

Real-Time Embedded System for Inattentive Driver Monitoring.

Saranyan, Angelo Clerence & Rizma Nalmi

Sri Lanka Institute of Information Technology
Malabe, Sri Lanka.

{saranyan.msundram,angelojoel2306,rizmanalmi}@gmail.com

Prabhath Buddhika

Sri Lanka Institute of Information Technology
Malabe, Sri Lanka.
prabhath.b@sliit.lk

ABSTRACT

One of the causes of motor vehicle accidents in Sri Lanka is driver inattention or drowsiness. In the field of intelligent transportation systems, continuous research and development are conducted to address this contemporary issue. Many approaches, such as driver assistance and drowsiness detection systems, have been proposed to overcome this fatality. The purpose of this research was to implement a product that can maximise road safety while improving the transport sector's efficiency and reliability of the logistics chain to reinforce the country's economic growth. In this paper, the correlation between the preprocessed vehicular parameters and visual features are used to analyse the driver state and make predictions of the driver's performance.

The proposed system uses computer vision and fuzzy logic inference implemented on the single-board computer Raspberry Pi to detect facial features and to determine the driver's drowsiness state, an ELM327 is used to read the vehicle parameters from the Electronic Control Unit (ECU) and motion sensors were used to obtain the steering angle. The data acquired is stored in a cloud platform using REST API. The database also contains driver details. The system uses a fingerprint scanner to identify the driver. An actuator was installed in the vehicle to alert the driver when the system detects inattentiveness. Overall the proposed project provided satisfying experimental results. It can be used as a solution to improve road safety and a supporting tool for the logistics sector to monitor vehicles and driver performance.

KEYWORDS: *Driver monitoring system, computer vision, fuzzy logic, vehicle telematics, steering angle, Rest API, cloud computing, RTOS.*

1 INTRODUCTION

The vehicle industry has seen a significant boost in the manufacturing of "intelligent" vehicles with active and passive safety systems. Grandjean (Grandjean, 1979) defines drowsiness as a "gradual and cumulative process associated with a loss of efficacy and disinclination for any kind of effort". According to Hossain et al (J. Hossain, 2003), drowsiness is "a state in which one's capacity or efficacy for work is reduced following physical and mental effort; it does not necessarily imply the irresistible desire for or tendency to fall asleep.

The rapid growth in the increase of motor vehicle use has raised many concerns across the country and the globe. According to the article released by the World Health Organization (WHO), in June 2021, approximately 1.3 million deaths are caused due to road accidents each year. The report also highlights that distracted driving is one of the leading causes among many reasons for road accidents. (World Health Organization, 2021)

The scope of this project is to create a non-intrusive device to analyse driver driving patterns across a fleet of vehicles and display precomputed information to the customer on a dashboard accessed through the world wide web. This research is divided into sub-parts, each of which contributes to monitoring driver inattentiveness and driver performance. Driver fatigue and driver inattentiveness are detected using computer vision technology, driving style and performance is monitored by obtaining

data from the OBD port of the vehicle and the active steering engagement. This data is entered into a relational database via an API, and fingerprint data is used for the identification of drivers to create a driver driving profile.

This research can be considered unique as it proposes a hybrid approach in driver behaviour monitoring systems by using both the facial features and vehicular information as opposed to using only one type of information to make the system more robust and suppress false positives, thereby providing the user with the frictionless experience.

2 RELATED WORK

According to the literature review of related works, multiple methods have been used in detecting driver behaviour. The forms can be considered invasive and noninvasive. A noninvasive approach uses behavioural measures to evaluate the state of the driver, whereas, in an invasive, a wearable device is used. The noninvasive method is more feasible to implement and cost-effective. All the below-cited work has taken different approaches towards detecting driver attentiveness. It is evident that in most of the research conducted based on non-invasive methods uses facial features or vehicular parameters to determine driver attentiveness.

In the researches (Warwick, 2015) (Li, 2015) (Jung, 2014), a wearable device is used to obtain physiological parameters like EEG, ECG and EoG signals. Jung et al (Jung, 2014) modified the steering wheel and attached a sensor to obtain the ECG signal of the driver to evaluate the drowsiness. This approach has many drawbacks. The cost and the approach demand the driver to wear devices while driving, and some techniques require modifications to the vehicle. Due to these reasons, this approach is not universally applicable.

In the noninvasive approach by Zhang and Hua (Zhang, 2015) local binary pattern features, and support vector machines are used to evaluate the state of the driver using facial information. Picot et al (Picot, 2012) proposed an approach that uses the blinking feature for drowsiness detection. In the research by Hariri et al (Abtahi, 2011), they presented a method for detection of driver drowsiness based on yawning behaviours. The modified Viola-Jones object detection algorithm is used for face and mouth detection. This method is based on several algorithms, which are insensitive to the changes in the lighting conditions, skin types and geometrical facial features.

In the research by Eyosiyas et al (Tadesse, 2014) a Hidden Markov Model based dynamic modelling of the facial emotions and the behavioral changes in addition to the detection of eye closures for drowsiness detection of the driver. The research concludes that instead of using frame-based approach to detect the drowsiness of the driver, temporal analysis of facial expressions maximized the accuracy and speed of detection.

Shaout et al (Shaout, 2018) presented a standalone embedded system to listen to CAN frames from the CAN bus and to predict possible future failure that may occur in the vehicle. Samson et al (Samson, 2019) presented a system that continuously monitors driver behaviour parameters such as ECG, speed, eye blinking, steering angle, turn signal usage to observe driver distractions. Castignani et al (Castignani, 2013) presented a fuzzy inference system that runs on a smartphone that uses the sensors of the phone to produce a score for driving behaviour.

