Real Time Driver Drowsiness Detection Hybrid Approach

Praveen Kumar S[1], Naveena K N[2], Saritha A N[3], Sushma T P[4], Vedhavathi N L[5]

[1,2,4,5] *Undergraduate Student, B.M.S. College of Engineering, Bengaluru, India*
[3]*Assistant Professor, B.M.S. College of Engineering, Bengaluru, India*

---***---

***Abstract* - Driver drowsiness detection is an important safety feature in modern vehicles, as it can help prevent accidents occurred by fatigue or falling sleep while driving. A hybrid approach to detecting driver drowsiness combines multiple methods to improve accuracy and reliability. One common method for detecting drowsiness is through the use of visual tracking systems, which use cameras to track the driver's eyes and facial expressions. Another method is through the use of physiological sensors, such as electrocardiography (ECG) or using pulse sensor, which will measure heart rate to detect changes indicative of drowsiness. A hybrid approach combines these methods, using the strengths of each to give a more reliable results of the driver's alertness. The visual tracking system can detect signs of drowsiness such as eye closure or nodding, while the physiological sensors can measure changes in brain activity or heart rate that may indicate drowsiness. This hybrid approach has the potential to improve the accuracy and reliability of drowsiness detection systems, helping to keep drivers and passengers safe on the road.**

I. INTRODUCTION

Driver fatigue is a major contributor to road accidents and traffic fatalities, with estimates suggesting that it may be a factor in up to 20% of all crashes. As a result, there has been a increase interest in developing drowsiness detection systems to detect when a driver is becoming drowsy and alert them to the need to rest or pull over.

One approach to driver drowsiness detection is using sensors to track physiological signs such as eye movement, blinking rate, and facial muscle activity. For instance, a system may use a camera to track the position and movement of the driver's eyes, or employ sensors to measure changes in skin conductance or heart rate. By analyzing these indicators, the system can identify when the driver is about to go in drowsy state and issue an alert.

Another strategy is to use machine learning algorithms to analyze patterns of vehicle movement, such as lane deviations, abrupt braking, or extended periods of time without steering input. The technology can detect these patterns, the system can infer when the driver is becoming drowsy and issue an alarm.

A hybrid approach incorporates multiple methods, such as using both physiological sensors and machine learning algorithms, to improve precision and reliability of the drowsiness detection system. This can be particularly useful in instances where one method may be less successful on its own, such as when a driver is wearing sunglasses or has a medical condition that affects their eye movement.

Real-time driver drowsiness detection systems can be integrated into a vehicle's existing monitoring and warning systems, or they can be installed as standalone devices. They can be used to alert the driver to the need for a break or to pull over and rest, or they can trigger alarms or alerts for other drivers on the road. In some cases, the system may even be able to automatically slow or stop the vehicle if the driver does not respond to the alert.

Overall, real-time drowsiness related automobile accidents have the potential to significantly reduce the number of accidents and fatalities caused by driver fatigue. By alerting drivers to the need to rest and take a break, these systems can help keep everyone on the road safer.

II. PROBLEM STATEMENT

Since drowsy driving can be particularly dangerous because it can affect a driver's response time, judgment, and decision-making ability, making it more difficult for them to respond to changing road conditions or avoid potential risks. It can also raise the risk of lane deviations, sudden braking, and other unpredictable driving behaviors that can put other drivers at risk.

The goal of driver drowsiness detection hybrid approach is to seek to identify when a driver is becoming drowsy or exhasuted and alert them to the need to rest or pull over, in order to reduce the risk of accidents caused by driver fatigue. By combining multiple methods, such as using both physiological sensors and machine learning algorithms, the system can improve the precision and reliability of the drowsiness detection process, helping to ensure that drivers are informned to rest before they become too exhausted to continue driving safely.

The use of Deep neural networks will make it possible for identification of the detection of the drowsiness to have knowledge of anomalies by considering multiple domain specific artifacts as input.

III. Literature Survey

A. Key Terminologies

1. **Driver drowsiness:** The state of being tired or fatigued while driving, which can impair a driver's reaction time, judgment, and decision-making ability.

2. **Alert:** A warning or notification issued to the driver to alert them to the need to rest or pull over.

3. **Physiological sensors:** Sensors that measure indicators such as eye movement, blinking rate, skin conductance, or heart rate to detect drowsy driving.

4. **Machine learning algorithms:** Algorithms that analyze patterns of vehicle movement, such as lane deviations, sudden braking, or prolonged periods of time without steering input, to infer when the driver is becoming drowsy.

