A project report on

DROWSINESS DETECTION SYSTEM USING IMAGE PROCESSING

Submitted in partial full fillment for the award of the degree of

Mtech Integrated Software Engineering

By

SIDDAMSETTY NARASIMHA MANOJ KUMAR (15MIS0183)

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology and Engineering

November,2019

DROWSINESS DETECTION SYSTEM USING IMAGE PROCESSING

Submitted in partial full fillment for the award of the degree of

Mtech Integrated Software Engineering

By

SIDDAMSETTY NARASIMHA MANOJ KUMAR (15MIS0183)

School of Information Technology and Engineering

November,2019

DECLARATION

I hereby declare that the thesis entitled "Drowsiness Detection System using Image processing " submitted by me, for the award of the degree of Specify the name of the degree VIT is a record of bonafide work carried out by me under the supervision of Assistant Prof. Rama Prabha K

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore

Date: 22nd Nov-2019

Signature of the Candidate

CERTIFICATE

This is to certify that the thesis entitled “Drowsiness Detection System using Image processing” submitted by SIDDAMSETTY NARASIMHA MANOJ KUMAR (15MIS0183) School of Information Technology and Engineering VIT, for the award of the degree of Name of the degree is a record of bonafide work carried out by him/her under my supervision.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The Project report fulfils the requirements and regulations of VIT and in my opinion meets the necessary standards for submission.

Signature of the Guide **Signature of the Hod**

Internal Examiner **External Examiner**

ABSTRACT

Recent time's major accidents occurred due to the sleepiness in drivers during driving on highways. The recent survey says that the number of accidents occurs due to the fatigue and unconscious state of the drivers in worldwide. No one can watch the driver at any time. In order to overcome that the proposed drowsiness detection system for drivers it is introduced in which the camera is used to monitor the face and eyes of the driver while driving. So, the video camera will capture the image of the driver if the person feels like sleepy or fainting. When the eyes of the driver are drowsed the image will automatically capture by camera by calculating the energy of each frame value one can determines whether the eyes or closed or not if the eyes are closed for certain time it intimate to the controller to display the message. In this project focusing on the video camera accessed by the MATLAB and the face of the user in real time is captured by every second is processed and stores in the database. Next the database checks with the SVM dataset that we are inserted into the project to train the system. Later the position of eyes is calculated and system verifies with the dimensions of the eyes, mouth. If the dimensions and energy values of the user is matched by the dataset we are dumped, it will display whether user feeling the drowsiness or not. If the user is in the drowsiness system warns the user and alert him/her to drive safely.

ACKNOWLEDGEMENT

It is my pleasure to express with deep sense of gratitude to Assistant Prof. Rama Prabha KP , Assistant Professor, School of Information Technology and Engineering, Vellore Institute of Technology, for his/her constant guidance, continual encouragement, understanding; more than all, he taught me patience in my endeavour. My association with him / her is not confined to academics only, but it is a great opportunity on my part of work with an intellectual and expert in the field of Department of Software and Systems Engineering.

I would like to express my gratitude to Dr. G Vishwanthan, Dr. Shekar Vishwanthan, Dr. Sankar Vishwanthan, Dr. Selvam Vishwanthan, Dr. Anand A Samuel and Bala Krushna Tripathy SITE for providing with an environment to work in and for his inspiration during the tenure of the course.

In jubilant mood I express ingeniously my whole-hearted thanks to Prof. Sree Dharinya S. Program Chair and Associate Professor, all teaching staff and members working as limbs of our university for their not-self-centred enthusiasm coupled with timely encouragements showered on me with zeal, which prompted the acquirement of the requisite knowledge to finalize my course study successfully. I would like to thank my parents for their support.

It is indeed a pleasure to thank my friends who persuaded and encouraged me to take up and complete this task. At last but not least, I express my gratitude and appreciation to all those who have helped me directly or indirectly toward the successful completion of this project.

Place: Vellore

Date: 22nd Nov-2019 **SN. MANOJ KUMAR**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATION

HMM- Hidden Markov model

EI- Eye Index

PA- Pupil Activity

SVM- Support Vector Machine

PLSR- Partial Least square Regression

PDF- Probability Density Function

GMS- Global Message Service

GPS- Global Positioning system

CHAPTER I

INTRODUCTION

1.1 GENERAL

The term digital image refers to processing of a two dimensional picture by a digital computer. In a broader context, it implies digital processing of any two dimensional data. A digital image is an array of real or complex numbers represented by a finite number of bits. An image given in the form of a transparency, slide, photograph or an X-ray is first digitized and stored as a matrix of binary digits in computer memory. This digitized image can then be processed and/or displayed on a high-resolution television monitor. For display, the image is stored in a rapid-access buffer memory, which refreshes the monitor at a rate of 25 frames per second to produce a visually continuous display.