Another method used in detecting driver attentiveness was the detection of the steering angle to analyse the driver engagement and thereby to monitor attentiveness. In the research (Zacharia, 2017), a system has been developed to detect driver engagement. The system was efficient but was complex in implementation. Here, the principle of giant magnetic resistance was used along with the MPC5567 microcontroller and CAN module.

3 METHODOLOGY

The overall overview diagram of the development conducted via this research is shown in Figure 1 below. The aim of the project is to monitor driving style and observe driver attentiveness. The facial features were detected using computer vision, the vehicular parameters were obtained from the Electronic Control Unit (ECU) via the OBDII port, the steering angle of the vehicle was determined by building a hardware system using a Micro Electro-Mechanical System (MEMS), an alarm was used to alert the driver in case of inattentiveness, a Rest API was implemented to store and retrieve data sent to

the database from the ECU, the driver details obtained from the fingerprint authentication and the facial details and vehicular parameters were stored in a centralised database incorporated to a cloud server.

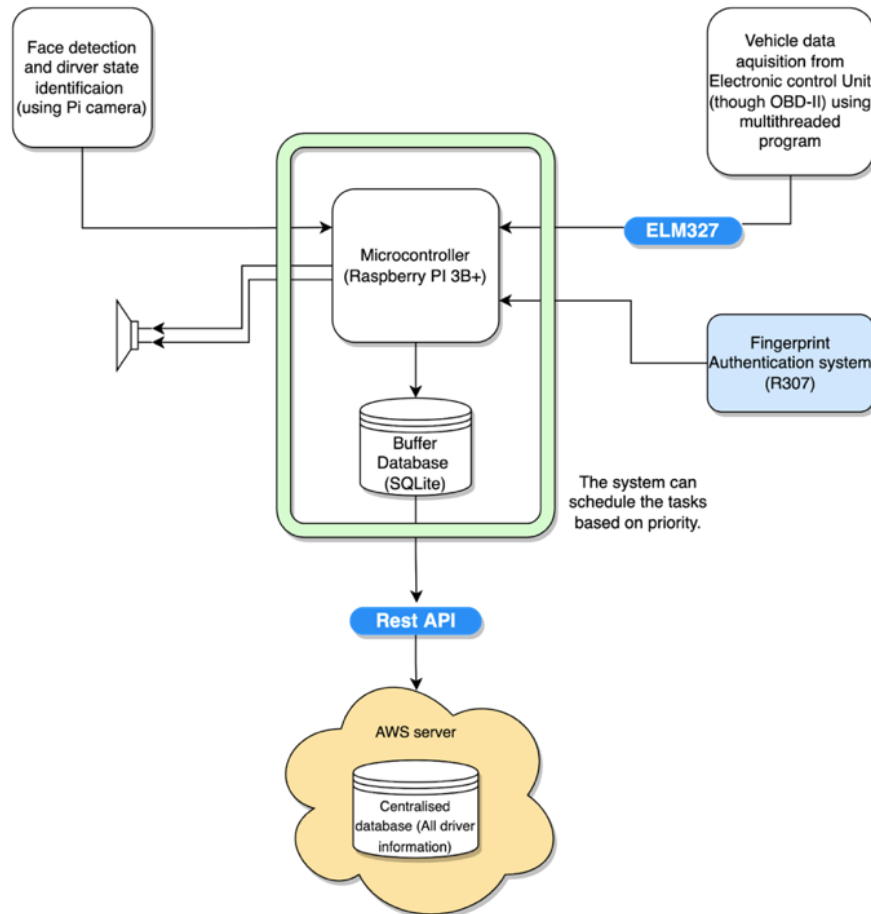


Figure 1. Overview Diagram

3.1 Attentiveness detection using facial features.

The proposed Driver Inattentiveness Monitoring System should be able to prevent the threats posed by the inattentiveness of drivers. The proposed solution should address the inattentiveness of the driver in real-time, prevent accidents, and it should be noninvasive.

In this proposed system, computer vision is used in capturing the facial features, which are accurate indicators of the driver's state and fuzzy logic is used in determining the state of the driver. This approach proves that this monitoring system is able to predict the state of the driver accurately compared to other existing solutions, and it can be implemented with minimal resources at an affordable cost. The driver state detection system consists of 3 stages. First is the facial features extraction stage: a CNN is used to detect the driver's face and compute the facial landmark points. The next stage is the computation of the drowsiness level indicators using the face landmark point information. The parameters used in determining the state of the driver are Head pose (HP), Blink frequency (BF), PERCLOS, Eye Closure Duration (ECD) and Mouth Opening Duration (MOD). The final stage is the driver state prediction; the calculated parameters are mapped to the driver's state using a fuzzy inference system.

Facial features extraction stage

In the proposed driver state detection system, CNN has been used to predict the face landmark points and the bounding box coordinates of the driver's face.

The CNN architecture consists of 28 convolutional layers, one Global average pooling layer and an output layer consisting of 3 different output units: one for predicting the confidence score, another 4 perceptrons for predicting the bounding box coordinates of the face and 68 perceptrons for predicting

the (x, y) coordinates of the face landmark points. The depthwise separable convolution for feature extractions using kernels was used. A bounded ReLU activation function is used for applying non-linearity to the output of the feature maps. Binary cross-entropy function and mean squared error functions are used to compute the error values of the CNN predictions. A stochastic gradient descent algorithm is used to train the CNN.

