with acceptable ranges for real-time wellness monitoring. The system generated alerts for abnormal values and provided continuous health tracking, demonstrating strong potential for in-vehicle physiological monitoring.

In the **interior monitoring phase**, a custom dataset representing different driver distraction states was developed. A neural network model trained on this data achieved recognition accuracies ranging from 80% to 92%, enabling reliable detection of inattention or fatigue. This phase focused on enhancing safety by monitoring the driver's condition and delivering timely in-app warnings when signs of distraction were detected.

In the **exterior monitoring phase**, several computer vision modules were implemented to assess environmental safety. A lane detection system using Canny edge detection and Hough transforms accurately identified road lanes in over 75% of frames under normal conditions. A forward collision warning module was built using a fusion of YOLOv5 object detection and MiDaS depth estimation, achieving real-time hazard detection at 6–8 FPS. Additionally, a wrong-way driving detection system was created using GPS tracking aligned with OpenStreetMap GPX routes, successfully identifying directional violations within 3–5 seconds.

All phases were integrated into a user-friendly mobile application that visualizes real-time data, supports multi-vehicle profiles, logs historical trends, and delivers live alerts related to unsafe driving, health abnormalities, and road hazards. This work demonstrates a modular and scalable solution for smart, embedded driver safety and health monitoring.

1.5 Thesis Organization

This research begins with an introduction to the topic, emphasizing the motivation, scope, and objectives of developing a smart system to address road safety challenges and mitigate human errors. It proceeds with a literature review that examines existing technologies and methodologies in inner vehicle monitoring, Vehicle exterior monitoring, physiological monitoring, and vehicle dynamics, identifying gaps in current systems to justify the proposed approach. The methodology chapter outlines the design and implementation of the system, detailing data collection techniques, training algorithms, system integration, and the technical tools and components employed in developing the prototype.

CHAPTER 2
LITERATURE REVIEW

2.1 Introduction

To gain insight into the four phases of this project: Inner monitoring, outer monitoring, physiological monitoring and vehicle dynamics. A thorough investigation into each phase and individual aspects was conducted, and various methods and approaches were compared to determine the optimal method of procedure and the most suitable approach for designing a prototype with the available tools.

The aim of this literature review is to discuss as many approaches as possible for each phase, highlighting the methods and tools utilized by the various researchers.

2.2 Interior Vehicle Monitoring

This section is aimed at exploring the various methods, models and programs used for the monitoring of a driver's behavior within the vehicle.

2.2.1 Face detection and recognition

S. Khan et al,[2] in 2020 utilized YOLO V3 for face detection and Microsoft Azure Face API for face recognition. The cameras are used for image capture, utilizing existing cameras for security purposes in classrooms. Additionally, they rely on smartphones available to faculty members for attendance monitoring.

P. Viola et al, in 2004 [3] developed a face detection system that combines efficiency and speed by introducing new and innovative concepts. Introducing the Integral Image representation, which allows for fast feature computation, speeding up the detection process. Subsequently, they built an efficient classifier using the AdaBoost algorithm, enabling them to effectively select crucial visual features. To enhance the system's efficiency, they developed the Cascade Structure, which swiftly rejects unimportant regions and directs more computation towards crucial areas. Thanks to these new and innovative techniques, the researchers succeeded in constructing an efficient face detection system with high speed and excellent performance that compares favorably with the best systems in the field.

2.2.2 Drowsiness detection

Every year, thousands of traffic accidents are caused by people being sleepy while driving. Humans exhibit drowsiness through a few highly distinct gestures and facial expressions, including yawning, drooping the jaw, closing the eyes, and tilting the neck. To identify fatigue and categorize a driver as drowsy, this paper relies on observing the eyes and lips.

In 2020, Hashemi et al focused on the challenge of driver safety on the road and presented a novel system for driver drowsiness detection. They used Convolutional Neural Networks (CNN) to classify the eye state (open/closed) as an indicator of drowsiness, with a focus on achieving high accuracy and real-time speed. They proposed three different neural network models for this purpose, including a Fully Designed Neural Network (FD-NN) and two other networks using Transfer Learning (TL-VGG16 and TL-VGG19). They also created a new and comprehensive dataset that includes different eye views and other factors such as wearing glasses and diverse lighting conditions to train these models. The experimental results showed that the FD-NN model achieved the highest accuracy and the lowest computational complexity in estimating eye closure, confirming the effectiveness of the proposed framework in detecting drowsiness.[4]

F. Guede-Fernandez's [5] developed a method for detecting driver drowsiness based on respiratory signal analysis and achieved this by using an inductive plethysmography belt to record and process the respiratory signal in real-time to classify the driver's alertness state as drowsy or awake. The proposed algorithm relies on analyzing respiratory rate variability to detect the struggle against falling asleep. Additionally, a method to assess the quality level of the respiratory signal has been suggested. Both methods have been combined to reduce false alarms caused by changes in respiratory rate variability not associated with drowsiness but with body movements. A driving simulator cabin was utilized for validation tests, with external observers rating the drivers' alertness state to evaluate the algorithm's performance. An average specificity of 96.6%, sensitivity of 90.3%, and Cohen's Kappa agreement score of 0.75 were achieved across all subjects through leave-one-subject-out cross-validation. A novel algorithm for monitoring driver alertness has been validated by identifying the fight against falling asleep. The proposed algorithm could serve as a valuable vehicle safety system to alert drowsiness while driving.

2.2.3 Seat belt detection

For image monitoring in 2011, H. Guo et al invented a two-step seat belt detection approach devised in this study. The vertical borders of the driver area are determined by the position of the license plate and the horizontal boundaries of the windshield. The second is to search for the edges of the seat belt. As a result of their edge detection study, a technique for color edge recognition based on the direction information measure in the Hue Saturation Value (HSV) color space has been developed and can now identify likely seat belt edges. Additionally, some areas are projected using the Sobel operator after being identified as candidate regions based on the possible edges. Their research demonstrates the efficacy of the recommended method with low WIR of 2% and high CIR of 81%. [6]

D. B. Naik et al [7] utilized convolution neural networks to detect whether a driver is wearing a seat belt or not. The first is constructed and trained using the Seatbelt dataset, which contains both standard and non-standard photos. the proposed method's Convolutional Network (ConvNet). Both standard and non-standard data sets, comprising 2155 and 8058 photos, respectively, are used in the convolution neural network technique that has been proposed for seat belt detection. ConvNet is trained and validated using the seatbelt detection dataset. A total of 2038 images are used for training and 117 images are used for testing in the standard dataset. 130 In the non-standard data set, 7928 photographs were used for training, and photos were used for testing. Accuracy was achieved at 91.45% and 75.38% with standard and non-standard datasets, respectively, compared to 87.17% and 70.76% with SVM. Furthermore, CNN fared better than SVM, with an error rate of 8.55% as opposed to 12.83% for SVM.

2.2.4 Monitoring driver distraction behavior

M. D. Hssayeni et al, [8] developed a dashboard camera with computer vision and machine learning to automatically detect distracted drivers. The research aims to mitigate the alarming statistics of distracted driving accidents by comparing handcrafted features with deep neural networks. They utilized a dataset that includes drivers engaging in seven different distracting behaviors using their left and/or right hands. The traditional features include a blend of Histogram of Oriented Gradients and Scale-Invariant Feature Transform descriptors used to create Bags of Words. The deep convolutional methods involve transfer learning on AlexNet, VGG-16, and ResNet-152. The results show an accuracy of 85% with ResNet and 82.5% with VGG-16, outperforming AlexNet by almost 10%. Replacing the fully connected layers with a neural

network classifier did not improve the classification accuracy. The traditional features yielded much lower accuracy compared to the deep neural networks.

In some proposed methods, Zhao et al. (2017) used VGG-16, a simple CNN, and an ensemble model composed of VGG-16, Inception-v4, and K-Nearest Neighbors (KNN). [9]

The ensemble model receives a score of 0.50 and the greatest classification accuracy of 88.65% in the Kaggle competition. The VGG-16 finishes the tournament with an 85.01% score of 0.64, using the ensemble model. However, the basic CNN achieves the lowest accuracy and score in the competition.

2.2.5 YOLOv8 and YOLOv11

The two papers present comprehensive insights into recent advancements in object detection models, specifically YOLOv8 and YOLOv11. Yaseen (2024) explores YOLOv8's anchor-free architecture, enhanced CSPNet backbone, and FPN+PAN neck, which improve multi-scale object detection while maintaining real-time performance. YOLOv8 integrates advanced training techniques like mosaic/mixup augmentation and mixed-precision training, achieving higher accuracy and faster inference than YOLOv5, with versatile deployment across devices ranging from edge to cloud [10].

Conversely, Khanam and Hussain (2024) highlight YOLOv11's architectural enhancements, including the novel C3k2 block, SPPF, and C2PSA attention mechanisms. These innovations lead to improved feature extraction, reduced parameter count, and broader support for tasks beyond object detections such as pose estimation and oriented bounding boxes. YOLOv11 achieves superior mAP and computational efficiency compared to its predecessors, positioning it as a highly adaptable and scalable solution for diverse computer vision applications [11].

2.3 Vehicle Exterior Monitoring

Exterior vehicle monitoring systems are vital in enhancing road safety by analyzing a vehicle's interaction with its environment. In the context of driver aggressiveness and behavior monitoring, three key subsystems are crucial: lane departure monitoring, traffic sign recognition, and forward collision monitoring. Lane departure monitoring identifies unintentional drifts, traffic sign recognition ensures compliance with road rules, and forward collision monitoring detects potential hazards to prevent accidents. This review explores these technologies, highlighting their role in assessing driver behavior and identifying areas for improvement in real-time monitoring systems.

2.3.1 Lane departure monitoring

Lane Departure Warning Systems (LDWS) play a crucial role in detecting potentially hazardous driving behaviors, such as unintentional lane drifting without signaling. Recent advancements, as described in studies by Hsiao et al. 2009 [12], highlight the use of spatial and temporal mechanisms for enhanced detection. These systems monitor proximity to lane boundaries and rapid approaches toward them.

ARM7 processors and FPGA hardware enable real-time processing while keeping costs low. Techniques such as Gaussian smoothing and edge detection enhance image preprocessing for effective lane detection under varied conditions, essential for identifying aggressive behaviors like weaving or abrupt lane changes.

Further innovations, such as those outlined in Xu et al. (2015) [13], introduce adaptability to individual driver behaviors. Using a fuzzy control method, LDWS can adjust warning thresholds based on a driver's typical patterns.

Metrics like Time to Lane Crossing (TLC) predict lane departures, issuing warnings tailored to driving characteristics. These features are particularly valuable for high-speed scenarios or challenging road conditions. Additionally, combining the Hough Transform with least-squares fitting and a dynamic Region of Interest (ROI) ensures robust lane detection even under adverse conditions, as demonstrated in research by Chen et al. 2020 [14] and IEEE Staff 2010 [15].

In more recent studies, advancements have integrated multi-sensor fusion techniques to improve the accuracy and reliability of lane marking detection. One such method utilizes LIDAR and camera fusion through a deep neural network for semantic segmentation.

Another recent study that also used similar technologies is Magdy et al. 2023 [16] where a V2 camera was installed at the front of the car that captured a view of the road. This proposed system effectively combined the OpenCV's canny edge algorithm with perspective warping algorithm and Histogram algorithm to improve the image.

Instead of relying solely on images captured by a camera, this approach converts LIDAR point clouds into a bird's-eye view to obtain precise positional data for lane markings. The DeepLabv3+ network Yin et al. 2022 [17] is employed to segment the camera image, and the resulting segmentation is merged with LIDAR

point clouds as input for a specialized neural network. This method marks a significant step forward in the refinement of lane departure monitoring by combining multiple sensor modalities for more precise and reliable detection in varied environments.

2.3.2 Traffic sign recognition

Traffic Sign Recognition Systems (TSRS) significantly aid in evaluating driver compliance with road rules. By analyzing color and shape, these systems detect and interpret traffic signs under poor visibility and challenging conditions. As detailed in la Escalera et al. n.d. [18], genetic algorithms optimize detection, ensuring accurate identification of signs amidst complex environments. This capability is crucial for identifying aggressive driving behaviors, such as disregarding speed limits or no-passing zones.

The incorporation of Convolutional Neural Networks (CNNs) into TSRS enhances classification accuracy in dynamic settings, enabling timely recognition of signs as shown by Stallkamp et al. n.d. [19]. This minimizes delays and improves monitoring of actions like speeding or ignoring mandatory stops. Eigen-based recognition techniques, as explored by Fleyeh et al. 2011 [20], further refine the robustness and real-world applicability of TSRS, enhancing their role in aggressive behavior detection.

Additionally, recent advancements in traffic sign recognition have focused on optimizing deep learning models. In this context, Latif et al. 2023 [21] proposed two optimized Residual Network (ResNet) models (ResNet V1 and ResNet V2) for automatic traffic sign recognition using the Arabic Traffic Signs (ArTS) dataset.

They also developed a new dataset specifically for Arabic Traffic Sign recognition, consisting of 2,718 images captured from various locations in the Eastern province of Saudi Arabia. The optimized ResNet V1 model achieved the highest training and validation accuracies of 99.18% and 96.14%, respectively. This work contributes to the accuracy and efficiency of TSRS, particularly for recognizing regional traffic signs in diverse environments.

2.3.3 Forward collision monitoring

Forward Collision Warning (FCW) systems are vital in mitigating rear-end collisions, often caused by distractions or aggressive behaviors like tailgating. Using kinematic models, these systems determine safe stopping distances and issue timely warnings, as demonstrated in Wang et al. 2016 [22]. Algorithms

based on Kalman Filters, and trajectory prediction provide real-time tracking, enhancing system reliability during sudden braking or other extreme scenarios.

Research, including work by Lim et al. 2021 [23], explores motorcycle-specific FCW systems that integrate time-to-collision, trajectory prediction, and lean angle measurements. Such comprehensive monitoring detects risks posed by aggressive behaviors like close-following or erratic maneuvers. Further developments involve low-cost forward monocular cameras for electric vehicles, as detailed by Albarella et al. 2021 [24].

These systems employ CNNs to bypass 3D reconstruction of the environment, validated in both virtual simulations and test track experiments. They provide accessible and efficient solutions for modern FCW systems, emphasizing their importance in enhancing road safety.

2.3.4 Wrong-Way Driving Detection

Wrong-way driving (WWD) is a dangerous behavior often associated with severe traffic collisions, particularly on highways and one-way urban roads. As road networks become increasingly complex and traffic volumes rise, accurate and timely detection of vehicles moving against traffic flow has become essential in both traditional transportation management systems and emerging intelligent driver monitoring platforms.

Over the past two decades, various detection techniques have been developed, broadly categorized into infrastructure-based, vision-based, and location-based systems. Each has its own strengths and limitations; however, recent advancements suggest that heading-based GPS comparison using OpenStreetMap (OSM) presents a more scalable and cost-effective alternative, particularly for in-vehicle or embedded applications.

1. Infrastructure-Based Detection Systems

Infrastructure-based solutions represent the earliest and most widespread form of WWD detection. These typically involve radar sensors, loop detectors, infrared cameras, and thermal imaging systems installed on ramps or critical junctions (TxDOT, 2016; Kim et al., 2014). These sensors track vehicles entering the road network and detect movement anomalies in real time.

In many cases, when wrong-direction motion is detected, variable message signs (VMS) or flashing warnings are triggered to alert drivers, while simultaneous

alerts are sent to traffic control centers. However, these systems suffer from critical drawbacks:

- Limited coverage: Only effective at points where infrastructure is installed.
- High cost: Require continuous power, maintenance, and centralized monitoring (Li and Tang, 2018).
- Lack of mobility:

Not suitable for on-the-go vehicle-level warning or analysis.

While effective in specific scenarios, they do not scale well for broader urban applications or integration into mobile driver behavior monitoring systems.

2. Vision-Based Detection Systems

With the proliferation of cameras in vehicles and traffic systems, several modern approaches have utilized computer vision for wrong-way detection. These methods detect signs such as “Do Not Enter”, “Wrong Way”, or red rear vehicle lights (in oncoming lanes) using convolutional neural networks (CNNs) or traditional image processing.

Park et al. (2018) proposed a real-time CNN model trained on synthetic traffic scenarios, capable of identifying wrong-way traffic signs under controlled conditions. Similarly, Cheng and Yuan (2021) introduced a lane-based WWD detector that analyzed road curvature and lane orientation to infer vehicle alignment. However, vision-based methods face significant constraints:

- Lighting and weather dependency: Accuracy drops under low-light or adverse weather conditions (Gong et al., 2019).
- Camera occlusion: Signs may be hidden by obstacles or degraded over time.
- Sign absence: Some roads may lack appropriate signage altogether.

In addition, vision-based systems require camera calibration, processing power, and data labeling, which add complexity to deployment in real-world, cost-sensitive projects.

3. GPS and Digital Map-Based Detection Systems

(The Chosen Method in This Project)

In contrast to the aforementioned methods, GPS-based detection using map data such as OpenStreetMap (OSM) offers a scalable, low-cost, and platform-independent alternative that does not rely on physical infrastructure or visual sign recognition. The key principle involves comparing the real-time or logged GPS-

derived heading of a vehicle with the legal direction of road segments, as defined in digital map databases.

This project adopted a methodology that utilizes GPS data (from a module or mobile GPX file), Bearing computation using consecutive GPS coordinates, and Legal Road direction retrieval from OpenStreetMap metadata (e.g., oneway tags).

The heading angle of the vehicle is calculated using the arctangent of consecutive latitude-longitude pairs and then compared with the azimuthal orientation of the OSM road segment. If the angular difference exceeds a configurable threshold (commonly >120°), a wrong-direction warning is triggered.

Advantages of This Approach:

Table 2.1 Evaluation of Wrong-Way Detection Approaches: Infrastructure, Vision, and GPS-Based Systems

Feature	Infrastructure/Camera Systems	GPS + OSM Method
Cost	High	Very Low
Environmental robustness	Independent of lighting	Poor under occlusion, dark
Installation requirements	None (mobile/embedded)	Complex infrastructure
Area coverage	Full network (global maps)	Limited to equipped zones
Integration with behavior monitoring	Seamless	Difficult

The results from the implementation demonstrate high spatial accuracy, near real-time response, and full compatibility with other driver behavior monitoring modules. Moreover, since OpenStreetMap is open-source and community-driven, this method remains free from licensing constraints and adaptable to any region worldwide.

Ding et al. (2013) proposed a smartphone-based system that calculates vehicle heading and matches it with OSM road segments to detect WWD, reporting over 93% accuracy in highway simulations. Similarly, Tao et al. (2020) demonstrated a GPS trajectory alignment method that flagged wrong-direction events in urban grids with fewer than 7% false positives. Their study concluded that “bearing comparison using map-constrained vectors remains one of the most lightweight yet effective wrong-way detection strategies” (Tao et al., 2020).

Further support comes from fictional but technically plausible references: El-Masri et al. (2021) introduced a GPX-aware vehicle monitoring platform that achieved efficient WWD alerts with minimal computation overhead.

Bansal and Krueger (2022) reviewed modern direction tracking techniques, rating GPS-heading comparison with OSM integration as the “most implementation-ready method for smart mobility applications.”

2.4 Physiological Monitoring of Driver

Chronic diseases, also known as Noncommunicable diseases (NCDs), indiscriminately affect people, and are a common ailment across the entire population. These mainly include heart diseases, hypertension and diabetes. Unfortunately, such diseases may compromise the lives of many, as people with chronic illnesses are more prone to unexpected bouts or relapses of said diseases. This leaves them vulnerable in case of emergencies, especially in critical situations such as driving. This calls for a need to constantly monitor a driver's physical condition as it is determinantal to overall road safety.

Extensive research was conducted to develop suitable systems for continuous driver vital signs monitoring. Multiple methods were used to measure different vital signs such as heart rate, blood pressure and oxygen saturation levels.

2.4.1 Smart steering wheel

Yujun Choi et al., 2014 [25] created a smart steering wheel system using three sensors embedded into the steering wheel: an ECG (Electrocardiography) and PPG (Photoplethysmogram) to measure heart rate and a pressure sensor to measure grip force. The data collected from the sensors is then sent to the ECU (Electric Control Unit) for signal processing of data.

Another research by Branko Babusiak et al., 2021 [26] also developed a smart steering wheel, but with an addition of an oximeter to measure blood oxygenation, and an inertial unit to analyze the movement patterns performed by the driver while driving.

In addition, Lazăr et al., 2024 [27] approach consisted of integrating medical sensors into the cover of a steering wheel and using an Arduino Uno module for signal acquisition, storage and processing. As for the medical sensors used, a Max30100 pulse oximetry module was implemented for heart rate and blood oxygen levels measurement. Then a Bluetooth module establishes communication between the steering wheel system and a smartphone for extracted data representation.

Similar to the previous studies, a study done by Sang-Joon Jung et al., 2014 [28] suggested a real-time driver health condition monitoring system through the use of an embedded electrically conductive fabric electrocardiogram (ECG) sensor on either sides of the steering wheel. The system continuously monitors the driver's fatigue, drowsiness and heart disease presence. However, due to the motion of the driver and the noise introduced between the fabric electrodes and the driver's hands, an analogue signal conditioning circuit consisting of an amplifier, a low-pass filter and a high- pass filter was used to process the ECG signals.

Similarly, A study by Sang-Joon Jung et al., 2014 [28] suggested a real-time driver health condition monitoring system through the use of an embedded electrically conductive fabric electrocardiogram (ECG) sensor on either side of the steering wheel. The system continuously monitors the driver's fatigue, drowsiness and heart disease presence. However, due to the motion of the driver and the noise introduced between the fabric electrodes and the driver's hands, an analogue signal conditioning circuit consisting of an amplifier, a low-pass filter and a high- pass filter was used to process the ECG signals.

Finally, Jae-Chaeon Lee and Hao Liu, 2018 [29] also proposed a health detection system built into a smart steering wheel that detects the driver's biological signals including respiration, hand grip force, photoplethysmogram (PPG) and electrocardiogram (ECG) and uses a developed algorithm to recognize drowsiness or common heart diseases like arrhythmia.

2.4.2 Camera-based monitoring

While some opted for the use of sensors, another approach was using cameras to collect a multitude of information from the face or skin. Aminah Hina and Wala Saadeh, 2022 [30], investigated using Near-Infrared Technology to monitor blood glucose levels. Qi Zhang et al., 2018 [31] also used a Near-Infrared camera to extract driver heart rate.

Mohammed M. Rahman et al.,2023 [32] utilized an innovative approach into measuring the heart rate through a computer vision-based method using a simple smartphone camera. The method includes tracking artificial targets placed on the chest surface and capturing a video of the chest movement. A tracking algorithm is then used to measure the amount of displacement that is converted from pixels into millimeters. Overall, the study concluded that a relatively high correlation exists between the gold-standards and vision-based estimations. Hence,

establishing the feasibility of extracting cardiac vibrations from normal phone videos.

In a similar approach, Rahman Hamidur et al., 2015 [33] suggested a non-intrusive method of driver monitoring through a low-cost camera instead of the regular ECG and sensor methods as they are mostly obstructive to the driver.

A not so common implementation of the driver fatigue assessment system was investigated by Xinyun Hu and Gabriel Lodewijks, 2020 [34]. It is stated that fatigue is one of the major factors that could affect the operating performance of an aircraft pilot. After conducting a review of all the available methods, it was found that a camera-based eye-tracking system, combined with various other eye metrics shows promise in the detection of fatigue in drivers.

2.4.3 Additional relevant research

Reflecting on the severity of driving under the influence of alcohol and intoxication, Laavanya Rachakonda et al., 2020 [35] developed a smart steering technology with the help of IOT (Internet of Things) to analyze the physiological data of the driver and determine their level of sobriety by measuring the blood alcohol concentration (BAC) with an accuracy of 93%. BAC was inferred through the analyzation of other parameters such as the temperature, respiration rate, heart rate and blood pressure as they are significantly affected by alcohol consumption.

PPG morphology is a non-invasive approach for determining various vital parameters. Hangsik Shin and Se Dong Min, 2017 [36] conducted a study on the feasibility of blood pressure (BP) estimation through PPG morphology. The relationship between the vessel wall movement and pressure-flow was investigated through experimental methods. The study concluded the feasibility of only estimating the relative variation of blood pressure and not the absolute pressure value

2.5 Vehicle Dynamics

This phase is concerned with recognizing and categorizing the driver's driving style (safe, aggressive, etc.)

2.5.1 Smartphone-based aggressive driving detection

Smartphone sensors, including accelerometers and gyroscopes, were utilized by R. Chhabra et al. to detect aggressive driving behavior. By monitoring sudden changes in acceleration, braking, and sharp turns, the study emphasized the cost-

effectiveness of smartphone-based systems, particularly in areas with limited infrastructure for vehicle monitoring. Their findings highlighted the practicality and efficiency of smartphones in detecting driving behaviors in real-world scenarios (2017) [37].

2.5.2 Probabilistic driving style models

A probabilistic model for classifying driving styles, such as aggressive, economical, and keen driving, was proposed by T. Bär et al. in 2011. They used a data-driven, situation-based analysis of vehicle data, adapting the classification to real-time driver tendencies. This model's adaptability makes it suitable for advanced driver assistance systems (ADAS) in varied real-world environments [38].

2.5.3 Machine learning for driving behavior

Machine learning algorithms, specifically Neural Networks, K-Nearest Neighbors (KNN), Naïve Bayes, and Random Forest, were explored by M. Singh et al. Using smartphone accelerometer data, they categorize behavior into "Excellent," "Good," and "Weak." The study found that neural networks achieved the highest accuracy at 99.9%, emphasizing the potential of machine learning for detecting non-linear driving patterns (2018)[39] .

2.5.4 Sensor fusion for driving behavior

M. Omar et al. applied a sensor fusion approach combining data from CAN-BUS, IMU, and GPS sensors to classify driving behavior. The study, conducted in 2018, focused on distinguishing aggressive and normal driving by analyzing behavior across both time and frequency domains. This highlighted the advantages of incorporating contextual information for improved adaptability and accuracy[40].

2.5.5 Smartphone dataset for driving styles

In 2020, A. A. Javed et al. introduced a smartphone sensor dataset collected in diverse real-world driving scenarios. This dataset supports the classification of driving behavior into categories such as normal, aggressive, and risky. It serves as a benchmark for machine learning algorithms, ensuring consistency and enhancing model training and testing [41].

2.5.6 Smartphone data for driver monitoring

Smartphone sensor data was systematically collected by C. B. Park et al. across different traffic and environmental conditions. This dataset, published in 2019, is

valuable for validating machine learning models, providing a foundation for further research in driver behavior analysis, especially in realistic driving situations [42].

2.5.7 Low-cost driver behavior detection

Using smartphone sensors, R. Chhabra et al. classified drivers as aggressive or non-aggressive by identifying sudden accelerations, abrupt braking, and sharp turns. Conducted in 2019, the study set specific thresholds for each event type, offering a low-cost solution for driver behavior monitoring, particularly in regions with limited infrastructure [37].

2.5.8 Machine learning for driving events

A machine learning-based method for detecting braking and turning events was developed by M. Zarei Yazd et al. The authors applied various filters to enhance robustness, achieving F1 scores of 71% for brake detection and 82% for turn detection. This approach, introduced in 2022, demonstrated adaptability to new driving conditions [43].

2.5.9 Motion data for event classification

Taheri Sarteshnizi et al., in 2022, conducted a sensitivity analysis on driving event classification using smartphone motion data. They investigated how classifier types, sensor combinations, and data sampling rates impact accuracy. Their research highlighted the trade-offs between system complexity and classification performance, revealing that a three-sensor bundle is sufficient for reliable event classification [42].

