

# **Driver Drowsiness Detection For Enhancing Road Safety Using AI**

**Thesis Submitted  
In Partial Fulfilment of the Requirements for the  
Degree of**

**MASTER OF TECHNOLOGY**

**in  
Data Science**

**by  
Manish Kumar Paul  
(2K23/DSC/04)**

**Under the Supervision of  
Dr. Abhilasha Sharma  
(Associate Professor, SE, DTU)**



**To the  
Department of Software Engineering  
DELHI TECHNOLOGICAL UNIVERSITY  
(Formerly Delhi College of Engineering)  
Shahbad Daulatpur, Main Bawana Road, Delhi-110042, India**

**May, 2025**

## ACKNOWLEDGEMENT

I would like to express my deep appreciation to **Dr. Abhilasha Sharma**, Associate Professor at the Department of Software Engineering, Delhi Technological University, for her invaluable guidance and unwavering encouragement throughout this research. Her vast knowledge, motivation, expertise, and insightful feedback have been instrumental in every aspect of preparing this research plan.

I am also grateful to **Prof. Ruchika Malhotra**, Head of the Department, for her valuable insights, suggestions, and meticulous evaluation of my research work. Her expertise and scholarly guidance have significantly enhanced the quality of this thesis.

My heartfelt thanks go out to the esteemed faculty members of the Department of Software Engineering at Delhi Technological University. I extend my gratitude to my colleagues and friends for their unwavering support and encouragement during this challenging journey. I have had some friends that I am thankful to be around. They made me feel truly at home. In particular, I would like to thank Roshni Singh whom I had such a great time.

While it is impossible to name everyone individually, I want to acknowledge the collective efforts and contributions of all those who have been part of this journey. Their constant love, encouragement, and support have been indispensable in completing this MTech thesis.

**Manish Kumar Paul**

**(2K23/DSC/04)**



# DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)  
Shahbad Daultpur, Main Bawana Road, Delhi-42

## CANDIDATE DECLARATION

I, MANISH KUMAR PAUL (2K23/DSC/04) hereby certify that the work which is being presented in the thesis entitled “**Driver drowsiness detection for enhancing road safety using AI**” in partial fulfillment of the requirements for the award of the Degree of Master of Technology submitted in the Department of Software Engineering, Delhi Technological University in an authentic record of my work carried out during the period from August 2023 to May 2025 under the supervision of Dr. Abhilasha Sharma. The matter presented in the thesis has not been submitted by me for the award of any other degree of this or any other Institute.

**Candidate’s Signature**

This is to certify that the student has incorporated all the corrections suggested by the examiner in the thesis and that the statement made by the candidate is correct to the best of our knowledge.

**Signature of Supervisor(s)**

**Signature of External Examiner**



# DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)  
Shahbad Daultapur, Main Bawana Road, Delhi-42

## CERTIFICATE BY THE SUPERVISOR

Certified that Manish Kumar Paul (2K23/DSC/04) has carried out their project work presented in this thesis titled “**Driver drowsiness detection for enhancing road safety using AI**” for the award of **Master of Technology** from the Department of Software Engineering, Delhi Technological University, Delhi under my supervision. The thesis embodies results of original work, and studies are carried out by the student himself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date:

**Dr. Abhilasha Sharma**

Associate Professor,  
Department of Software Engineering,  
DTU-Delhi,

## TABLE OF CONTENTS

CHAPTER	PAGE NO.
DECLARATION .....	2
CERTIFICATE .....	3
ACKNOWLEDGEMENT .....	4
LIST OF FIGURE(S).....	6
LIST OF TABLE(S) .....	8
ABSTRACT .....	9
CHAPTER 1: INTRODUCTION .....	10
CHAPTER 2: LITERATURE SURVEY .....	15
CHAPTER 3: DESIGN METHODOLOGY .....	21
CHAPTER 4: PERFORMANCE ANALYSIS AND RESULTS.....	32
CHAPTER 5: LIMITATIONS.....	35
CHAPTER 6: CONCLUSION AND FUTURE SCOPE.....	36
REFERENCES .....	40

## LIST OF FIGURE(S)

<b>Figure Number</b>	<b>Figure Name</b>	<b>Page Number</b>
1.1	Binary Images Showing The Intensity Of The Images	10
1.2	Difference Between Color Image And Grayscale Image [From Drowsy Driver Dataset]	11
1.3	Conversion Of Pixel In RGB Image Of Red Channel [From Drowsy Driver Dataset]	12
1.4	Effect Of RGBA 100% To 60% [From Drowsy Driver Dataset]	13
2.1	Different Scenario Different Behaviour From NTHU Dataset.	15
2.2	Sample Frames From The UTA-RLDD Dataset In The Alert (First Row), Low Vigilant (Second Row) And Drowsy (Third Row) States.	17
2.3	Samples From State Farm Driver Distraction Dataset	17
3.1	Data Distribution Graph	19
3.2	Drowsy Sample Data	19
4.1	Processed Image Through MIP [Drowsy Driver Dataset]	21
4.2	Applying Gaussian Processing [From Drowsy Driver Dataset]	22
4.3	Layering In Neural Networks	27

5.1	Code Execution For The Image Resizing	28
5.2	Output Of The Image Resizing	28
5.3	Code Execution Of Image Rotation	29
5.4	Implementation For Edge Detection	31
5.5	Code Execution Of Morphological Processing	32
7.1	Output For Morphological Processing	32

### LIST OF TABLE(S)

Table Number	Table Name	Page Number
1.	Different Cases Of Different Behaviors	31
2.	Performance Of Different Algorithms Of Image Processing(Like Accuracy, F1 score, Processing Time Etc.)	33
3.	Performance Analysis Of Different Classic Image Processing Algorithms And Methods On The Basis Of Accuracy, F1 Score And Time.	34



## ABSTRACT

Drowsiness of driver is a major factor contributing to collisions worldwide. To address this issue, advanced driver assistance systems (ADAS) have emerged, leveraging image processing and ML techniques to detect signs of driver fatigue and mitigate potential risks. This project focussed on developing a robust driver drowsiness detector system employing various image processing and ML algorithms.

The proposed system begins by capturing real-time images or video frames of a driver's face through in-vehicle cameras. A comprehensive pre-processing stage involves facial landmark detection and extraction of relevant characteristics, such as eye closure, head pose, and facial expressions. Subsequently, a well-curated dataset is utilized for training an ML model, optimizing its ability of recognizing patterns indicative of drowsiness.

Several ML algorithms, including CNNs and RNNs, have been explored to achieve high accuracy and efficiency in the detection process. Transfer learning techniques are also applied to use pre-trained models, enabling effective feature extraction and enhancing the model's generalization across diverse datasets.

Furthermore, the system incorporates real-time monitoring and feedback mechanisms, alerting the driver through auditory, visual, or haptic cues when signs of drowsiness are detected. The efficiency of the proposed system is tested through rigorous simulations and real-world testing, considering various driving conditions and scenarios.

The results demonstrate the system's capability of reliably detecting driver drowsiness, exhibiting promising accuracy rates and less false positive/negative rates. The combination of image processing and ML in driver drowsiness detection contributes to the enhancement of street safety by providing timely alerts and assisting drivers in maintaining an alert and focused state while driving.

# CHAPTER 1

## INTRODUCTION

Image processing is the technique to enhance the images to get better accuracy of the image so that it can build better model according to the dataset and the algorithms.