Computation of the drowsiness level indicators stage

The parameters used to detect drowsiness is computed using the facial feature information obtained from the CNN. Parameters used for the evaluation are:

1. Head pose (HP) - The head pose of the person is a prevalent indicator of drowsiness in behavioural-measures-based systems. If the head pose of the person is identified as down or at different sides rather than looking forward to a reasonable amount of time for a given temporal window, then the system will identify the person as drowsy and not focusing on the driving.

2. PERCLOS - is the proportion of closed eye frames to the total number of frames in a one-minute time window.

3. Average Eye Closure Duration (AECD) - This metric is the mean duration of intervals in which eyes are closed in the defined temporal window. According to (Kaplan, 2015) (Satzoda, 2014), when the driver is awake, the proportion of closed-eye frames is less than 30%. The eye closure time for the driver is less when they are awake than the eye closure time when the driver is in a drowsy state. So if the eye closure time exceeds a certain threshold value, the system can determine the driver as drowsy.

4. Frequency of Blinking - is the number of times the driver blinks in a minute. When people are awake, they blink 8 to 15 times a minute. But when the driver is in a low vigilant state and in a mild drowsiness state, the frequency of blinking will increase. But when they are extremely tired or drowsy, the frequency of blinking will drastically decrease.

5. Average Mouth Opening Duration (AMOD) - is defined as the mean duration of time intervals for which the driver kept their mouth open during a defined temporal window. According to (Barbizet, 1958), it is known that the whole yawning process lasts for 7 seconds.

These metrics are complementary to each other. Therefore, if one or more indicators gives false-positive results, the other indicators will act as the proper validators of the driver's state. From the trials performed using these features, it was found that the last metric, which is the AMOD, is the metric that gives the most false-positive results. Because the mouth of the person is kept open while the person is talking. So, it can be mistakenly taken for yawning, which is an obvious indicator of drowsiness. So by combining multiple feature values, this problem can be solved.

Driver state prediction stage

The mapping of the drowsiness indicators to the state of the driver is performed by the fuzzy inference system (Mamdani, 1975). This provides a non-linear, interpretable glass box like algorithm for this task. It maps the HP, PERCLOS, AECD, blink frequency and AMOD to the state of the driver output, which are awake, low vigilant and severe drowsy state.

In this proposed driver state detection system, the driver state identification process is done by the Mamdani FIS, which uses triangular and trapezoidal membership functions while the output consists of singleton membership functions. The reason for the choice of singleton membership functions for the output is because this FIS has been modelled like a multinomial classification system. So the singleton MFs are the suitable choice because each output MFs consists of only one element with membership degree equal to one and zero for others.

The details of the fuzzy operators used in this FIS are as given below; Figure 2 shows the membership function of the system:

- OR operator(T-conorm) is used as the connecting operator for the rules in the rule-base
- Mamdani Implication operator is minimum
- Aggregation operator is maximum
- Largest of the maximum is used as the defuzzification method.

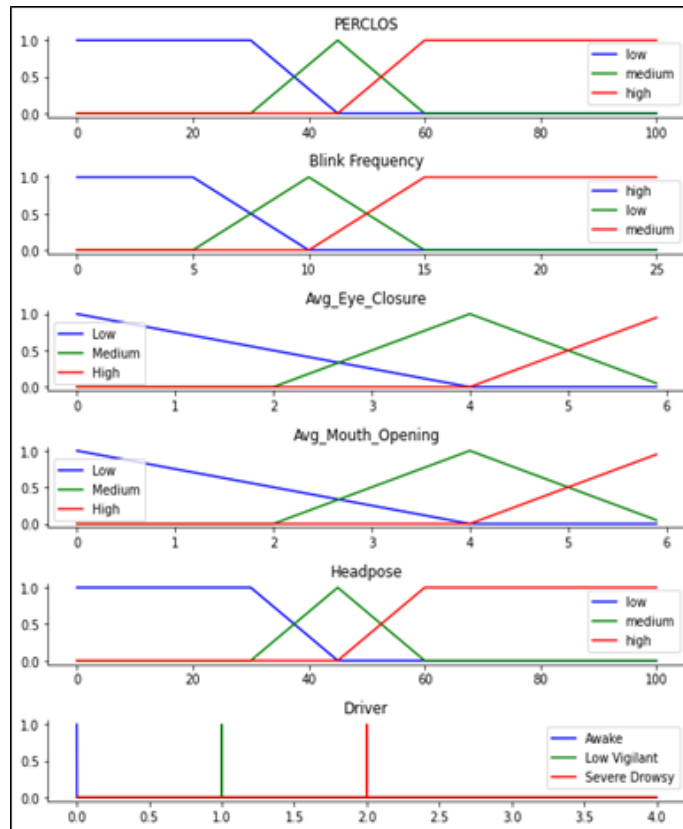


Figure 2 Input and Output membership functions of the Mamdani FIS

Dataset

The CNN was trained using a wild dataset (Huang, 2008) to detect the face landmark points. The dataset consists of 13,000 images collected from different sources. To obtain data for drowsy state indicators, the dataset "Real-life Drowsiness Dataset" (Ghoddosian, 2019) was used. The dataset was used in detecting multi-stage drowsiness. The main goal of this dataset is to detect the early symptoms of drowsiness in addition to detecting visible cues of drowsiness. This dataset consists of videos of 60 participants under the awake, low vigilant and severe drowsy state.