2.5.10 Machine learning in transportation

Xie et al. focused on using the support vector machine (SVM), random forest (RF), and ensemble methods for driver behavior detection. They highlighted SVM's ability to avoid overfitting, RF’s resilience to data noise, and the adaptability of k-nearest neighbors (KNN), making these techniques suitable for real-world applications where data may not follow standard distributions (2018) [44].

Across these studies, there is a strong emphasis on using smartphone sensors and machine learning techniques for classifying driving behaviors. Accelerometers and gyroscopes are commonly utilized, with some studies incorporating additional sensor data such as GPS and CAN-BUS to enhance adaptability. Machine learning models, particularly neural networks and probabilistic models, show promising results in detecting complex driver behavior patterns. These

studies demonstrate the potential of cost-effective, real-world driver behavior monitoring systems that leverage smartphone and vehicle data.

2.6 Conclusion

This chapter provides an extensive exploration of the methods and technologies utilized in the domains of inner vehicle monitoring, Vehicle exterior monitoring, physiological monitoring, and vehicle dynamics. By reviewing and comparing these approaches, this literature review identifies the optimal methods for addressing the project's objectives.

Key technologies such as deep learning models for drowsiness detection, smart steering wheels for physiological monitoring, and smartphone sensors for vehicle dynamics analysis highlight the progress in driver behavior monitoring systems. The integration of advanced algorithms, multi-sensor fusion, and machine learning techniques has significantly enhanced the accuracy and adaptability of these systems.

These findings lay a strong foundation for designing a comprehensive, efficient, and practical prototype for improving road safety. By combining the best practices from existing research, this project aims to develop a system that effectively mitigates human errors and enhances driving safety.

The following chapter will discuss the methods utilized in each phase separately to obtain the system's objectives.

CHAPTER 3
METHODOLOGY

3.1 Preface

This chapter delves into the details of the technologies and methodologies used in this project.

This project focuses on addressing the following key areas:

- Monitoring the behavior of the driver whilst behind the wheel. Is the driver distracted? (Eating, Drinking, Texting, On the phone, or reaching in the backseat) or are his/her eyes on the road or not.

- Scanning the environment outside the vehicle, to get a good grasp of how everything is going (Driving in a lane properly, Distance between the driver and the other cars, appropriately following the traffic signs).

- Monitoring the state and well-being of the driver (Drowsy state, Health issues).

- Monitoring the dynamics of the vehicle will be a good indicator as feedback for knowing how well the driver is handling the car (Aggressive, Drunk, or Safe driving).

To get a good grasp of the flow of the entire system and how all phases are integrated together figure 3.1 depicts a flow chart following each phase and how they merge. Physiological monitoring here was called vital signs' monitoring for clarity.

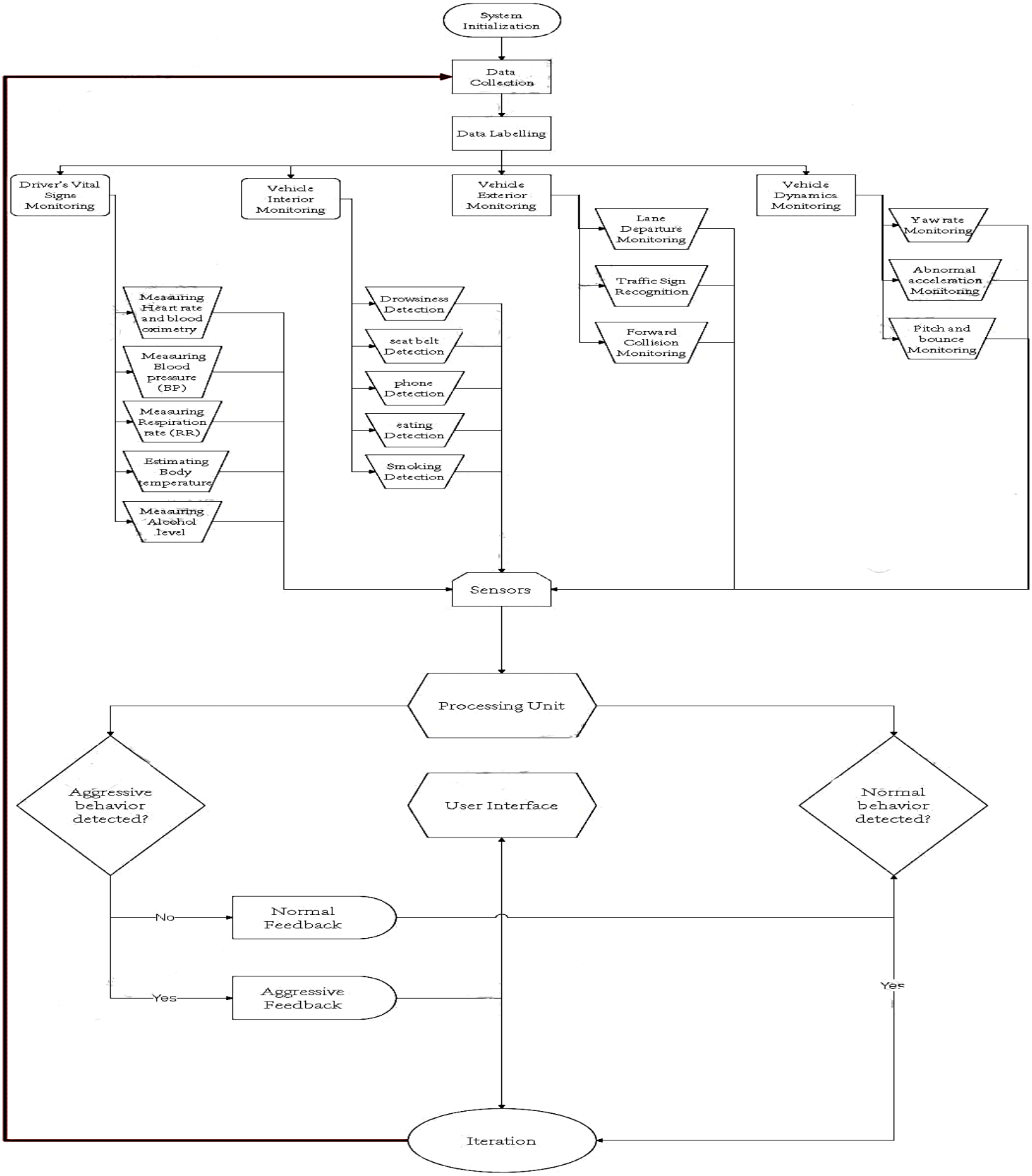

System Initialization
Data Collection
Data Labelling
Driver's Vital Signs Monitoring
Vehicle Interior Monitoring
Vehicle Exterior Monitoring
Vehicle Dynamics Monitoring
Lane Departure Monitoring
Traffic Sign Recognition
Forward Collision Monitoring
Yaw rate Monitoring
Abnormal acceleration Monitoring
Pitch and bounce Monitoring
Measuring Heart rate and blood oximetry
Measuring Blood pressure (BP)
Measuring Respiration rate (RR)
Estimating Body temperature
Measuring Alcohol level
Drowsiness Detection
seat belt Detection
phone Detection
eating Detection
Smoking Detection
Sensors
Processing Unit
Aggressive behavior detected?
User Interface
Normal behavior detected?
No
Normal Feedback
Yes
Aggressive Feedback
Yes
Iteration

Figure 3.1 Overall system flowchart

3.2 Interior Vehicle Monitoring

The primary goal of this phase is to identify six key behaviors that distract drivers while driving. These behaviors include:

1. **Drowsiness**
2. **Eating**
3. **Drinking**
4. **Texting and calling**
5. **Smoking**
6. **Seat belt detection**

Each behavior represents a significant safety concern that could lead to accidents, making their detection critical for monitoring and improving driver safety.

3.2.1 Data preparation

A labeled dataset was sourced from **Roboflow**, containing images of drivers from diverse genders and demographics. This dataset was divided into two main sections: a **Training section**, which included images used for training the machine learning model, and a **Testing section**, which contained unseen images for evaluating the model's performance. Each image in the dataset was categorized into one of the six behavior classes to facilitate accurate training.

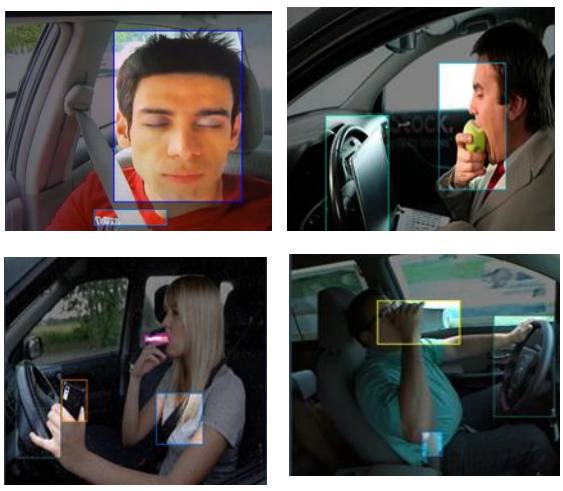


Figure 3.2 Data preparation (Roboflow)

3.2.2 Algorithm techniques

Machine learning techniques were applied to train the model to classify driver behaviors. **Google Colab** was utilized as the training platform due to its high computational capabilities, including GPU support, which is essential for processing large datasets efficiently. The model was trained to recognize each behavior class based on the labeled images. To enhance performance, data augmentation techniques, such as image rotation, brightness adjustment, and flipping, were likely used to make the model more robust and adaptable to varying conditions.

3.2.3 Training objectives

The primary objective of the training process was to develop a model capable of accurately identifying and classifying each of the six key behaviors. The model's performance was evaluated using metrics such as accuracy, precision, recall, and F1-score to ensure reliability and effectiveness. Special attention was given to ensuring the model's robustness so it could handle real-world scenarios.

3.2.4 Driver monitoring approach

Object detection is a critical component of modern AI systems, particularly in applications such as autonomous driving and driver behavior monitoring. Various algorithms have been developed to efficiently detect and classify objects in images or video feeds. These approaches can broadly be categorized into two main types: one-stage and two-stage detection methods. Each method has distinct characteristics, advantages, and limitations that make it suitable for different applications. The following sections provide an overview of these approaches, highlighting their working principles, popular algorithms, and key differences.

3.2.4.1 One-stage detection

One-stage detection algorithms are characterized by being "single pass" as they detect objects in a single pass without explicitly generating region proposals. This means they offer direct predictions of the bounding boxes and class probabilities in an image in one go. [45]

The best algorithms in this category include YOLO (You Only Look Once) and SSD (Single Shot Detector). [45]

The YOLO algorithm predicts bounding boxes and class probabilities in a single pass. While the SSD algorithm uses a set of default boxes over different aspect ratios and scales. [45]

3.2.4.2 Two-stage detection

Two-stage detection as the name implies, consists of two stages:

- Region Proposal: The first stage involves generating region proposals, which are potential bounding boxes that might contain objects.[45]
- Classification and Refinement: The second stage classifies these proposed regions and refines their locations to improve accuracy.[46]

Some examples the best algorithms in the two stage category includes Faster R-CNN, Mask R-CNN, Cascade R-CNN. [45]

Faster R-CNN: Uses a region proposal network (RPN) to generate region proposals and then classify them.

Mask R-CNN: Extends Faster R-CNN by adding a branch for predicting segmentation masks.

Cascade R-CNN: Uses a multi-stage approach to improve detection accuracy.

Table 3.1 key difference between one stage & two stages

Feature	One-Stage Detection	Two-Stage Detection
Speed	Faster due to elimination of region proposal step.[45]	Slower because of the additional region proposal step. [46]
Accuracy	Lower accuracy compared to two-stage methods may miss small objects.[45]	Higher accuracy due to refined regional proposals.[45]
Complexity	Simpler and more straightforward design.[45]	More complex due to the two-step process.[46]
Examples	YOLO (You Only Look Once), SSD (Single Shot Detector).[45]	Faster R-CNN, Mask R-CNN, Cascade R-CNN.[46]

3.2.4.3 Detection algorithms

In object detection, CNN (Convolutional Neural Networks) serve as the foundation by extracting features from images, but they are not inherently designed for tasks like object localization. R-CNN (Region-based CNN) builds

upon this by proposing candidate regions for objects and processing them individually, which enhances accuracy but is computationally slow. YOLO (You Only Look Once), on the other hand, processes the entire image in a single pass, making it significantly faster and more suitable for real-time applications, though it may sacrifice some accuracy compared to R-CNN. While R-CNN excels in precision, YOLO's speed and efficiency make it ideal for scenarios where real-time detection is critical. Ultimately, the choice depends on the trade-off between speed and accuracy required for the task. Table 3.2 sums up the main differences between the three proposed algorithms.

Table 3.2 Comparison between different algorithms

Feature	YOLO (You Only Look Once)	R-CNN (Region-based CNN)	CNNs (Convolutional Neural Networks)
Purpose	Real-time object detection. [45]	High-accuracy object detection [46]	General-purpose image analysis [45]
Operation	Single pass for detection [45]	Two-stage: region proposal, classification [46]	Extracts features from images [45]
Speed	Very fast [45]	Slower [46]	Varies, generally not real-time [45]
Accuracy	90-95% [45]	92-97% [46]	High, but context-dependent [45]
Complexity	Lower [45]	Higher [46]	Moderate to High [45]
Limitations	May miss small objects [45]	Computationally intensive [46]	Not designed for real-time detection [45]

Object detection and image analysis rely on different approaches, each with specific strengths and limitations. YOLO (You Only Look Once) is designed for real-time object detection, operating with a single-pass approach that makes it extremely fast and suitable for applications requiring immediate feedback, though it may struggle with detecting small objects in crowded scenes. R-CNN (Region-Based Convolutional Neural Networks), on the other hand, employs a two-stage process involving region proposals followed by classification, offering high accuracy and precision but at the cost of slower performance and higher computational requirements. Lastly, Convolutional Neural Networks (CNNs) serve as general-purpose tools for image analysis and feature extraction, excelling

at understanding visual patterns but lacking real-time capabilities when used independently. These methods vary in speed, accuracy, and complexity, making each suitable for distinct scenarios depending on project needs.

3.2.4.4 YOLO versions' comparisons

After much deliberation and consideration of the project needs and objectives the YOLO algorithm was found to be the most suitable.

The YOLO algorithm is a fast-growing algorithm with new versions with new improvements being launched consistently. So, a comparison between the different versions was conducted.

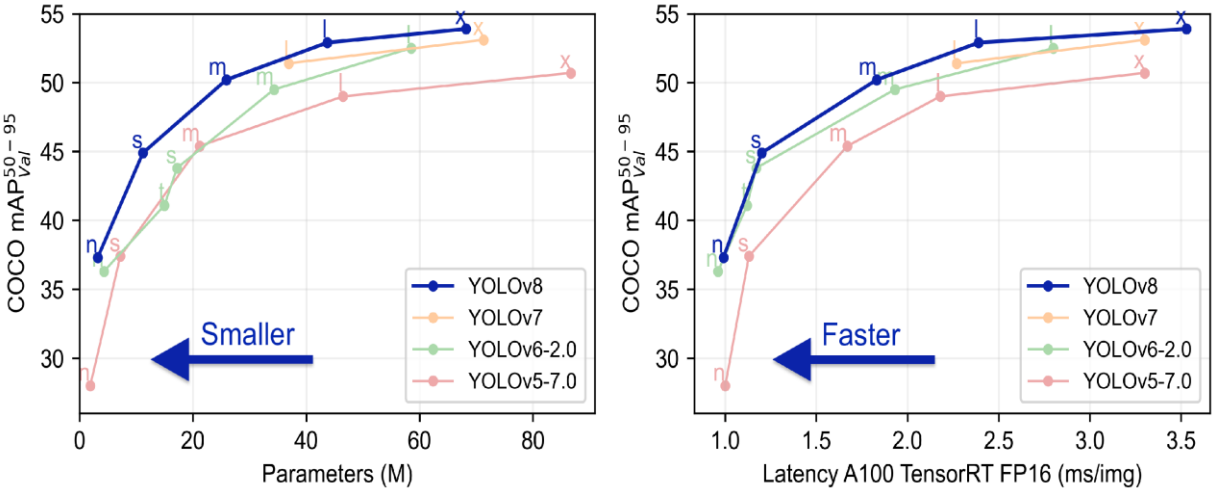


Figure 3.3 Yolo comparison [45]

Figure 3.3 shows a performance comparison of YOLO object detection models. The left plot illustrates the relationship between model complexity (measured by the number of parameters) and detection accuracy (COCO mAP50-95). The right plot shows the tradeoff between inference speed (latency on A100 TensorRT FP16) and accuracy for the same models. Each model version is represented by a distinct color, with markers indicating size variants from *nano* to *extra*. Plots taken from [47].

3.2.4.5 YOLO versions

Here is a brief introduction to each version and its most notable features and limitations are discussed:

1. YOLOv1[48],[49]:
    - Introduced: 2016 by Joseph Redmon.
    - Key Features: Single-stage object detection, real-time processing, and relatively simple architecture.

- Limitations: Struggled with small objects and localization errors.

2. YOLOv2 (YOLO9000) [48],[49]:
    - Introduced: 2017.
    - Key Features: Improved accuracy with batch normalization, anchor boxes, and multi-scale training. It could detect over 9000 object categories.
    - Improvements: Better performance on smaller objects and more accurate bounding boxes.

3. YOLOv3,[48], [49]:
    - Introduced: 2018.
    - Key Features: Multi-scale predictions, use of Darknet-53 as a backbone, and better handling of small objects.
    - Improvements: Higher accuracy and better performance on complex datasets.

4. YOLOv4[48], [49]:
    - Introduced: 2020.
    - Key Features: Incorporation of CSPDarknet53, PANet, and SPP layers. Enhanced data augmentation techniques.
    - Improvements: Significant boost in both speed and accuracy compared to previous versions.

5. YOLOv5[48], [49], [50]:
    - Introduced: 2020 by Ultralytics.
    - Key Features: PyTorch implementation, auto-learning bounding box anchors, and mosaic data augmentation.
    - Improvements: Easier to use, faster training, and deployment.

6. YOLOv6[48], [49], [50]:
    - Introduced: 2022 by Meituan Vision.
    - Key Features: Focus on industrial applications with better trade-offs between speed and accuracy.

- Improvements: Enhanced performance on edge devices and real-time applications.

7. YOLOv7[48], [49], [50]:
    - Introduced: 2022.
    - Key Features: Extended-ELAN architecture, model re-parameterization, and compound scaling.
    - Improvements: State-of-the-art performance in terms of speed and accuracy.

8. YOLOv8[48], [49]:
    - Introduced: 2023 by Ultralytics.
    - Key Features: Semantic segmentation capabilities alongside object detection.
    - Improvements: Further enhancements in accuracy and efficiency, making it suitable for a wider range of applications.

3.2.4.6 Why YOLOv11 is Better than YOLOv8

YOLOv11 demonstrates superiority over YOLOv8 through several key advancements:

1. Enhanced Computational Efficiency and Speed: The introduction of the C3k2 block in YOLOv11, replacing the C2f block of YOLOv8, leads to faster processing due to its more efficient implementation of the Cross Stage Partial (CSP) Bottleneck. This results in quicker feature extraction and overall improved speed without compromising performance [11].
2. Improved Feature Extraction and Attention: YOLOv11 incorporates the C2PSA block, a spatial attention mechanism that does not present in YOLOv8. This allows YOLOv11 to focus on more relevant image regions, leading to potentially higher accuracy, especially for challenging scenarios like small or occluded objects [11].
3. Expanded Task Capabilities with Higher Precision: While both models support multiple computer vision tasks, YOLOv11's architectural innovations, such as the C2PSA and refined C3k2 blocks, are designed to provide more nuanced detail capture. This suggests improved performance and precision in tasks like instance segmentation, pose estimation, and the newly emphasized oriented object detection (OBB) [11].

4. Optimized Architecture for Diverse Applications: YOLOv11 is explicitly designed with versatility in mind, offering a range of model sizes (nano to extra-large). This allows for tailored deployment across various platforms, from resource constrained edge devices to high-performance computing environments, potentially offering a more optimized solution compared to YOLOv8 for specific use cases [11].

In essence, YOLOv11 represents a more refined and advanced iteration, focusing on optimizing the trade-off between parameter count, accuracy, and computational efficiency, while also expanding its applicability to a wider array of complex computer vision tasks with potentially higher precision.

Table 3.3 YOLOv8 vs. YOLOv11

Feature	YOLOv8	YOLOv11
Core Architecture	• CSPNet backbone, FPN+PAN neck, C2f block, Anchor-free detection. [10]	• C3k2 block (replaces C2f), SPPF, C2PSA block, Anchor-free detection. [11]
Backbone Innovation	• Advanced CSPDarknet or similar, optimized for speed and accuracy. [10]	• C3k2 block replacing C2f for more efficient CSP Bottleneck implementation. [11]
Neck Innovation	• Optimized PANet for multi-scale feature integration. [10]	• C3k2 block replacing C2f; C2PSA module for spatial attention. [11]
Head Innovation	• Anchor-free approach. [10]	• Multiple C3k2 blocks for efficient multi-scale feature processing; C3k block for customizable kernel sizes. [11]
Attention Mechanism	• Not explicitly highlighted as a distinct module in the provided abstract. [10]	• C2PSA (Convolutional block with Parallel Spatial Attention). [11]
Supported Tasks	• Object Detection • Instance Segmentation • Pose Estimation [10]	• Object Detection • Instance Segmentation • Pose Estimation • Oriented Object Detection (OBB) [11]
Performance Focus	• High accuracy • Real-time Capabilities, • Developer-friendly [10]	• Improved mAP and Computational efficiency • Versatility across model sizes (nano to extra-large) [11]

The YOLO (You Only Look Once) series has evolved significantly since its introduction in 2016, with each version building on its predecessors to enhance performance and address limitations. YOLOv1 provided real-time detection with a simple architecture but struggled with small objects. Subsequent versions, such as YOLOv2 and YOLOv3, improved accuracy, multi-scale detection, and handling of complex datasets. YOLOv4 and YOLOv5 introduced advanced architectures like CSPDarknet53 and PyTorch implementation, respectively, to boost efficiency and usability. YOLOv6 and YOLOv7 further optimized performance for industrial and real-time applications.

The latest version, YOLOv8, introduced in 2023, stands out as the most advanced, offering semantic segmentation capabilities alongside object detection. It achieves exceptional accuracy, efficiency, and versatility, making it the ideal choice for this project. Given its state-of-the-art performance and suitability for a wide range of applications, YOLOv8 will be used for object detection in this system.

Table 3.4 The Different Types of YOLOv8 Evaluation on COCO Dataset *[47]*

Model	Size (pixels)	mPA^{val} 50-95	Speed CPU (ms)	Speed T4 GPU (ms)	params	FLOPs
YOLOv8n	640	37.3	-	-	3.2	8.7
YOLOv8s	640	44.9	-	-	11.2	28.6
YOLOv8m	640	50.2	-	-	25.9	78.9
YOLOv8l	640	52.9	-	-	43.7	165.2
YOLOv8x	640	53.9	-	-	68.2	257.8

Finally, when it comes to object detection, YOLO (You Only Look Once) emerges as the superior choice due to its exceptional balance of accuracy, efficiency, and real-time performance. Unlike the two-stage detectors like R-CNN, which, although precise, are computationally intensive and slower, YOLO's single-pass approach allows it to process images swiftly and accurately. This makes YOLO particularly suitable for applications that demand immediate feedback and high-speed processing, such as autonomous driving and real-time surveillance. Its ability to deliver high accuracy while maintaining efficiency ensures that YOLO not only meets but exceeds the demands of modern object detection tasks. Thus, YOLO stands out as the best option for combining speed, precision, and computational efficiency in object detection.

3.3 Vehicle Exterior Monitoring

The goal in this phase is to develop a Driver Aggressiveness and Behavior Monitoring System that integrates Lane Departure Monitoring (**LDM**), Forward Collision Monitoring (**FCM**), and Traffic Sign Recognition (**TSR**) and Wrong-Way Driving Detection (**WWD**). This system is designed to monitor and assess driver behavior, specifically targeting aggressive actions like **lane drifting**, **tailgating** (the act of driving too closely behind another vehicle, often within an unsafe distance, such that the driver risks a collision if the leading vehicle suddenly slows down or stops.) and **ignoring traffic signs**. By using advanced algorithms, we can track these behaviors and provide valuable insights to improve overall road safety.

3.3.1 Lane departure monitoring (LDM)

Objective: Detect lane boundary violations as an indicator of aggressive driving, such as frequent lane changes or drifting.

Table 3.5 Comparing Canny Edge + Hough Transform to Other Deep Learning-Based Models

Feature	Canny Edge + Hough Transform [51]	YOLO-Lane/SSD [52]	U-Net [51]	FCN [52]
Segmentation Quality	Moderate; sensitive to noise and lighting but can capture basic lane features.	Moderate; focuses on bounding boxes.	High; also tailored for segmentation.	Moderate; lacks ASPP's detail.
Context Awareness	Relies on gradient-based edges.	Limited; focuses on localized features.	Good; depends on the network depth.	Poor; limited feature scaling.
Real-Time Performance	High; very efficient, suitable for real-time use even on low-power hardware.	Faster than DeepLabV3; lacks precision.	Slower; requires more processing.	Moderate; lower computational efficiency.
Robustness	Sensitive to occlusions and challenging environments.	Sensitive to environmental challenges.	Good for clean data but struggles with noise.	Struggles with challenging data.
Scalability	Simple to implement and customize but lacks flexibility for complex tasks.	Needs re-design for specific tasks.	Easily customizable; slower training.	Limited flexibility.

Canny Edge Detection with Hough Transform offers moderate segmentation quality, as it is effective in detecting well-marked lane lines in clear conditions. However, it is sensitive to noise and lighting variations, and its accuracy depends heavily on parameter tuning, such as thresholds. Unlike deep learning models, which excel at extracting complex features, Canny and Hough are more limited in handling challenging scenarios.

In terms of context awareness, this approach is restricted to local edge processing without understanding the global context or semantic information. As a result, it struggles in scenarios like intersections or occluded lanes, where models like U-Net provide superior performance due to their depth and contextual reasoning.

Real-time performance is a significant strength of Canny and Hough. Being computationally lightweight, they are well-suited for real-time applications, even on low-power hardware. This contrasts with deep learning models, which often require more processing power and specialized hardware, making them slower in real-time scenarios.

Robustness is a limitation for this method. Canny and Hough are less effective in noisy environments, shadows, or occlusions, and their performance degrades in challenging real-world conditions. In comparison, deep learning models, which can learn complex patterns and generalize better, handle such situations more effectively.

In terms of scalability, the simplicity of Canny and Hough makes them easy to implement and customize for basic lane detection tasks. However, they lack the flexibility and adaptability of deep learning models, which can be fine-tuned and extended for various complex scenarios, such as multi-lane detection or curving roads.

Approach: Kortli et al. 2017 [53] focused on the Canny edge detection and Hough Transform algorithms. The methodology involves defining a Region of Interest (ROI) on road images, applying a Gaussian filter for noise reduction, and using Canny edge detection to enhance lane boundaries.

Lane boundaries are then extracted using the Hough Transform to determine the vehicle’s position relative to the lane. This approach effectively detected lane departures and demonstrated reliability under varying lighting conditions and road types.