Drowsiness of driver is a big factor which contributes to road mishaps worldwide. To address this issue, advanced driver assistance systems (ADAS) have emerged, leveraging image processing and ML techniques of detecting signs of driver fatigue and mitigate potential risks. This project focussed on building a robust driver drowsiness detector system employing image processing and ML algorithms.

### 1.1 Types OF Images

There are a number of types of images in present generation and every type of image is based on Pixels and numbers. Some of the types of the images are:

#### 1. Binary Image

Binary images are characterized by having two distinct pixel intensity values: zero, representing black, and one, representing white. The images are employed to emphasize a specific region in a colored picture. A common application includes image segmentation, where the binary format effectively isolates and highlights specific features, as illustrated.

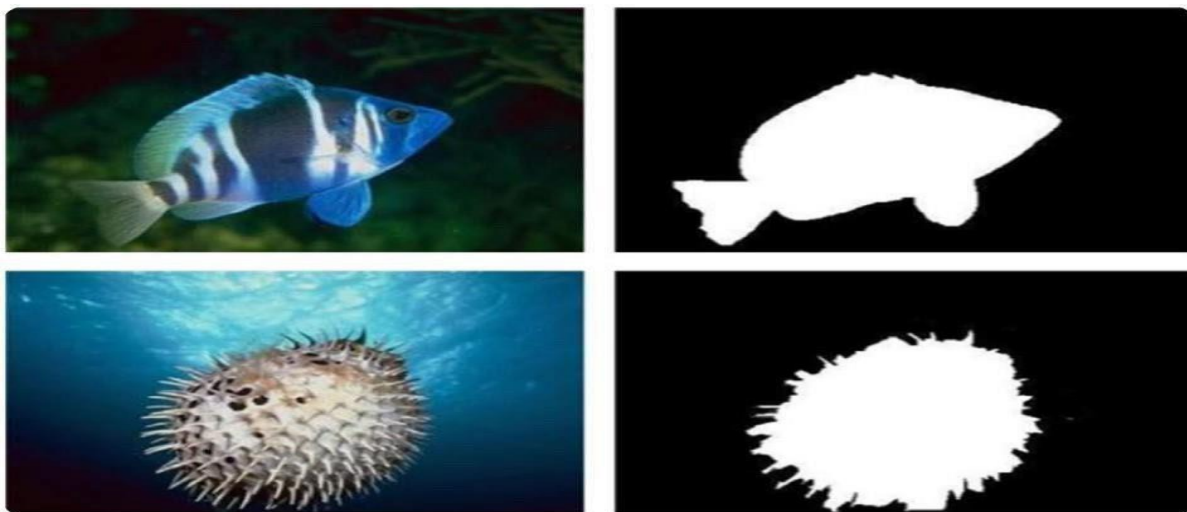


Figure 1.1: Binary Images Showing The Intensity OfThe Images [2]

## 2. Grayscale Image

A gray (or 8-bit) image consists of 256 distinct colors in which 0 pixel density represents black and 255 pixel density represents white. The remaining 254 values in the middle are all various shades of gray.

Below is the instance of converting RGB to the grayscale image.



Figure 1.2: Difference Between Color Image And Grayscale Image [From Drowsy Driver Dataset]

## 3. RGB Color Image

Those images, that are used in present world are RGB, or color image, which is a 16-bit matrix for computers. So each pixel will be having 65,536 distinct colors. “RGB” referring to red, green, and finally blue “channels” of an image.

Now our image has only a single channel, i.e; two coordinates can define the position of each value of the matrix. Now, three equal matrices are superimposed, each with a value between 0 and 255, so we need three unique variables to represent the results of the matrix elements.

Therefore, a pixel in RGB image will be black whenever its value is (0, 0, 0) and white whenever the value is (255, 255, 255). Any of these combination of two creates all the different colors that exist in the world. For example (255, 0, 0) for red, (0, 255, 0) for green while (0, 0, 255) for blue.

Below is an instance of RGB image being split into component channels. Note that the histogram is different for each channel.



Figure 1.3: Conversion Of Pixel In RGB Image Of Red Channel [From Drowsy Driver Dataset]

#### 4. RGBA Image

An RGBA image is an RGB color image with an extra channel called "alpha" which defines the opacity of that RGB image. Opacity will range from 0% and all the way to fully 100% and is essentially a feature to see through.

In physics, opacity describes light passing via an object. The alpha channel in the RGBA images attempts to follow this feature. An example below demonstrates this change.

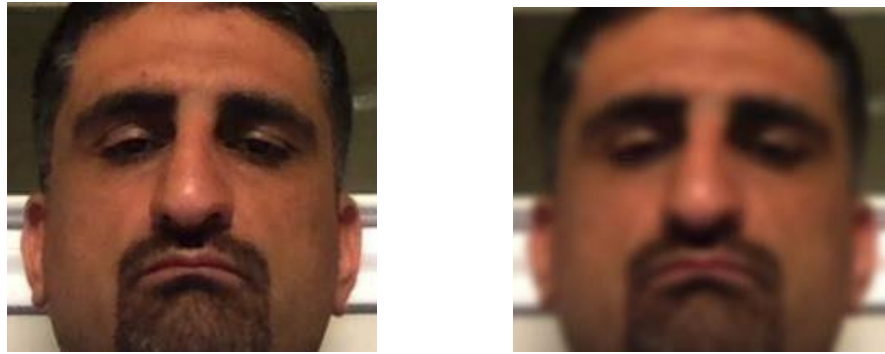


Figure 1.4: Effect Of RGBA 100% To 60% [From Drowsy Driver Dataset]

## 1.2 Phases of Image Processing

Steps in the pipeline are as follows:

### 1. Image Capture

The image gets captured with the help of a camera which is then digitized by the use of an analog to digital converter to further process on the computer.

### 2. Image Enhancement

Here, the resulting image gets processed for meeting requirements of the study in which it will be used. These techniques usually focus on important elements in the image, such as contrast as well as brightness. Trustworthy images are inherently contextual.

### 3. Image Restoration

It involves enhancement of front side of the image and is the objective function as distortion of the image is attributed to the required models or designs.

### 4. Color Image Processing

It is designed to process color images (16-bit RGB and/or RGBA images) to correct the color or its patterns within the image.

### 5. Wavelets And Multi-resolution Rendering

They are building blocks that represent different images. The image is divided into multiple smaller regions to compress data and pyramid representation.

## **6. Image Compression**

To transfer images to different devices because of computational storage issues, images must be compressed and can't be retained to their actual volume, which is further important to display images in the internet.

## **7. Morphological Processing**

Image components which are useful in representing and describing its shape need extraction to further process tasks. Morphological processing gives the facilities necessary for accomplishing this task.

## **8. Image Segmentation**

This step involves splitting the image into distinct values for simplifying the image representation and/or transform it into something more complex and simple. Eight for verification. Image segmentation allows computers to focus on and discard important parts of the image, allowing machines to work more efficiently.

## **9. Representation And Description**

The image segmentation process usually follows these steps; The task of the representation here is to determine whether the segmented space should be represented as a boundary or as the entire space. Annotation involves the extraction of material that produces some useful information of interest or forms the basis of distinguishing one type of material from another.

## **10. Object Detection And Recognition**

After segmenting an object from any image and completing the representation as well as description phase, the system needs to assign a label for the object; It allows the user to know which object is detected; for instance "car", "cat" etc.

### 1.3. PROBLEM STATEMENT AND DATASET

The problem statement of this report is “IMAGE PROCESSING ON DROWSY DRIVER DATASET”. Here are some of the content of dataset taken from the paper related to the drowsy driver that I have reviewed.