3.2 Vehicle Data

ECU Data Acquisition

The vehicle data acquisition was made using an ELM327 microcontroller through the DLC. The program for data extraction was created using Python3 on a Linux environment. Vehicle data would be extracted from the ECU in real-time and uploaded to a local "buffer" database before being uploaded to the cloud at the end of the trip. The system is equipped with an actuator (alarm) to sound when a driving anomaly is detected. The fingerprint scanner is used to identify the driver of the vehicle in a fleet of vehicles, so the driving details would be saved under the particular driver's profile.

The flow diagram for the interconnection of all the systems is shown in Figure 3 below. Extracting vehicle parameters from the running vehicle is done using two threads in Python3. The operation of the threads is shown in the diagram below.

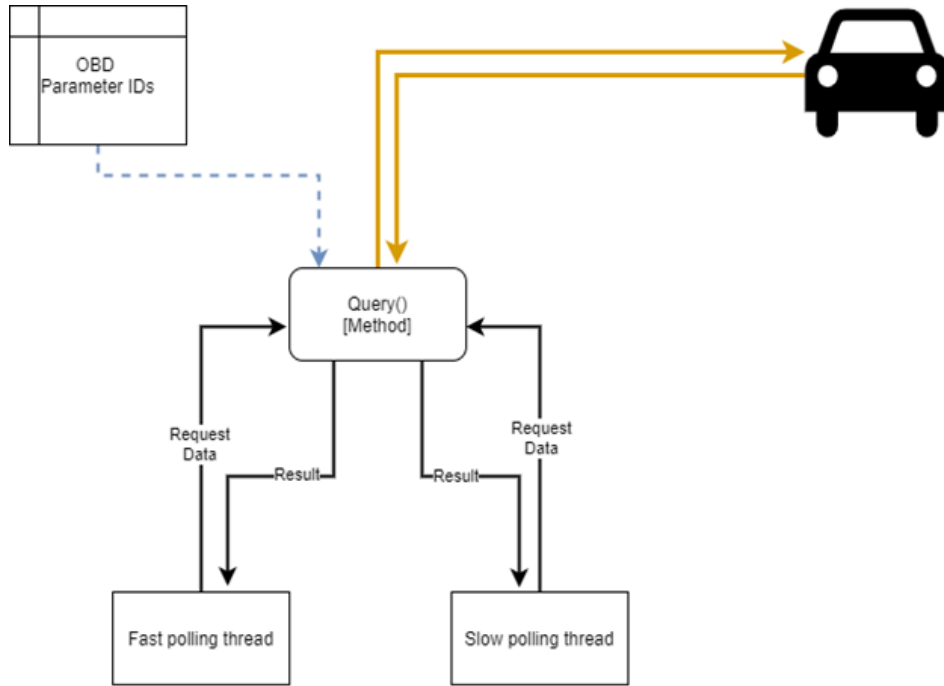


Figure 3. Flow diagram for data acquisition

The python program uses serial communication to communicate with the ECU of the vehicle. After serial communication is set up in the operating system, the python script uses the *Query()* method to send and receive data serially.

The vehicle uses multiple sensors which collect many different types of data and constantly feed this data to the vehicle ECU, which uses this information to control critical actuators in the vehicle. This data can be obtained by using Parameter IDs (PIDs). When a certain vehicle parameter is required, the respective parameter ID must be passed into the ECU through the Data Link Connector (DLC). The response will be sent containing the required parameter.

Figure 4 below displays the structure of the Python program. The block diagram is described below:

1. Thread 1 (Main thread) - This thread is responsible for initiating the rest of the threads and is also responsible for exception handling and terminating the program if any keyboard interrupt is detected.

2. Thread 2 (Fast thread) - This thread is responsible for polling the frequently varying parameters. This thread polls 'Engine load', 'RPM', 'Speed', and 'Throttle position' every second, and the contents will be stored in lists created for each parameter. Thirty data samples will be collected through a while loop and stored in individual buffers. The data was collected through the OBD-II port of the vehicle using the *Query()* method, which is shared between the fast and the slow threads.

Shared memory is used to temporarily save the contents of the fast and slow threads while preventing race conditions (SQLite, n.d.) by deploying access control methods. The *notify_all* command is used to prevent deadlocks.

3. Thread 3 (Slow thread) - This thread is similar to the fast-polling thread, but fewer vehicle parameters are acquired in the thirty-second window. This thread will poll 'Engine coolant temperature' and 'Intake air temperature' every 15 seconds taking a total of 30 seconds, similar to the fast-polling thread.

4. Thread 4 (Upload thread) - This thread is set as a daemon thread as it must run in the background, collecting data sent by the fast and the slow threads and uploading it to the database at set time intervals. The upload thread will upload the contents of the *buffer* into the database after the upload flags of fast and slow polling threads have been set. The upload thread uses the *sqlite3* database API to upload the contents to the database.