5. **Hybrid system:** A system that combines multiple methods, such as using both physiological sensors and deep learning algorithms, to improve the precision and reliability of the drowsiness identification process.

6. **Real-time:** A system that operates in real-time, meaning that it can detect and respond to drowsy driving in the moment, rather than after the fact.

7. **Standalone device:** A device that is not integrated into a vehicle's existing systems, but is instead installed separately.

8. **Automatic slowing or stopping:** The ability of a system to automatically slow or stop the vehicle if the driver does not respond to an alert.

B. Existing Systems

There are several existing systems for detecting driver drowsiness in real-time. These systems can be split into different categories: the one that physiological sensors to measure eye movement and blinking rate, and those that use machine learning algorithms to analyze patterns of vehicle movement. Physiological sensor-based systems often use cameras or sensors to track the position and movement of the driver's eyes, or to measure changes in skin conductance, heart rate, or other physiological indicators. By analyzing the system will be able to detect when the driver will be going into the drowsy state and issue an notifications. Machine learning-based systems use algorithms to analyze patterns of vehicle movement, such as lane deviations, sudden braking, or prolonged periods of time without steering input. By detecting these patterns, the system can infer when the driver is becoming drowsy and issue an alert. Hybrid systems use many multiple methods to increase the precison of sleepiness detection, such as use of both physiological sensors and machine learning algorithms, to improve the accuracy and reliability of the drowsiness detection process.

Some examples of existing driver drowsiness detection systems include:

1. Nissan's "Fatigue Detection System," which employs a camera to monitor the driver's eye movements and facial expressions to detect drowsiness.

2. Volvo's "Driver Alert Control," which uses machine learning algorithms to analyze the vehicle's movements and issue an alert if it detects patterns indicative of drowsy driving.

3. Drowsy Drive, a standalone device that uses a combination of physiological sensors and machine learning algorithms to detect drowsy driving and issue an alert.

These systems can be integrated into a vehicle's existing monitoring and warning systems, or they can be installed as standalone devices. They can be used to alert the driver to the need for a break or to pull over and rest, or they can trigger alarms or alerts for other drivers on the road. In some cases, the system may even be able to automatically slow or stop the vehicle if the driver does not respond to the alert.

IV. PROPOSED SOLUTION

A. *High Level Design*

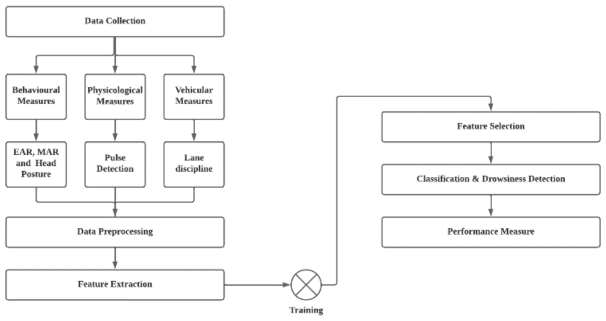


Fig.1: High level design

V. IMPLEMENTATION

A. Behavioral features

The main aim is to first combine all three facial features like Eyes, Yawning & Head Posture to predict the drowsiness for live camera. To be simple it is a regression issue, hence the model should be predicting 136 scalar as an outcome. The problem is in the performance. Since the will be hundreds of faces in a photo, Real time prediction is going to be very difficult unless out model is fast and quick enough.

Analyzing all these, We used CNN architecture called Xception Net. The geometry of a filter in a typical CNN layer is (output channels, input channels, k, k), where k stands for the filter size. The filter may take into account spatial correlations concurrently across many channels and within a single channel thanks to this convolution process.

The XceptionNet work, "Xception: Deep Learning with Depth wise Separable Convolutions," however, presents a new strategy by disentangling channel correlations and spatial correlations into independent operations. Since the InceptionNet also conducts a comparable decoupling, this idea is not entirely novel. The main distinction is the stronger decoupling assumption made by the XceptionNet, which gave it the nickname "Xception" ("Extreme Inception").

The creation of the Inception module was primarily driven by the need to more effectively separate spatial and cross-channel correlations. The fact that the Inception module accomplishes this by explicitly dividing the channels into a series of operations that independently take into account spatial and cross-channel correlations justifies the use of the word "efficient".