1.1.1 THE IMAGE PROCESSING SYSTEM

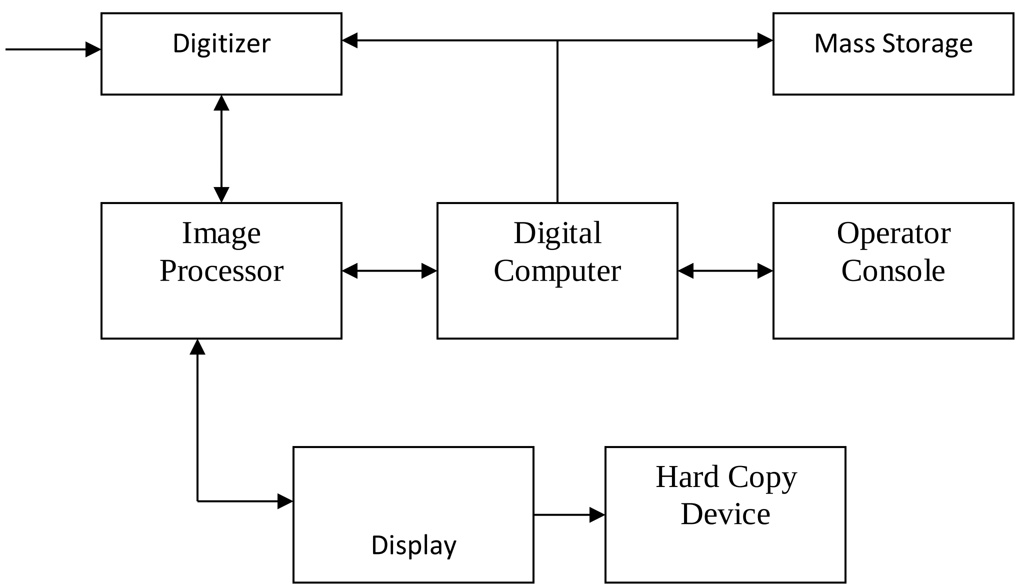


FIG 1.1 BLOCK DIAGRAM FOR IMAGE PROCESSING SYSTEM

DIGITIZER:

A digitizer converts an image into a numerical representation suitable for input into a digital computer. Some common digitizers are

1. Microdensitometer
2. Flying spot scanner
3. Image dissector
4. Videocon camera
5. Photosensitive solid- state arrays.

IMAGE PROCESSOR:

An image processor does the functions of image acquisition, storage, pre-processing, segmentation, representation, recognition and interpretation and finally displays or records the resulting image. The following block diagram gives the fundamental sequence involved in an image processing system

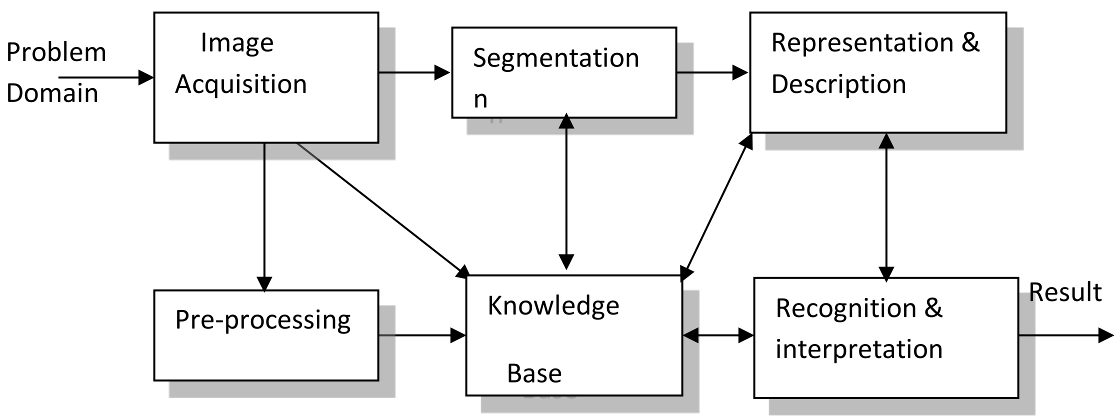


FIG 1.2 BLOCK DIAGRAM OF FUNDAMENTAL SEQUENCE INVOLVED IN AN IMAGE PROCESSING SYSTEM

As detailed in the diagram, the first step in the process is image acquisition by an imaging sensor in conjunction with a digitizer to digitize the image. The next step is the preprocessing step where the image is improved being fed as an input to the other processes. Pre-processing typically deals with enhancing, removing noise, isolating regions, etc. Segmentation partitions an image into its constituent parts or objects. The output of segmentation is usually raw pixel data, which consists of either the boundary of the region or the pixels in the region themselves. Representation is the process of transforming the raw pixel data into a form useful for subsequent processing by the computer. Description deals with extracting features that are basic in differentiating one class of objects from another. Recognition assigns a label to an object based on the information provided by its descriptors. Interpretation involves assigning meaning to an ensemble of recognized objects. The knowledge about a problem domain is incorporated into the knowledge base. The knowledge base guides the operation of each processing module and also controls the interaction between the modules. Not all modules need be necessarily present for a specific function. The composition of the image processing system depends on its application. The frame rate of the image processor is normally around 25 frames per second.