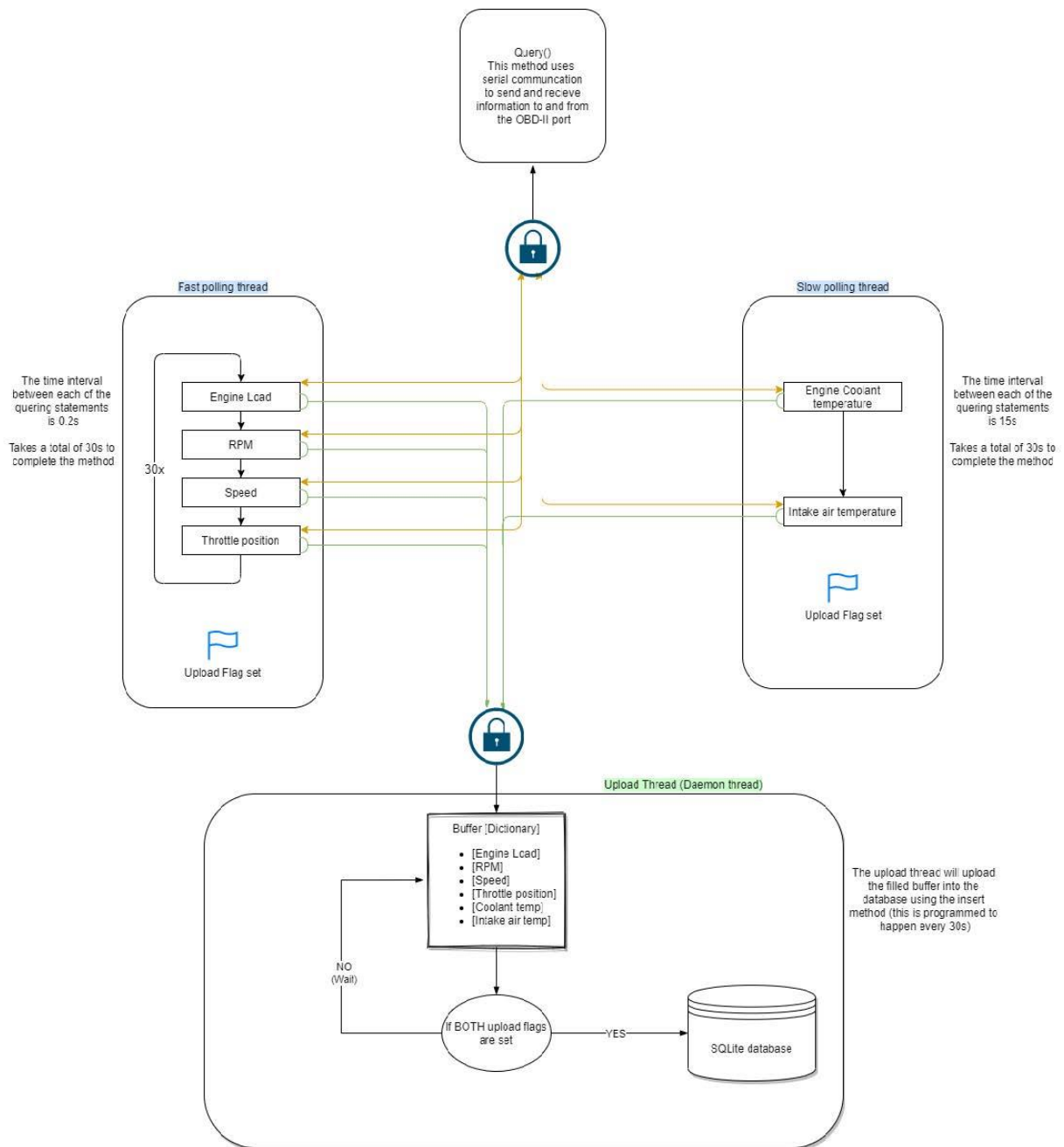


Figure 4 Structure of the multithreaded program

Implementation of steering angle detection mechanism.

The inattentiveness of drivers can be measured by their engagement with the steering wheel. Apart from the computer vision used to analyse the facial feature in detecting the driver attentiveness, the steering angle and vehicle parameters were used to detect the driver engagement.

The system to detect the steering angle was designed using the ESP3266, MPU6050 and HC-05 Bluetooth module. The prototype was built using Arduino, as shown in Figure 5. The circuit was planned to be placed at the back of the steering wheel, as shown below in Figure 6.

The data obtained from the sensor was processed, and the driver state was detected as active, moderately active and inactive. SPI communication was used in obtaining the data, and fuzzy logic was used in predicting the driver state. The driver state data combined with the vehicle data was used for driver profiling and driver inattentiveness measures.