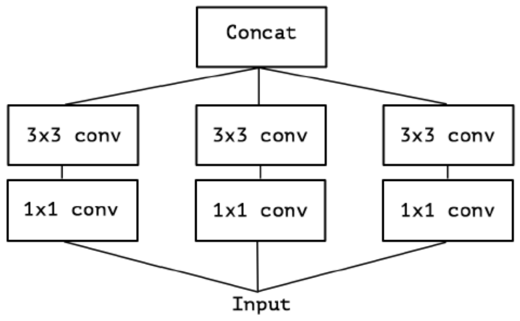


Fig.2: Simplified version of an inception module

Cross-channel correlations are handled by the standard Inception module in two steps. Multiple 1x1 convolutional neural networks (CNNs), also known as pointwise operations, are used in the initial processing of these correlations. The 1x1 CNNs interpret the cross-channel correlations by operating on parts of the input channels. The module then interprets the correlations inside these smaller 3D channel segments by employing standard CNN layers.

A 1x1 CNN is used in the initial operation of the original Inception module to record cross-channel correlations. Then, conventional CNN layers with kernel height and breadth greater than 1 are applied separately to each segment in the ensuing processes.

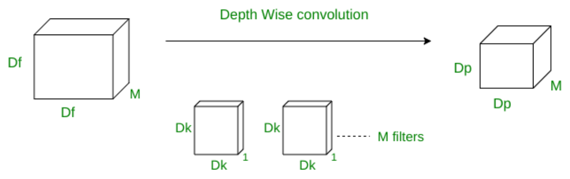


Fig.3: Depth wise convolution.

The extreme form of the Inception module, on the other hand, makes a greater assumption and presupposes that cross-channel correlations and spatial correlations may be totally dissociated. To do this, an extreme Inception module first maps the cross-channel correlations using a 1x1 CNN. The spatial correlations of each output channel are then individually mapped. Since each kernel is applied to a channel individually in the second operation, the kernel's form would be (output channels, 1, k, k).

Depth wise Separable Convolutions do a pointwise convolution first, then a channel-wise spatial convolution.

By isolating spatial and cross-channel correlations, this technique increases computing efficiency.

The Inception module, on the other hand, starts with a pointwise convolution and then performs following operations. In this module, a rectified linear unit (ReLU) non-linearity is frequently present during both the pointwise convolution and the following operations. The network's expressive capacity is increased by the introduction of non-linear transformations brought about by the integration of ReLU non-linearities.

Though there are no non-linearities between the pointwise and channel-wise spatial convolutions when depth wise separable convolutions are used to carry out the operations. A distinguishing feature of this technique is the depth wise separable convolutions, which further minimize computing costs by removing non-linearities.

Efficiency with depth wise Separable Convolution. We need to check many times a vanilla CNN layer needs to be multiplied to get an output shape of Co x Ho x Wo using kernels of size Ci x Kh x Kw, where Co is the number of kernels. Ci x Kh x Kw (which is the form of a single kernel) gives the number of multiplications for a single CNN operation using a single kernel. The total number of multiplications by a single kernel following the CNN procedure is (Ci x Kh x Kw) x (Ho x Wo). This operation is carried out Ho x Wo times for each kernel.

A hypothetical input layer might have the following shape: Ci x H x W, where Ci stands for the quantity of input channels and H and W are the height and breadth, respectively. Using kernels of size 1 x Kh x Kw, where Ci specifies the number of such kernels, our objective is to create an output layer with the form of Ci x Ho x Wo. Lets note that the height and breadth of the input layer are unaffected by pointwise convolution. As a result, the depth wise convolution ought to provide results with the appropriate end form. The number of multiplications needed in a single depth wise convolution by a single kernel is (1 x Kh x Kw), illustrating the fact that each kernel is applied individually to each input channel. Given that there are Ci total kernels, we can compute the total number of multiplications performed by all the kernels in a depth wise convolution layer as (1 x Kh x Kw) x (Ci x Ho x Wo). We may further reduce the equation when Ho equals Wo (both equal O) and Kh equals Kw (both equal K): The total number of multiplications in a depth wise convolution layer is represented by the expression (K2) x (Ci x O2). The number of multiplications in a depth wise separable convolution operation is now calculated using the formula Ci * O2 * (K2 + Co), and the result is 2,117,477,376. This figure is around nine times less than what a typical CNN layer would be. Furthermore, this depth wise separable convolution only has 134,144 parameters, which is around nine times fewer.

Each kernel is applied to each input channel separately in this case. In order to express the form of a single kernel, the quantity of multiplications needed for a single pointwise convolution operation by a single kernel is (1 x Kh x Kw). The total number of multiplications by a single kernel following the pointwise convolution procedure is (1 x Kh x Kw) x (Ho x Wo), with each kernel undergoing this operation Ho x Wo times. The total number of multiplications performed by all the kernels in this pointwise convolution layer may be estimated as (1 x Kh x Kw) x (Co x Ho x Wo) given that we have Co total kernels.