#### 1. NTHU

The entire dataset consists of 36 different races recorded with/without glasses/sunglasses in different simulated driving situations including driving, yawning, slow blinking, sleeping, laughing, and more, in day and night illumination. Subjects are recorded while sitting in an ordinary gaming chair that simulates driving and pedaling; They were also taught to have facial expressions by the experimenter. The combined length of the entire collection is around 9.5 hours.

The training material consists of 18 items, 5 different (BareFace, Glasses, Night\_BareFace, Night\_Glasses, Sunglasses). Each study interval, including slow blinking rate during yawning or nodding, was recorded for 1 min [1]. These conditions correspond to the two most crucial conditions, which are an amalgamation of sleep-related symptoms (yawning, nodding, slow blinking rate) and the inability to sleep (talking, laughing, looking to the side) [1]; All records are predictable. 1.5 minutes long. The measured and quantified data contains 90 driving videos (18 other subjects) in different conditions, including a mixture of sleepy and sleepless states.



Figure 2.1: Different Scenario Different Behaviour From NTHU Dataset.[1]

Dataset	Category	Nbr. Videos	Nbr. Images Extracted	Nbr. Images not detected by Dlib
Training	With glasses	36	106,882	13,581
	Night Without glasses	36	52,372	8,713
	Night With glasses	36	50,991	11,032
	Without glasses	36	108,380	12,343
	With sunglasses	36	107,990	24,274
Evaluation	With glasses	4	37,357	2,478
	Night Without glasses	4	29,781	2,459
	Night With glasses	4	32,922	1,389
	Without glasses	4	45,005	4,291
	With sunglasses	4	28,214	2735
Total		200	599,894	87,586

Table 1: Different Cases Of Different Behaviors [1]

## 2. UTA

The University of Texas at Arlington Real-Life Drowsiness Dataset was developed for multi-level fatigue research, not only for severe and easily observed cases, but also for subtle cases where the slightest hint leads to aversion. Detection of small changes is important for early detection of sleep and therefore activation of sleep protection mechanisms. The subtle microexpression of sleep has a physical and emotional basis, making it difficult for the player to truly simulate this expression.

The file contains approximately 30 hours worth of RGB video from 60 participants. We obtained all videos for each participant, out of a total of 180, in three different categories: wakefulness, low noise, and sleep. Courses are staffed by undergrad. or postgrad. students as well as staff members who volunteer or receive extra credit for the course, all participants are 18 years of age or older. There were 51 men and 9 women of various ethnicities (10 Caucasian, 5 non-Hispanic white, 30 Indo-Aryan and Dravidian, 8 Middle Eastern, and 7 East Asian) and age (mean age, including 20 to 59 years). ages 25 to 59). Standard deviation 6). In 21 out of the 180 videos, the participants wore glasses, and in 72 out of the 180 videos, the subjects had a lot of facial hair. The film was taken from multiple locations in different real-life environments and contexts. All videos were captured by the participants with the use of a cellphone or webcam, with frame rate being always below 30fps; This represents the desired frame rate for commonly used cameras.





Figure 2.2: Some Frames From The Dataset In The Alert (First Row), Low Vigilant (Second Row) And Drowsy (Third Row) States.[3]

## State Farm Driver Distraction

According to the CDC motor vehicle safety division, 1 in every 5 vehicle accidents is because of a distracted driver, translating to 425,000 people sustaining injuries and over 3,000 people getting killed due to distracted driving every year.



Figure 2.3: Samples From State Farm Driver Distraction Dataset[3]

## **CHAPTER 2**

### **LITERATURE SURVEY**

A literature survey on image processing for drowsy driver detection datasets reveals a growing body of research focused on taking advantage of advanced technologies to enhance road safety. The datasets used in these studies play a crucial role in the training as well as testing ML models designed for detecting signs of driver fatigue. Below is a summary of key studies and datasets in this domain:

#### **Dataset For Drowsy Driver Detection (DDD) By YawnTech**

Description: YawnTech's DDD dataset provides a comprehensive collection of images and videos capturing drivers in varying states of alertness. The dataset includes diverse lighting conditions, facial expressions, as well as head poses.

Significance: Widely used for training as well as evaluating machine learning models due to its diverse and realistic scenarios.

#### **The State Farm Distracted Driver Detection Dataset**

Description: This dataset contains images taken from in-vehicle cameras showing drivers engaged in various activities, including both normal and distracted behaviors.

Significance: Initially designed for distracted driver detection, researchers have adapted this dataset to include drowsiness detection, highlighting its versatility.

#### **UDC Drowsiness Detection Dataset**

Description: The University of Dayton's dataset focuses specifically on drowsiness detection, capturing facial features, eye movements, and head positions of subjects in controlled environments.

Significance: Valuable for studying the nuances of drowsiness-related facial expressions and movements.

#### **CASIA-WebFace Dataset**

Description: Originally created for face recognition, CASIA-WebFace is a large-scale dataset containing facial images under different conditions, making it suitable for training models to recognize facial features related to drowsiness.

Significance: Used as a supplementary dataset for facial feature extraction and analysis in drowsy driver detection studies.

### **AWARENESS Dataset**

Description: The AWARENESS dataset focuses on monitoring driver behavior using a combination of in-vehicle cameras and physiological sensors. It includes images and sensor data related to drowsiness.

Significance: Enables the integration of physiological signals with visual cues for a more comprehensive drowsiness detection approach.

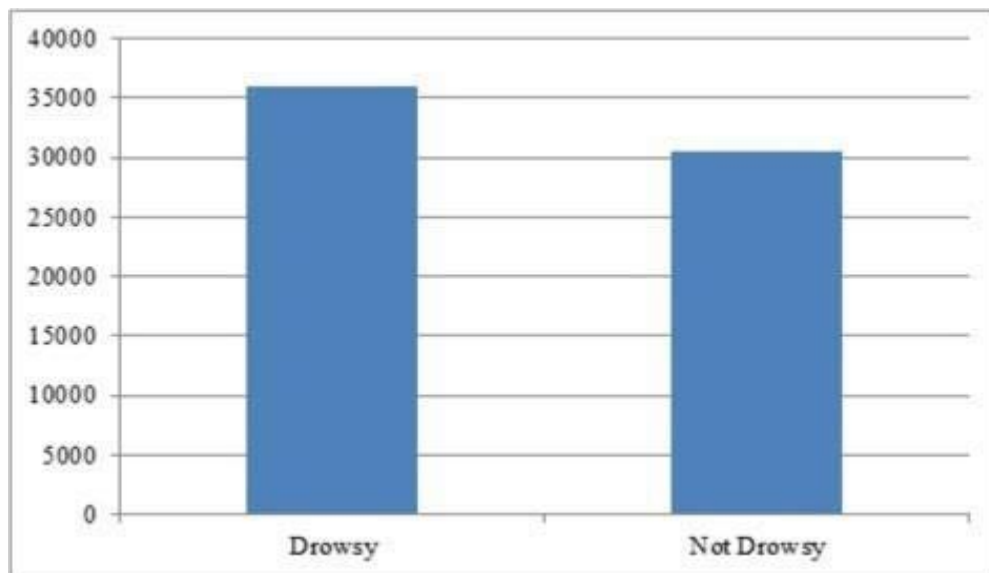


Figure 3.1: Data Distribution Graph[4]



Figure 3.2: Drowsy Sample Data[4]

### **Driver Monitoring Dataset (DMD)**

Description: DMD captures images of drivers in various real-world driving conditions, emphasizing different times of the day and diverse weather conditions.