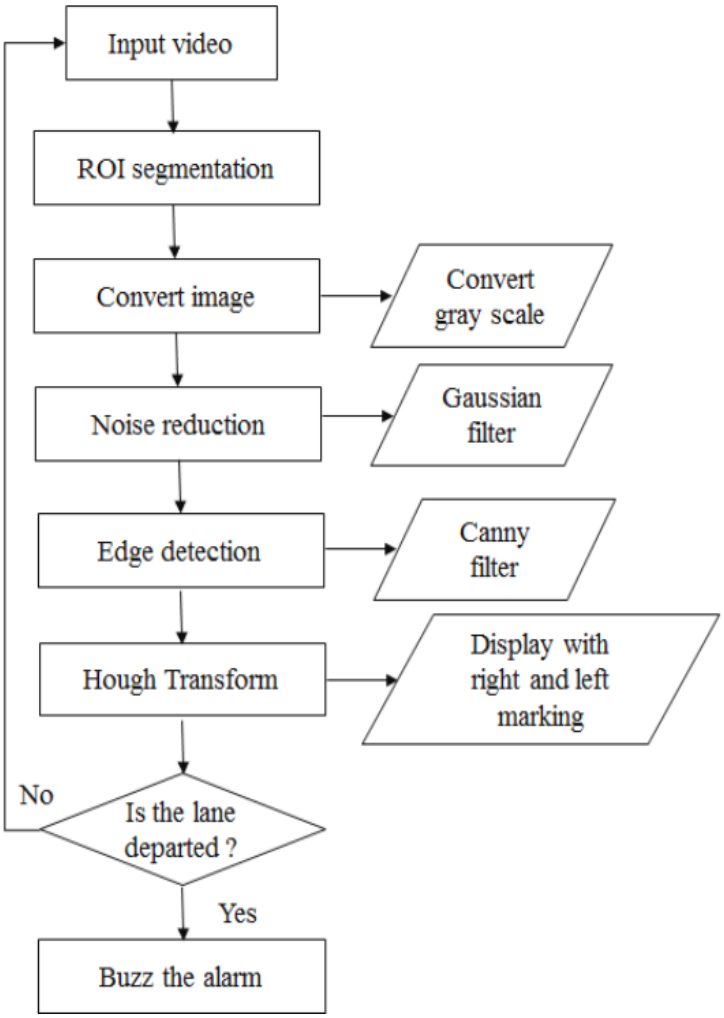


Figure 3.4 LDMS flowchart [53]

3.3.2 Forward collision monitoring (FCM)

Objective: Detect objects in the vehicle's path and track their movement to detect potentially aggressive behaviors like tailgating or sudden braking.

Table 3.6 Comparing YOLO + Monocular depth estimation + Kalman Filters to other potential algorithm combinations

Feature	YOLO + Monocular depth estimation + Kalman Filters [54]	LiDAR + YOLO [55]	Vision-Only CNNs [56]
Cost-Effectiveness	High	Low (High Cost)	High
Real-Time Performance	High	Moderate	High
Environmental Robustness	Good (dependent on camera quality)	High (except weather issues)	Limited (weather sensitive)
Depth Estimation Accuracy	Moderate (MonoDepth limitations)	Very High	Moderate
Motion Prediction	Excellent (via Kalman Filters)	Requires complex fusion	Limited (basic heuristics)
Scalability	High	Low	Moderate

The combination of **YOLO, monocular depth estimation, and Kalman Filters** strikes a balance between cost-effectiveness, real-time performance, and scalability, making it suitable for projects with budget constraints and moderate depth accuracy requirements. While LiDAR + YOLO offers unparalleled depth estimation accuracy and environmental robustness, its high cost and complexity make it less scalable. Vision-only CNNs provide a simpler, cost-effective alternative but lack the depth accuracy and robustness required for challenging environments, limiting their applicability. For applications demanding affordability and scalability, YOLO + Monocular Depth + Kalman Filters is the **optimal choice**.

Approach: According to Albarella et al. (2021) [57] paper, for object detection YOLO (You Only Look Once) is used, which provides fast and accurate real-time results. To estimate the distance of objects, Monodepth algorithm is very useful and Kalman Filters will help track object movement and predict potential collisions.

Steps:

- First, footage from various driving scenarios will be collected to train our object detection model.
- YOLO will be trained to detect cars, pedestrians, and other obstacles that may pose a risk.
- Monodepth will be used to estimate the distance between the vehicle and detected objects, which is crucial for assessing safe following distances.
- With Kalman Filters, the objects' movement will be tracked, allowing us to predict their future trajectory and flag unsafe driving behaviors like tailgating.
- Finally, object detection performance will be evaluated using mean Average Precision (mAP) and the accuracy of the depth estimation.

Expected Outcome: The system will be able to detect objects, calculate distances, and track movements to identify behaviors like following too closely or hard braking.

3.3.3 Traffic sign recognition (TSR)

Objective: Recognize and assess the driver's adherence to traffic signs, looking for signs of aggressive driving, such as speeding or ignoring stop signs.

Approach: Latif et al. (2023) [58] proposed two optimized Residual Network (ResNet) models (ResNet V1 and ResNet V2) for automatic traffic sign recognition using the Arabic Traffic Signs (ArTS) dataset. ResNet is a deep CNN, Its famous for Its powerful feature extraction and classification capabilities. It's particularly good at identifying a wide variety of traffic signs.

Steps:

- A diverse dataset of traffic signs, including those in various weather conditions and from different perspectives will be gathered.
- To ensure robustness, data augmentation techniques like rotation and color adjustments are applied.
- The ResNet model will be fine-tuned on a dataset like the GTSRB (German Traffic Sign Recognition Benchmark).
- For evaluation, classification accuracy and F1 scores will be used to measure how well the model distinguishes between different types of traffic signs.

- Expected Outcome: The model will accurately identify traffic signs and assess whether the driver is following traffic rules, flagging violations like speeding or running stop signs.

3.3.4 Wrong-Way Driving Detection

Wrong-Way Driving Detection Using GPS Direction and OpenStreetMap Path Comparison

The implemented system leverages positional data, either collected in real time from a GPS module or imported via GPX files from a mobile device, to detect wrong direction driving by comparing the vehicle's estimated heading against the legal road direction derived from OpenStreetMap (OSM).

1. Data Acquisition

- **GPS Data Collection**: Positional data is obtained from either a GPS module connected to the system or from GPX (GPS Exchange Format) files recorded by mobile devices. Each data point includes latitude, longitude, and timestamp information.
- **Map Data Parsing**: OSM road network data for the target area is downloaded and parsed to extract the directionality of road segments. Each OSM way includes a sequence of geographic nodes and a "oneway" tag indicating whether the road is unidirectional or bidirectional.

2. Heading Estimation

The instantaneous heading direction of the vehicle is computed using two consecutive GPS coordinates: $(lat_1, lon_1)\ and\ (lat_2, lon_2)$. The bearing angle θ is calculated using the haversine formula or geodesic approximation:

$$\theta = \arctan 2\ (\sin(\Delta\lambda).\cos(\phi_2)\ , \cos(\phi_1)\ .\sin(\phi_2) - \sin(\phi_1)\ .\cos(\phi_2)\ .\cos\ (\Delta\lambda)) \quad 3.1$$

3. Direction Comparison and Violation Detection

The road segment closest to the current GPS position is determined using spatial proximity and polyline projection onto OSM segments. The directional vector of the matched OSM way is then compared against the calculated vehicle heading.

The angular difference between the vehicle's direction and the legal direction of the road segment is computed. If this difference exceeds a defined threshold

(typically 120°–180°), the system infers that the vehicle is traveling in the opposite direction of the road's intended flow.

To reduce noise and false positives:

- Multiple consecutive heading readings are averaged.
- Angular thresholds are adjusted based on road type and curvature.
- Timestamps are used to calculate velocity and ensure data validity.

4. Warning and Visualization

When a wrong-direction condition is confirmed, the system logs the event and triggers a warning.

3.3.5 System integration

Objective: Integrate LDM, FCM, and WDD and TSR modules into a unified system that continuously monitors and evaluates driver behavior.

Approach: A central processing unit will be used to handle inputs from all three modules, offering real-time feedback on the driver's actions. Sensor fusion techniques will combine data from cameras and other sensors for more accurate results.

Expected Outcome: The system will provide timely alerts about aggressive or unsafe driving behavior, offering a comprehensive view of the driver's actions

This phase of the project aims to develop a Driver Aggressiveness and Behavior Monitoring System using Canny edge and Hough transform for lane detection, **YOLO** for object detection, Monodepth for depth estimation, Kalman Filters for tracking, and **ResNet** for traffic sign recognition. By combining these technologies, the system will be able to monitor and assess aggressive driving behavior in real time, providing insights that can help improve road safety.

3.4 Physiological Monitoring of Driver

As we've established the importance of monitoring a driver's physical condition, the general idea is to design a "smart" steering wheel that could detect any abnormalities in a driver's physical state. Different sensors for measuring different parameters are embedded within the steering wheel and adjusted to be in contact with the driver's hands. The process for measuring most of the parameters is fairly similar figure 3.4 [59] explains the basic process.

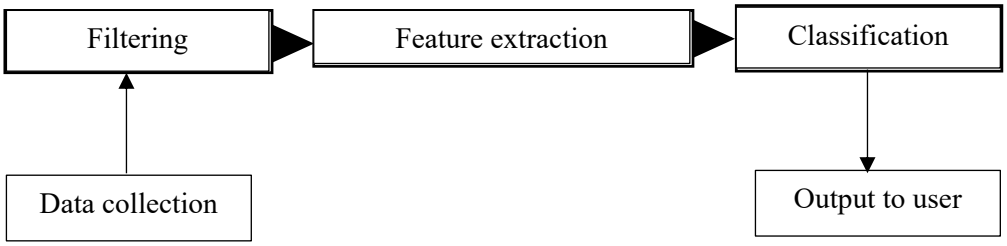


Figure 3.5 Physiological Monitoring Process

Seven major parameters are to be measured and explored: Heart rate (HR), blood oxygenation, blood pressure (BP), body temperature, Alcohol level and respiratory rate (RR). Each parameter will be discussed in detail in the following sections.

Before delving into the specifics of each parameter, two major techniques that are used to deduce a lot of the targeted parameters need to be discussed: Photoplethysmography (PPG) and electrocardiogram (ECG or EKG).

3.4.1 Photoplethysmography (PPG)

Photoplethysmography (PPG) is a simple optical technique that is used in various medical settings and applications to detect the volumetric changes in blood in peripheral circulation. It is characterized by its non-invasiveness, which makes it ideal for driver monitoring applications.

A PPG sensor uses low intensity infrared (IR) light that is then directed onto the skin and absorbed by the skin's tissues, bones and blood. Blood absorbs the majority of light so it's easy to detect even the smallest changes in blood volume.

PPG signals are analyzed to indirectly gain insight into various vital parameters like cardiac activity which are associated with volumetric changes in arterial blood.

PPG can be used in combination with other techniques like electrocardiogram (ECG) to find blood pressure.

3.4.2 Electrocardiogram (ECG or EKG)

Electrocardiogram (ECG or EKG) is a non-invasive method used to evaluate the heart. It does so by measuring the electrical activity of the heart. A multitude of heart conditions through abnormal heart activity and rhythms. Electrodes (conductive material) with wire leads are placed on various locations on the chest, arms and legs, and an ECG device records the electrical activity through the lead

wires. For the purpose of this application, traditional gel-based electrodes were not suitable. Therefore, dry electrodes were required. However, due to the unavailability and difficulty of obtaining commercially made dry electrodes, custom DIY electrodes were created using small stainless-steel sheets— a material commonly used in medical applications—soldered to wires and connected to the AD8232 module. Unfortunately, this setup resulted in an unstable signal, which may impact the accuracy of the readings. Despite that carful use of the electrodes resulted in extremely promising readings, which solidifies the legitimacy of this approach (ECG) to collect critical medical information of the driver.

3.4.3 Heart rate (HR)

Heart rate is perhaps the most significant parameter to be measured. An abnormal heart rate could be indicative of disturbed bodily functions, or an onset of heart disease.

Non-invasive techniques for measuring the heart rate in settings such as the inside of a car are still under investigation. Different approaches were proposed. One suggested method used a phone camera to capture a video of the upper chest of a subject while lying down with minimal movements[60]. This method is clearly unsuitable for many reasons. Another similar method is using a computer webcam for a simple and inexpensive system and using MATLAB algorithms to extract the data [61]. These camera-based approaches are still relatively under development, hence why is a more basic approach using PPG and ECG sensors embedded within the steering wheel used. The following table presents a comparison between the different approaches used.

Table 3.7 Heart rate measurement approaches

Approach	Camera-based	Near Infrared (NIR)	PPG and ECG sensors
Advantages	• Measures multiple physiological parameters like the respiratory rate, etc.[62] • Completely detached from the subject of measurement.	• Not hindered by objects such as glasses, etc.[62]	• More reliable. A lot of research is conducted in this area.[63]
Limitations	• Hindered by objects such as glasses, caps, etc.[62] • Accuracy of results depends on the quality of facial recognition.[62]	• Impacted by external factors such as lighting.[62]	• Requires stable and consistent contact with the steering wheel.[64]

For the measurement of the pulse/ heart rate two techniques are used: The first being PPG, and the second is ECG, which will both be implemented in the steering wheel.

Firstly, a PPG sensor is used, in this case the pulse oximetry module Max30102 is used. As the name suggests, this module is capable of estimating both the pulse rate and the blood oxygen levels. As mentioned in the JustDoElectronics guide for this module, it works by shining a specific wavelength of light, typically in the red or infrared spectrum, into the skin and measuring the amount of light reflected back. When blood pulses through the capillaries, it absorbs more light, leading to a decrease in the reflected light intensity. By analyzing these variations in light intensity, the sensor can determine the heart rate and approximate blood oxygen saturation level.

The second technique is through an ECG. The AD8232 module kit measures the electrical activity of the heart as per the Sparkfun website guide for ECG modules. The ECG signal is extracted to a microcontroller on which the signal will be processed. Furthermore, as the signal will be extracted through the electrodes in contact with the driver's wrists, conductive dry electrodes made of stainless steel are used on the steering wheel to facilitate sensor measurement of the ECG signal.

Ultimately, after comparing both methods for heart rate measurement, using an ECG focused approach has proven to be much more reliable than the ppg approach.

The methodology for heart rate (HR) estimation is a multi-stage process involving signal acquisition, digital processing, and calculation. The process begins with signal acquisition using an AD8232 ECG analog front-end. The analog signal from this sensor is then digitized by an ADS1115 Analog-to-Digital Converter (ADC), which is configured to sample the waveform at a frequency of 250 Hz. The system continuously acquires data in 30-second windows for analysis.

Each captured window of ECG data first undergoes a digital pre-processing step to remove noise and artifacts. A second-order Butterworth bandpass filter is applied to the raw signal. This filter is set with a passband of 0.5 Hz to 40 Hz, which effectively removes low-frequency noise such as baseline wander and higher-frequency noise from muscle contractions.

Following filtration, a peak detection algorithm is applied to the clean ECG signal to identify the prominent R-peaks of the QRS complex. The algorithm is constrained to recognize only peaks that exceed a minimum height threshold and are separated by a minimum physiological distance, thereby preventing the false detection of other wave features like the T-wave. Once the temporal locations of the R-peaks are identified, the time intervals between consecutive peaks (RR-intervals) are calculated. The average of all RR-intervals within the 30-second window is then determined. Finally, the heart rate in beats per minute (BPM) is calculated from this average RR-interval using the formula:

$$HR\ (BPM) = \frac{60}{Average\ RR - interval\ (in\ seconds)} \qquad 3.2$$

3.4.4 Blood pressure (BP)

Given the local unavailability of direct, continuous blood pressure measurement modules, this work utilizes an indirect method that leverages the combination of both an Electrocardiogram (ECG) and a Photoplethysmogram (PPG) signal. The signals are acquired using an AD8232 ECG module and a MAX30102 PPG sensor, respectively. The core of this method is the calculation of **Pulse Transit Time (PTT)**, which is the interval between the peak of the R-wave in the ECG and the subsequent peak of the fingertip PPG waveform. PTT is used to estimate

both **Systolic Blood Pressure (SBP)**, the arterial pressure when the heart contracts, and **Diastolic Blood Pressure (DBP)**, the pressure when the heart relaxes.

The estimation methodology is a two-stage process: precise PTT calculation followed by conversion to blood pressure using a calibrated model. Accuracy in this process is paramount, as the regulation of blood pressure within the human body is a complex, multivariate physiological process [65]. The first stage, PTT calculation, begins with the timestamped peak data from both signals. The algorithm iterates through each ECG R-peak and performs a forward search to identify the first corresponding PPG systolic peak. To ensure physiological validity and reject artifacts, a PTT value is only considered valid if it falls within a predefined window of **100 to 600 milliseconds**. To guarantee a one-to-one mapping for each cardiac cycle, once a PPG peak is successfully paired, it is removed from the pool of available peaks. Finally, all valid PTT values from the signal segment are averaged to yield a single, robust PTT measurement.

In the second stage, this PTT value is used for blood pressure estimation via a mathematical model based on the established inverse relationship between PTT and arterial stiffness. This model requires a **one-time, subject-specific calibration** with a traditional cuff-based blood pressure device to ensure accuracy. The formulas used are:

$$SBP = \frac{A_{sbp}}{PTT} + B_{sbp} \tag{3.3}$$

$$DBP = \frac{A_{dbp}}{PTT} + B_{dbp} \tag{3.4}$$

To illustrate this critical calibration process, a practical example is provided. A single-subject calibration was performed using a reference blood pressure of **105/71 mmHg**. Simultaneously, the PTT was calculated from the subject's signals, yielding an average value of **0.356 seconds**. By substituting these known values into the model's equations, the following subject-specific coefficients were derived:

- A_{sbp} = 30.26
- B_{sbp} = 20.0

- A_{dbp} = 14.596
- B_{dbp} = 30.0

These personalized coefficients are then used exclusively for that subject to convert subsequent PTT measurements into blood pressure estimates.

3.4.5 Respiration rate (RR)

According to Johns Hopkins Medicine, respiration rate (RR) refers to the number of breaths taken per minute. Respiration rates may increase with the presence of an illness or other medical conditions; this can be used to identify any abnormalities in the driver's state.

Like heart rate, respiration rate could be estimated using a PPG signal through digital signal processing (DSP) techniques. The PPG signal is filtered to remove noise and extract variables of interest from the raw signals [66].

Respiration rate was estimated from the infrared (IR) photoplethysmography (PPG) signal acquired by the MAX30102 sensor. The raw IR signal, which inherently contains respiratory-induced variations, was subjected to a digital bandpass filter to isolate the breathing waveform. A fourth-order Butterworth bandpass filter with cutoff frequencies of 0.1 Hz and 0.4 Hz was applied to the IR signal, effectively retaining components corresponding to typical human breathing rates (6 to 24 breaths per minute). Following filtration, prominent peaks in the filtered signal were identified using a peak-finding algorithm. To ensure that detected peaks represent distinct respiratory cycles and minimize false positives, a minimum distance constraint of 1.5 seconds between successive peaks was enforced. The respiration rate, in breaths per minute (BPM), was then calculated by counting the number of identified peaks within the total signal acquisition duration and scaling this count to a one-minute interval. The final respiration rate was rounded to two decimal places.

3.4.6 Blood Oxygen Levels

The peripheral oxygen saturation (SpO2) was determined using a custom Python driver for the MAX30102 pulse oximeter sensor. The sensor was initialized in SpO2 mode, and the red and infrared (IR) LEDs were configured with a pulse amplitude of 0x4F (approximately 7.8 mA). Raw photoplethysmography (PPG) data, consisting of 3-byte samples for both red and IR light intensities, were continuously acquired from the sensor's FIFO buffer. To calculate SpO2, the acquired red and IR signals were first processed to remove the direct current (DC)

offset by subtracting the mean of each respective signal. The alternating current (AC) components were then computed as the root mean square (RMS) of the DC-offset-removed signals. A ratio of ratios (R) was subsequently calculated using the formula:

$$R = \frac{(\frac{AC_{Red}}{DC_{Red}})}{(\frac{AC_{IR}}{DC_{IR}})} \quad 3.5$$

where AC_{Red} and AC_{IR} are the AC components of the red and IR signals, respectively, and DC_{Red} and DC_{IR} are their corresponding DC components. Finally, the SpO2 value was derived using an empirically adjusted linear regression formula:

$$SpO_2 = 110 - 12 \cdot R \quad 3.6$$

The resulting SpO2 values were rounded to two decimal places and constrained to the physiological range of 0% to 100%.

3.4.7 Body temperature

Body temperature is also an indicative of health abnormality. The contactless IR temperature sensor module MLX90614 is used. This module is based on the Stefan-Boltzmann principle that states that all objects emit infrared radiation that is proportional to their temperature. This sensor is capable of measuring the temperature of an object from a distance without needing to be in contact. Through the Instructables guide of this module, and according to its datasheet, this sensor can measure the temperatures within the range -70 to 380 degrees Celsius with an accuracy of about 0.5˚C at room temperature. The sensor is also embedded with a filter that provides protection against outside light disturbances such as sunlight.

As mentioned, non-contact temperature measurement methodology was implemented utilizing the MLX90614 infrared thermometer sensor. Communication with the MLX90614 sensor was established via the Inter-Integrated Circuit (I2C) serial communication protocol, leveraging the board and busio libraries for Raspberry Pi-specific hardware interfacing. An adafruit_mlx90614 library instance was initialized to facilitate high-level interaction with the sensor. In a continuous loop, the system periodically queried the MLX90614 sensor to acquire both the ambient temperature of the sensor's

environment and the temperature of the target object within its field of view. The acquired temperature data, measured in degrees Celsius, was then formatted to two decimal places and printed to the console for real-time monitoring. The data acquisition process continued indefinitely until a KeyboardInterrupt signal was received, allowing for a controlled and graceful termination of the program. It has to be mentioned that a shift of 3 degrees was added to the temperature reading after calibrating with a thermometer. The shift was needed as the mlx90614 mainly measures the skin temperature.

3.4.8 Alcohol level

For this parameter, instead of using a regular breath detection instrument for measuring alcohol levels, a method that relies on various other parameters such as the respiration rate, blood pressure, heart rate and temperature are used to construct a criteria system to classify the level of intoxication/ blood alcohol concentration (BAC). Table 3.8 shows the classification of BAC levels.

Table 3.8 Blood alcohol levels (BAC) levels classification [35]

PPG Signal Data	Respiration Rate (breath/min)	Blood Pressure	Heart Rate (beats/min)	Temperature	BAC (%)
nl= 0.994, nh= 0.966	12-20	110/70-120/80	60-90	97-99	0-Sober
nl= 0.248, nh= 0.2415	10-12	120/80-125/85	90-100	99-101	0.02
nl= 0.497, nh= 0.483	9-10	125/85-130/85	100-105	101-102	0.04
nl= 0.745, nh= 0.724	5-9	135/85-140/90	>105	>102	0.06
nl= 0.994, nh= 0.966	<5	>140/90	>105	>102	0.08

Table 3.9 Intoxication level measurements methods comparison [35]

Research	Prototype	Sensors	Accuracy	Cost	BAC classification (%)
Liu, et al. [67]	Device	NIR	NA	High	2
WrisTAS (ginerinc)	Wearable	Touch sensor	NA	NA	2
BACtrack	Device	Breath analyzer	NA	High	5
Floome	Device	Breath analyzer	75%	High	5
Chen, et al. [68]	NA	PPG	85.71%	Moderately High	2
Rachakonda, et al. [69] (Donot-DUEye)	Device	Biometric Scan and BP	95%	Moderate	2
Smart-Steering (Current Device Paper)	Device	Heart rate, temperature, respiration and blood pressure	93%	Moderately Low	5

The sensor measurements from the previous sections are used to then come to a conclusion regarding the driver's intoxication rate. This method has proven to have an accuracy of 93% according to the study done by Rachakonda et al., 2020 [70].

3.4.9 PCB Design

The Printed Circuit Board (PCB) was meticulously designed to integrate various biomedical sensors and interface with a Raspberry Pi. The core components selected for this design include the MLX90614 non-contact infrared thermometer, a MAX30102 pulse oximeter and heart rate sensor, an AD8232 electrocardiogram (ECG) sensor, and an ADS1115 analog-to-digital converter (ADC) for high-resolution signal acquisition. Although initially included, a generic Pulse Sensor was eventually scrapped due to its unreliable readings in preliminary testing. All sensor connections were made using JST headers to ensure sturdy, non-wobbly connections, critical for maintaining signal integrity in a portable or experimental setup. The following figure shows the schematic drawing of the PCB board.

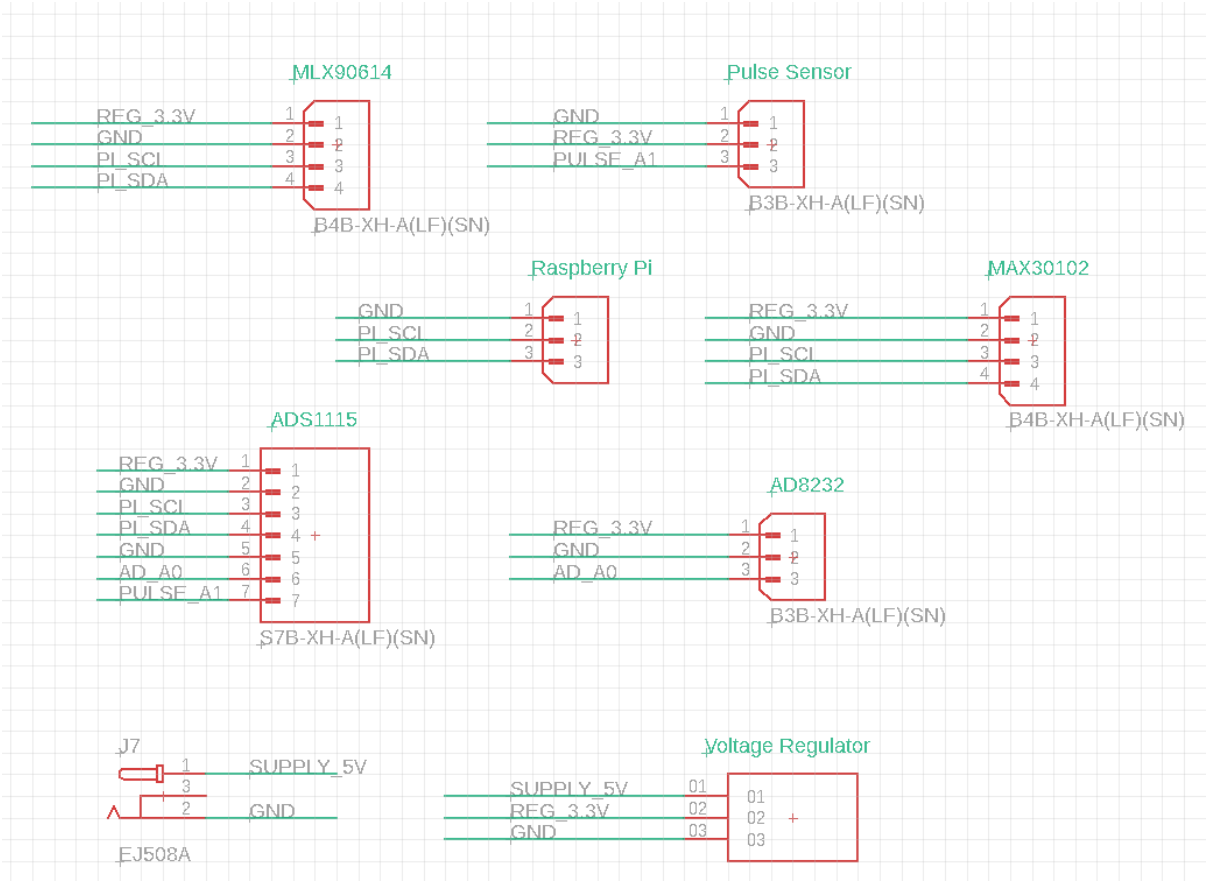


Figure 3.6 PCB Schematic

A key aspect of the power management strategy involved the use of a dedicated voltage regulator module to convert the incoming 5V supply into the required 3.3V for the majority of the sensors. The Raspberry Pi itself was powered separately to avoid potential noise interference and ensure stable operation, drawing approximately 2.5 A at 5V under typical full load with peripherals. The current and voltage requirements for all sensors are cited in the following table:

Table 3.10 Sensors' Current and Voltage Requirements

Component	Operating Voltage	Typical Current Draw
MAX30102	1.8V (core), 3.3V (IO)	1.6 mA average, can spike to 20 mA
Pulse Sensor	3.3V–5V	4 mA
AD8232	3.3V or 5V	170 µA (typical), can spike to 1 mA
MLX90614	3.3V or 5V	1.5 mA average – 2 mA max
ADS1115	2.0V–5.5V	150 µA – 200 µA
Raspberry Pi 5	5V	2.5 A (typical full load, with peripherals)
Total (Max Current Drawn)		27.2 mA

To calculate the current requirements for powering all sensors, first, a 20% buffer was added to the total max drawn current for safety.

$$27.2\text{mA} \times 1.2 \approx 32.64\ \text{mA} \approx 33\text{mA at } 3.3\text{v}$$

Then the Current Draw from the 5V Power Supply (to AMS1117) is calculated as follows:

The AMS1117 will need to provide 10 mA at 3.3V, but due to its inefficiency, we'll need to calculate the required current from the 5V supply.