The sum of the multiplications in a depth-wise separable convolution layer is provided by (Ci x K2 x O2) + (Ci x O2 x Co), which is abbreviated to Ci * O2 * (K2 + Co) based on the aforementioned formulas. Let's look at an example where we want to run a standard CNN operation with a 3x3 kernel size, producing 512 output channels. The input layer has 256 channels, a height and width of 128, and these parameters. Accordingly, the output height and width, represented as O, may be calculated using the convolution formula, which results in 126. This suggests that Ci = 256, H = W = 128, K = 3, and Co = 512. The total number of multiplications in a CNN operation is 18,728,091,648 when these numbers are substituted into the formula (Ci x K2) x (Co x O2), which calculates the number of multiplications in a CNN operation. Additionally, there are 1,180,160 trainable parameters in this layer. In depth wise separable convolutions, the number of multiplications and parameter size are significantly reduced, which not only makes them quicker but also significantly more memory-efficient.

The ibug 300W Large Face Landmark Dataset, which is roughly 1.7GB in size and comprises over 7,000 faces with 68 landmark coordinate values per face, is a fantastic dataset for our project.

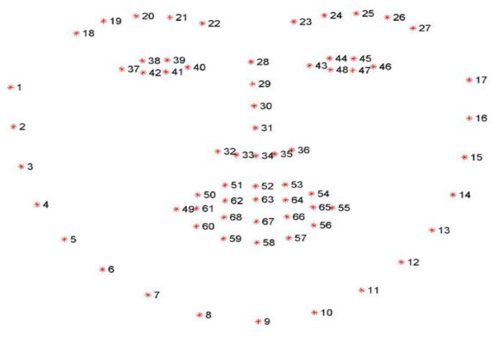


Fig.4: 68 points mark-up

Let's dissect the augmentation procedure into its component pieces. We will read, preprocess, and add augmentation to the faces in the first section before adding augmentation to the landmarks. The dataset is currently being enhanced using a variety of augmentations. With some additional border padding, the offset crop function randomly selects a face picture from the training data. The random face crop function then chooses a cropped area at random from the padded face picture. Random rotation is applied to the face and the related landmarks for landmark augmentation. The preprocessing function then scans the image, preprocesses it, and adds enhancements to the landmarks and face. After completing all of the augmentation stages, this yields a series of updated photos.

Fig.5: Visualizing few samples from the training batch

The author's recommended architecture, which we are executing exactly, is as follows: The model is not only a substitute for Inception modules in an Inception Net with depth wise separable convolutions. The Xception design, on the other hand, is made up of a linear stack of depth-wise separable convolution layers that incorporate residual connections. The supplied code snippet illustrates how depthwise separable convolutions are implemented in PyTorch. The input layer is divided into sections using the "groups" option. In this instance, we set it to the number of input channels, causing a kernel of form (1 x k x k) to be convolved with each input channel. It is crucial to note that this method uses depth wise separable convolutions with residual connections, which is consistent with the Xception architecture's original design.