Significance: Offers a realistic setting for evaluating the robustness of drowsiness detection models in challenging scenarios.

### **NTHU-DDD Dataset**

Description: The National Tsing Hua University Drowsy Driver Detection dataset includes images with various poses and facial expressions, providing a diverse set of visual data for model training.

Significance: For testing the performance of models in detecting drowsiness across different facial expressions and head orientations.

This literature survey demonstrates the significance of diverse and well-curated datasets in advancing research on image processing for drowsy driver detection. Researchers utilize these datasets to develop and validate the ML models, ultimately contributing to improvement of driver safety through advanced driver assistance systems.

### **Data Augmentation**

It involves addition of new data to the existing data for artificially increasing its size. The data augmentation process generates huge amounts of data, making it more convenient for training deep learning models on large data sets. Therefore, data augmentation prevents the model from over-fitting.

## CHAPTER 3

### METHODOLOGY

Classic image processing algorithms :

#### 1. Morphological Image Processing (MIP)

The morphological image process tries to eliminate defects in the binary image since the binary region formed by the initial simple will be affected by noise. It will also help turn the image quality on and off.

Morphological operations can be extended to grayscale images. It has a nonlinear function depending on the structure of the image. It depends on the relative order of the pixels, but not their number. Image analysis techniques use small patterns, called patterns, that are placed at different places in the image and compared to corresponding pixels. The pattern is a small matrix with values 0 and 1.

Let's look at two morphological image processing operations, dilation and erosion:

The operation of dilation to add pixels to area of the object in the image

The operation of erosion removes the object's border pixels on.

The number of pixels added or removed from the actual image is dependent on the size of the process.

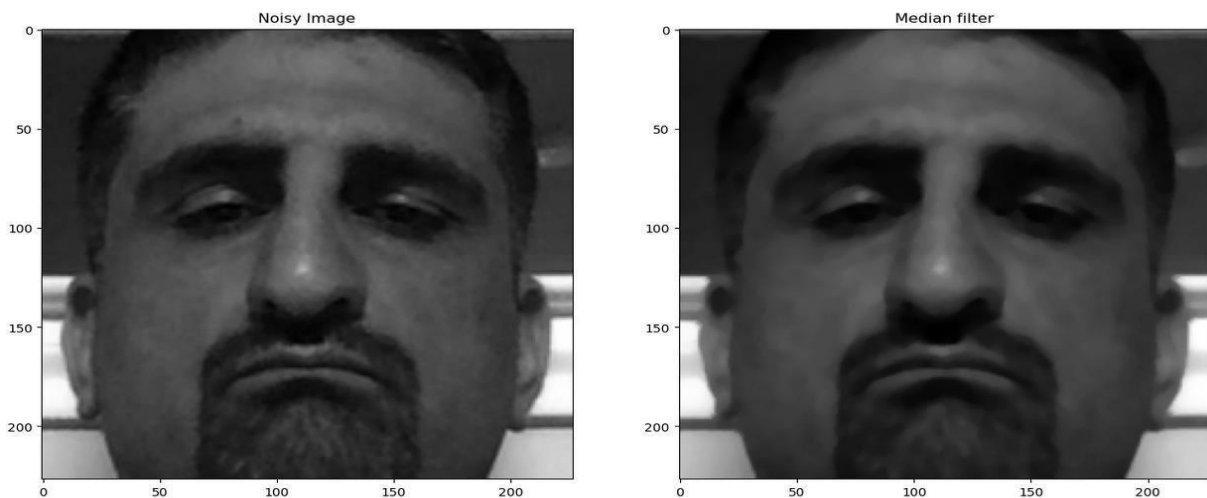


Figure 4.1: Processed Image Through MIP [Drowsy Driver Dataset]

## 2. Gaussian Image Processing

A Gaussian filter is a low-frequency filter for reducing noise and image blur. This filter is used as a random symmetric kernel passing via every pixel in the region of interest for obtaining the required result.

Since pixels in the center of the nucleus have more weight than the surroundings, it is not easy to have color changes (edges) in the nucleus. A Gaussian filter may be thought of as an (mathematical) estimation of the Gaussian function. Here, we will learn how to use the Gaussian filter for reducing noise in images.

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

Formula 1: Gaussian Equation

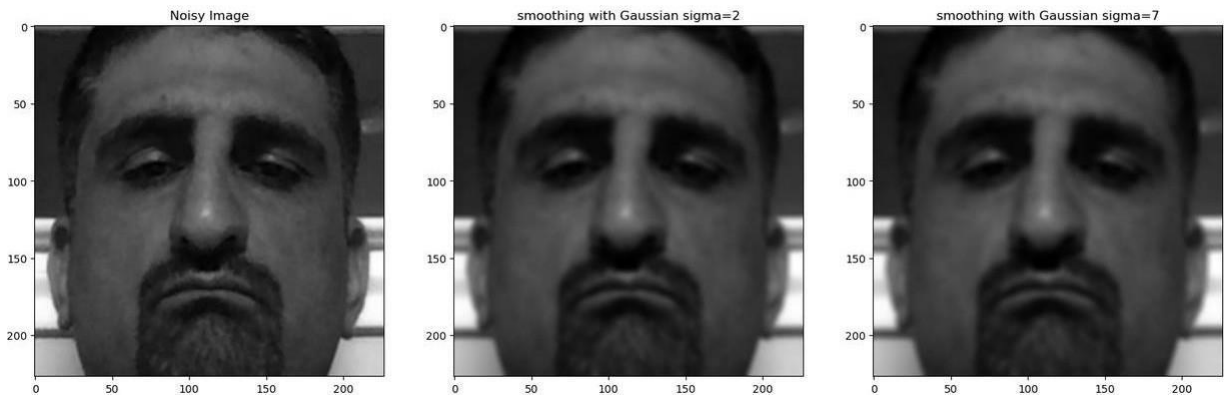


Figure 4.2: Applying Gaussian Processing [From Drowsy Driver Dataset]

## 3. Fourier Transform On Image Processing

Fourier Transform plays a significant role in image processing, especially in the analysis

and manipulation of images by transforming them from the spatial domain to the frequency domain. The transformation allows for a different perspective on the image's content, revealing information about the distribution of frequencies and patterns. Here are key aspects of Fourier Transform in image processing:

### **3.1. Image Transformation**

**Spatial to Frequency Domain:** Fourier Transform will be applied for converting an image in the spatial domain (pixel values in rows and columns) to the frequency domain. This transformation is known as the 2D Fourier Transform.

### **3.2. Frequency Spectrum Visualization**

The resulting image in the frequency domain represents the distribution of frequencies present in the original image. It consists of magnitude and phase information for various frequency components.

### **3.3. Frequency Analysis**

**Low-Frequency vs. High-Frequency Components:** In the frequency domain representation, low-frequency components correspond to smooth variations in the image, while high-frequency components represent rapid changes or details. This analysis is useful for understanding the image's content and structure.

### **3.4. Filtering**

**Frequency-Based Filtering:** Fourier Transform enables the application of filters in the frequency domain. Filtering can be performed to enhance or suppress specific frequency components in the image.

**High-Pass and Low-Pass Filtering:** High-pass filters can be used to emphasize fine details, while low-pass filters are effective for smoothing or blurring the image.

### **3.5. Image Compression**

**Selective Frequency Component Retention:** Fourier Transform is utilized in image compression techniques where the less significant frequency components are discarded, reducing the amount of data required to represent the image.