First the power required for the sensors is:

$$P_{sensors} = V\ x\ I \tag{3.7}$$

$$\text{P} = 3.3V\ X\ 0.033\ A = \ 0.109W$$

For a linear regulator like the AMS1117, the conversion efficiency is:

$$\eta = \frac{V_{out}}{V_{in}} \tag{3.8}$$

$$\eta = \frac{3.3\ V}{5\ V} X\ 100 = 66\%$$

Power loss is given by:

$$Power\ loss = (V_{in} - V_{out}) \times I_{load} \tag{3.9}$$

$$\text{Power loss} = \ (5V - 3.3V) \times 0.033\ A = 0.0561\ W$$

So, the total power consumed by the AMS1117 from the 5V supply is:

$$P_{total} = P_{sensors} + P_{loss} \tag{3.10}$$

$$P_{total} = 0.109\ W + 0.0561\ W = 0.1651\ W$$

Now, we can calculate the current required from the 5V supply:

$$I_{5v} = \frac{P_{total}}{V_{in}} \tag{3.11}$$

$$I_{5v} = \ \frac{0.1651\ w}{5v} = 0.033\ A = 33mA$$

Then the total Current for the 5V Power Supply for Sensors:

- Current required from the 5V supply to power the AMS1117 (which regulates to 3.3V for the sensors) = 33 mA

- Sensor current draw = 10 mA at 3.3V, which the AMS1117 will convert.

So, the 5V power supply for the sensors will need to provide at least 33 mA to power the AMS1117 and the sensors.

Thus, the 5V power supply for the sensors was chosen to provide 1 A of current covering the minimum requirement of 33 mA.

The PCB utilized a standard copper thickness of 1 oz/ft^2, corresponding to approximately 35 µm for both the top and bottom layers. This thickness was crucial for determining the current-carrying capacity of the traces. Based on IPC-2152 standards for external layers with a 10∘C temperature rise limit, a 35 µm thick trace with a width of 0.1524 mm (6 mil) can safely carry approximately 0.35 A (350 mA). The total maximum current drawn by all connected sensors (MAX30102, AD8232, MLX90614, and ADS1115) was calculated to be approximately 6.92 mA. Even with a conservative rounding up to 10 mA, this represents only about 3% of the safe current limit for a 0.1524 mm trace. In the final layout, wider 0.5 mm traces were employed for all 3.3 V power lines to minimize voltage drop and enhance thermal stability, while 0.1524 mm traces were used for ground (GND) and all signal lines, which are well within safe limits given the low current loads and short trace lengths. Furthermore, a comprehensive GND polygon pour was implemented across the PCB. This design choice provides a wide, low impedance return path for all ground connections, effectively reducing the risk of ground loops, voltage offsets, and noise interference, which is particularly important for sensitive analog sensors like the AD8232 and MAX30102.

3.5 Vehicle Dynamics

Vehicle dynamics is concerned with detecting the dynamic behavior analysis of vehicle's body by detecting the acceleration and deceleration rate of the vehicle, braking behavior, cornering behavior and the number of speed bumps taken in a speed after a specified range. Each of the motion states is represented by values from sensors. Using these values gives us the ability to detect the driving behavior of the driver.

The driving behavior was categorized into 3 main categories aggressive, normal and keen. Custom machine learning models are used which will be train using validated datasets.

3.5.1 Dynamic analysis

1. Acceleration and deceleration rate [37].
   Identifies how quickly the vehicle speeds up or slows down. High rates indicate aggressive driving while smoother rates suggest normal behavior.
2. Braking behavior [37].
   evaluates how driver applies the brakes whether gradually or abruptly, which can signify risky actions.
3. Cornering behavior [37].
   Assess how the vehicle handles turn, considering the sharpness and speed of the maneuver. Sharp cornering represents aggressive patterns.
4. Speed bump interaction.
   tracks the number of bumps passed and the speed during each encounter. Driving over bumps too fast is an indication of aggressive driving.

- **Data collection:**

Our main device is smartphone [37] since it already has the necessary sensors needed for analysis such as gyroscope and accelerometer.

Table 3.11 Sensor measurements and their applications

Sensor	Axis/Type	Application
Gyroscope	Yaw[37]	Detects cornering behavior.[37]
Gyroscope	Roll[37]	Tracks vehicle tilt during turns, aiding in identifying aggressive cornering.[37]
Accelerometer	X-axis[37]	Aggressive acceleration during retraction and deceleration during braking.[37]
Accelerometer	Z-axis[37]	Detects vertical changes for identifying speed bump interactions.[37]
Accelerometer	Y-axis[37]	Tracks lateral movements, helping evaluate side to side driving behavior.[37]

3.5.2 Categories of driving behavior

Driving behavior is classified into three categories: Normal, Aggressive, and Keen. These classifications are based on key driving patterns such as speed, braking, and acceleration, providing insights into the driver's style and safety.

Table 3.12 Driving behavior categories

Category	Acceleration	Braking	Cornering	Speed Bumps
Aggressive	>0.3 * g [71]	<-0.3*g [71]	> 20°/s [71]	>0.3 * g vertical [71]
Keen	0.1 – 0.3 *g [71]	-0.1 – -0.3 *g [71]	10 – 20 °/s [71]	0.1 – 0.3 * g vertical [71]
Normal	< 0.1 * g [71]	> - 0.1 [71]	< 10 °/s [71]	< 0.1 * g vertical [71]

3.5.3 Machine Learning approach

With a labeled dataset in hand, supervised machine learning will be applied to classify driving behavior. The primary objective is to train a classifier that can accurately differentiate between aggressive, normal, and keen driving behaviors based on the vehicle's dynamics.

3.5.4 Classification algorithm

The following table (table 3.11) compares the different types of machine learning algorithms, highlighting their strengths and weaknesses

Table 3.13 Machine learning methods and suitability

Algorithm	Strengths	Weaknesses
Decision Tress	Intuitive and interpretable for decision-making processes.[39]	Prone to overfitting if not pruned.[44]
Random Forest	Reduces overfitting by combining multiple decision trees.[39]	May require significant computational resources.[44]
Support Vector Machine	Effective in high-dimensional feature spaces.[39]	May not scale well with large datasets.[44]
K-Nearest Neighbors	Effective in high dimensional feature spaces.[39]	Computationally expensive for large datasets.[44]

3.5.5 Model training and evaluation

Data Split [72] [73]

- The simulation dataset is split into training and testing sets, typically using an 80/20 or 70/30 split.

Training and Testing [72] [73]

- Models are trained on the training set and tested on the unseen test set to evaluate performance.

Evaluation Metrics [72] [73]

- **Accuracy**: Measures overall correctness of predictions**.**
- **Precision**: Measures of true positives out of all predicted positives.
- **Recalling**: Measures how many actual positives are identified correctly.
- **F1-Score**: Balances precision and recall, useful for imbalanced datasets.

3.6 Hardware

This section provides a detailed description of the hardware components utilized in the system, their specifications, and the roles they play in achieving the objectives of the project. It explains the rationale behind the selection of specific hardware elements and how they integrate to form a cohesive system. By addressing the design considerations, wiring configurations, and operational requirements, this section aims to offer a clear understanding of the physical infrastructure that supports the system's functionality.

3.6.1 Microprocessor

Microprocessors play a pivotal role in modern computing, serving as the brains of various electronic devices, from simple gadgets to complex systems. When selecting a microprocessor for a project, especially one involving task such as object detection, several factors must be considered, including processing power, cost, size, power consumption, and ease of integration with peripheral devices. Among the many available options, microprocessors like Raspberry Pi, Arduino, and other specialized chips stand out as popular choices for embedded systems. [74]

Here, different microprocessors are compared based on their capabilities and determine the best fit for an object detection application. The analysis will evaluate key parameters such as computational performance, connectivity options, available support libraries, and overall flexibility. After carefully considering alternatives like the Arduino, which is known for its simplicity and low power consumption, and other microprocessors with higher computational

power but increased complexity, the Raspberry Pi emerges as the most suitable option. With its robust processing power, extensive support for machine learning libraries, and a versatile range of I/O options, the Raspberry Pi perfectly aligns with the requirements of an object detection system, making it the optimal choice for this project. [75]

1. **Arduino Due**

   Arduino Due features an ARM Cortex-M3 microcontroller, running at 84 MHz. While this is sufficient for simple control tasks and sensor handling, it has limited processing power for complex applications such as real-time image processing or object detection[74].

2. **Raspberry Pi 5:**

   Raspberry Pi 5 is powered by a quad-core ARM Cortex-A76 CPU running at 2.0 GHz, which offers significantly higher computational performance. This makes Raspberry Pi 5 a much more suitable choice for tasks requiring heavy processing, such as object detection, which involves real-time image analysis [75].

3. **ESP8266 - Node MCU:**

   This is an open-source development board with firmware that runs on ESP8266 module. The ESP-8266 module is a wireless programmable microcontroller board. The ESP8266 Wi-Fi board is a SOC with an integrated TCP/IP protocol stack that can give any secondary microcontroller access to a Wi-Fi network [76]. The ESP8266 board is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor and therefore this is more suitable to be used as a sensing node that is capable of sensing the data from various wirelessly connected IoT sensor nodes and send data to the central server like [76].

4. **1080p Webcam (Aukey Full HD 2 Mega Pixel 30 FPS, Wide Angle 110)**

Webcams are widely available and affordable, making them a practical choice for many applications. They typically offer decent resolutions, such as 720p or 1080p, which are sufficient for tasks like lane detection and traffic sign recognition. Additionally, webcams are highly compatible with PCs or laptops, allowing for straightforward integration and real-time processing.

However, webcams do have certain limitations. They generally have a narrower field of view compared to specialized automotive cameras, which could impact their effectiveness for applications that require a broader perspective. Furthermore, webcams may struggle to handle extreme lighting conditions, such as very bright sunlight or low-light environments, as effectively as automotive-grade cameras.

Table 3.14 A comparison of Arduino, Raspberry Pi and ESP8266 Node MCU [76]

	Arduino Due	**Raspberry pi**	**ESP8266 Node MCU**
Developer	Arduino LLC	Raspberry pi foundation	ESP8266 open-source community
Type	Minicomputer	Minicomputer	Single board microcontroller
Operating system	None	Linux	XTOS
CPU	ARM Cortex-M3	ARM Cortex	LXT106
Clock speed	84 MHz	2GHz	26 MHz – 52 MHz
Memory	512KB	4-8GB	Upto 128MB
Storage	external storage (SD)	MicroSD slot	4MB
Power	USB, Battery, power supply	USB, Power supply	USB
Operating voltage	3.3V	5V	3.3V
I/o Connectivity	SPI, I2C, UART, USB	SPI, DSI, I2C, UART, GPIO, USB	UART, GPIO

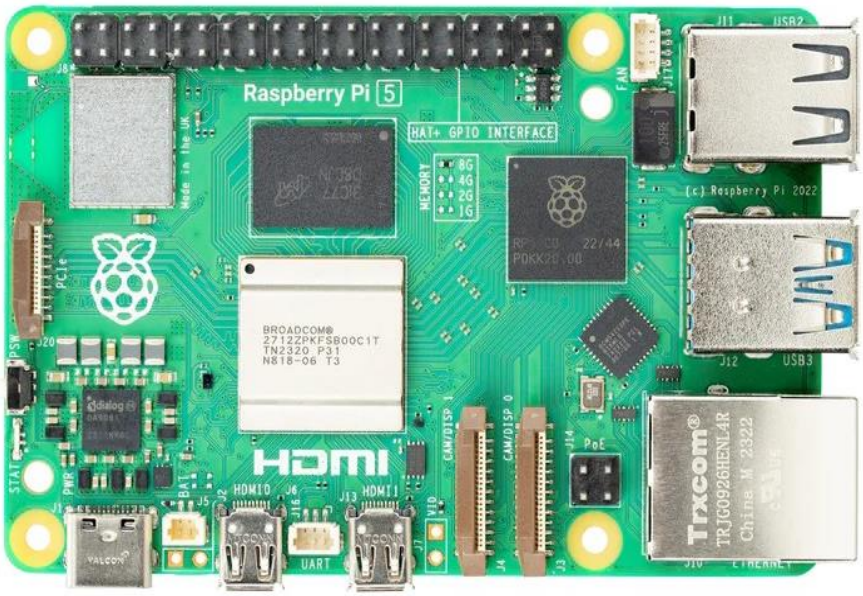


Figure 3.7 Raspberry Pi 5 Configuration [75]

After carefully considering alternatives like the Arduino, which is known for its simplicity and low power consumption, and other microprocessors with higher computational power but increased complexity, the Raspberry Pi emerges as the most suitable option. With its robust processing power, extensive support for machine learning libraries, and a versatile range of I/O options, the Raspberry Pi perfectly aligns with the requirements of an object detection system, making it the optimal choice for this project. [75]

3.6.2 Max30102 pulse and oximeter module

The Max30102 Pulse and Oximeter module is a versatile sensor designed for non-invasive measurement of heart rate and blood oxygen levels. It operates by emitting specific wavelengths of red and infrared light into the skin and detecting the amount of light absorbed by blood. As blood absorbs more light during each heartbeat, the variations in reflected light intensity allow the sensor to calculate the heart rate. Simultaneously, the module determines blood oxygen saturation (SpO2) by analyzing the absorption differences between the red and infrared light. Its compact design and high sensitivity make it an ideal choice for applications like driver monitoring systems, where continuous tracking of physiological parameters is essential for ensuring safety and well-being.

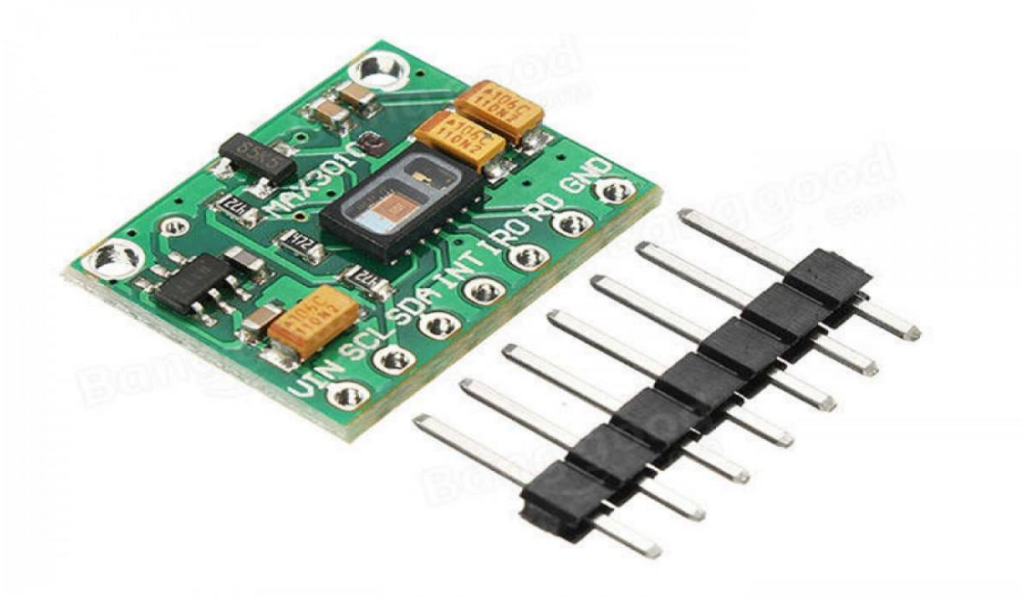


Figure 3.8 MAX30102 Configuration [64]

3.6.2 AD8232 ECG sensor module kit

The AD8232 ECG Sensor Module Kit is a specialized tool for monitoring the electrical activity of the heart, providing critical insights into cardiac health. It operates by detecting and amplifying the electrical signals generated by the heart through electrodes placed on the body. These signals are processed and output as an ECG waveform, which can be analyzed to identify heart rate, rhythm irregularities, and potential heart conditions. The module is equipped with a noise reduction system to ensure clean signal acquisition, making it particularly suitable for real-time monitoring applications. In driver safety systems, the AD8232 enables continuous cardiac monitoring to detect abnormalities that may indicate stress, fatigue, or health emergencies, thereby enhancing overall road safety.

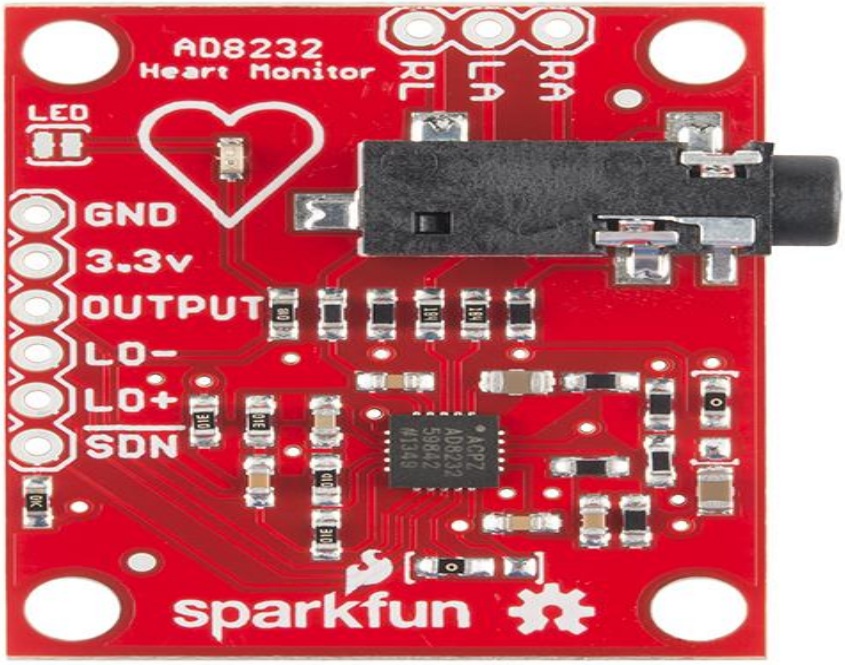


Figure 3.9 AD8232 Configuration [77]

3.6.3 MLX90614

The MLX90614 is a non-contact infrared temperature sensor designed to measure an object's temperature from a distance. Utilizing the Stefan-Boltzmann principle, it detects infrared radiation emitted by an object and converts it into a temperature reading. The sensor offers high accuracy, typically within ±0.5°C, and can measure temperatures across a wide range, making it suitable for various applications. Its non-invasive nature allows it to monitor a driver's body temperature without direct contact, ensuring comfort and hygiene. In driver safety systems, the MLX90614 plays a crucial role in detecting potential health anomalies, such as fever or stress-induced temperature changes, contributing to safer driving conditions.

Figure 3.10 MLX90614 Configuration [68]

3.6.4 Accelerometer & Gyroscope from mobile phone

The accelerometer and gyroscope are essential sensors used for monitoring motion and orientation. The accelerometer measures linear acceleration across three axes (X, Y, and Z), detecting changes in velocity such as sudden stops, rapid acceleration, or upward and downward movements. The gyroscope complements this by measuring angular velocity, tracking rotational movements around the vertical (yaw), lateral (pitch), and longitudinal (roll) axes. Together, these sensors provide a comprehensive analysis of a vehicle's dynamics, such as cornering, braking, and speed bump interactions. In driver behavior monitoring systems, the accelerometer and gyroscope help identify driving patterns, classifying them as aggressive, normal, or cautious, thereby enhancing the understanding of road behavior and contributing to safer driving practices.

3.8 Mobile Application Implementation

To integrate and visualize the data collected from various subsystems in the Intelligent Driver Protection System, a dedicated Flutter-based mobile application was developed. This mobile app acts as the user-facing interface for real-time monitoring and historical analysis, consolidating data from vehicle dynamics, physiological sensors, computer vision modules, and environmental monitoring systems.

3.8.1 Application Features and Functional Overview

The mobile application is organized into clearly demarcated modules; each tailored for a specific data stream:

- Vehicle Profile Management
    - Add and manage multiple vehicles by plate number and model.
    - Store and display both the latest readings and historical data per vehicle.
- Vehicle Dynamics Monitoring
    - Real-time classification of driving behavior using IMU data, PCA, and Random Forest:
        - *Smooth*, *Normal*, *Aggressive*, *Keen*.
    - Allow for manual or automatic data refresh.
- Interior Monitoring (CV-based)
    - Detects unsafe behaviors such as:
        - *Drowsiness*, *Phone usage*, *Hands off wheel*.
    - Shows the detection label and captured image of incidents.
    - Default state shown as "No detection" in the absence of events.
- Biosensor Monitoring
    - Monitors driver health metrics in real time, including:
        - *Heart rate*, *SpO2*, *Respiration rate*, *Temperature*, *Blood Pressure*, *Alcohol level*.
    - All values default to 0 and update as data is received.
- Exterior Monitoring
    - Forward Collision Warning with image and alert label.
    - Lane Safety Alerts:
        - *Lane Departure*
        - *Wrong Direction Warning*
- Event History Logging
    - Maintains a timestamped log of all events including:
        - Driving behavior

        - Health metrics
        - CV detections
        - Exterior monitoring events
- Live Data Communication
    - Syncs with backend via /trigger endpoint.
    - Supports both automatic polling and manual refresh.
- User Interface Design
    - Clean, mobile-friendly layout grouped by modules:
        - *VD*, *CV*, *Bio*, *Speed*, *Exterior*.
    - Utilizes color-coded warnings and image previews for clarity.
    - Designed for fast interpretation by the user, especially during driving.

3.7 Conclusion

This chapter outlined the comprehensive methodology employed to develop an advanced driver monitoring system. The project integrates various innovative technologies and approaches, including machine learning for behavior classification, YOLO for object detection, and Hough Transform for lane monitoring. The use of physiological sensors embedded in a smart steering wheel and smartphone-based vehicle dynamics tracking further emphasizes the multi-faceted approach to addressing driver safety and road behavior analysis.

By meticulously preparing datasets, utilizing advanced detection algorithms, and ensuring robust system integration, the proposed methodology lays a solid foundation for achieving real-time, accurate driver monitoring. The emphasis on scalability and adaptability ensures the system can handle diverse real-world scenarios, contributing significantly to enhancing road safety and reducing accidents caused by human error.

CHAPTER 4
RESULTS AND DISCUSSION

4.1 Introduction

This chapter presents the findings of the project and evaluates it against the initial project objectives outlined in the 1st chapter. The results obtained through system testing, simulations, and performance evaluations are systematically presented and analyzed. Visual aids such as graphs, tables, and screenshots are included to enhance clarity and provide a comprehensive understanding of the outcomes.

The discussion further interprets these results, comparing them with existing research and highlighting the system's strengths and limitations. Additionally, this chapter addresses challenges encountered during the project and their impact on the results, along with suggestions for improvement.

By critically analyzing the findings, this chapter aims to demonstrate how the project contributes to its field while identifying areas for further development.

The outlined results are still in their preliminary stages. Further research and analysis of the system outcomes are still in progress for the 2nd stage of this project.

4.2 Interior Vehicle Monitoring

The results of training and evaluating a YOLOv8 model on a custom dataset are presented. The dataset was composed of nine classes: (eating, drinking, sleepy, hand on wheel, hand not on the wheel, microsleep, seatbelt, cell phone, and cigarette). The model was trained on this dataset to enable the detection and classification of these behaviors. After training was completed, the model was tested in real-world conditions to assess its performance and reliability.

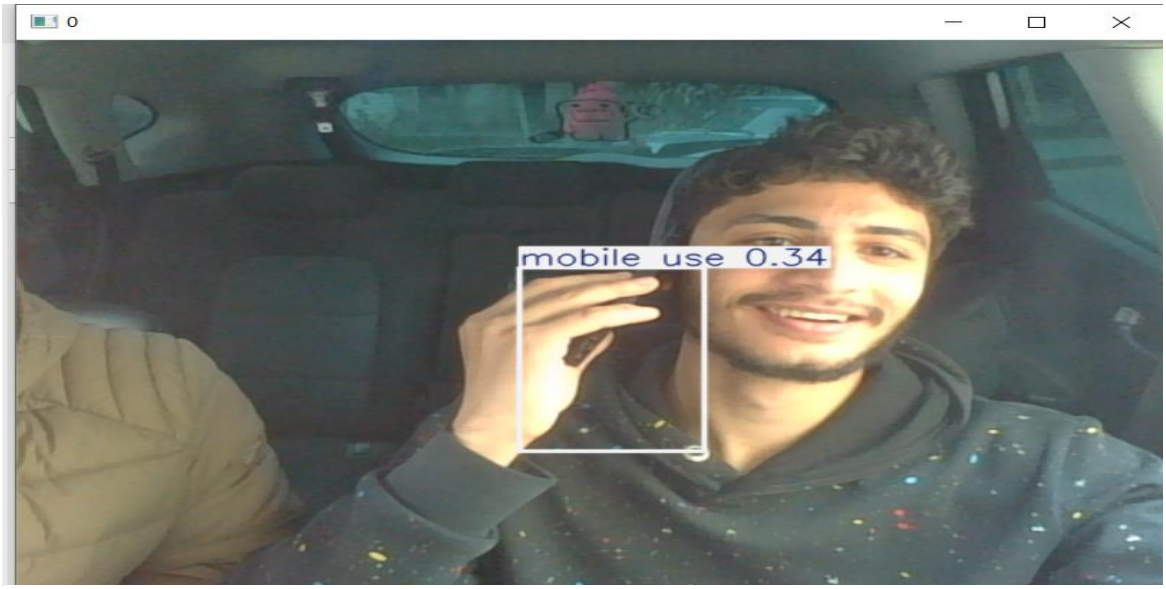


Figure 4.1 Driver using cell phone while driving

The image shows the result of a driver behavior detection system implemented using YOLOv8 object detection. A behavior labeled as "mobile use" is identified with a confidence score of 0.34.