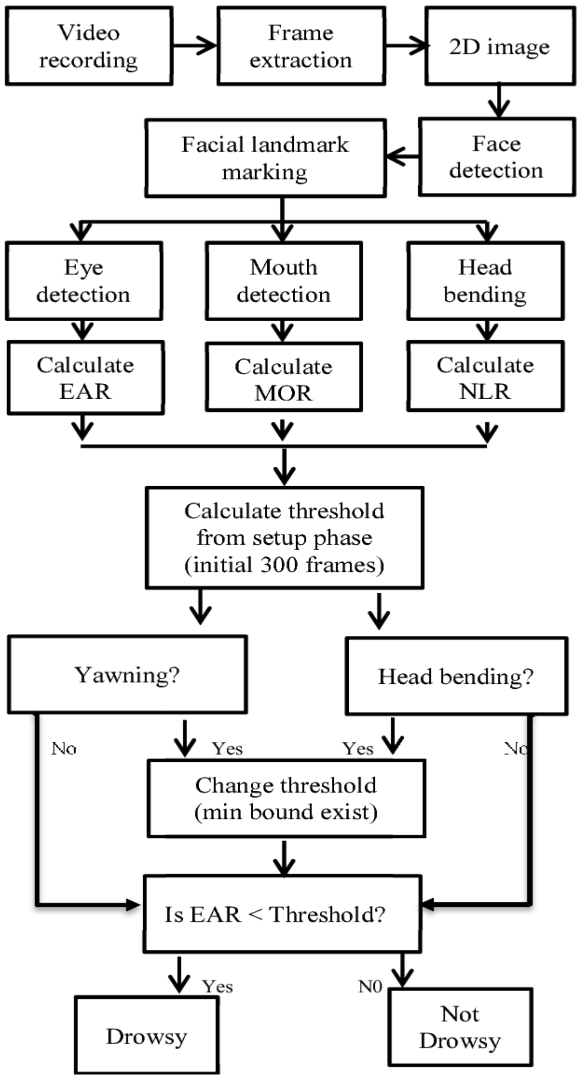


Fig.6: Behavioral Approach Detailed Design

B. Lane Detection

The main goal of the Hough transform is to find lines in a picture. It is a method for figuring out a shape's parameters based on its border points. The Hough transform has been developed throughout time to locate various forms, especially circles or ellipses. The equation y = mx + c is changed in the original Hough transform for straight lines and represented as x cos(e) + y sin(e) = r in its parametric version. As a result, lines may be represented in terms of their angle (e) and length (r) from the origin. Each point in an image corresponds to a sinusoidal wave in the parameter space in the context of the Hough transform. The lines in the picture are represented by each point in the parameter space. When two locations in the picture are collinear, the sinusoids that correspond to those points in the parameter space meet at that specific location. A similar process may be used to detect circles using the Hough transform. When several sites around the circle's circumference are known, it enables the estimate of circle parameters. The parametric equations

$$\begin{aligned} x &= a + R\cos(\theta) \\ y &= b + R\sin(\theta) \end{aligned}$$

can be used to describe a circle with radius R and centers (a, b). The points (x, y) trace the circumference of the circle as the angle changes from 0 to 360 degrees.

The following method is used to find the lane markings:

- For the parameters p and e, the parameter space is discretized across an appropriate range.
- The members of an accumulator array are all initialized to 0.
- The accumulator array is filled with each point in the gradient picture that exceeds a set threshold. These points are recognized, incremented, and saved.
- The value in the accumulator array represents the number of points on the associated line, and local maxima in the accumulator array correspond to collinear points in the image.

C. Sensor Based Approach

The sensor based approach consists of 4 sensors to provide necessary alerts to the driver when he is driving in the drowsy state. The sensors used are InfraRed Sensor, Ultrasonic Sensor, Gas Senor MQ-3 and Pulse sensor.

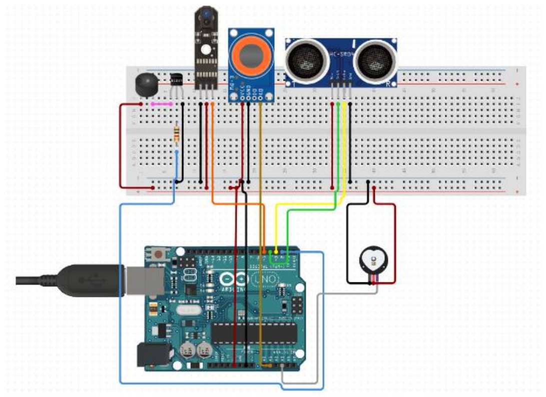


Fig.7: Circuit Diagram

The Arduino Uno microcontroller is used to interconnect all the sensors together and the data is collected from the sensors it is analyzed and the necessary alerts are given to the driver. The infrared sensor consists of IR emitter and IR photodiode. The emitter production the radiation, if the eye is open the radiation is not reflected back and if the eye is closed the radiation is reflected back to the IR photodiode. The IR photodiode will capture the radiation and the eye closeness is detected. This will be used to detect when the driver tends to fall a sleep.

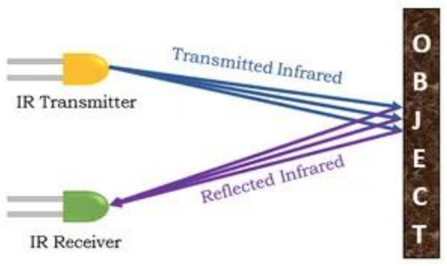


Fig.8: Infrared sensor

The Ultrasonic sensor works by transferring the sound waves and receiving it through transducer. The time taken to detect the sound waves by transducer determines the distance between the sensor and the object. This is used to avoid crashes when the driver is in the drowsy state.