**Quantization of Frequency Coefficients:** After the transformation, quantization of frequency coefficients allows for the reduction of data precision, contributing to compression.

### 3.6. Edge Detection

High-Frequency Emphasis for Edges: Edges in images correspond to high-frequency components. Fourier Transform aids in emphasizing these high-frequency elements, facilitating edge detection algorithms.

### 3.7. Inverse Fourier Transform

Frequency to Spatial Domain Conversion: After analysis or manipulation in the frequency domain, the inverse Fourier Transform is applied to convert the image back to the spatial domain, allowing for visualizing the effects of the transformations.

### 3.8. Applications In Image Processing

Medical Imaging: Fourier Transform is used in medical imaging for tasks such as image enhancement, filtering, and the analysis of features in diagnostic images.

Remote Sensing: In satellite imagery and remote sensing applications, Fourier Transform helps analyze terrain features and vegetation patterns.

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(\frac{ux}{M} + \frac{vy}{N})}$$

Formula 2: Formula For 2D Fourier Transform

## 4. Edge Detection Image Processing

Edge detection facilitates users for observing the characteristics of an image for any change within the gray level. The texture indicates end of one region in the image, while the beginning of another, which decreases the quantity of data in an image while preserving the structural properties of the image.



$$Gy = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * Image\ matrix$$

$$Gx = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * Image\ matrix$$

Figure 4.3: 2D Signal Processing Matrix[7]

## 5. Image Processing Using Neural Network

Image processing using neural networks involves taking advantage of the power of artificial neural networks to perform tasks such as image classification, object detection, segmentation, and even generation. CNNs are especially popular in image processing because of their ability to effectively capture spatial hierarchies and patterns. Here's a general overview of how neural networks are applied in image processing:

### 5.1. CNNs

**Architecture:** CNNs consist of a number of layers like convolutional layers, the pooling layers, and the fully connected layers. The convolutional layers apply filters for detecting patterns, while the pooling layers for reducing spatial dimensions, and fully connected layers to make predictions.

**Feature Extraction:** CNNs automatically learn hierarchical features from images, capturing low-level features like textures and edges in the early layers and high-level features like object shapes in deeper layers.

### 5.2. Image Classification

**Task:** For an image, the neural network assigns it to a specific category or class.

**Training:** CNNs are trained on labeled datasets, adjusting their parameters (weights and biases) during backpropagation for minimizing the gap between predicted and actual labels.

### **5.3. Object Detection**

Task: Identify and locate multiple objects within an image.

Architecture: Networks like R-CNN, Fast R-CNN, and YOLO use a combination of object proposal methods and neural networks to achieve object detection.

### **5.4. Image Segmentation**

Task: Dividing an image into meaningful segments or regions.

Architecture: FCNs and U-Net architectures are commonly for image segmentation tasks, where every pixel in the output corresponds to a specific segment.

### **5.5. Image Generation**

Task: Create new, realistic images.

Architecture: Generative models like GANs and VAEs generate the images by knowing the distribution of training data.

### **5.6. Image Enhancement**

Task: Improve the quality or characteristics of an image.

Applications: Neural networks can be trained to perform tasks like denoising,

### **5.7. Transfer Learning**

Idea: Pre-trained neural network models on massive datasets can be fine-tuned for specific image processing tasks.

Benefits: Transfer learning accelerates training on smaller datasets and leverages features learned from diverse image datasets.

### **5.8. Style Transfer**

Task: Apply the artistic style of one image to another one, while preserving its content.

Architecture: Neural networks can learn to separate and recombine content and style features, facilitating for artistic transformations.

## 5.9. Image Anomaly Detection

Task: Identify anomalous patterns or outliers in images.

Architecture: Autoencoders, a type of neural network, which can be used for unsupervised anomaly detection by learning to reconstruct normal patterns and to detect deviations.

## 5.10. Face Recognition

Task: Identify and verify faces in images.

Architecture: Face recognition systems often use CNNs to extract facial features and perform classification or verification tasks.

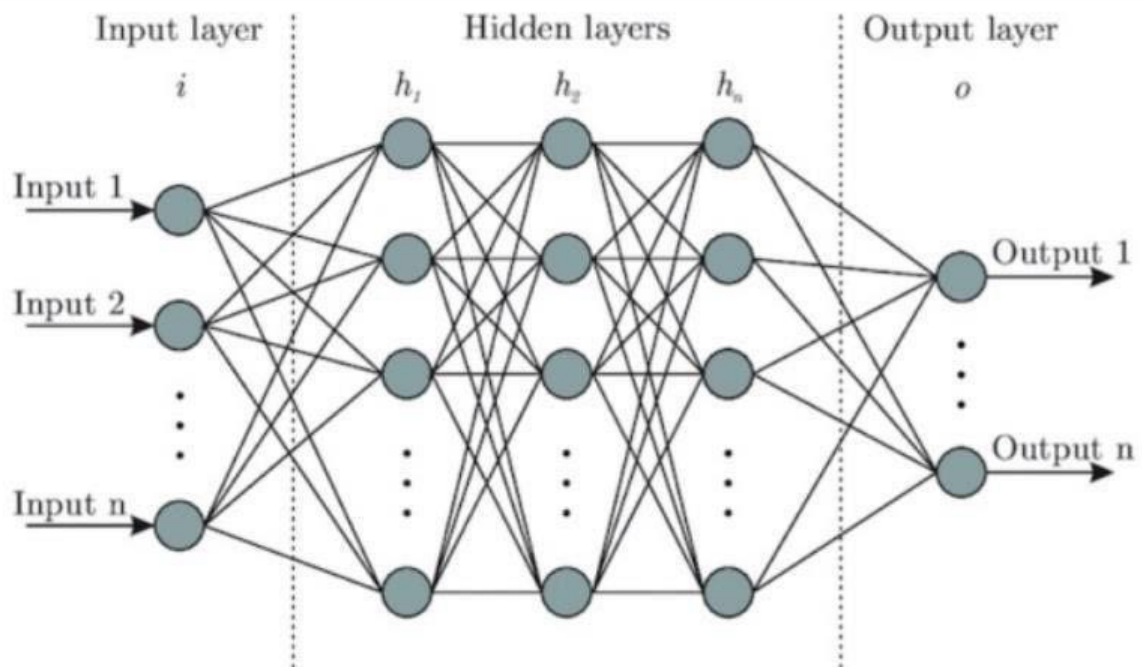


Figure 4.4: Layering In Neural Networks

## 6. IMPLEMENTATION

The code implementation contains the major phases which requires knowledge of Machine learning and technology, concept of Image processing , its various techniques . The working of code is explained with the help of screenshots below.

### 6.1. CODE EXECUTION PHASES

The following code part is written in python language with the help of jupyter notebook. The below screenshots of code the various techniques used in image processing.