Figure 5 Components used and prototype



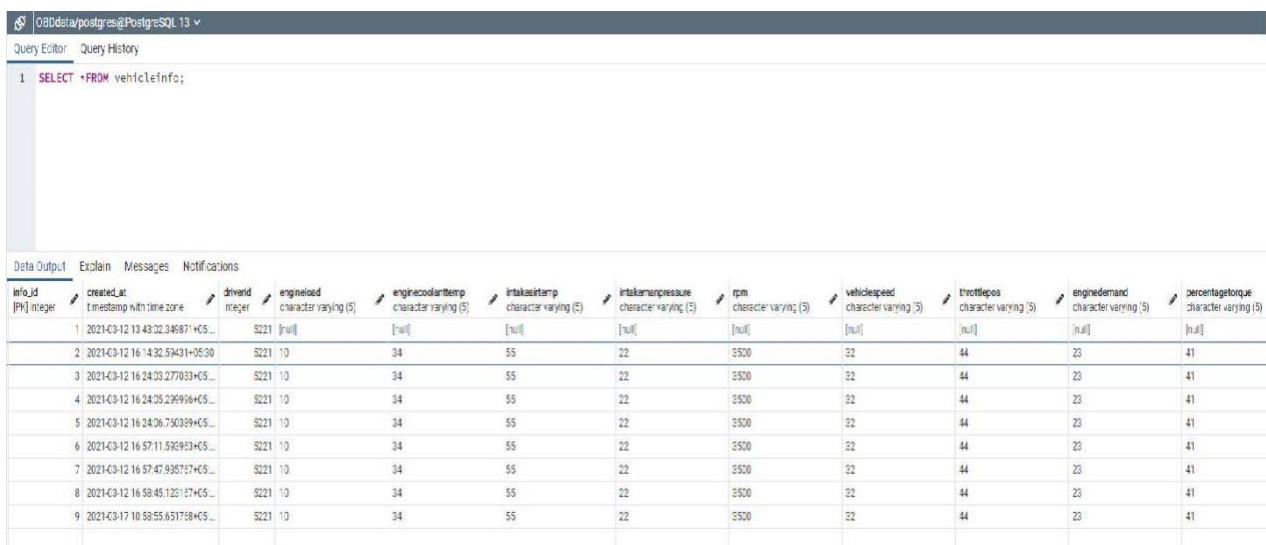
Figure 6 Placement of the device

Database

Local Database

The local database was implemented in raspberry pi to save the data collected from the vehicle in a structured and organised manner. The advantage provided by introducing a local database buffer is that a constant connection with the cloud is not required.

Figure 7 is an image of the local database with the vehicular parameters collected with the timestamp information.



| info_id | driver | created_at | drivem | engineoil | enginecoolanttemp | intakeairtemp | intakeairpressure | rpm | vehiclespeed | throttlepos | enginedemand | percentagetorque |
|---------|--------|----------------------------------|--------|-----------|-------------------|---------------|-------------------|--------|--------------|-------------|--------------|------------------|
| 1 | | 2021-03-12 13:43:22.346871+05... | S221 | [null] | [null] | [null] | [null] | [null] | [null] | [null] | [null] | [null] |
| 2 | | 2021-03-12 16:14:32.59431+05:30 | S221 | 10 | 34 | 55 | 22 | 3500 | 32 | 44 | 23 | 41 |
| 3 | | 2021-03-12 16:24:03.277033+05... | S221 | 10 | 34 | 55 | 22 | 3500 | 32 | 44 | 23 | 41 |
| 4 | | 2021-03-12 16:24:25.236996+05... | S221 | 10 | 34 | 55 | 22 | 3500 | 32 | 44 | 23 | 41 |
| 5 | | 2021-03-12 16:24:26.730339+05... | S221 | 10 | 34 | 55 | 22 | 3500 | 32 | 44 | 23 | 41 |
| 6 | | 2021-03-12 16:57:11.526953+05... | S221 | 10 | 34 | 55 | 22 | 3500 | 32 | 44 | 23 | 41 |
| 7 | | 2021-03-12 16:57:47.936757+05... | S221 | 10 | 34 | 55 | 22 | 3500 | 32 | 44 | 23 | 41 |
| 8 | | 2021-03-12 16:58:45.125157+05... | S221 | 10 | 34 | 55 | 22 | 3500 | 32 | 44 | 23 | 41 |
| 9 | | 2021-03-17 10:58:55.651759+05... | S221 | 10 | 34 | 55 | 22 | 3500 | 32 | 44 | 23 | 41 |

Figure 7 Local Database

The vehicular parameters from *Figure 7* are used to analyse patterns in driving styles and thereby determine the attentiveness of the driver using data in conjunction with visual parameters to increase the accuracy of prediction.

Authenticate

The authentication table of the database consists of the driver information and the fingerprint authentication. The table comprises a RowID; this is the first field in the table. This field also contains the PRIMARY KEY and is set to AUTOINCREMENT. This field is mainly to monitor the number of entries to the Authenticate table, a DriverID: this field stores the identification number of the driver and is of INTEGER datatype, the Name: The name of the driver is stored and the FingerTemplate: this is the most important column and it is used to store the fingerprint template of the drivers. This column is of BLOB datatype (Binary Large Object). The fingerprint features of the drivers are converted into a hexadecimal format in a process called serialisation and stored in this field. This value is used by the fingerprint scanner module to authenticate the user (driver).

3.3 REST API

To develop the Rest API the spring framework and Java programming language was used. There are several factors to be considered when developing a Rest API, and its implementation is done in several—identifying the model objects, creating model URLs, determining the representation type and assigning methods.

According to the first step, the objects that need to be represented as resources are configured. The resources are the endpoint of the API, and the URL models connect the resources with the sub-resources. The common representation is of two types: XML and JSON. In this application, a JSON representation was used. Assigning the HTTP methods is a crucial part. These methods are used to create, retrieve, update and delete (CRUD) data to the API.

Table 1 below shows the HTTP and CRUD representation. Initially, the API was built using an array list. Thereafter it was modified to apache derby and finally implemented in the PostgreSQL database.

This API was further developed to implement multiple-table databases. The database comprises three tables. One to store the collected vehicle data and another to store the driver information. The main table for mapping data the many to many mappings approach was used in the implementation.

Table 1 REST API Methods

| Method | Description |
|--------|--|
| GET | Retrieve information about the REST API resource |
| POST | Create a REST API resource |
| PUT | Update a REST API resource |
| DELETE | Delete a REST API resource or related component |

4 RESULTS AND DISCUSSION

The implemented system consists of facial feature detection, vehicular parameter analysis and an implemented DBMS to record and interpret data. Table 2 below shows the facial feature data used. Table 3 below shows the vehicle parameters obtained from the ECU at the same instance as the data obtained for the facial feature.