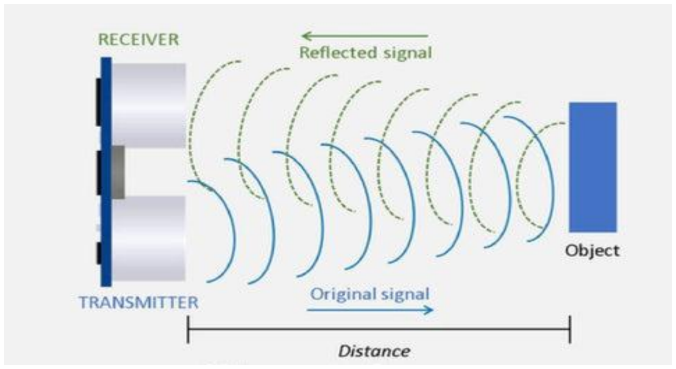


Fig.9: Ultrasonic sensor

A semiconductor gas sensor called the MQ3 smoke sensor is used to find when the driver is alcoholic . The sensor measures how a sensing material's resistance changes in the presence of alchohol. Tin dioxide (SnO2), which has a decreased electrical conductivity in clean air, serves as the detecting component of the MQ3 sensor. The SnO2 conductivity rises when smoke is present. The amount of smoke in the air directly relates to the change in resistance. Between 25 and 500 ppm of alchohol may be detected with the MQ3 sensor. A specific temperature is reached on the sensor. The sensor is open to the atmosphere. The conductivity of SnO2 rises in alcohol. The concentration of alcohol is calculated and the driver is warned for driving in the alcoholic state.

By flashing a light through the skin and measuring the quantity of light reflected, a pulse sensor operates. With every heartbeat, the blood flow alters, which affects how much light is reflected. The sensor can then determine the heart rate using this data. A light is projected through the skin by the sensor. The sensor counts how much light is reflected. Based on variations in the amount of light reflected, the sensor determines the heart rate. When the driver is in the drowsy state the pulse rate reduces drastically, we capture the pulse rate from the driver and the alert is provide when he is in the drowsy state.

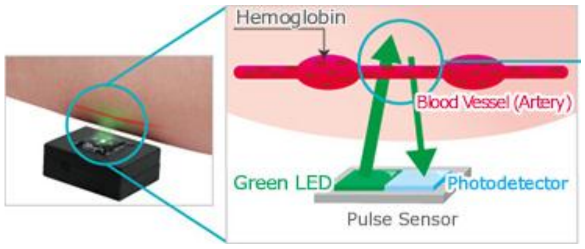


Fig.10: Pulse sensing

All the above mentioned sensor are integrated together to provide the necessary alerts to the driver in real time.