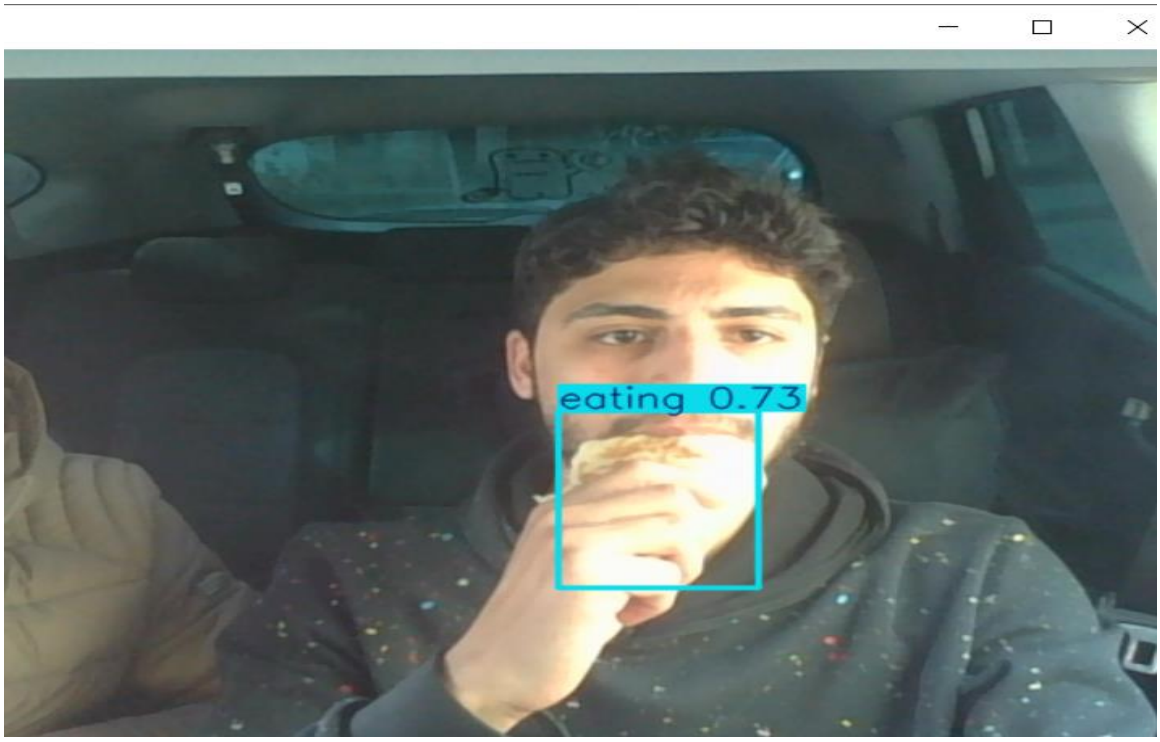


Figure 4.2 Driver eating while driving

The image shows the result of a driver behavior detection system implemented using YOLOv8 object detection. A behavior labeled as "eating" is identified with a confidence score of 0.73, and a bounding box is drawn around the driver's hand and mouth to indicate the detected activity.

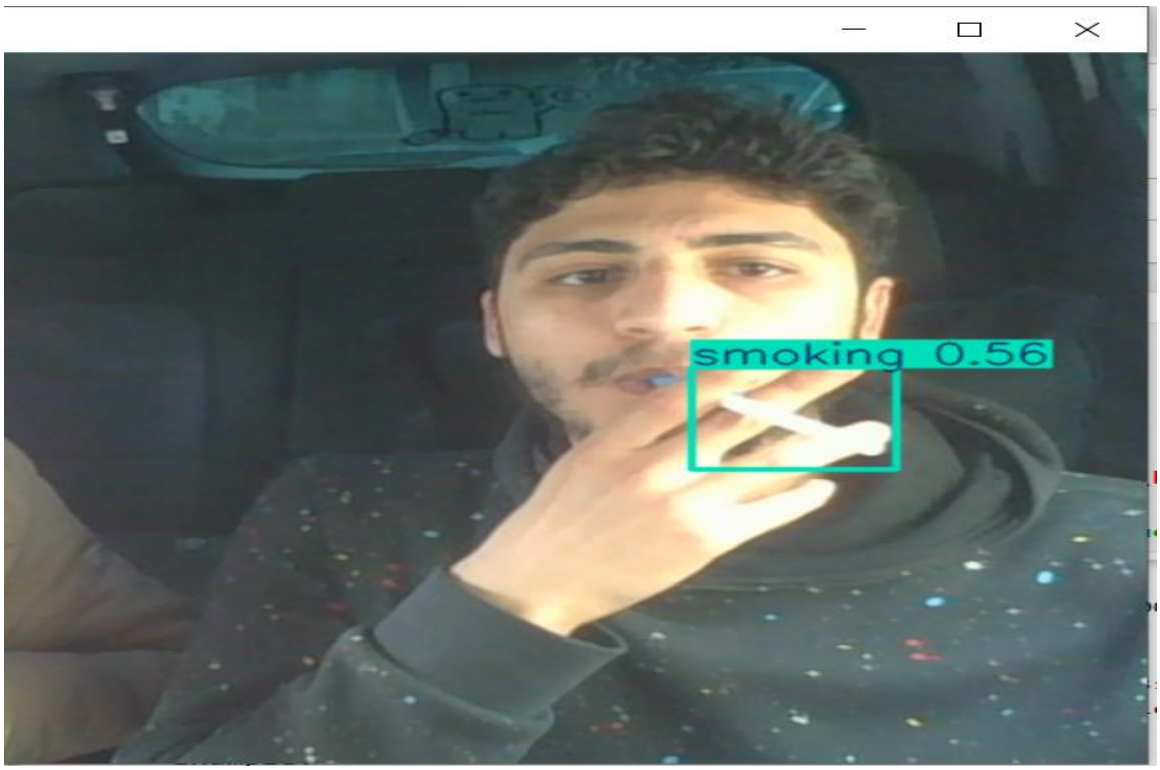


Figure 4.3 Driver is smoking while driving

The image demonstrates the output of a driver behavior detection system developed using YOLOv8 object detection. In this instance, a behavior labeled as "smoking" is identified with a confidence score of 0.56. A bounding box is drawn around the driver's hand and mouth area to highlight the detected activity.

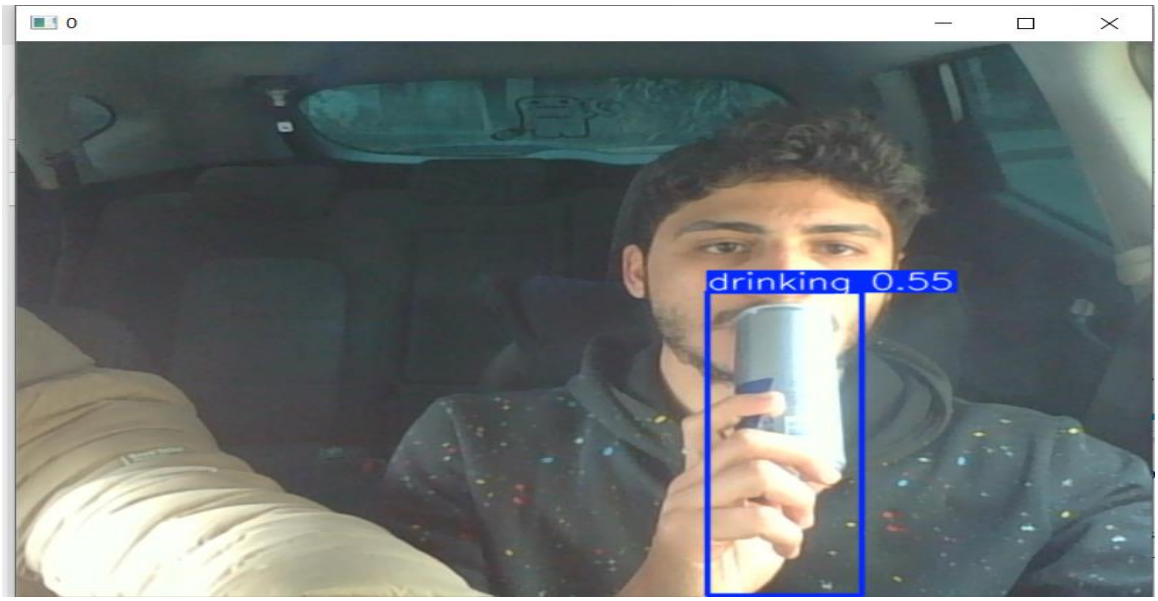


Figure 4.4 Driver drinking while driving

The image illustrates the output of a driver behavior detection system utilizing YOLOv8 object detection. In this case, a behavior labeled as "drinking" is identified with a confidence score of 0.55. A bounding box is drawn around the driver's hand and the object near their mouth to highlight the detected activity.

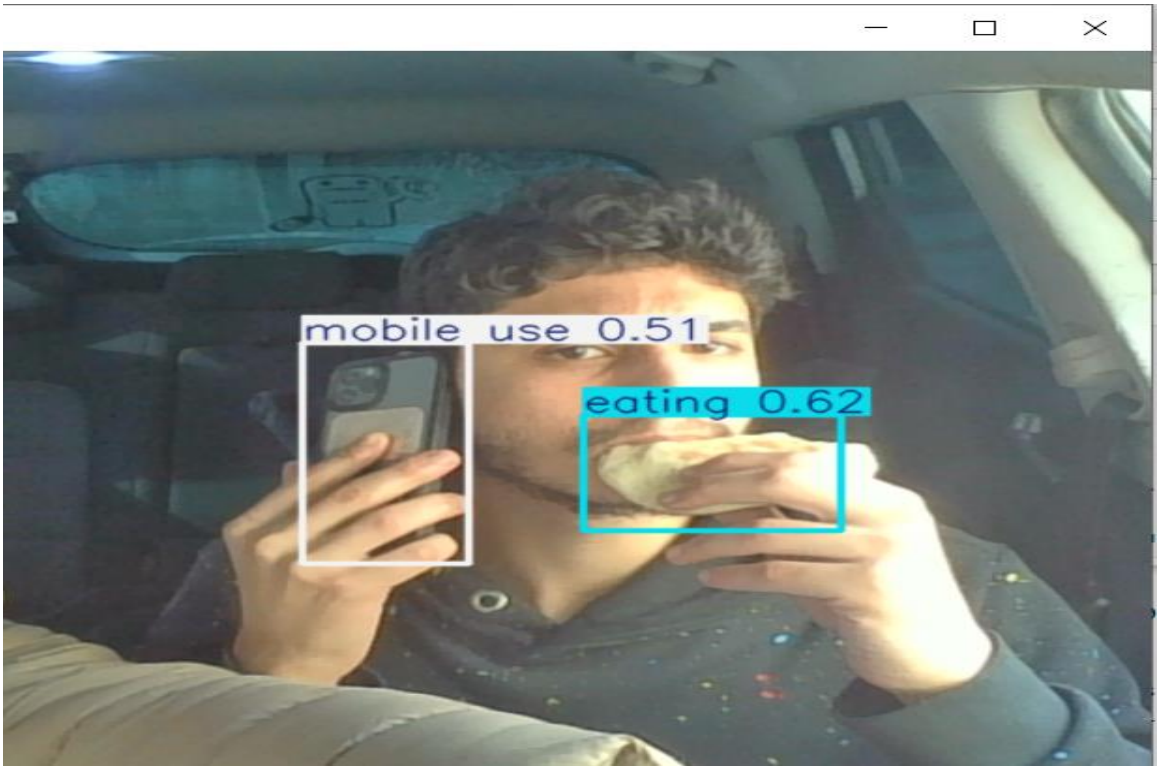


Figure 4.5 Driver using cellphone and drinking while driving

The image demonstrates the output of a driver behavior detection system developed using YOLOv8 object detection. In this instance, a behavior labeled as "eating" is identified with a confidence score of 0.62, and holding a mobile phone to their ear, labeling the action as "mobile use" is identified with a confidence score of 0.51.

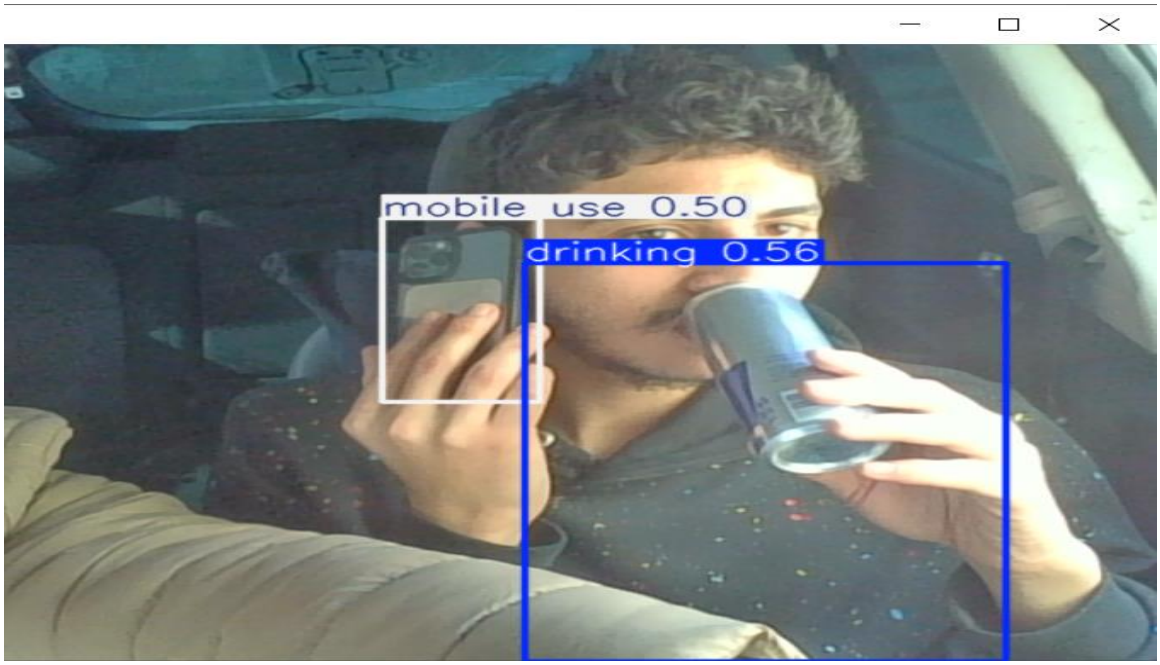


Figure 4.6 Driver using cellphone and drinking while driving

The image demonstrates the output of a driver behavior detection system developed using YOLOv8 object detection. In this instance, a behavior labeled as "drinking" is identified with a confidence score of 0.56, and holding a mobile phone to their ear, labeling the action as "mobile use" is identified with a confidence score of 0.50.

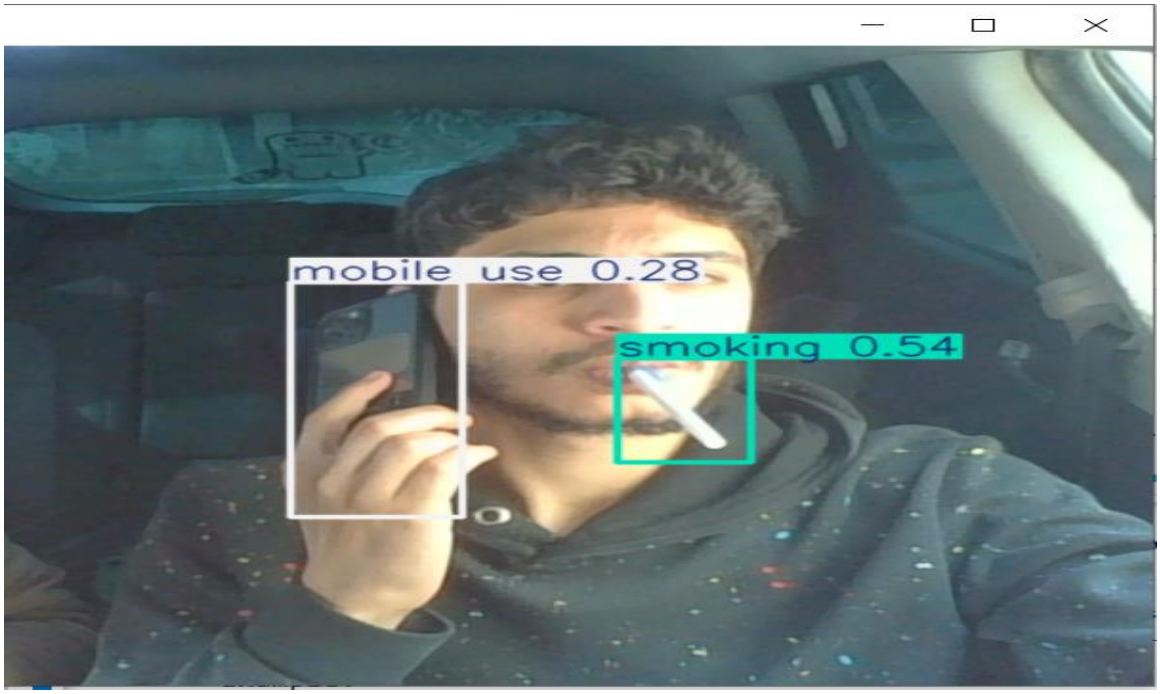


Figure 4.7 Driver using cellphone and smoking while driving

The image demonstrates the output of a driver behavior detection system developed using YOLOv8 object detection. In this instance, a behavior labeled as "smoking" is identified with a confidence score of 0.54, and holding a mobile phone to their ear, labeling the action as "mobile use" is identified with a confidence score of 0.28.

4.2.1 YOLOv8m Model optimization

The YOLOv8m model was trained on **8143 images** with **60 epochs** to detect various driver behaviors. The evaluation results indicate that strong overall performance was achieved, though some areas require improvement.

4.2.1.1 Overall Model Performance

A **high precision (95.2%)** and **good recall (93.1%)** were obtained, meaning that a large proportion of the relevant objects were correctly identified while a relatively low false positive rate was maintained. The **mAP@50 (96%)** suggests that good performance was exhibited in detecting objects when a loose Intersection over Union (IoU) threshold was used. However, the **mAP@50-95 (59.6%)** indicates that a decline in detection performance was observed under stricter IoU thresholds, meaning that some predictions may not have been highly precise in terms of bounding box placement.

Table 4.1 Overall Model Performance

Metric	Value	Reference Value
Precision (P)	(95.2%) ✅ Good	>80%
Recall (R)	(93.1%) ✅ Good	>75%
mAP@50	(96%) ✅ Good	90%–97%
mAP@50-95	(59.6%) ✅ Good	50–65%

4.2.1.2 Interpretation:

- The model demonstrates strong overall performance, with high precision and recall values indicating effective object detection and localization capabilities.
- An mAP@0.5 of 96% reflects excellent detection accuracy at a moderate IoU threshold, confirming the model's reliability in identifying objects correctly.
- The mAP@0.5:0.95 score of 59.6%, while lower, is expected. This metric assesses performance across a range of IoU thresholds, placing greater emphasis on precise localization.

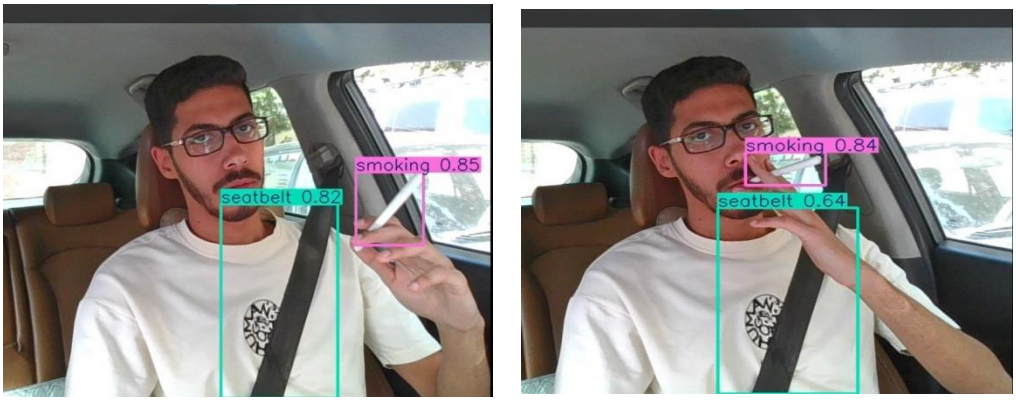


Figure 4.8 Smoking Detection

In the detected result, an object classified as "Cigarette" has been identified and enclosed within a blue bounding box. A confidence score of 0.83 has been assigned, indicating that the model is 85% certain about the detection.

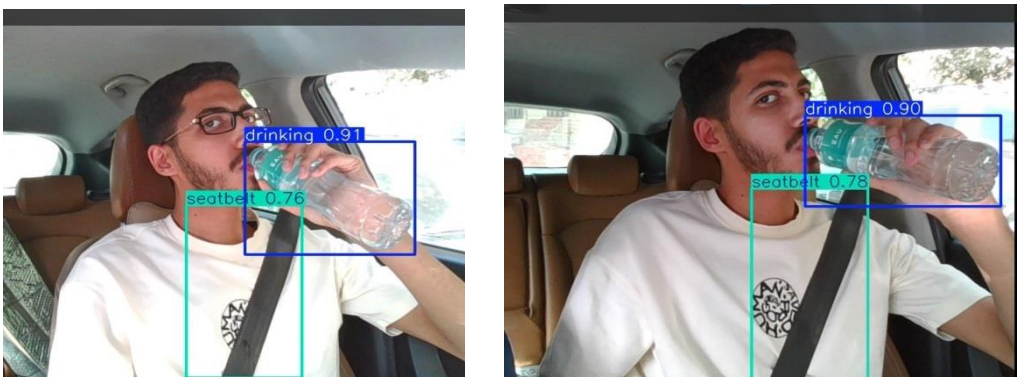


Figure 4.9 Drinking Detection

The image illustrates the output of a driver behavior detection system utilizing YOLOv8 object detection. In this case, a behavior labeled as "drinking" is identified with a confidence score of 0.91. A bounding box is drawn around the driver's hand and the object near their mouth to highlight the detected activity.

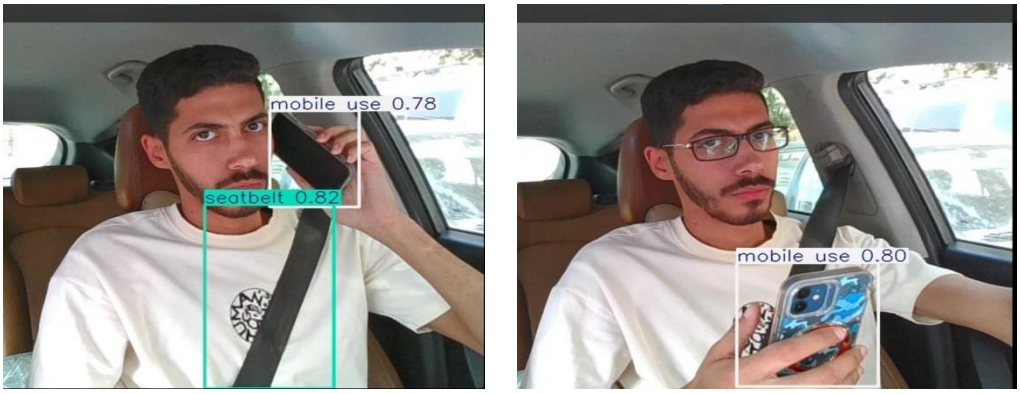


Figure 4.10 Cellphone Detection

The image shows the result of a driver behavior detection system implemented using YOLOv8 object detection. A behavior labeled as "mobile use" is identified with a confidence score of 0.83.

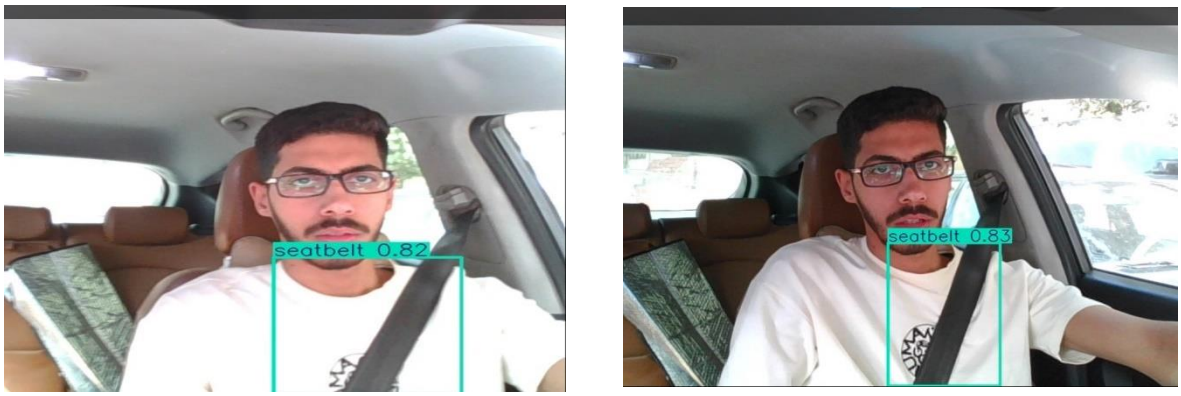


Figure 4.11 Seat-Belt Detection

In the captured detection result, the object identified as "Seatbelt" has been highlighted with a red bounding box, and a confidence score of 0.83 has been assigned, indicating that the model is 83% certain ab out the detection.

Figure 4.12 Sleeping Detection

The image shows the result of a driver behavior detection system implemented using YOLOv8 object detection. A behavior labeled as "sleep" is identified with a confidence score of 0.73.

Figure 4.13 Eating Detection

The image shows the result of a driver behavior detection system implemented using YOLOv8 object detection. A behavior labeled as "sleep" is identified with a confidence score of 0.87.

4.3 Vehicle Exterior Monitoring

This section outlines the results obtained from each implemented part of the exterior monitoring phase, including Lane Departure Monitoring (LDM), Forward Collision Monitoring (FCM), and Wrong-Way Driving Detection (WWD). The performance of each system was evaluated through various tests using image, video data, real-world webcam live stream, as well as real-world GPS recordings.

4.3.1 Traffic Sign Detection using YOLOv11

Data

The Self Driving Car Dataset is used to train the traffic sign detection model. It contains **4969** total images split into train, validation and test sets with **3530**, **801** and **638** images of dimension 416x416 respectively. The dataset contains images of 15 different traffic signs.

The classes available in the dataset are:

1. Green Light
2. Red Light
3. Speed Limit (120-110-100-90-80-70-60-50-40-30-20-10)
4. Stop

Model

The yolo11n version of the model is used to fine-tune on the dataset. The model was trained for **50** epochs with batch size **16**.

YOLO11n Evaluation

Table 4.2 YOLO11n Evaluation

Class	Images	Instances	Box(P)	R	mAP50	mAP50-95)
all	801	944	0.95	0.905	0.959	0.836
Green Light	87	122	0.901	0.743	0.851	0.525
Red Light	74	108	0.891	0.722	0.844	0.529
Speed Limit 100	52	52	0.95	0.942	0.989	0.889
Speed Limit 110	17	17	0.916	1	0.986	0.915
Speed Limit 120	60	60	1	0.943	0.995	0.908
Speed Limit 20	56	56	0.981	0.93	0.985	0.871
Speed Limit 30	71	74	0.963	0.959	0.984	0.924
Speed Limit 40	53	55	0.935	0.945	0.988	0.887
Speed Limit 50	68	71	0.973	0.915	0.98	0.886
Speed Limit 60	76	76	0.92	0.912	0.96	0.89
Speed Limit 70	78	78	0.987	0.962	0.981	0.9
Speed Limit 80	56	56	0.96	0.929	0.973	0.866
Speed Limit 90	38	38	0.954	0.789	0.924	0.784
Stop	81	81	0.975	0.982	0.988	0.929

Detections

Figure 4.14 Speed Limit 40 Sign Detection

In the image, the YOLOv11 object detection model has successfully identified a "Speed Limit 40 KPH" sign with high accuracy, as shown by the bounding box and the label "**Speed Limit 40**" with a confidence score of **0.96**. This demonstrates the model's excellent performance in:

Accurate text and symbol recognition on road signs.