Table 2 Performance Evaluation Data

| HP | PERCLOS | BF | ECD | MCD | Subject State |
|----|---------|----|-----|-----|---------------|
| 15 | 20 | 10 | 1.1 | 0.9 | 0 |
| 70 | 89 | 4 | 3.9 | 3.5 | 2 |
| 65 | 80 | 3 | 3.4 | 2.7 | 2 |
| 40 | 52 | 23 | 2 | 2 | 1 |
| 40 | 48 | 20 | 2.3 | 1.8 | 1 |
| 15 | 45 | 17 | 1.7 | 1.2 | 0 |
| 10 | 30 | 12 | 1.9 | 1.5 | 0 |
| 10 | 35 | 14 | 1.5 | 1.7 | 0 |
| 50 | 90 | 2 | 3.9 | 3.6 | 1 |
| 64 | 90 | 3 | 4 | 3.7 | 1 |

Table 3 Shows the vehicle parameters recorded at the same instance as Table 2

| info_id | created_at | driverID | Engine Load | Engine Coolant Temp | Intake Air Temp | RPM | Vehicle Speed | Throttle Position |
|---------|---------------|----------|-------------|---------------------|-----------------|------|---------------|-------------------|
| 455 | 09-07-21 8:50 | 5221 | 58 | 82 | 50 | 1506 | 23 | 22 |
| 456 | 09-07-21 8:51 | 5221 | 68 | 82 | 50 | 1190 | 27 | 18 |
| 457 | 09-07-21 8:51 | 5221 | 22 | 83 | 50 | 942 | 25 | 17 |
| 458 | 09-07-21 8:51 | 5221 | 25 | 83 | 50 | 773 | 11 | 18 |
| 459 | 09-07-21 8:51 | 5221 | 39 | 83 | 50 | 882 | 6 | 17 |
| 460 | 09-07-21 8:51 | 5221 | 44 | 84 | 50 | 839 | 4 | 18 |
| 461 | 09-07-21 8:51 | 5221 | 62 | 84 | 50 | 1640 | 8 | 23 |
| 462 | 09-07-21 8:51 | 5221 | 69 | 83 | 50 | 1621 | 20 | 23 |
| 463 | 09-07-21 8:51 | 5221 | 69 | 83 | 50 | 1537 | 27 | 22 |
| 464 | 09-07-21 8:51 | 5221 | 74 | 83 | 50 | 1734 | 24 | 23 |

The CNN model was trained for 20 epochs and achieved 97% accuracy on the training set and 95.7% accuracy on the validation data set. The overall loss of the CNN was 1.6341 at the end of the 20 epochs and stopped decreasing. So the training was stopped at that point. Figure 8 shows the accuracy of the CNN after 20 epochs and the loss of the CNN after 20 epochs.

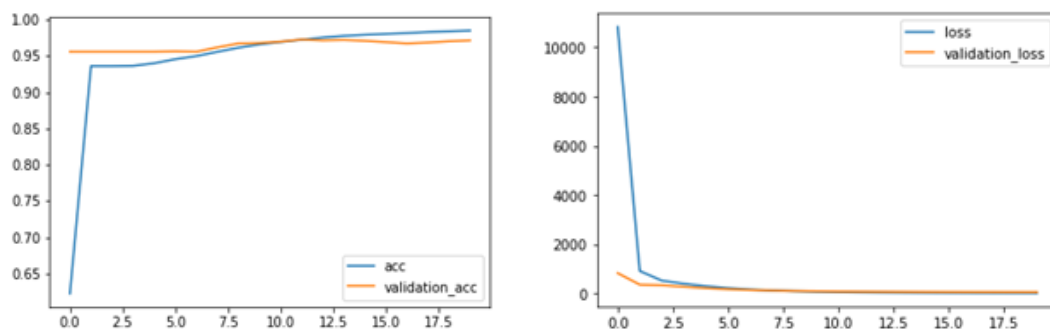


Figure 8 Training and validation accuracy and loss of CNN

The use of depth wise convolution reduces the amount of computation and the number of learnable parameters needed to implement the same CNN topology using the standard convolution. For the convolutional layer with the input size of 14x14x512, kernel size of 3x3x512x512, Table 4 shows the amount of Mult-Adds and parameters needed for those implementations.

Table 4 Memory and Computation resource usage for standard convolution and depth wise convolution

| Layer/Modification | Million Mult-Adds | Million Parameters |
|--------------------------|-------------------|--------------------|
| Convolution | 462 | 2.36 |
| Depthwise Seperable Conv | 52.3 | 0.27 |

Evaluation results of Mamdani FIS

The Mamdani fuzzy inference system is able to capture the non-linear relationship between the drowsiness indicators and the state of the driver in a very interpretable way. Its decision-making process

is more transparent compared to the other computational learning algorithms. But the performance of the Mamdani FIS is subjected to varying parameters: the choice of input and output membership functions, the shape of the membership functions, the number of rules in the rule base, the antecedents and consequents used in the rule base. These are the hyperparameters that need to be adjusted in order to get the optimal algorithm for this proposed driver state detection system.

There are many tradeoffs in changing these parameters

- Increasing the number of rules in the rule base will increase the performance of the FIS but it will increase the amount of computation needed to evaluate the output.
- The knowledge base can be optimised by the subject experts related to the problem, but there are difficulties in finding the right people for the task.
- Changing the membership functions to be more non-linear will increase the accuracy but will increase the amount of computation and time needed to make predictions in the defuzzification unit.
- Table 5 shows the performance evaluation metrics of the Mamdani fuzzy inference system.

