DRIVER DROWSINESS DETECTION REPORT

Objective

The primary objective of this project is to develop a real-time, automated system capable of detecting driver drowsiness or fatigue through non-intrusive visual monitoring. Road accidents caused by drowsy driving are a major concern worldwide, often leading to serious injuries, fatalities, and significant economic losses. Traditional methods for detecting fatigue, such as self-reporting or vehicle-based measures (like steering behavior), are either unreliable or dependent on external conditions. Therefore, the goal of this project is to implement a more effective and real-time solution by leveraging computer vision and machine learning techniques to analyze the driver's facial features—particularly their eye behavior. The system aims to recognize early signs of drowsiness by tracking the Eye Aspect Ratio (EAR), which decreases significantly when a person's eyes are closing or remain closed for extended periods. By continuously monitoring EAR using a live video stream from a webcam, the system can determine if the driver is becoming drowsy and trigger an immediate alert, both visually and audibly, to help prevent potential accidents. This project is designed to be lightweight, costeffective, and easily deployable in various environments, including personal vehicles, fleet management systems, and public transportation, ultimately contributing to improved road safety and driver awareness.

Tools and Technologies Used

This project integrates a diverse set of technologies, each playing a critical role in the successful implementation of the Driver Drowsiness Detection System. The chosen technologies are optimized for real-time performance, accuracy, and ease of use, ensuring that the system operates efficiently with minimal hardware requirements.

1. Python

Python is the primary programming language used for this project due to its simplicity, readability, and the vast ecosystem of libraries available for computer vision and machine learning. Python enables rapid prototyping and provides seamless integration with third-party

libraries necessary for image processing, sound playback, and mathematical computations. It also supports cross-platform deployment, making the system easily portable.

2. OpenCV (Open Source Computer Vision Library)

OpenCV is a powerful and widely-used open-source computer vision and machine learning library. It is responsible for handling all the image and video processing tasks in the project, such as reading frames from the webcam, converting images to grayscale, drawing contours on the eyes, and displaying output in real time. Its performance efficiency and support for a wide range of functions make it ideal for developing real-time vision applications like drowsiness detection.

3. dlib

The dlib library provides state-of-the-art machine learning tools and is especially known for its facial landmark detection capabilities. In this project, dlib is used to detect the driver's face and extract 68 facial landmarks, including those around the eyes. These landmarks are crucial for calculating the Eye Aspect Ratio (EAR), which serves as the foundation of the drowsiness detection logic. The pre-trained facial landmark model (shape_predictor_68_face_landmarks.dat) used by dlib is accurate and efficient even in varying lighting conditions.

4. SciPy (Scientific Python)

SciPy is used in this project to perform Euclidean distance calculations, which are needed to compute the EAR. By measuring the distances between specific eye landmarks, SciPy helps determine whether the eyes are open or closed. Its fast and reliable mathematical functions support real-time performance and numerical precision.

5. imutils imutils is a convenience library that simplifies many common image processing tasks. In this project, it is mainly used to convert the dlib shape object into a NumPy array of coordinates using its face_utils module. This makes it easier to extract and manipulate specific facial landmarks, such as those around the eyes.

6. pygame pygame is a Python module designed for writing video games but is used here to implement the alarm system. When drowsiness is detected, pygame is used to load and continuously play a sound file (alarm.wav) to alert the driver. It is lightweight and easy to integrate, providing a simple way to handle audio playback within the Python environment.

7. Webcam (Input Device)

A basic webcam serves as the primary input device, capturing live video of the driver's face. This video feed is then processed frame-by-frame by the system. The project is hardware-light and does not require any specialized sensors or high-end cameras, making it cost-effective and widely deployable.

Methodology

The methodology of the Driver Drowsiness Detection System revolves around real-time facial feature analysis, particularly focusing on the driver's eye movements. The detection logic is rooted in a metric called **Eye Aspect Ratio (EAR)**, which is computed from the positions of specific facial landmarks. The process begins by capturing a live video feed using the system's webcam. Each frame from this feed is processed individually in a continuous loop to monitor the driver's alertness.

The first step in analysing each video frame involves **face detection**. This is accomplished using the `dlib` library's frontal face detector, which provides accurate detection of human faces in the grayscale representation of the frame. Once a face is detected, a **facial landmark predictor** (`shape_predictor_68_face_landmarks.dat`) is employed. This predictor locates 68 specific landmarks on the detected face, including the outlines of the eyes, eyebrows, nose, mouth, and jawline. For the purpose of drowsiness detection, the landmarks corresponding to the left and right eyes are extracted.

With the eye coordinates now available, the system proceeds to calculate the **Eye Aspect Ratio (EAR)** for each eye. EAR is a mathematical formula that quantifies the openness of the eye using Euclidean distances between certain eye landmarks. It is calculated by measuring the vertical distances between the eyelids and dividing it by the horizontal width of the eye. A high EAR indicates an open eye, while a significantly low EAR implies a closed or nearly closed eye. By taking the average of both eyes' EAR values, the system gains a robust metric for determining the driver's eye state in each frame.