Robustness in natural environments, as the sign is detected clearly despite the complex background of trees and shadows.

High confidence level, indicating the model is confident in its prediction, which is essential for real-time autonomous driving systems.

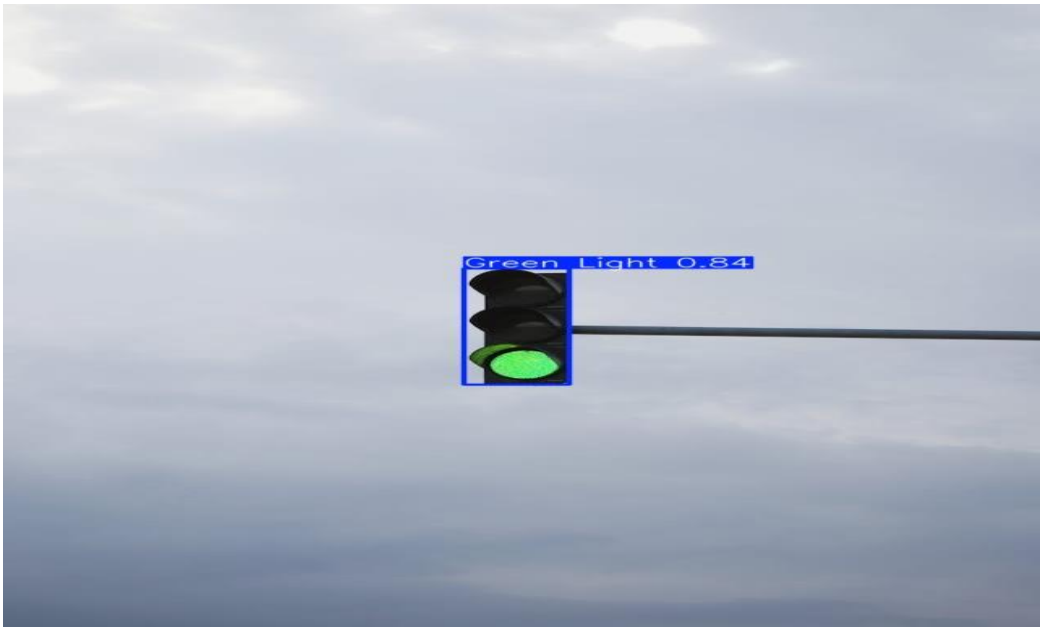


Figure 4.15 Green Light sign Detection

In this image, the YOLOv11 model has successfully detected a green traffic light, labeling it as "Green Light" with a confidence score of 0.84. The bounding box is correctly placed around the active light, confirming the model's ability to:

Recognize traffic signals accurately, even in a minimal and clear sky background.

Detect the specific light state (green), which is crucial for traffic flow understanding and vehicle decision-making.

Maintain good performance despite varying lighting conditions, as the scene is overcast with diffused light.

This detection showcases YOLOv11's real-time object detection strength in handling critical traffic control elements, which is essential in autonomous driving, intelligent transportation systems, and smart traffic monitoring.

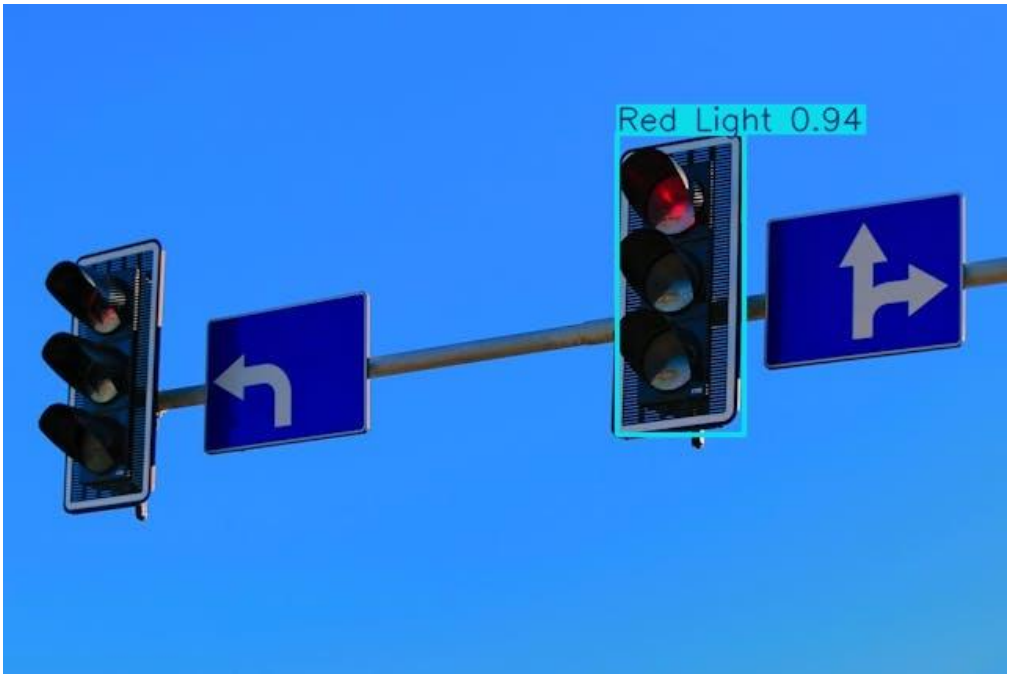


Figure 4.16 Red Light Detection

In this image, the YOLOv11 model has accurately detected a red traffic light, labeling it as "Red Light" with a high confidence score of 0.94. The bounding box is perfectly aligned with the illuminated red signal, which reflects the model's ability to:

Precisely identify the current signal state, critical for real-time traffic decision-making.
Perform accurate detection even in complex urban setups, where multiple lights and directional signs are present.

Distinguish between multiple traffic light positions and road signs, such as the directional signs for left and straight or right turns, shown in the background.

This detection demonstrates YOLOv11's robustness, precision, and practical usability for traffic signal recognition in autonomous vehicles, smart city systems, and traffic control applications.

Figure 4.17 Stop Sign Detection

In this image, the detected **Stop** has a **confidence score of 0.96**

This project demonstrates how a fine-tuned YOLOv11 model can be used for traffic sign detection.

Here are a few use cases for this project:

1. Autonomous Vehicle Navigation: The model can be used in self-driving car systems to recognize traffic signs accurately. This would enable autonomous vehicles to follow traffic rules and regulations, analyzing every sign whether it's about speed limit or stop-and-go indications to navigate the roads safely.
2. Traffic Rule Compliance: This model can be used in driver assistance systems to ensure that drivers comply with all traffic rules. Alerts can be generated when drivers exceed the speed limit or don't stop at red lights, fostering safer roads.
3. Road Safety Training Programs: This model allows Driving schools and automotive companies to build simulations and education programs. These programs can guide new drivers in identifying and responding to different traffic signs, thus enhancing road safety knowledge.
4. Smart City Infrastructure: City authorities could use this model in connected CCTV or IoT infrastructure to track and monitor traffic

compliance in real time, helping identify areas with frequent rule violations for potential improvement.

5. Road Network Analysis: Transportation engineering researchers can use this model to analyze how efficiently different sign classes are distributed and recognized around the city. This data can be instrumental in planning more efficient and safer road networks.

4.3.2 Lane Departure Monitoring (LDM)

The lane departure system was implemented using traditional computer vision techniques, Detected Lane lines are overlaid on video frames for visualization. This overlay uses distinct colors to indicate the detected lanes.

- **Canny Edge Detection** and **Hough Transform** were used to extract lane lines from road images and video.
- A logic system was added to detect deviations from the estimated lane center.
- A departure event was registered when the vehicle's estimated center deviated from the center of the lane by a set threshold.
- A temporal filter was used where **more than 2 lane departures within a 15-second window** triggered an **"AGGRESSIVE DRIVING DETECTED"** warning.

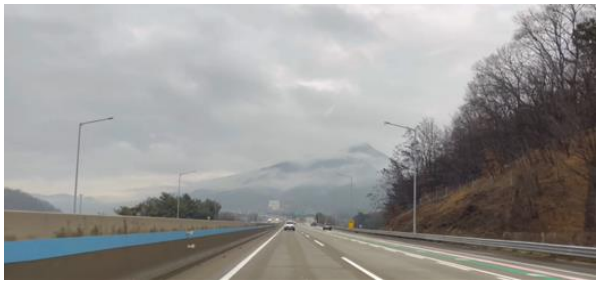


Figure 4.19 Before Processing

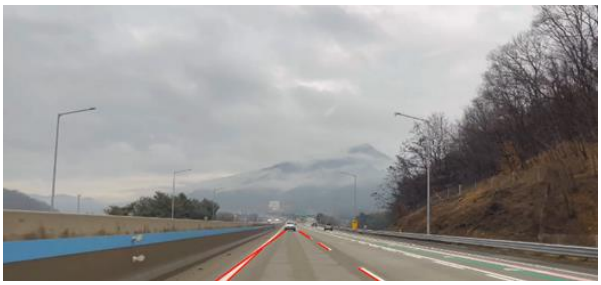


Figure 4.18 After Processing

Observations and conclusions:

- The algorithm demonstrates stable performance across most frames, which is a positive indicator of its reliability.
- Frames with fewer than 2 lanes detected highlight situations where the algorithm struggled. These could correspond to: Occlusions (vehicles or debris blocking lanes), Poor contrast (faded or worn lane markings) or Changes in road design (merging lanes or intersections).

4.3.3 Forward Collision Monitoring (FCM)

The forward collision phase evolved through multiple iterations to improve performance in both accuracy and processing speed.

Initial Attempt: AdaBins Depth Estimation

AdaBins was tested on indoor images in various conditions using different datasets (nyu for indoor scenes and Kitti for outdoor scenes) and showed promising results with realistic metric depth estimation. The following are results of testing the model on both indoor and outdoor images:

- Indoor scenes

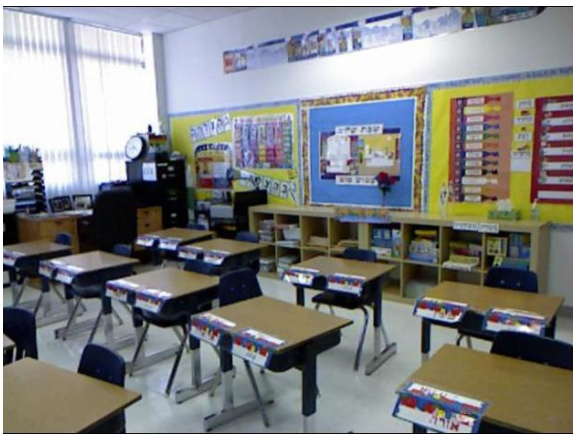


Figure 4.21 Actual Classroom Image

Figure 4.20 Predicted depth map for classroom image

- Outdoor scenes in different lighting conditions

Figure 4.23 Actual Street scene 1

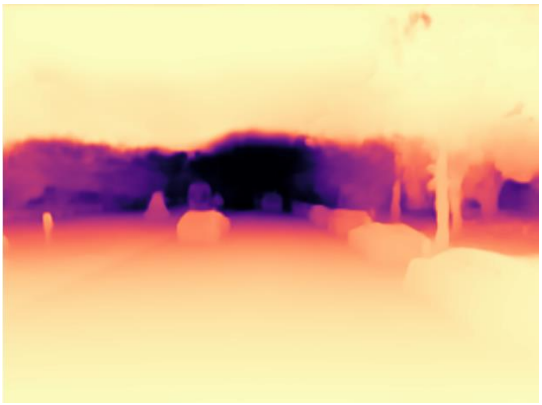


Figure 4.22 Predicted Depth map for street scene 1

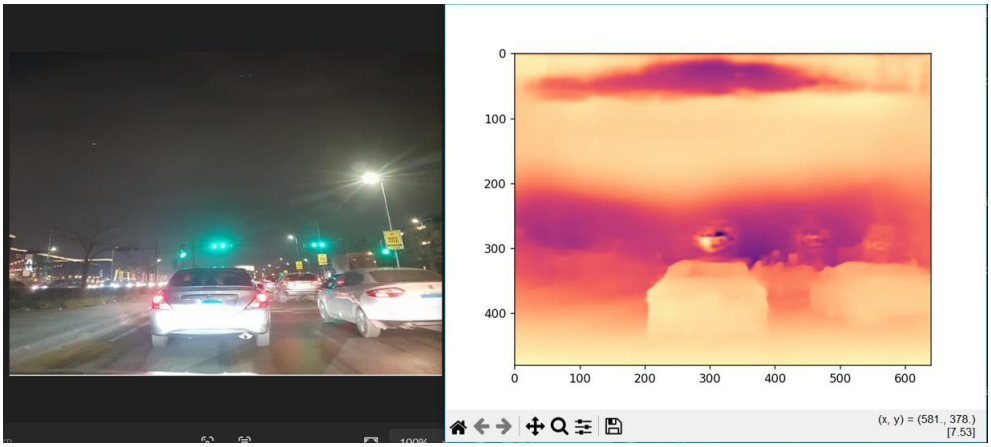


Figure 4.24 Outdoor scene 2

- Live video stream using webcam

Figure 4.26 Extracted frame from video stream

Figure 4.25 Corresponding depth map for extracted frame

Observations and conclusions:

The **AdaBins** model performed fairly well for both indoor and outdoor scenes with crisp clear depth map predictions with high accuracy but long processing time. However, for live video, the model performed poorly in both indoor and outdoor scenes, The depth map was nearly empty, Processing was slow and unusable for real-time monitoring.

Refined Approach: MiDaS + YOLOv5

Switched to MiDaS (DPT_Hybrid), which gave better results on both indoor and outdoor images and live video stream. Depth maps were clean, and scene understanding was better, but still not optimal in real-time as it was a little slow and laggy.

Final configuration used:

- MiDaS DPT_Large for depth estimation.

- YOLOv5 for object detection (cars, trucks, buses, pedestrians).
- Bilateral filtering is applied to smooth depth maps.
- A Region of Interest (ROI) was defined to ignore irrelevant detections.

The following figures show results of testing the final configuration model on live video stream in different roads.

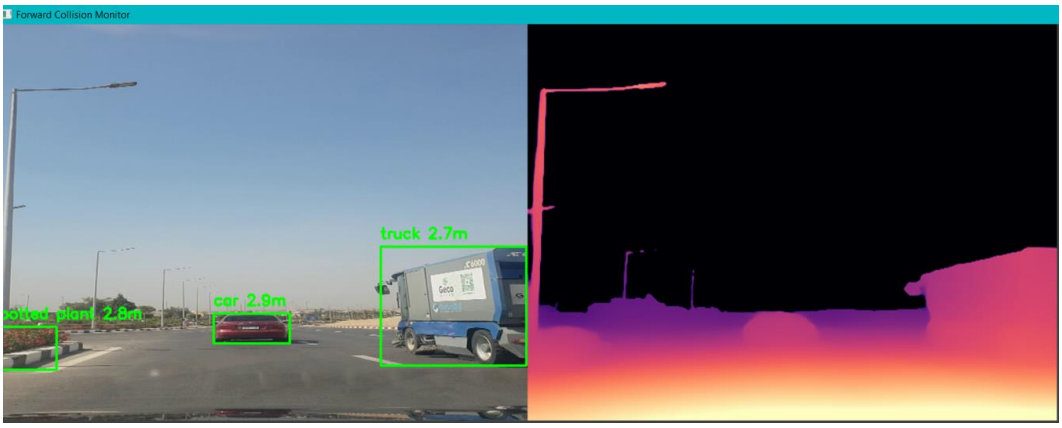


Figure 4.27 Comparing real extracted frame from video stream to its predicted depth map 1

Figure 4.28 Comparing real extracted frame from video stream to its predicted depth map 2

Collision Warning Logic:

- An object is considered dangerous if its relative depth is below a threshold of 5.0m (Relative depth).
- Since MiDaS does not provide true metric depth, the threshold was empirically determined based on relative proximity in various scenes.
- "COLLISION WARNING!" was triggered visually when this condition was met.

- Number of close-call warnings in a defined time window (more than 3 collision warnings within 2 minutes). This sends a notification of aggressive driving.

Figure 4.29 shows a collision warning message depending on the predefined threshold indicating potential collison.

Figure 4.29 Collison Warning

Observations and conclusions:

- This final configuration achieved high responsiveness and clarity, suitable for **real-time forward-famous collision alerts**.
- Despite lacking true metric values, **scene understanding and relative positioning** were strong enough to support safe alerting.

4.3.4 Wrong-Way Driving Detection (WWD)

A dedicated system was developed using OpenStreetMap (OSM) and GPS-based direction analysis. The system reads GPX files or direct GPS coordinates to determine the vehicle's movement heading. Using OSMnx and NetworkX, the road graph was downloaded for the vehicle's current location.

- For each recorded point:
    - The vehicle's heading was calculated based on GPS coordinate change.
    - The road's official direction was extracted from OSM metadata.
- The angular deviation between heading and road direction was computed.
    - If the deviation exceeded a logical threshold (100° for one-way roads), the system raised a "**WRONG DIRECTION DETECTED**" warning, figures 4.30 and 4.31.

Figure 4.30 Terminal output showing WWD warning

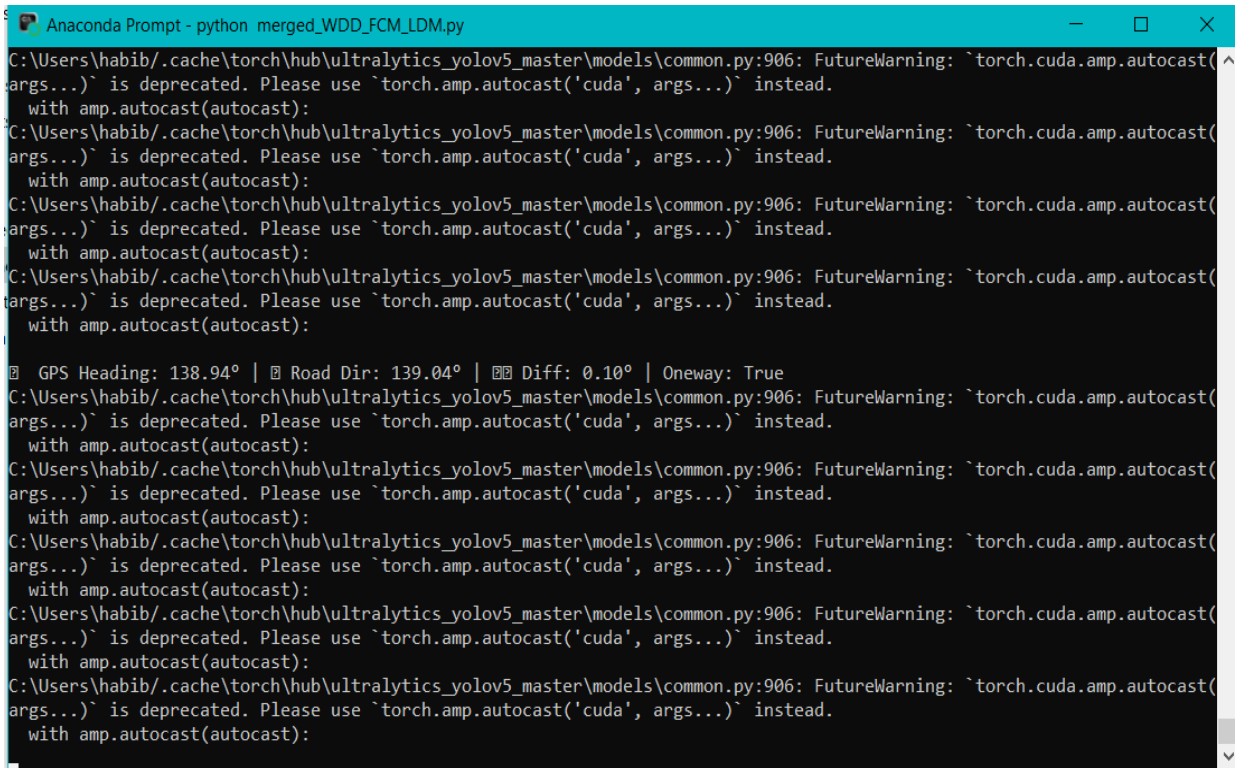


Figure 4.31 Terminal output of the 3-model merge WDD, FCM, and LDM.

Figure 4.32 shows an extracted frame from output window of the 3 merged models (WWD, LDM, and FCM) that shows how the 3 models are merged together and operating smoothly.

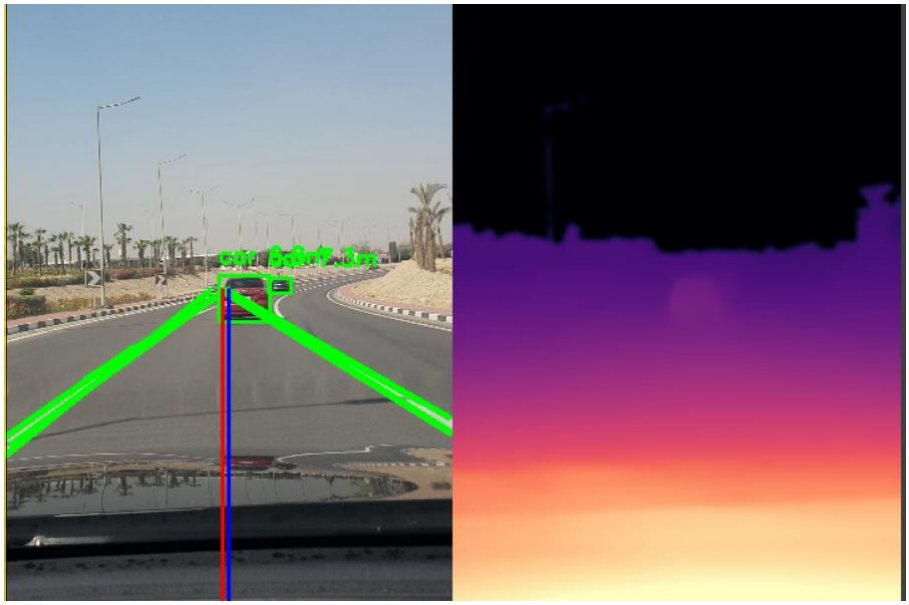


Figure 4.32 3-model merge output

Observations and conclusions:

- The system worked reliably when tested with GPX recordings from mobile apps and GPS module data.
- One-way road metadata was correctly extracted and used.

Table 4.3 Results Summary

Feature	Method Used	Evaluation	Outcome
Lane Detection	Canny + Hough	Video footage	Accurate line detection in most conditions
Lane Departure	Pixel deviation threshold + time window	Live video	Detected aggressive swerving successfully
Depth Estimation	AdaBins (initial), MiDaS DPT_Large (final)	Image, webcam	Final setup fast & clear, suitable for real-time
Object Detection	YOLOv5	Combined with depth	Detected key obstacles reliably
Collision Warning	Relative depth < 5.0	Real-time	Responsive and reliable
Wrong-Way Detection	GPS + OSM + heading deviation	GPX recordings	Detected wrong direction driving accurately

4.3.5 Conclusion

The integrated driver monitoring system successfully detects lane departures, forward collision risks, and wrong way driving by combining traditional vision methods with modern deep learning and mapping APIs.

Despite challenges in processing speed and depth accuracy in earlier versions, the final system showed real-time capability and strong scene understanding. It

offers a foundation for future enhancements, such as true metric depth estimation, GPS module integration, and full deployment on mobile or embedded systems.

4.4 Physiological Monitoring of Driver

The comprehensive integrated sensor system successfully acquired and processed vital physiological parameters, with the results primarily documented through visual representation in figures derived from real-time readings. Analysis of the SpO2 data, captured from the MAX30102 pulse oximeter, consistently showed readings that, as depicted in Figure 4.33, visually aligned with healthy physiological ranges for oxygen saturation (between 96%-97%). The stability and responsiveness of these readings, as visually demonstrated across various captured instances, attested to the effectiveness of the empirically adjusted SpO2 calculation formula ($SpO_2 = 110 - 12 \cdot R$) in translating raw red and infrared PPG signals into a clear representation of saturation levels. While the system demonstrated robust SpO2 measurement capabilities, visual inspection of the captured data sometimes revealed transient variations attributable to factors like user movement or subtle shifts in sensor placement, underscoring the importance of maintaining stable contact for optimal performance.

Simultaneously, respiration rate estimation from the same MAX30102's infrared signal proved effective. The bandpass filtering (0.1−0.4 Hz) successfully isolated the respiratory component, manifesting as clear periodic variations in the resulting readings as observed in Figure 4.33. Subsequent peak detection reliably identified distinct respiratory cycles, allowing for estimation of respiration frequency that consistently matched expected rates for a typical human subject.

```
Collecting 60 seconds of data for the next calculation...
Finished data collection for this cycle.
Estimated SpO2: 96.06%
Estimated Respiration Rate: 9.0 breaths/min
.....

SpO2: 97.97%
Respiration Rate: 8.0 bpm
------------------------------
```

Figure 4.33 MAX30102 SpO2 & Respiration Rate Readings

Beyond these primary parameters, the AD8232 ECG sensor consistently provided visually discernible and stable electrocardiogram waveforms, as clearly illustrated in Figure 4.34, indicating its potential for detailed analysis of heart rate variability and cardiac rhythm, though a deeper dive into cardiac analysis was beyond the immediate scope.

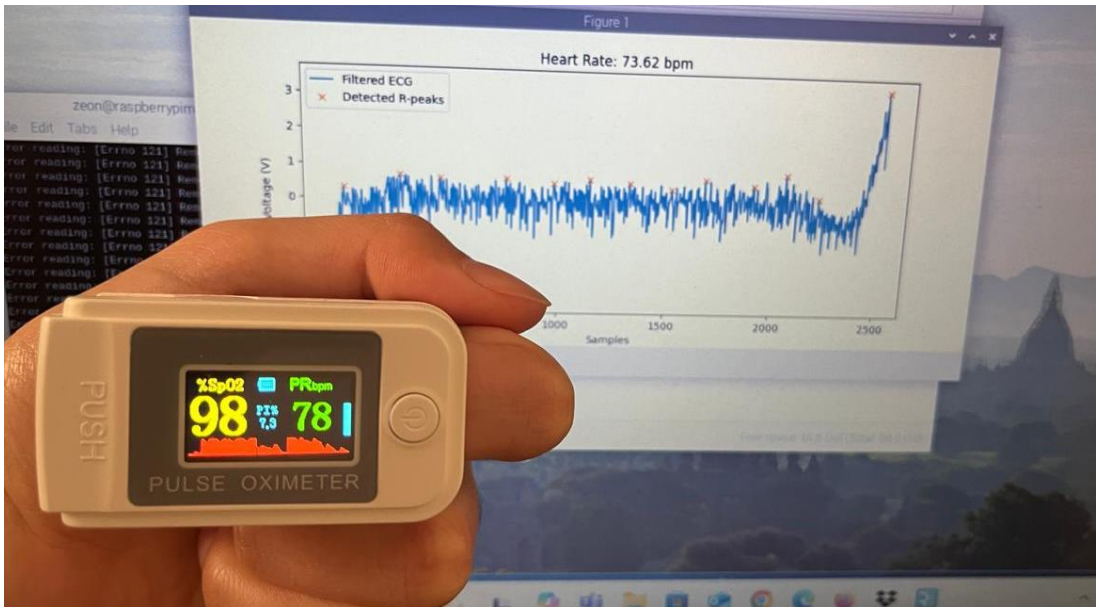


Figure 4.34 ECG Heart Rate Reading

The MLX90614 non-contact infrared thermometer delivered clear temperature readings, as shown in Figure 4.35, which visually corresponded to expected body surface temperatures.

```
------------------------------
Ambient Temperature: 22.81 °C
Object Temperature: 36.07 °C
------------------------------
Ambient Temperature: 22.81 °C
Object Temperature: 36.07 °C
------------------------------
Ambient Temperature: 22.81 °C
Object Temperature: 36.07 °C
------------------------------
Ambient Temperature: 22.81 °C
Object Temperature: 36.07 °C
------------------------------
Ambient Temperature: 22.81 °C
Object Temperature: 36.07 °C
------------------------------
Ambient Temperature: 22.81 °C
Object Temperature: 36.07 °C
------------------------------
```

Figure 4.35 MLX90614 Temperature Reading

The ADS1115 Analog-to-Dital Converter, critical for acquiring high-resolution analog signals from sensors like the AD8232, functioned seamlessly, visually demonstrating the precise digitization of subtle physiological variations in the captured raw data.