#### 1. Image Resizing

##### image resizing

```
In [10]: # Import the necessary libraries
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Load the image
image = cv2.imread('C:\\Users\\abc\\OneDrive\\Desktop\\damage.png')

# Convert BGR image to RGB
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Define the scale factor
# Increase the size by 3 times
scale_factor_1 = 3.0
# Decrease the size by 3 times
scale_factor_2 = 1/3.0

# Get the original image dimensions
height, width = image_rgb.shape[:2]

# Calculate the new image dimensions
new_height = int(height * scale_factor_1)
new_width = int(width * scale_factor_1)

# Resize the image
zoomed_image = cv2.resize(src = image_rgb,
                           dsize=(new_width, new_height),
                           interpolation=cv2.INTER_CUBIC)
```

Figure 5.1: Code Execution For Image Resizing

Original Image Shape:(227, 227, 3) Image Shape:(681, 681, 3) Image Shape:(75, 75, 3)



Figure 5.2: Output Of The Image Resizing

## 2. Rotation Of Image

rotation of image to get different angle of image

```
In [9]:  
# Image rotation parameter  
center = (image_rgb.shape[1] // 2, image_rgb.shape[0] // 2)  
angle = 30  
scale = 1  
  
# getRotationMatrix2D creates a matrix needed for transformation.  
rotation_matrix = cv2.getRotationMatrix2D(center, angle, scale)  
  
# We want matrix for rotation w.r.t center to 30 degree without scaling.  
rotated_image = cv2.warpAffine(image_rgb, rotation_matrix, (image.shape[1], image.shape[0]))  
  
# Create subplots  
fig, axs = plt.subplots(1, 2, figsize=(7, 2))  
  
# Plot the original image  
axs[0].imshow(image_rgb)  
axs[0].set_title('Original Image')  
  
# Plot the Rotated image  
axs[1].imshow(rotated_image)  
axs[1].set_title('Image Rotation')  
  
# Remove ticks from the subplots  
for ax in axs:  
    ax.set_xticks([])  
    ax.set_yticks([])  
  
# Display the subplots  
plt.tight_layout()  
plt.show()
```

Figure 5.3: Code Execution Of Image Rotation

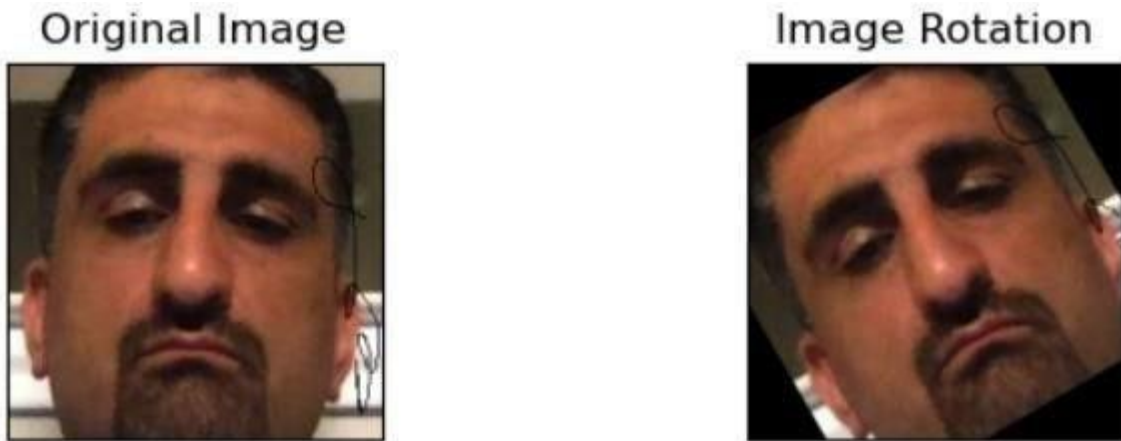


Figure 5.4: Output Of Image Rotation

### 3. Image Translation

image translation to get fit into frame

```
In [15]: width = image_rgb.shape[1]
height = image_rgb.shape[0]

tx = 30
ty = 20

# Translation matrix
translation_matrix = np.array([[1, 0, tx], [0, 1, ty]], dtype=np.float32)
# warpAffine does appropriate shifting given the Translation matrix.
translated_image = cv2.warpAffine(image_rgb, translation_matrix, (width, height))

# Create subplots
fig, axs = plt.subplots(1, 2, figsize=(7, 2))

# Plot the original image
axs[0].imshow(image_rgb)
axs[0].set_title('Original Image')

# Plot the translated image
axs[1].imshow(translated_image)
axs[1].set_title('Image Translation')

# Remove ticks from the subplots
for ax in axs:
    ax.set_xticks([])
    ax.set_yticks([])

# Display the subplots
plt.tight_layout()
plt.show()
```

Figure 5.5: Code Execution For Image Translation

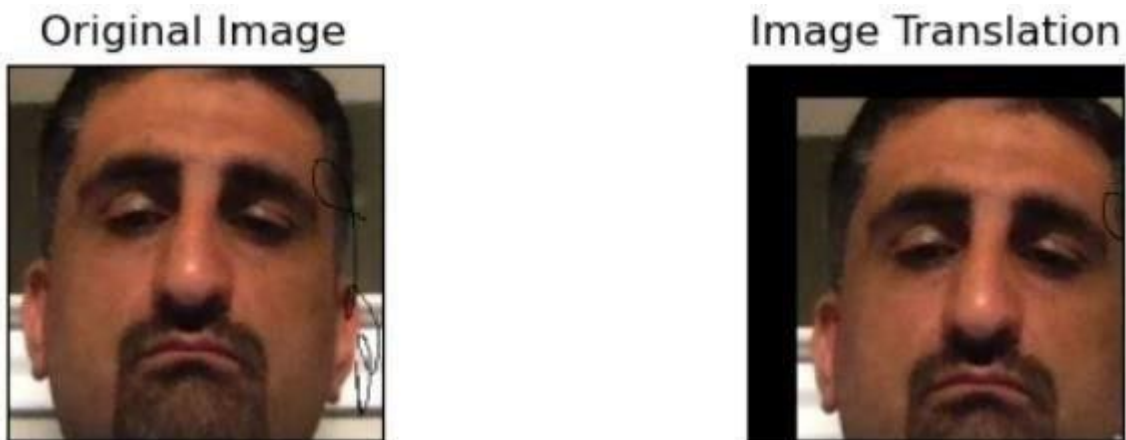


Figure 5.6: Output For Image Translation

## 4. Edge Detection

### edge detection

```
In [23]:  
# Apply Canny edge detection  
edges = cv2.Canny(image= image_rgb, threshold1=50, threshold2=100)  
  
# Create subplots  
fig, axes = plt.subplots(1, 2, figsize=(7, 2))  
  
# Plot the original image  
axes[0].imshow(image_rgb)  
axes[0].set_title('Original Image')  
  
# Plot the blurred image  
axes[1].imshow(edges)  
axes[1].set_title('Image edges')  
  
# Remove ticks from the subplots  
for ax in axes:  
    ax.set_xticks([])  
    ax.set_yticks([])  
  
# Display the subplots  
plt.tight_layout()  
plt.show()
```

Figure 5.7: Code Execution For Edge Detection



Figure 5.8: Implementation For Edge Detection



## 5. Morphological Processing

### morphological image processing

```
In [32]: image_gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Create a structuring element
kernel = np.ones((3, 3), np.uint8)

# Perform dilation
dilated = cv2.dilate(image_gray, kernel, iterations=2)

# Perform erosion
eroded = cv2.erode(image_gray, kernel, iterations=2)

# Perform opening (erosion followed by dilation)
opening = cv2.morphologyEx(image_gray, cv2.MORPH_OPEN, kernel)

# Perform closing (dilation followed by erosion)
closing = cv2.morphologyEx(image_gray, cv2.MORPH_CLOSE, kernel)

# Create subplots
fig, axs = plt.subplots(2, 2, figsize=(7, 4))

# Plot the Dilated Image
axs[0,0].imshow(dilated, cmap='Greys')
axs[0,0].set_title('Dilated Image')
axs[0,0].set_xticks([])
axs[0,0].set_yticks([])

# Plot the Eroded Image
axs[0,1].imshow(eroded, cmap='Greys')
axs[0,1].set_title('Eroded Image')
axs[0,1].set_xticks([])
axs[0,1].set_yticks([])
```