To detect drowsiness reliably, the system compares the EAR value to a predefined threshold (e.g., 0.25). However, to avoid false positives from natural eye blinks, the system does not immediately trigger an alert. Instead, it maintains a **frame counter** to track how many consecutive frames the EAR remains below the threshold. Only if the EAR stays consistently

low across a specified number of frames (e.g., 20 frames), the system concludes that the driver is likely drowsy or asleep. This method ensures that short, intentional blinks do not activate the alarm.

Once the drowsiness condition is met, the system initiates both **visual and auditory alerts**. A warning message such as "DROWSINESS ALERT!" is overlaid on the video stream to provide a visual cue. Simultaneously, an alarm sound (`alarm.wav`) is played using the `pygame` library, ensuring the driver is audibly alerted as well. The alarm continues until the driver opens their eyes again and the EAR rises above the threshold for a few frames, at which point the alarm is stopped and the system resets the counter.

Throughout the process, **OpenCV** is used extensively for capturing and displaying video, drawing contours around the eyes, and annotating frames with EAR values and alert messages. This real-time feedback loop ensures that the driver is continuously monitored with minimal delay. The combination of computer vision techniques, facial landmark analysis, and a responsive alarm system makes this methodology highly effective for addressing driver fatigue and promoting road safety.

- **Face Detection:**

`dlib`'s `get_frontal_face_detector()` is used to detect the face in real-time from webcam input.

- **Facial Landmark Detection:**

- `shape_predictor_68_face_landmarks.dat` is used to detect 68 facial landmarks.
- Landmarks corresponding to the eyes are extracted.

- **Eye Aspect Ratio (EAR):** • EAR is a value that represents the openness of the eye. When a person is drowsy, the eyes begin to close, and the EAR decreases.
- EAR is computed using Euclidean distances between key eye landmarks.
- Formula:

$$EAR=\frac{||p2-p6||+||p3-p5||}{2\times||p1-p4||}EAR=\frac{2\times||p1-p4||}{||p2-p6||+||p3-p5||}$$

- **Drowsiness Detection:**

- If the EAR remains below a threshold (e.g., `0.25`) for a number of consecutive frames (e.g., `20`), an alarm is triggered using `pygame`.

System Workflow

---

1. Start video capture.

2. Detect face and facial landmarks.
3. Extract eye coordinates.
4. Calculate EAR.
5. If EAR < threshold for continuous frames:
    - Trigger a visual alert.
    - Play alarm sound.
6. Reset alarm when eyes are open again.

Parameters Used

Parameter	Value	Description
EAR Threshold	0.25	Below this value, eye is considered closed
Consecutive Frames	20	Number of frames before triggering alarm

Alarm System

- **pygame** is used to load and play an alarm sound (`alarm.wav`).
- The alarm runs in a loop and stops only when the driver opens their eyes again.

Applications

The Driver Drowsiness Detection System has a wide range of practical applications, especially in fields where human alertness is critical for safety and operational efficiency. The most prominent and direct application lies in the automotive and transportation sector, where the system can be integrated into personal vehicles, commercial trucks, buses, and public transport systems. Long-haul truck drivers and night-shift bus operators are especially prone to fatigue, and an automated alert system can serve as a lifesaving tool by preventing accidents due to microsleeps or unrecognized drowsiness. The system can function as an add-on safety module for modern Advanced Driver Assistance Systems (ADAS), enhancing the safety features already present in smart vehicles.

- Driver safety systems in cars and trucks
- Monitoring fatigue in industrial environments
- Assistive tools for night shift workers.

Limitations

- Requires good lighting conditions
- Might not work well if the face is not fully visible
- Glasses or poor camera quality may affect performance

Future Improvements

- Add yawning detection (using mouth landmarks)
- Integrate with IoT (send alert to connected car systems)
- Use deep learning models for higher accuracy
- Deploy on mobile or embedded systems (Raspberry Pi)

Conclusion

The Driver Drowsiness Detection System developed in this project presents a practical and efficient solution to one of the leading causes of road accidents—driver fatigue. By leveraging computer vision and facial landmark analysis, the system is capable of monitoring the driver's eye movements in real time and determining their level of alertness based on the Eye Aspect Ratio (EAR). The implementation successfully demonstrates that EAR is a reliable metric for identifying closed or drowsy eyes, which can be used as an early indicator of driver fatigue.Through the integration of widely-used libraries such as OpenCV, dlib, and pygame, the system provides a low-cost, non-intrusive, and easily deployable solution that can be incorporated into a variety of vehicles without requiring expensive hardware or advanced sensors. The use of visual and auditory alerts ensures that once signs of drowsiness are detected, the driver is immediately made aware of the danger, allowing them to take corrective action—such as pulling over to rest or taking a break. This real-time response capability is critical in reducing the likelihood of accidents due to microsleeps or impaired attention.

Moreover, the modular and open-source nature of the project ensures that it can be further improved and customized to meet the needs of specific users or industries. For instance, the system can be enhanced by incorporating other fatigue indicators such as yawning detection, head nodding, or eye closure duration tracking. With further development, it could even be integrated into IoT-based vehicle management systems or used in autonomous vehicles as a safety redundancy feature.In conclusion, this project provides a strong foundation for building AI-powered safety systems aimed at protecting drivers and passengers from the risks associated with fatigue. It proves that with the right combination of computer vision and intelligent

algorithms, technology can play a crucial role in saving lives, promoting safe driving practices, and potentially transforming the future of transportation safety.