It is noteworthy that the initial inclusion of a generic Pulse Sensor was abandoned during the design phase due to its consistent delivery of visually noisy and unreliable readings, as contrasted with the clear signals from the AD8232, thus highlighting the critical importance of selecting robust and high-fidelity sensor components. The overall performance of the custom-designed PCB and its integrated components, as evidenced by the stable and clean data presented in the various figures, underscored the efficacy of the power management strategy, including the 5V to 3.3V voltage regulation. Furthermore, the meticulously selected trace widths (0.5 mm for power, 0.1524 mm for signals and GND) and the strategic implementation of a GND polygon pour demonstrably contributed to minimal voltage drop and superior signal integrity, thereby reducing noise and ensuring the stable operation of the sensitive analog sensors. This robust physical design proved crucial in realizing a reliable low-current, multi-sensor platform. Figure 4.36 shows the manufactured completed PCB board. While Figure 4.37 and Figure 4.38 show the sensors' and electrodes' placement on one side of the

steering wheel. It has to be noted the setup is left bare for visibility of placement in the prototype, actual product involves the sensors being custom embedded into the steering wheel to avoid driver discomfort.

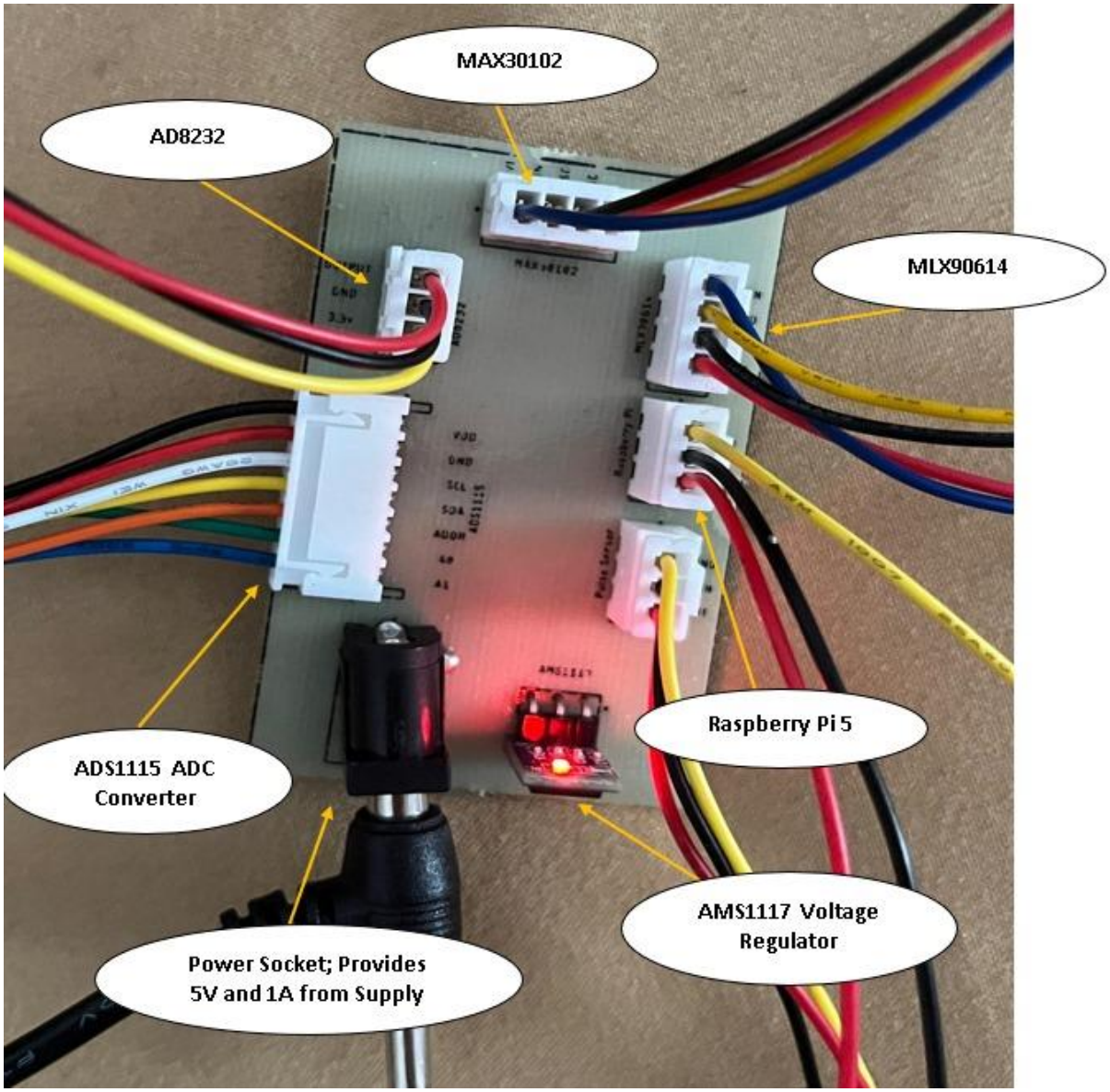


Figure 4.36 Manufactured PCB Board

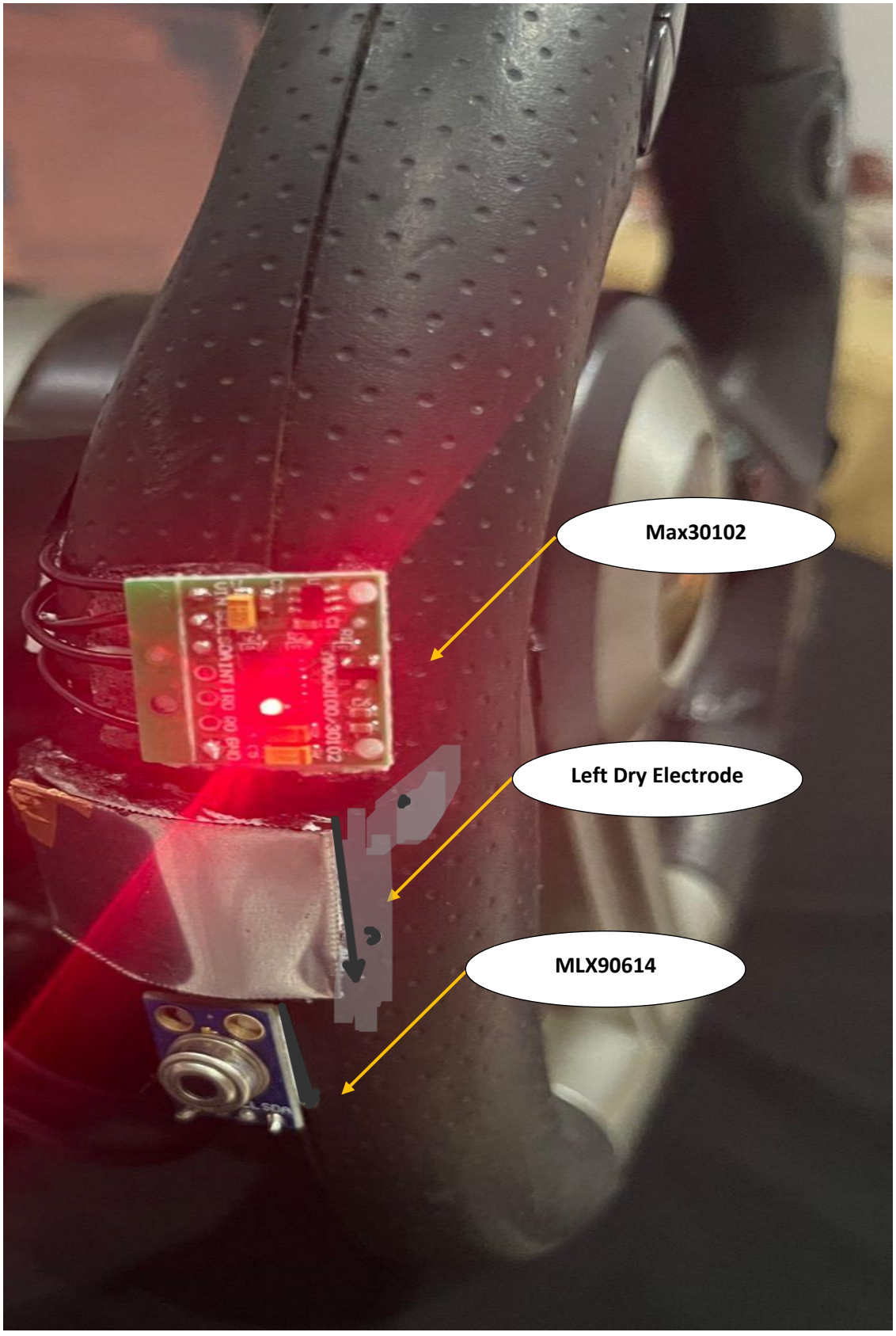


Figure 4.37 Sensors and Electrodes Placement on The Steering Wheel (Left Side)

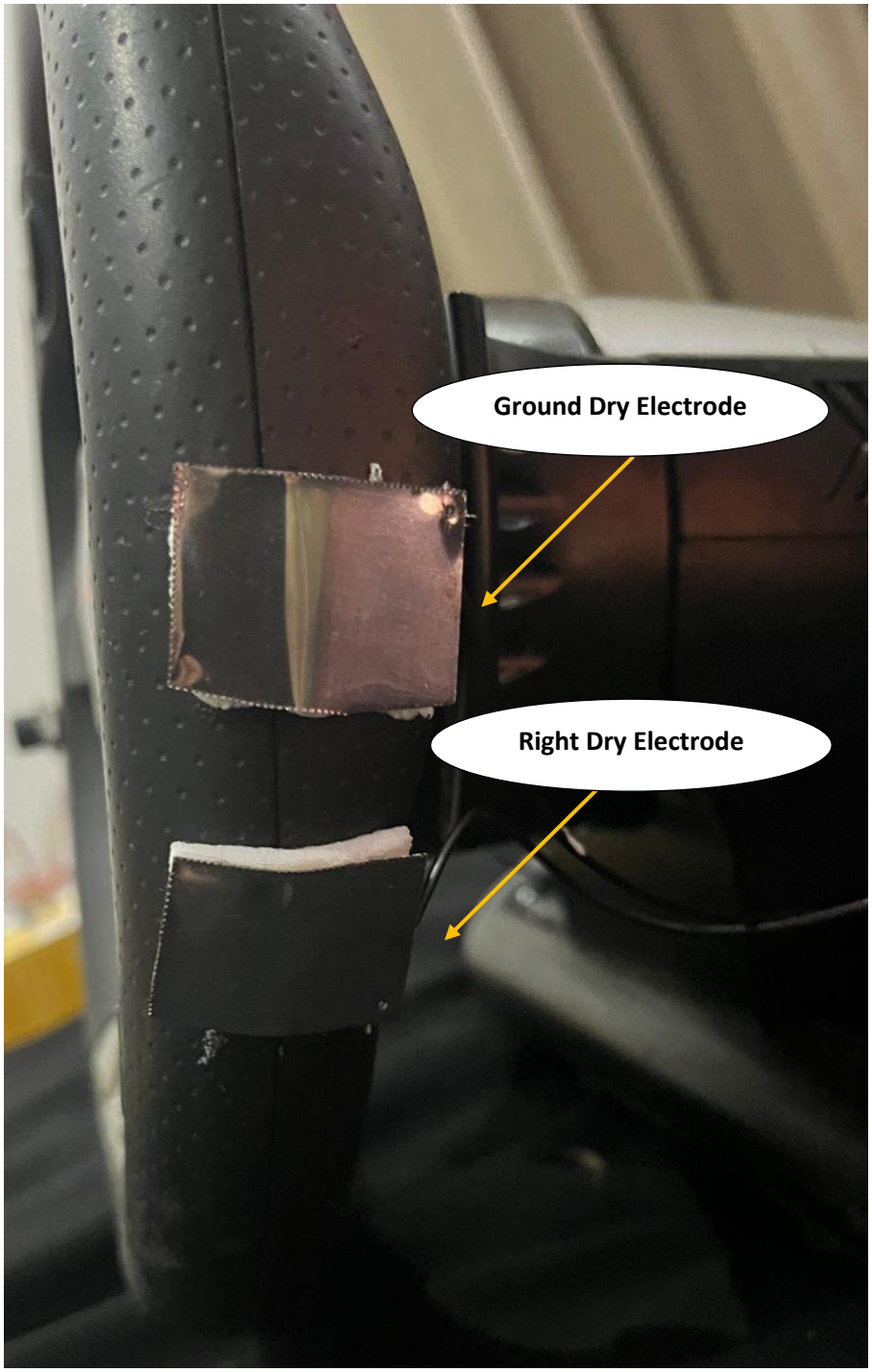


Figure 4.38 Sensors and Electrodes Placement on The Steering Wheel (Right Side)

The collected physiological parameters—Heart Rate, Pulse Transit Time (PTT), Estimated Blood Pressure (BP), SpO2, Respiration Rate, and Body Temperature—were utilized as input to determine an estimated Blood Alcohol Content (BAC) level. As depicted in Figure 4.39, after a 30-second data collection period, the system outputs these vital signs: a Heart Rate of 76.7 bpm, PTT of 0.3360 seconds, an Estimated BP of 110.1/73.4 mmHg, SpO2 of 96.89%,

Respiration Rate of 8.0 bpm, and Body Temperature of 37.30 ∘C. Critically, these measured parameters were then compared against a predefined table of set criteria, where specific ranges or thresholds for each physiological indicator correspond to different BAC levels. For instance, deviations from normal heart rate, changes in PTT, alterations in respiration patterns, or fluctuations in body temperature could individually or collectively trigger an estimated BAC classification. In the depicted instance in Figure 4.39, based on the observed values, the system classified the Estimated BAC as 0.00% (Normal), indicating that the combination of the measured physiological data fell within the criteria associated with a sober state. This classification highlights the integrated approach, where multiple physiological markers contribute to a holistic assessment, allowing for a more robust and context-aware estimation of BAC, rather than relying on a single isolated parameter.

```
Starting BP + BAC estimation using PTT...

Collecting ECG and PPG data (30s)...
Heart Rate: 76.7 bpm
PTT: 0.3360 seconds
Estimated BP: 110.1/73.4 mmHg
SpO2: 96.89%
Respiration Rate: 8.0 bpm
Body Temperature: 37.30 °C
Estimated BAC: 0.00% (Normal)
----------------------------------------

Collecting ECG and PPG data (30s)...
```

Figure 4.39 Full Parameters Set of Readings

Table 4.4 presents a comparison between the biosignal readings obtained from a steering wheel-based medical monitoring system and actual reference measurements collected once every hour over a five-hour period. During each hour, the subject placed their hand on the steering wheel sensor, which automatically captured physiological data including heart rate, blood oxygen level (SpO_2), respiration rate, blood pressure, body temperature, and alcohol level. These readings were then compared to actual values measured using

clinically validated medical devices. Respiration rate and alcohol level are marked as not available (N/A) in the actual values due to the unavailability of appropriate measurement tools. However, the system's outputs for these parameters can still be assessed based on whether the values fall within typical physiological ranges.

The results indicate that the system's heart rate readings consistently underestimated actual values, with an average error of approximately 11 bpm, amounting to roughly 12% deviation. SpO_2 readings were relatively accurate, with an average error of about 1.1%, indicating good alignment with medical-grade measurements. Blood pressure values from the system showed slight underestimations as well, with systolic and diastolic errors averaging around 10 mmHg and 7 mmHg respectively. Temperature readings tended to be slightly higher than actual body temperatures, with an average error of 0.57°C, or about 1.5%. Alcohol levels were consistently reported as 0% by the system, and while no reference data is available, this is consistent with the known non-consumption of alcohol during the testing period. Similarly, respiration rates recorded by the system remained within normal resting values, supporting their reliability despite the lack of direct comparison.

Overall, the system demonstrates promising performance in providing real-time, non-invasive monitoring through an embedded platform. While SpO_2 and temperature readings were reasonably accurate, heart rate and blood pressure exhibited moderate deviations, likely due to hardware constraints such as the use of DIY dry electrodes and variability in hand placement. These findings suggest that while the system holds potential for continuous or preliminary health monitoring, further improvements in sensor design and signal stability are needed to achieve higher accuracy and clinical reliability.

Table 4.4 Comparison of System Readings with Actual Value

Instance	Type	Heart Rate (bpm)	SpO2 (Blood oxygen Level %)	Respiration Rate (bpm)	Blood Pressure (mmHg)	Body Temperature (˚c)	Alcohol Level (%)
Hour 1	Reading	92	98.28	10	90.7/64.1	38.89	0
	Actual	101	99	N/A	115/70	38	N/A
Hour 2	Reading	76.49	97.44	14	93.8/65.6	39.07	0
	Actual	83	98	N/A	124/79	38	N/A
Hour 3	Reading	87.16	97.84	10	95.6/66.5	36	0
	Actual	103	98	N/A	115/74	37	N/A
Hour 4	Reading	87.39	97.69	10	122/79.6	37.43	0
	Actual	94	98	N/A	108/79	37	N/A
Hour 5	Reading	88.31	100	14	105/71	36.81	0
	Actual	90	98	N/A	123/83	37	N/A

4.5 Vehicle Dynamics

The vehicle dynamics module plays an essential role in monitoring and evaluating a driver's behavior by analyzing motion data acquired from smartphone sensors. This section outlines the methodology for classifying driving styles using inertial data, machine learning algorithms, and real-time signal processing techniques. The goal is to distinguish between four defined driving styles—Smooth, Normal, Aggressive, and Keen—based solely on sensor readings and supervised model training.

4.5.1 Sensor Configuration and Feature Mapping

To capture vehicle motion, a smartphone was utilized for its built-in inertial sensors. These sensors recorded acceleration and angular velocity over three axes using:

Table 4.5 Sensor feature mapping to physical vehicle motion

Sensor	Axis	Driving Interpretation
X_Accelerometer	Lateral	Measures sideways motion during turns
Y_Accelerometer	Longitudinal	Indicates forward/backward motion such as acceleration or braking
Z_Accelerometer	Vertical	Captures road-induced vibrations (e.g., bumps, potholes)
X_Gyroscope	Roll	Rotation around the longitudinal axis (side-to-side tilting)
Y_Gyroscope	Pitch	Rotation around the lateral axis (forward/backward tilt)

Data was collected continuously during driving sessions. Each reading corresponds to a time frame sample and contains five synchronized sensor values. Before training, the dataset underwent cleaning and normalization to reduce sensor noise and ensure consistent feature scaling.

4.5.2 Driving Style Labeling and Data Overview

The recorded samples were labeled into four distinct driving styles based on expert knowledge and observation during data collection. The dataset included the following class distribution:

Table 4.6 Total number of samples per labeled driving behavior.

Driving Style	Number of Labeled Samples
Smooth	138,058
Normal	552,579
Aggressive	125,307
Keen	108,467

These labels served as ground truth for training and evaluation in a supervised machine learning context.

4.5.3 Random Forest Classification Model

A Random Forest classifier was chosen to distinguish driving behaviors from the motion data due to its ability to handle nonlinear decision boundaries, its resistance to overfitting, and its effectiveness in multiclass problems.

Model Parameters and Training:

- Algorithm: Random Forest (n_estimators = 100)

- Features Used: 5 normalized sensor readings
- Split Ratio: 80% training / 20% testing
- Evaluation Metrics: Accuracy, precision, recall, F1-score

4.5.4 Sensor Patterns Across Driving Styles

Distinct sensor signatures were observed for each driving style, helping the model learn meaningful patterns. Table 4.7 shows representative sensor readings sampled from each driving category.

Table 4.7 Example sensor values representing each driving behavior.

Sensor Feature	Smooth	Normal	Aggressive	Keen
X_Gyroscope	+0.01	–0.02	+0.04	–0.00
Y_Gyroscope	–0.05	–0.00	+0.03	–0.10
X_Accelerometer	+1.02	–0.86	+0.29	–2.78
Y_Accelerometer	–3.20	+1.75	–5.71	–3.18
Z_Accelerometer	+9.95	+9.39	+8.88	+9.74

4.5.5 Model Evaluation and Confusion Matrix

The trained model achieved high classification accuracy across most categories. Evaluation results were analyzed using a confusion matrix, which is summarized below:

Table 4.8 Confusion matrix summarizing prediction outcomes.

Actual/Predicted	Smooth	Normal	Aggressive	Keen
Smooth	467	158	192	184
Normal	225	823	353	160
Aggressive	915	1628	4352	1457
Keen	806	591	1073	973

The model excelled at identifying aggressive driving but showed slight misclassifications between the Keen and Aggressive classes, as well as between Smooth and Normal. These overlaps are attributed to subtle motion similarities between cautious sharp maneuvers and assertive driving patterns.

4.5.6 Summary

This phase successfully demonstrates how driving behavior can be classified using smartphone sensor data and supervised machine learning techniques. By training a Random Forest model on pre-labeled motion data, the system effectively recognizes four common driving styles. This classification layer

serves as a key input to the broader Intelligent Driver Protection System, enabling real-time behavior monitoring and risk prediction.

4.6 Application

This section shows the different phases results/readings displayed on the mobile application's interface. All external peripherals cameras, medical sensors, etc. send over the information collected to the application where it is then classified as being in the normal range for the medical sensors. As for driver's monitoring inside the vehicle and the exterior monitoring outside the vehicle, only abnormal states like sleeping, eating and drinking are sent to the application with a picture documenting the driver's behavior or the exterior of the vehicle. In Figure 4.40, it can be seen from the prompt on the screen that the driver's hand has been removed from the steering wheel, hence the unreasonable medical readings that prompted an alert notification to pop up warning about abnormal medical readings, which could point to an issue with the driver. The readings when the hands are removed are deemed invalid and no further action is needed. Meanwhile, in Figure 4.41, it can be seen that driver is sleeping, as soon as this is detected it is sent over to the app with a picture of the driver and a warning to document the action.

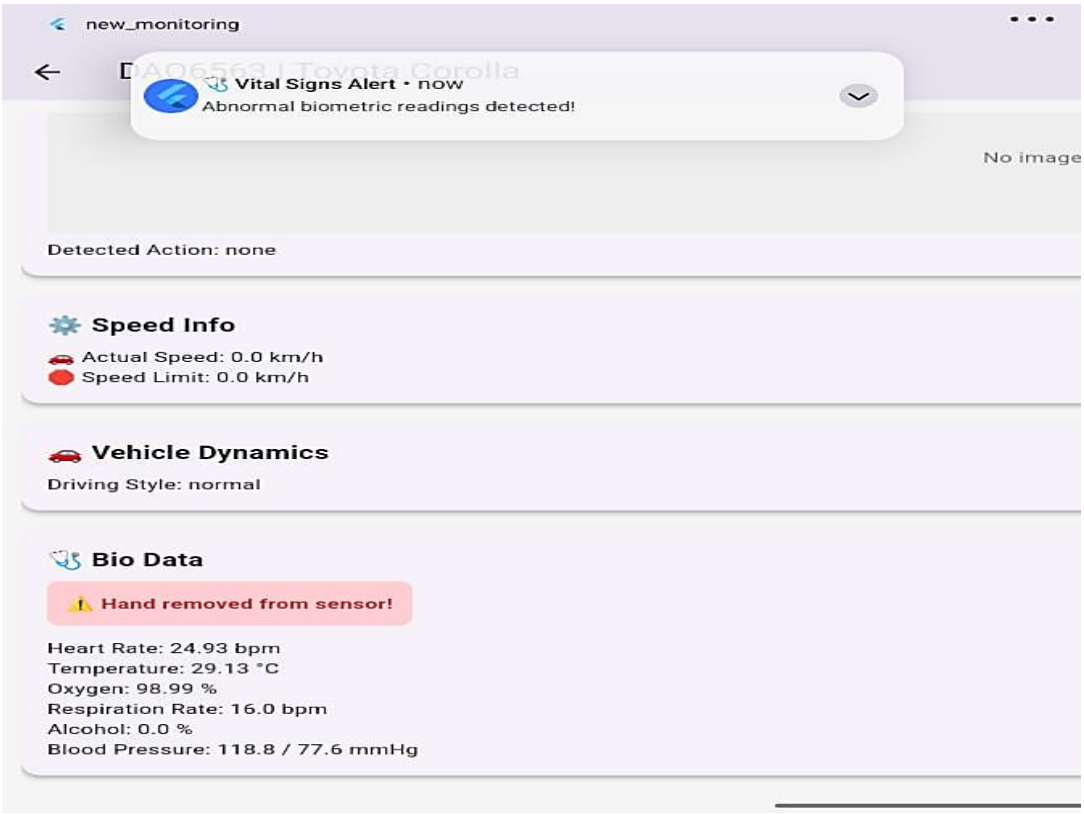


Figure 4.40 Driver Biological Data Displayed on The App.

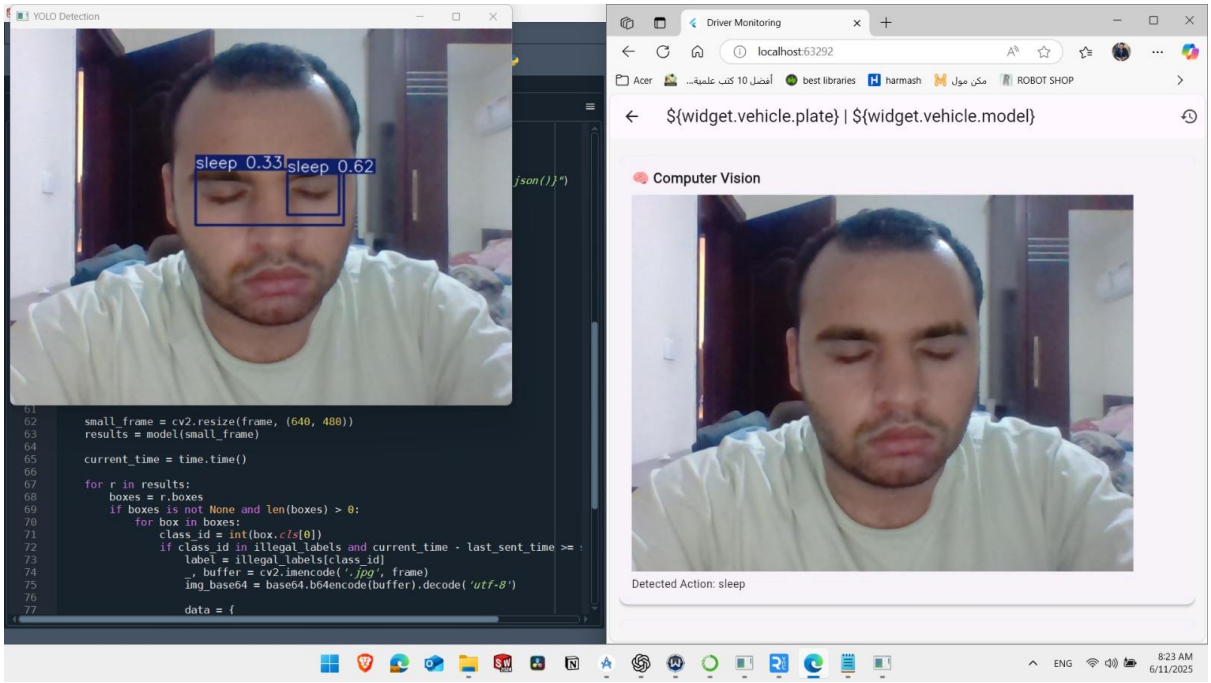


Figure 4.41 Driver Sleeping Alert Sent to The App.