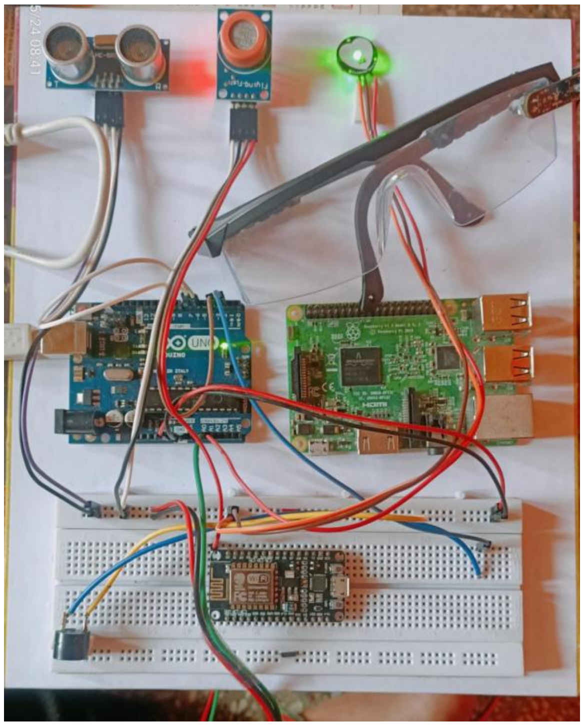


Fig.11: Prototype

VI. RESULTS

A. Behavioral Approach

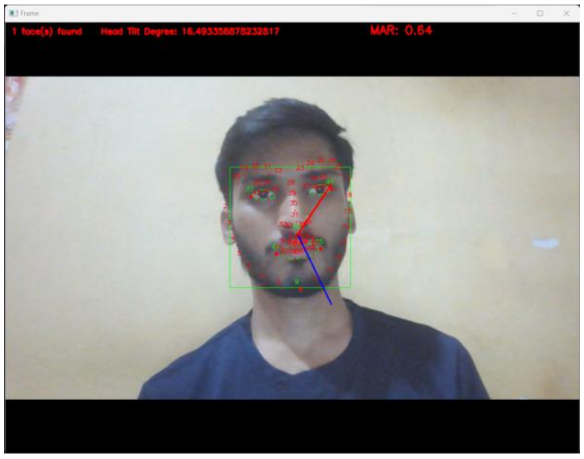


Fig.12: Image of Idle Face

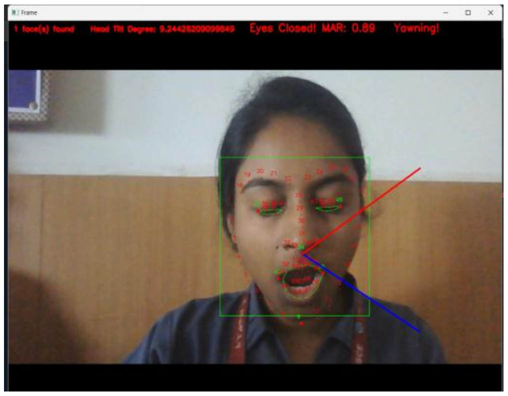


Fig.13: Detection of Drowsiness

B. Vehicular Approach

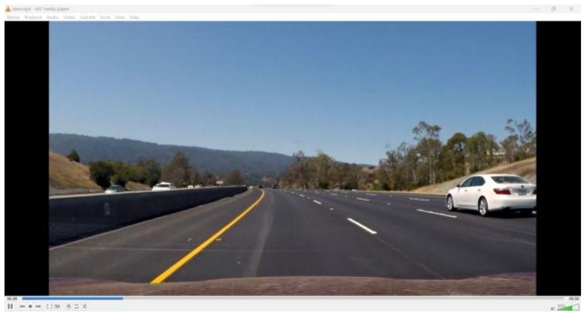


Fig.14: Frame of Road

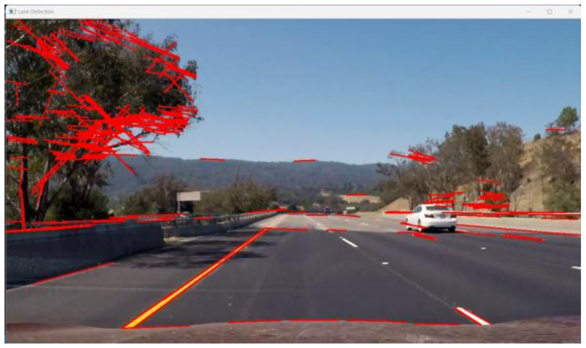
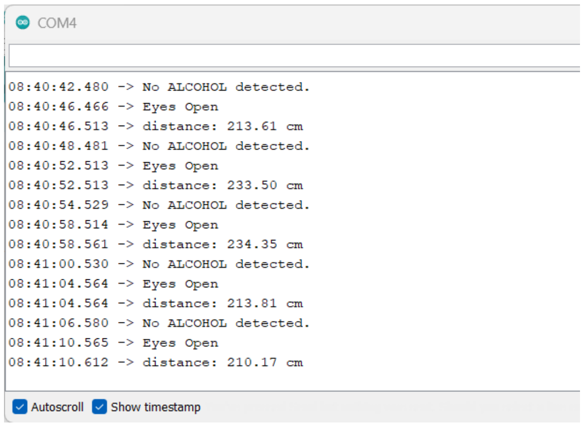


Fig.15: Detection of lane

C. Sensor Based Approach



```
08:40:42.480 -> No ALCOHOL detected.
08:40:46.466 -> Eyes Open
08:40:46.513 -> distance: 213.61 cm
08:40:48.481 -> No ALCOHOL detected.
08:40:52.513 -> Eyes Open
08:40:52.513 -> distance: 233.50 cm
08:40:54.529 -> No ALCOHOL detected.
08:40:58.514 -> Eyes Open
08:40:58.561 -> distance: 234.35 cm
08:41:00.530 -> No ALCOHOL detected.
08:41:04.564 -> Eyes Open
08:41:04.564 -> distance: 213.81 cm
08:41:06.580 -> No ALCOHOL detected.
08:41:10.565 -> Eyes Open
08:41:10.612 -> distance: 210.17 cm
```

Fig.16: Data captured from sensors

VII. CONCLUSION

In conclusion, real-time driver drowsiness detection is a important technology for reducing traffic fatalities occurred by driver exhaustion. By detecting when a driver is becoming drowsy or fatigued and alerting them to the need to rest or pull over, these systems can help keep everyone on the road safer.