Table 5 Performance evaluation results

| Algorithm | Precision | Recall | F1-Score |
|--------------------------------|-----------|--------|----------|
| Mamdani fuzzy inference system | 0.9333 | 1.0 | 0.9655 |

REST API

The REST API developed was tested using the postman software to ensure the functioning of the methods. The multi-table database was created and generated using the API. The implementation of the HTTPS methods was successful, and data from the local database was sent to the centralised database via the API.

5 CONCLUSION

Over time support technologies have proven themselves to be effective in preventing motor vehicle accidents caused by the inattentiveness of the driver. However, most systems either use the vehicular parameters or facial features whereas, this research proposes a hybrid system which improves the accuracy and reliability. The proposed system consists of a non-intrusive computer vision and fuzzy logic-based system to detect the early drowsiness symptoms of the drivers and warn them beforehand to avoid accidents and loss of lives. The proposed approach uses the facial feature information obtained from a camera to evaluate the state of the driver, their ability to drive and alert them if necessary.

Rather than using the in-car sensor's information and vehicle technical condition solely for assessing the state and the driver's ability to drive, using facial information that reflects the person's true state and reduces false positives. The use of fuzzy logic makes the proposed system well equipped to handle the uncertainties and subjectiveness present in the conditions for evaluating the drowsiness condition of a person. Further studies are needed to find more facial behavioral indicators of drowsiness robust to false positives and provide clinical validations to support those findings.

A real-time driver state detection system has been implemented successfully using a low-cost System on Chip Raspberry Pi 3B+, Pi camera, some additional electronics components, deep learning frameworks and fuzzy logic to detect the state of the driver and evaluate their ability to drive. The Mamdani fuzzy inference system used for the detection of driver state using facial features had a precision of 0.9333, a recall of 1.0 and F1-score of 0.9655 which proves the system to be reliable. The more important vehicle parameters and steering angle were obtained every 30 seconds to obtain accurate details of the parameters.

The use of multiple parameters and the hybrid approach makes the system more reliable. Also, this system is low cost, portable and can be used in any vehicle as there are no technical constraints in implementation. Another main advantage of this system is it is noninvasive. Thereby, it can be concluded that the proposed system can fulfill the objective of inattentive driver monitoring.

REFERENCES

- Grandjean, E. (1979). Fatigue in industry. *Br J Ind Med*, 175-186.
- J. Hossain, L. R. (2003). Underlying sleep pathology may cause chronic high fatigue in shift-workers. *Journal of sleep research*, 223-230.
- World Health Organization. (2021, June 21). From <https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries..>
- Warwick, B. a. (2015). Detecting driver drowsiness using wireless wearables. *IEEE 12th international conference on mobile ad hoc and sensor systems* (pp. 585-588). IEEE.
- Li, G. a.-L.-Y. (2015). Smartwatch-based wearable EEG system for driver drowsiness detection. *IEEE Sensors Journal*, 7169 - 7180.
- Jung, S.-J. a.-S.-Y. (2014). Driver fatigue and drowsiness monitoring system with embedded electrocardiogram sensor on steering wheel. *IET Intelligent Transport Systems*, 43 - 50.
- Zhang, Y. a. (2015). Driver fatigue recognition based on facial expression analysis using local binary patterns. *Optik*, 4501-4505.
- Picot, A. a.-S. (2012). Using retina modelling to characterise blinking: comparison between EOG and video analysis. *Machine Vision and Applications*, 1195-1208.
- Abtahi, S. a. (2011). Driver drowsiness monitoring based on yawning detection. *IEEE International Instrumentation and Measurement Technology Conference* (pp. 1-4). IEEE.
- Shan, S.-E. a. (2018). IoT and Computer Vision Based Driver Safety Monitoring System with Risk Prediction. *International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2)* (pp. 1-4). IEEE.
- Tadesse, E. a. (2014). Driver drowsiness detection through HMM based dynamic modeling. *IEEE International conference on robotics and automation (ICRA)* (pp. 4003-4008). IEEE.
- Zacharia, S. a. (2017). Implementation of steering wheel angle sensor system with Controlled Area Network. *International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)* (pp. 54-60). IEEE.
- Kaplan, S. a. (2015). Driver behavior analysis for safe driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 3017 - 3032.
- Satzoda, R. K. (2014). Drive analysis using vehicle dynamics and vision-based lane semantics. *IEEE Transactions on Intelligent Transportation Systems*, 9-18.
- Barbizet, J. (1958). Yawning. *J. Neurol. Neurosurg. Psychiatry*, 203-209.
- Mamdani, E. H. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International journal of man-machine studies*, 1-13.
- Ghoddosian, R. a. (2019). A realistic dataset and baseline temporal model for early drowsiness detection. *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*.
- SQLite. (n.d.). From [sqlite.org: https://www.sqlite.org/index.html](https://www.sqlite.org/index.html)
- Shaout, A. a. (2018). CAN Sniffing for Vehicle Condition, Driver Behavior Analysis and Data Logging. *International Arab Conference on Information Technology (ACIT)* (pp. 1-6). IEEE.
- Samson, S. a. (2019). Analysing Abnormalities While Driving To Avoid Accidents. *Fifth International Conference on Science Technology Engineering and Mathematics* (pp. 187-191). IEEE.
- Castignani, G. a. (2013). Driver behavior profiling using smartphones. *16th International IEEE Conference on Intelligent Transportation Systems* (pp. 552-557). IEEE.