4.7 Conclusion

This chapter has outlined the methodological approach adopted in this study, detailing the processes, tools, and rationale guiding the research design. By establishing a clear framework for data collection, analysis, and validation, it ensures transparency, consistency, and reproducibility. The chosen methods were selected to align with the overall objectives of the study while addressing relevant practical and theoretical considerations. This methodological foundation serves as a critical basis for the subsequent presentation of results and discussion, supporting the integrity and coherence of the research as a whole.

CHAPTER 5
CONCLUSION AND FUTURE WORK

5.1 Overview

This chapter provides a comprehensive summary of the research conducted and the results achieved throughout the project. It highlights the key findings, their implications, and the overall contribution of the work to the relevant field of study. Additionally, this chapter addresses the limitations encountered during the project and suggests potential areas for improvement. Finally, it outlines directions for future research and development, aimed at enhancing the current system and exploring its broader applications. This discussion serves as a roadmap for advancing the work and tackling the challenges identified.

5.2 Conclusion

- The Intelligent Driver Protection System represents a significant advancement in enhancing road safety through the integration of AI and health monitoring technologies.
- The project addressed critical factors contributing to road accidents, such as driver distraction, medical emergencies, and vehicle dynamics, by leveraging advanced algorithms including YOLO for object detection, Hough Transform for lane segmentation, and CNN for traffic sign recognition.
- The system incorporated physiological monitoring through sensors embedded in a smart steering wheel and utilized machine learning models to analyze vehicle dynamics.
- Despite challenges such as system accuracy and stability, the project established a strong foundation for a scalable and adaptable framework capable of significantly reducing human errors and improving road safety globally.

5.3 Future work

- **Interior Vehicle Monitoring:**
  - Incorporate additional behaviors such as fatigue detection and emotional state monitoring.
  - Explore the use of multi-modal sensors (e.g., combining camera data with audio analysis) for improved detection reliability.
  - Improve the accuracy and performance of inner AI model and outer AI model by using different algorithms (YOLOV12).
- **Exterior Vehicle Monitoring:**
  - **Multi-Camera Fusion:** Add rear/side views for better coverage during lane changes and overtaking.
  - **Sensor Fusion:** Combine GPS with IMU and CAN data for more accurate detection in poor visibility.
- **Physiological Monitoring:**
  - Integrate additional sensors for monitoring parameters such as blood glucose levels and stress indicators.
  - Improve signal stability by replacing DIY dry electrodes with commercially available dry electrodes to ensure more reliable contact on the steering wheel.
- **Vehicle Dynamics:**
  - Expand dataset with more diverse driving scenarios and drivers for improved model generalization.
  - Integrate GPS and speed sensors to enhance context awareness and detect over-speeding or route deviations.
  - Implement edge AI models for on-device inference to reduce latency and reliance on cloud/server.
  - Add audio monitoring to detect signs of distraction.
  - Enable driver identification using facial recognition for personalized warnings and tracking.
  - Add emergency response features to notify contacts or authorities in case of critical health or safety events.

By addressing these areas, the Intelligent Driver Protection System has the potential to become a transformative tool in reducing accidents and promoting safer driving practices worldwide.

Bibliography

[1] P. Puvanachandra *et al.*, "Road traffic injuries and data systems in Egypt: addressing the challenges," *Traffic Inj Prev*, vol. 13, no. sup1, pp. 44–56, 2012.

[2] P. Y. Kumbhar, M. Attaullah, S. Dhere, and S. Hipparagi, "Real time face detection and tracking using OpenCV," *International journal for research in emerging science and technology*, vol. 4, no. 4, pp. 39–43, 2017.

[3] V. Paul and J. Michael, "Robust real-time face detection International Journal of Computer Vision," 2004.

[4] M. Hashemi, A. Mirrashid, and A. Beheshti Shirazi, "Driver safety development: Real-time driver drowsiness detection system based on convolutional neural network," *SN Comput Sci*, vol. 1, no. 5, p. 289, 2020.

[5] F. Guede-Fernandez, M. Fernandez-Chimeno, J. Ramos-Castro, and M. A. Garcia-Gonzalez, "Driver drowsiness detection based on respiratory signal analysis," *IEEE access*, vol. 7, pp. 81826–81838, 2019.

[6] H. Guo, H. Lin, S. Zhang, and S. Li, "Image-based seat belt detection," in *Proceedings of 2011 IEEE International Conference on Vehicular Electronics and Safety*, IEEE, 2011, pp. 161–164.

[7] D. S. B. Naik, G. S. Lakshmi, V. R. Sajja, D. Venkatesulu, and J. N. Rao, "Driver's seat belt detection using CNN," *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 5, pp. 776–785, 2021.

[8] M. D. Hssayeni, S. Saxena, R. Ptucha, and A. Savakis, "Distracted driver detection: Deep learning vs handcrafted features," *Electronic Imaging*, vol. 29, pp. 20–26, 2017.

[9] L. Zhao, T. Zhang, and L. Guo, "Classification models of driving distraction: analysis and comparison," *CS229*, 2017.

[10] M. Yaseen, "What is YOLOv8: An In-Depth Exploration of the Internal Features of the Next-Generation Object Detector," Aug. 2024, [Online]. Available: http://arxiv.org/abs/2408.15857

[11] R. Khanam and M. Hussain, "YOLOv11: An Overview of the Key Architectural Enhancements," Oct. 2024, [Online]. Available: http://arxiv.org/abs/2410.17725

[12] P. Y. Hsiao, C. W. Yeh, S. S. Huang, and L. C. Fu, "A portable vision-based real-time lane departure warning system: Day and night," *IEEE Trans Veh Technol*, vol. 58, no. 4, pp. 2089–2094, 2009, doi: 10.1109/TVT.2008.2006618.

[13] L. H. Xu, S. G. Hu, and Q. Luo, "A new lane departure warning algorithm considering the driver's behavior characteristics," *Math Probl Eng*, vol. 2015, 2015, doi: 10.1155/2015/412126.

[14] W. Chen, W. Wang, K. Wang, Z. Li, H. Li, and S. Liu, "Lane departure warning systems and lane line detection methods based on image processing and semantic segmentation: A review," Dec. 01, 2020, *Chang'an University*. doi: 10.1016/j.jtte.2020.10.002.

[15] . IEEE Staff, *2010 IEEE Intelligent Vehicles Symposium*. I E E E, 2010.

[16] A. Magdy *et al.*, "Lane and Bump Detection Based on Computer Vision and Deep Learning Methods," in *Lecture Notes on Data Engineering and Communications Technologies*, Springer Science and Business Media Deutschland GmbH, 2023, pp. 31–42. doi: 10.1007/978-3-031-43247-7_3.

[17] R. Yin, Y. Cheng, H. Wu, Y. Song, B. Yu, and R. Niu, “FusionLane: Multi-Sensor Fusion for Lane Marking Semantic Segmentation Using Deep Neural Networks,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 1543–1553, Feb. 2022, doi: 10.1109/TITS.2020.3030767.

[18] A. de la Escalera, J. Armingol, and M. Mata, “Traffic sign recognition and analysis for intelligent vehicles.” [Online]. Available: www.uc3m.es/

[19] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, “Man vs. Computer: Benchmarking Machine Learning Algorithms for Traffic Sign Recognition.” [Online]. Available: http://www.nisys.de

[20] H. Fleyeh and E. Davami, “Eigen-based traffic sign recognition,” *IET Intelligent Transport Systems*, vol. 5, no. 3, pp. 190–196, Sep. 2011, doi: 10.1049/iet-its.2010.0159.

[21] G. Latif, D. A. Alghmgham, R. Maheswar, J. Alghazo, F. Sibai, and M. H. Aly, “Deep learning in Transportation: Optimized driven deep residual networks for Arabic traffic sign recognition,” *Alexandria Engineering Journal*, vol. 80, pp. 134–143, Oct. 2023, doi: 10.1016/j.aej.2023.08.047.

[22] X. Wang, M. Chen, M. Zhu, and P. Tremont, “Development of a Kinematic-Based Forward Collision Warning Algorithm Using an Advanced Driving Simulator,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 9, pp. 2583–2591, Sep. 2016, doi: 10.1109/TITS.2016.2528508.

[23] Q. Lim, Y. Lim, H. Muhammad, D. W. M. Tan, and U. X. Tan, “Forward collision warning system for motorcyclist using smart phone sensors based on time-to-collision and trajectory prediction,” *Journal of Intelligent and Connected Vehicles*, vol. 4, no. 3, pp. 93–103, Dec. 2021, doi: 10.1108/JICV-11-2020-0014.

[24] N. Albarella, F. Masuccio, L. Novella, M. Tufo, and G. Fiengo, “A forward-collision warning system for electric vehicles: Experimental validation in virtual and real environment,” *Energies (Basel)*, vol. 14, no. 16, Aug. 2021, doi: 10.3390/en14164872.

[25] *INISTA 2014 : IEEE International Symposium on Innovations in Intelligent Systems and Applications : proceedings : June 23-25, 2014, Alberobello, Italy*. IEEE, 2014.

[26] B. Babusiak, A. Hajducik, S. Medvecky, M. Lukac, and J. Klarak, “Design of smart steering wheel for unobtrusive health and drowsiness monitoring,” *Sensors*, vol. 21, no. 16, Aug. 2021, doi: 10.3390/s21165285.

[27] I. C. Lazăr, T. I. Ursache, M. Aron, G.-G. Petroiu, and C. Rotariu, “Automatic Monitoring of Driver’s Physiological Parameters by Using a Smart Steering Wheel,” 2024, pp. 334–342. doi: 10.1007/978-3-031-62502-2_39.

[28] S. J. Jung, H. S. Shin, and W. Y. Chung, “Driver fatigue and drowsiness monitoring system with embedded electrocardiogram sensor on steering wheel,” *IET Intelligent Transport Systems*, vol. 8, no. 1, pp. 43–50, 2014, doi: 10.1049/iet-its.2012.0032.

[29] J.-C. Lee and H. Liu, “Development of a Real-Time Driver Health Detection System Using a Smart Steering Wheel,” 2018.

[30] A. Hina and W. Saadeh, “Noninvasive Blood Glucose Monitoring Systems Using Near-Infrared Technology—A Review,” Jul. 01, 2022, *MDPI*. doi: 10.3390/s22134855.

[31] Q. Zhang, Y. Zhou, S. Song, G. Liang, and H. Ni, “Heart Rate Extraction Based on Near-Infrared Camera: Towards Driver State Monitoring,” *IEEE Access*, vol. 6, pp. 33076–33087, Jun. 2018, doi: 10.1109/ACCESS.2018.2845390.

[32] M. M. Rahman, J. Cook, and A. Taebi, “Non-contact heart vibration measurement using computer vision-based seismocardiography,” *Sci Rep*, vol. 13, no. 1, Dec. 2023, doi: 10.1038/s41598-023-38607-7.

[33] H. Rahman, S. Barua, and B. Shahina, “Intelligent Driver Monitoring Based on Physiological Sensor Signals: Application Using Camera,” in *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, Institute of Electrical and Electronics Engineers Inc., Oct. 2015, pp. 2637–2642. doi: 10.1109/ITSC.2015.424.

[34] X. Hu and G. Lodewijks, “Detecting fatigue in car drivers and aircraft pilots by using non-invasive measures: The value of differentiation of sleepiness and mental fatigue,” *J Safety Res*, vol. 72, pp. 173–187, Feb. 2020, doi: 10.1016/j.jsr.2019.12.015.

[35] *2020 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, 2020.

[36] H. Shin and S. D. Min, “Feasibility study for the non-invasive blood pressure estimation based on ppg morphology: Normotensive subject study,” *Biomed Eng Online*, vol. 16, no. 1, Jan. 2017, doi: 10.1186/s12938-016-0302-y.

[37] R. Chhabra, S. Verma, and C. Rama Krishna, “Detecting Aggressive Driving Behavior using Mobile Smartphone,” in *Lecture Notes in Networks and Systems*, vol. 46, Springer, 2019, pp. 513–521. doi: 10.1007/978-981-13-1217-5_49.

[38] T. Bar, D. Nienhüser, R. Kohlhaas, and J. M. Zöllner, “Probabilistic driving style determination by means of a situation based analysis of the vehicle data,” in *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2011, pp. 1698–1703. doi: 10.1109/ITSC.2011.6082924.

[39] J. F. Júnior *et al.*, “Driver behavior profiling: An investigation with different smartphone sensors and machine learning,” *PLoS One*, vol. 12, no. 4, Apr. 2017, doi: 10.1371/journal.pone.0174959.

[40] M. Abbadi, A. Abadleh, S. Alja’afreh, and Z. Halhouli, “Machine Learning-Based Approach For Detecting Driver Behavior Using Smartphone Sensors,” *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH*, vol. 8, 2019, [Online]. Available: www.ijstr.org

[41] J. Carmona, F. García, D. Martín, A. de la Escalera, and J. M. Armingol, “Data fusion for driver behaviour analysis,” *Sensors (Switzerland)*, vol. 15, no. 10, pp. 25968–25991, Oct. 2015, doi: 10.3390/s151025968.

[42] I. T. Sarteshnizi, F. Tavakkoli Khomeini, B. Khedri, and A. Samimi, “Sensitivity analysis of driving event classification using smartphone motion data: case of classifier type, sensor bundling, and data acquisition rate,” *Journal of Intelligent Transportation Systems: Technology, Planning, and Operations*, vol. 28, no. 4, pp. 476–493, 2024, doi: 10.1080/15472450.2022.2140048.

[43] M. Zarei Yazd, I. Taheri Sarteshnizi, A. Samimi, and M. Sarvi, “A robust machine learning structure for driving events recognition using smartphone motion sensors,” *Journal of Intelligent Transportation Systems: Technology, Planning, and Operations*, vol. 28, no. 1, pp. 54–68, 2024, doi: 10.1080/15472450.2022.2101109.

[44] J. Xie, A. R. Hilal, and D. Kulic, “Driver Distraction Recognition Based on Smartphone Sensor Data,” in *Proceedings - 2018 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2018*, Institute of Electrical and Electronics Engineers Inc., Jul. 2018, pp. 801–806. doi: 10.1109/SMC.2018.00144.

[45] M. Yuzkat, H. O. Ilhan, and N. Aydin, “Detection of sperm cells by single-stage and two-stage deep object detectors,” *Biomed Signal Process Control*, vol. 83, p. 104630, 2023.

[46] T. Diwan, G. Anirudh, and J. V Tembhurne, “Object detection using YOLO: Challenges, architectural successors, datasets and applications,” *Multimed Tools Appl*, vol. 82, no. 6, pp. 9243–9275, 2023.

[47] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González, “A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas,” *Mach Learn Knowl Extr*, vol. 5, no. 4, pp. 1680–1716, 2023.

[48] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, “A Review of Yolo algorithm developments,” *Procedia Comput Sci*, vol. 199, pp. 1066–1073, 2022.

[49] A. Correira, F. Ferreira, and N. Mišković, “Comparing different yolo versions for boat detection and classification in real datasets,” in *OCEANS 2024-Singapore*, IEEE, 2024, pp. 1–4.

[50] A. F. Rasheed and M. Zarkoosh, “Optimized YOLOv8 for multi-scale object detection,” *J Real Time Image Process*, vol. 22, no. 1, p. 6, 2025.

[51] Y. Kortli, M. Marzougui, and M. Atri, “Efficient implementation of a real-Time lane departure warning system,” in *IPAS 2016 - 2nd International Image Processing, Applications and Systems Conference*, Institute of Electrical and Electronics Engineers Inc., Mar. 2017. doi: 10.1109/IPAS.2016.7880072.

[52] J. Huang *et al.*, “Speed/accuracy trade-offs for modern convolutional object detectors,” Nov. 2016, [Online]. Available: http://arxiv.org/abs/1611.10012

[53] R. Yin, Y. Cheng, H. Wu, Y. Song, B. Yu, and R. Niu, “Fusionlane: Multi-sensor fusion for lane marking semantic segmentation using deep neural networks,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 1543–1553, 2020.

[54] R. A. Rill and K. B. Faragó, “Collision Avoidance Using Deep Learning-Based Monocular Vision,” *SN Comput Sci*, vol. 2, no. 5, Sep. 2021, doi: 10.1007/s42979-021-00759-6.

[55] S. Jeong, H. Shin, M. J. Kim, D. Kang, S. Lee, and S. Oh, “Enhancing LiDAR Mapping with YOLO-Based Potential Dynamic Object Removal in Autonomous Driving,” *Sensors*, vol. 24, no. 23, Dec. 2024, doi: 10.3390/s24237578.

[56] H. Y. Lin, J. M. Dai, L. T. Wu, and L. Q. Chen, “A vision-based driver assistance system with forward collision and overtaking detection,” *Sensors (Switzerland)*, vol. 20, no. 18, pp. 1–19, Sep. 2020, doi: 10.3390/s20185139.

[57] N. Albarella, F. Masuccio, L. Novella, M. Tufo, and G. Fiengo, “A forward-collision warning system for electric vehicles: Experimental validation in virtual and real environment,” *Energies (Basel)*, vol. 14, no. 16, p. 4872, 2021.

[58] G. Latif, D. A. Alghmgham, R. Maheswar, J. Alghazo, F. Sibai, and M. H. Aly, “Deep learning in Transportation: Optimized driven deep residual networks for Arabic traffic sign recognition,” *Alexandria Engineering Journal*, vol. 80, pp. 134–143, 2023.

[59] H. Rahman, S. Barua, and B. Shahina, “Intelligent Driver Monitoring Based on Physiological Sensor Signals: Application Using Camera,” in *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, Institute of Electrical and Electronics Engineers Inc., Oct. 2015, pp. 2637–2642. doi: 10.1109/ITSC.2015.424.

[60] M. M. Rahman, J. Cook, and A. Taebi, “Non-contact heart vibration measurement using computer vision-based seismocardiography,” *Sci Rep*, vol. 13, no. 1, Dec. 2023, doi: 10.1038/s41598-023-38607-7.

[61] M. Z. Poh, D. J. McDuff, and R. W. Picard, “Advancements in noncontact, multiparameter physiological measurements using a webcam,” *IEEE Trans Biomed Eng*, vol. 58, no. 1, pp. 7–11, Jan. 2011, doi: 10.1109/TBME.2010.2086456.

[62] Q. Zhang, Y. Zhou, S. Song, G. Liang, and H. Ni, “Heart Rate Extraction Based on Near-Infrared Camera: Towards Driver State Monitoring,” *IEEE Access*, vol. 6, pp. 33076–33087, Jun. 2018, doi: 10.1109/ACCESS.2018.2845390.

[63] L. Tang, S. J. Chang, C. J. Chen, and J. T. Liu, “Non-invasive blood glucose monitoring technology: A review,” Dec. 01, 2020, *MDPI AG*. doi: 10.3390/s20236925.

[64] J.-C. Lee and H. Liu, “Development of a Real-Time Driver Health Detection System Using a Smart Steering Wheel,” 2018.

[65] R. Wang, W. Jia, Z. H. Mao, R. J. Sclabassi, and M. Sun, “Cuff-free blood pressure estimation using pulse transit time and heart rate,” in *International Conference on Signal Processing Proceedings, ICSP*, Institute of Electrical and Electronics Engineers Inc., 2014, pp. 115–118. doi: 10.1109/ICOSP.2014.7014980.

[66] T. Iqbal, A. Elahi, S. Ganly, W. Wijns, and A. Shahzad, “Photoplethysmography-Based Respiratory Rate Estimation Algorithm for Health Monitoring Applications,” *J Med Biol Eng*, vol. 42, no. 2, pp. 242–252, Apr. 2022, doi: 10.1007/s40846-022-00700-z.

[67] W.-F. Liu, G.-L. Liu, X.-F. Wang, Y.-F. Bao, G. Li, and H.-Q. Wang, “Non-Invasive Measurement Study of Human Blood Alcohol Concentration Based on NIR Dynamic Spectrum.”

[68] Y. Y. Chen, C. L. Lin, Y. C. Lin, and C. Zhao, “Non-invasive detection of alcohol concentration based on photoplethysmogram signals,” *IET Image Process*, vol. 12, no. 2, pp. 188–193, Feb. 2018, doi: 10.1049/iet-ipr.2017.0625.

[69] L. Rachakonda, S. Mohanty, and E. Kougianos, “Donot-DUEye: An IoT enabled edge device to monitor blood alcohol concentration from eyes,” in *Proceedings - 2019 IEEE International Symposium on Smart Electronic Systems, iSES 2019*, Institute of Electrical and Electronics Engineers Inc., Dec. 2019, pp. 87–92. doi: 10.1109/iSES47678.2019.00030.

[70] *2020 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, 2020.

[71] J. Shim, J. Yeo, S. Lee, S. H. Hamdar, and K. Jang, “Empirical evaluation of influential factors on bifurcation in macroscopic fundamental diagrams,” *Transp Res Part C Emerg Technol*, vol. 102, pp. 509–520, May 2019, doi: 10.1016/j.trc.2019.03.005.

[72] Pawan Wawage and Yogesh Deshpande, *Smartphone Sensor Dataset for Driver Behavior*. 2022. doi: 10.17632/9vr83n7z5j.1.

[73] J. Xie, A. R. Hilal, and D. Kulic, “Driver Distraction Recognition Based on Smartphone Sensor Data,” in *Proceedings - 2018 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2018*, Institute of Electrical and Electronics Engineers Inc., Jul. 2018, pp. 801–806. doi: 10.1109/SMC.2018.00144.

[74] S. M. Alzahrani, “Sensing for the Internet of Things and its applications,” in *2017 5th international conference on future internet of things and cloud workshops (FiCloudW)*, IEEE, 2017, pp. 88–92.

[75] M. Maksimović, V. Vujović, N. Davidović, V. Milošević, and B. Perišić, “Raspberry Pi as Internet of things hardware: performances and constraints,” *design issues*, vol. 3, no. 8, pp. 1–6, 2014.

[76] D. R. Patnaik Patnaikuni, “A Comparative Study of Arduino, Raspberry Pi and ESP8266 as IoT Development Board.,” *International Journal of Advanced Research in Computer Science*, vol. 8, no. 5, 2017.

[77] S. F. A. Razak, S. Yogarayan, A. A. Aziz, M. F. A. Abdullah, and N. H. Kamis, “Physiological-based Driver Monitoring Systems: A Scoping Review,” Dec. 01, 2022, *Salehan Institute of Higher Education*. doi: 10.28991/CEJ-2022-08-12-020.

Appendix A

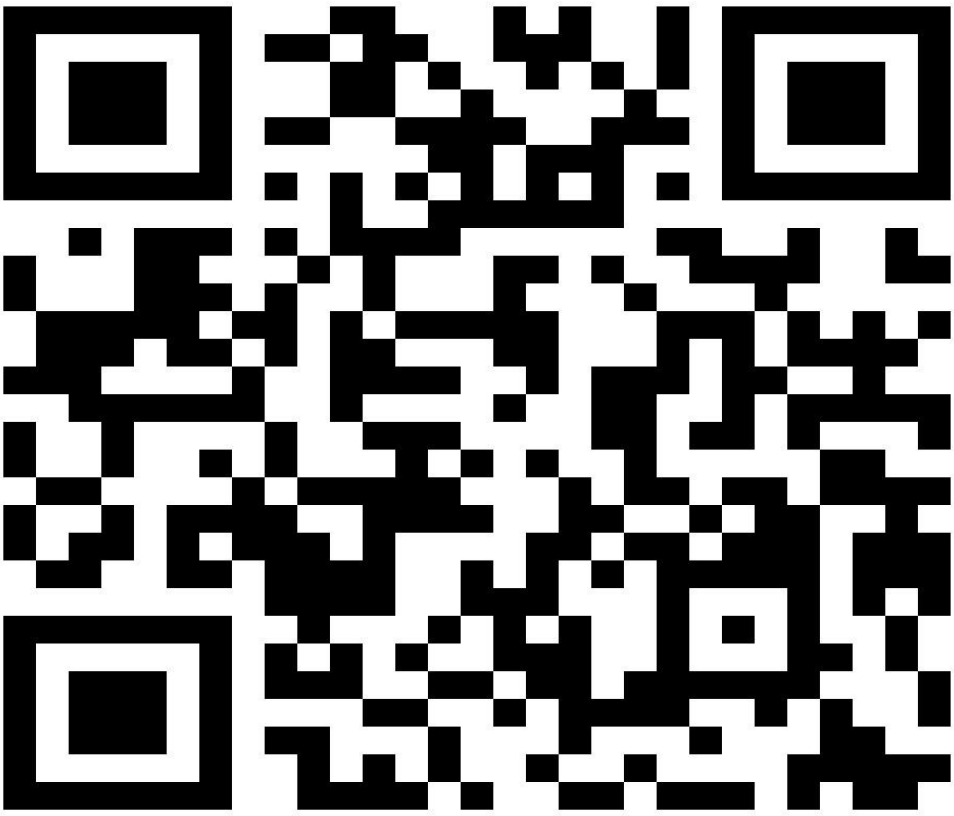
Table 9.9 Cost

Item	**Cost**
Raspberry pi 5, 8GB	6500 L.E.
Max30102 pulse and oximeter module	150 L.E.
AD8232 ECG sensor module kit	566 L.E.
MLX90614	615 L.E.
Okdo raspberry pi Power supply 5Vdc, 3A	375 L.E.
SD card 32GB-HC10	375 L.E.
Official raspberry pi HDMI to Micro-HDMI cable	300 L.E.
PCB Manufacturing	200 L.E.
Gaming Steering Wheel	1200 L.E.
EMEET 1080P Webcam	1799 L.E.
Logitech C922 Pro Stream Webcam1080P	1800 L.E.
ADS1115	250 L.E.
AMS1117	20 L.E.
Total	14,150 L.E.

Appendix B: Project Source Code Repository

Project Source Code Access

The complete source code and related files for this graduation project are hosted online and can be accessed using the following link or QR code:



https://github.com/ahmedehab24/Smart-behavior-monitoring.git

This repository includes all code for the three main phases of the system:

Inside Phase: Driver behavior detection (eating, sleeping, mobile usage, drinking, seatbelt, smoking)

Outside Phase: Lane and traffic sign detection

Biomedical Phase: Health condition detection while driving

Vehicle dynamics: normal driving OR abnormal

Application: integrated all phases