Real-time driver drowsiness detection systems can be integrated into a vehicle's existing monitoring and warning systems, or they can be installed as standalone devices. They can be used to alert the driver to the need for a break or to pull over and rest, or they can trigger alarms or alerts for other drivers on the road. In some cases, the system may even be able to automatically slow or stop the vehicle if the driver does not respond to the alert. Overall, these systems have the potential to significantly reduce the number of car accidents and fatalities caused by driver fatigue.

REFERENCES

[1] M. Ramzan, H. U. Khan, S. M. Awan, A. Ismail, M. Ilyas and A. Mahmood, "A Survey on State-of-the Art Drowsiness Detection Techniques," in IEEE Access, vol. 7, pp. 61904-61919, 2019, doi: 10.1109/ACCESS.2019.2914373.

[2] R. Ahmad and J. N. Borole, "Drowsy driver identification using eye blink detection," Int. J. Comput. Sci. Inf. Technol., vol. 6, no. 1, pp. 270–274, Jan. 2015.

[3] C. Yan et al., "Video-based classification of driving behavior using a hierarchical classification system with multiple features," Int. J. Pattern Recognit. Artif. Intell., vol. 30, no. 5, 2016, Art. no. 1650010.

[4] I. Teyeb, O. Jemai, M. Zaied, and C. B. Amar, ''A novel approach for drowsy driver detection using Head posture estimation and eyes recognition system based on wavelet network,'' in Proc. 5th Int. Conf. Inf., Intell., Syst. Appl. (IISA), Jul. 2014, pp. 379–384.

[5] O. Khunpisuth, T. Chotchinasri, V. Koschakosai and N. Hnoohom, "Driver Drowsiness Detection Using Eye-Closeness Detection," 2016 12th International Conference on Signal- Image Technology & Internet-Based Systems (SITIS), 2016, pp. 661-668, doi: 10.1109/SITIS.2016.110.

[6] Y. Katyal, S. Alur and S. Dwivedi, "Safe driving by detecting lane discipline and driver drowsiness," 2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies, 2014, pp. 1008-1012, doi:10.1109/ICACCCT.2014.7019248.

[7] G. Zhenhai, L. DinhDat, H. Hongyu, Y. Ziwen and W. Xinyu, "Driver Drowsiness Detection Based on Time Series Analysis of Steering Wheel Angular Velocity," 2017 9th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), 2017, pp. 99-101, doi: 10.1109/ICMTMA.2017.0031.

[8] Z. Li, S. E. Li, R. Li, B. Cheng, and J. Shi, ''Online detection of driver fatigue using steering wheel angles for real driving conditions,'' Sensors, vol. 17, no. 3, p. 495, Mar. 2017.

[9] H. A. Rahim, A. Dalimi, and H. Jaafar, ''Detecting drowsy driver using pulse sensor,'' J. Technol., vol. 73, no. 3, pp. 5–8, Mar. 2015.

[10] B. Warwick, N. Symons, X. Chen, and K. Xiong, ''Detecting driver drowsiness using wireless wearables,'' in Proc. 12th Int. Conf. Mobile Ad Hoc Sensor Syst. (MASS), Oct. 2015, pp. 585–588.

[11] H. S. AlZu'bi, W. Al-Nuaimy, and N. S. Al-Zubi, ''EEG-based driver fatigue detection,'' in Proc. 6th Int. Conf. Develop. Syst. Eng. (DESE), Dec. 2013, pp. 111–114

[12] M. Awais, N. Badruddin, and M. Drieberg, ''A hybrid approach to detect driver drowsiness utilizing physiological signals to improve system performance and wearability,'' Sensors, vol. 17, no. 9, p. 1991, Aug. 2017.

[13] Gustavo A. Peláez C., Fernando García, Arturo de la Escalera, and José María Armingol," Driver Monitoring Based on Low-Cost 3-D Sensors." IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, VOL. 15, NO. 4, Page(s): 1855 – 1860 AUGUST 2014.

[14] A. Picot, S. Charbonnier, and A. Caplier, "On-line automatic detection of driver drowsiness using a single electroencephalographic channel," in Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE, 2008, pp. 3864-3867.

[15] G. Borghini, L. Astolfi, G. Vecchiato, D. Mattia, and F. Babiloni, "Measuring neurophysiological signals in aircraft pilots and car drivers for the assessment of mental workload, fatigue and drowsiness," Neuroscience & Biobehavioral Reviews, 2012.