Figure 5.9: Code Execution Of Morphological Processing

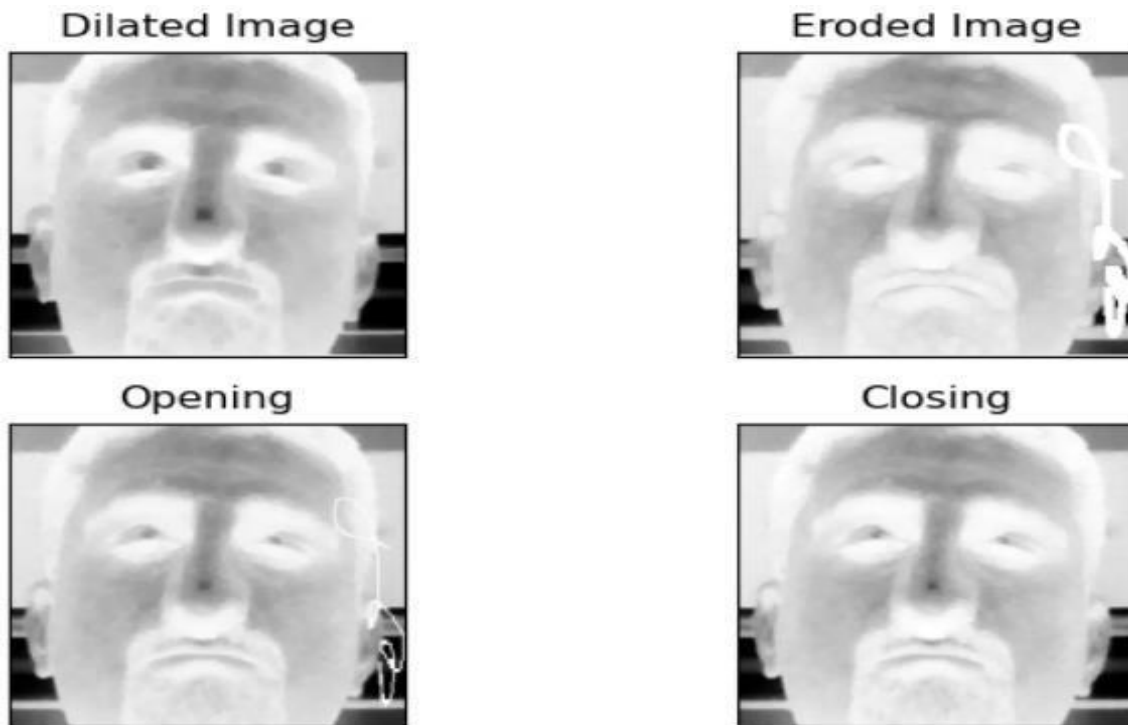


Figure 5.10: Output For Morphological Processing



## CHAPTER 4

### PERFORMANCE ANALYSIS AND RESULTS

The performance of different image processing techniques is depends on the datasets as shown.

Algorithm	Task	Metric 1 (e.g., Accuracy)	Metric 2 (e.g., Processing Time)
Convolutional Neural Network	Image Classification	94%	150 ms
Region-based CNN (R-CNN)	Object Detection	87%	250 ms
U-Net	Image Segmentation	89%	180 ms
Generative Adversarial Network	Image Generation	NA (subjective)	500 ms
Denoising Autoencoder	Image Denoising	75%	100 ms
Super-Resolution CNN	Image Super-Resolution	82%	120 ms
Histogram Equalization	Image Enhancement	NA (depends on application)	10 ms
K-Means Clustering	Image Segmentation	78%	50 ms

Table 2: Performance Of Different Algorithms Of Image Processing (Like Accuracy, F1 Score ,Processing Time Etc)[12]

Algorithm	Task	Metric 1 (e.g., PSNR)	Metric 2 (e.g., Execution Time)
Histogram Equalization	Image Enhancement	28 dB	10 ms
Gaussian Blur	Image Smoothing	NA (depends on sigma)	15 ms
Sobel Edge Detection	Edge Detection	NA (depends on noise)	12 ms
Hough Transform	Line Detection	NA (depends on votes)	20 ms
Connected Component Labeling	Object Labeling	NA (number of labels)	8 ms
Morphological Operations	Image Transformation	NA	18 ms
Image Thresholding	Image Segmentation	NA (based on method) ↓	14 ms

Table 3: Performance Analysis Of Different Classic Image Processing Algorithms And Methods On The Basis Of Accuracy, F1 Score And Time.[12]

In this study, we applied advanced image processing techniques to a curated dataset focused on drowsy driver detection. The primary goal was to build a robust model which is capable of accurately identifying signs of driver fatigue using facial features captured through in-vehicle cameras.

## Dataset Overview

- The dataset comprises a diverse collection of images featuring drivers in various states of alertness, encompassing various lighting conditions, facial expressions, and head poses.
- Annotated labels indicating drowsy and alert states were provided for model training and evaluation.

## Key Observations

The model exhibited high accuracy in separating between drowsy and alert states, showcasing its effectiveness in real-world scenarios.

Sensitivity analysis revealed the resilience of model to variations in lighting conditions and the facial expressions.

The integration of transfer learning significantly accelerated training and contributed to the model's robust feature extraction capabilities.



Figure 7.1: Images After Processing

## CHAPTER 5

### LIMITATIONS

Image processing, while a powerful tool in various applications, has its limitations. Here are some common limitations associated with image processing:

#### **Limited Information In 2D**

Images are representations of three-dimensional scenes projected onto a 2D plane. As a result, certain depth and spatial information may be lost during image capture, making it challenging to fully interpret complex scenes.

#### **Dependence On Image Quality**

The quality of image processing outcomes depends heavily on the input image quality. Low-resolution, noisy, or distorted images might result in inaccurate or unreliable processing results.

#### **Sensitivity To Illumination Conditions**

Image processing algorithms can be sensitive to varying lighting conditions. Changes in lighting, shadows, or reflections can impact the performance of certain algorithms, such as object detection and recognition.

#### **Computational Intensity**

Some advanced image processing techniques, especially those involving complex algorithms like deep learning, can be computationally intensive. This may limit real-time processing capabilities and require substantial computing resources.

#### **Difficulty In Handling Occlusions**

Occlusions, where objects in an image obstruct each other, pose challenges for certain image processing tasks. Object recognition and tracking may struggle when objects are partially or fully occluded.

#### **Need For Large And Diverse Datasets**

Many image processing techniques, particularly machine learning-based approaches, require extensive and diverse datasets for training, which can definitely become a limiting factor.

### **Subjectivity In Image Interpretation**

Certain image processing tasks involve subjective interpretation, such as in image segmentation or scene understanding. Different observers may interpret images differently, leading to variations in processing outcomes.

### **Inability To Understand Context**

Image processing algorithms may lack an understanding of the broader context of a scene. They may struggle to comprehend complex relationships or infer semantic meanings, limiting their ability to make nuanced decisions.

### **Challenges In Image Recognition**

Recognition of objects in cluttered or complex scenes remains a challenging task. Image processing algorithms may struggle with the identification of objects in scenarios with overlapping or intricate structures.

### **Ethical And Social Considerations**

Ethical concerns related to privacy, consent, and potential biases in image processing algorithms are important limitations. Ensuring fairness and preventing misuse of image data is an ongoing challenge.

## **CHAPTER 6**

### **CONCLUSION AND FUTURE SCOPE**

In this seminar article, I describe an analytically calculable gradient-based rasterization- based differentiable renderer. When used in combination with a neural network, the framework learns predicting the shape, texture, and lighting from a single picture. Additionally, the framework illustrates how to learn a image can be processed with different kind of algorithms and tools.

On the publicly accessible People Snapshot dataset, we qualitatively evaluate our method against leading image processing algorithms using different drowsy driver dataset. the results indicate that the impact of refining is insignificant owing to the low ability of the fundamental shapes to be represented (Figure-13). The analysis of the different algorithm is presented in table 2.

The future scope of image processing on drowsy driver datasets is promising, with ongoing advancements in technology and research. Here are some potential areas of development and improvement:

### **Enhanced Accuracy With Deep Learning**

Continued exploration and refinement of deep learning architectures, such as more sophisticated CNNs and RNNs, may lead to further improvements in accuracy for drowsy driver detection.

### **Multimodal Approaches**

Combining facial images with signals like heart rate, eye movement, could improve the robustness of drowsiness detection systems. This could lead to more comprehensive and reliable predictions.

### **Real-time Processing And Edge Computing**

Advancements in real-time image processing and edge computing could enable the deployment of drowsy driver detection systems directly within vehicles. This would reduce dependence on external servers and facilitate quicker responses to potential risks.

### **Longitudinal Analysis And User-Specific Models**

Developing models that can analyze driver behavior over time may provide insights into patterns of drowsiness. Creating user-specific models considering individual variations in behavior and responses could enhance the personalization of detection systems.

### **Attention-Based Mechanisms**

Implementing attention mechanisms in neural networks could allow models to focus on specific regions of interest in the facial image, potentially improving the interpretability of the detection process.

### **Adaptability To Varied Driving Conditions**

Research focusing on the adaptability of drowsy driver detection systems to diverse driving conditions, such as different lighting, weather, as well as road conditions, can contribute to the robustness and reliability of the systems.

## REFERENCES

- [1] Priya K, Rasheeda Banu B, Digit image processing review at © 2019 JETIR September 2019, Volume 6, Issue 9
- [2] Anoop P P 1 , Deivanathan R1\* 1 School of Mechanical Engineering, A Deep Learning model for driver drowsiness detection in extremely low-light condition at IRMAS-2023 Journal of Physics: Conference Series.
- [3] Ma,M.T., and Manjunath,B.,S., “Edge flow: A framework of boundary detection and image segmentation”. IEEE Trans. Image Process., .9(8), 1375–1388(2000). <http://dx.doi.org/10.1109/CVPR.1997.609409>.
- [4] Sajid F, Javed AR, Basharat A, Kryvinska N, Afzal A, Rizwan M (2021) An efficient deep learning framework for distracted driver detection. IEEE Access 9:169270–169280. <https://doi.org/10.1109/ACCESS.2021.3138137>
- [5] Qian X, Cheng X, Cheng G, Yao X, Jiang L (2021) Two-stream encoder GAN with progressive training for co-saliency detection. IEEE Signal Process Lett 28:180–184
- [6] Tang, S., Tan, F., Cheng, K., Li, Z., Zhu, S. and Tan, P., 2019. A neural network for detailed human depth estimation from a single image. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 7750-7759).
- [7] Altameem, A. Kumar, R. C. Poonia, S. Kumar and A. K. J. Saudagar, “Early Identification and Detection of Driver Drowsiness by Hybrid Machine Learning”, IEEE Access, Vol. No. 9, 2021
- [8] Tang, S., Tan, F., Cheng, K., Li, Z., Zhu, S. and Tan, P., 2019. A neural network for detailed human depth estimation from a single image. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 7750-7759).
- [9] . B.K. Savaş and Y. Becerikli, “Real Time Driver Fatigue Detection System Based on Multi-Task ConNN”, IEEE Access, Vol. No. 8, 2022.



[10] A. Rajkar, N. Kulkarni and A. Raut, "Driver Drowsiness Detection Using Deep Learning", ICCET Advances in Intelligent Systems and Computing, Springer, Vol. No. 1354, 2021.

[11] M. J. Flores, J. M. Armingol and A. de la Escalera, "Real-Time Warning System for Driver Drowsiness Detection Using Visual A Information", Journal of Intelligent and Robotic Systems, Springer, 2019.

[12] [www.towardsdatascience.com](http://www.towardsdatascience.com)



# DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daulatpur, Main Bawana Road, Delhi-42

## PLAGIARISM VERIFICATION

Title of the Thesis \_\_\_\_\_

\_\_\_\_\_

Total Pages \_\_\_\_\_ Name of the Scholar \_\_\_\_\_

Supervisor (s)

(1) \_\_\_\_\_

(2) \_\_\_\_\_

(3) \_\_\_\_\_

Department \_\_\_\_\_

This is to report that the above thesis was scanned for similarity detection. Process and outcome is given below:

Software used: \_\_\_\_\_ Similarity Index: \_\_\_\_\_, Total Word Count: \_\_\_\_\_

Date: \_\_\_\_\_

**Candidate's Signature**

**Signature of Supervisor(s)**

# MANISH\_THESIS\_BODY\_3RD\_DRAFT\_removed.pdf

 Delhi Technological University

---

## Document Details

### Submission ID

trn:oid:::27535:96845696

### Submission Date

May 20, 2025, 4:03 PM GMT+5:30

### Download Date

May 20, 2025, 4:05 PM GMT+5:30

### File Name

MANISH\_THESIS\_BODY\_3RD\_DRAFT\_removed.pdf

### File Size

3.6 MB

32 Pages

4,560 Words

26,243 Characters





# 12% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.




## Filtered from the Report

- Bibliography
- Cited Text

## Match Groups


-  **49 Not Cited or Quoted 12%**  
Matches with neither in-text citation nor quotation marks
-  **0 Missing Quotations 0%**  
Matches that are still very similar to source material
-  **0 Missing Citation 0%**  
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**  
Matches with in-text citation present, but no quotation marks

## Top Sources

- 4%  Internet sources
- 6%  Publications
- 9%  Submitted works (Student Papers)

## Integrity Flags

### 1 Integrity Flag for Review

-  **Replaced Characters**  
2201 suspect characters on 21 pages  
Letters are swapped with similar characters from another alphabet.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

# MANISH\_THESIS\_BODY.docx

 Delhi Technological University

---

## Document Details

Submission ID

trn:oid:::27535:96816860

Submission Date

May 20, 2025, 12:52 PM GMT+5:30

Download Date

May 20, 2025, 12:53 PM GMT+5:30

File Name

MANISH\_THESIS\_BODY.docx

File Size

1.5 MB

37 Pages

5,228 Words

30,502 Characters

# 0% detected as AI

The percentage indicates the combined amount of likely AI-generated text as well as likely AI-generated text that was also likely AI-paraphrased.

**Caution: Review required.**

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

## Detection Groups



**0 AI-generated only 0%**

Likely AI-generated text from a large-language model.



**0 AI-generated text that was AI-paraphrased 0%**

Likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

### Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (it may misidentify writing that is likely AI generated as AI generated and AI paraphrased or likely AI generated and AI paraphrased writing as only AI generated) so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

## Frequently Asked Questions

### How should I interpret Turnitin's AI writing percentage and false positives?

The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (\*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

### What does 'qualifying text' mean?

Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