DIGITAL COMPUTER:

Mathematical processing of the digitized image such as convolution, averaging, addition, subtraction, etc. are done by the computer.

MASS STORAGE:

The secondary storage devices normally used are floppy disks, CD ROMs etc.

HARD COPY DEVICE:

The hard copy device is used to produce a permanent copy of the image and for the storage of the software involved.

OPERATOR CONSOLE:

The operator console consists of equipment and arrangements for verification of intermediate results and for alterations in the software as and when require. The operator is also capable of checking for any resulting errors and for the entry of requisite data.

1.1.2 IMAGE PROCESSING FUNDAMENTAL:

Digital image processing refers processing of the image in digital form. Modern cameras may directly take the image in digital form but generally images are originated in optical form. They are captured by video cameras and digitalized. The digitalization process includes sampling, quantization. Then these images are processed by the five fundamental processes, at least any one of them, not necessarily all of them.

IMAGE PROCESSING TECHNIQUES:

This section gives various image processing techniques.

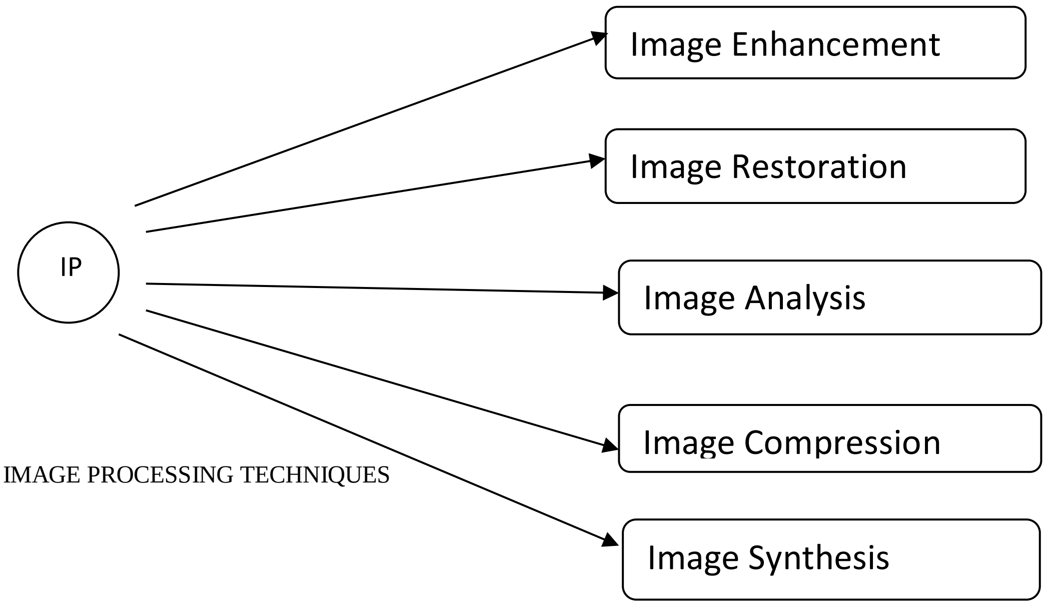


FIG1.3: IMAGE PROCESSING TECHNIQUES

IMAGE ENHANCEMENT:

Image enhancement operations improve the qualities of an image like improving the image's contrast and brightness characteristics, reducing its noise content, or sharpen the details. This just enhances the image and reveals the same information in more understandable image. It does not add any information to it.

IMAGE RESTORATION:

Image restoration like enhancement improves the qualities of image but all the operations are mainly based on known, measured, or degradations of the original image. Image restorations are used to restore images with problems such as geometric distortion, improper focus, repetitive noise, and camera motion. It is used to correct images for known degradations.

IMAGE ANALYSIS:

Image analysis operations produce numerical or graphical information based on characteristics of the original image. They break into objects and then classify them. They depend on the image statistics. Common operations are extraction and description of scene and image features, automated measurements, and object classification. Image analyze are mainly used in machine vision applications.

IMAGE COMPRESSION:

Image compression and decompression reduce the data content necessary to describe the image. Most of the images contain lot of redundant information, compression removes all the redundancies. Because of the compression the size is reduced, so efficiently stored or transported. The compressed image is decompressed when displayed. Lossless compression preserves the exact data in the original image, but Lossy compression does not represent the original image but provide excellent compression.

IMAGE SYNTHESIS:

Image synthesis operations create images from other images or non-image data. Image synthesis operations generally create images that are either physically impossible or impractical to acquire.

APPLICATIONS OF DIGITAL IMAGE PROCESSING:

Digital image processing has a broad spectrum of applications, such as remote sensing via satellites and other space crafts, image transmission and storage for business applications, medical processing, radar, sonar and acoustic image processing, robotics and automated inspection of industrial parts.

MEDICAL APPLICATIONS:

In medical applications, one is concerned with processing of chest X-rays, cineangiograms, projection images of transaxial tomography and other medical images that occur in radiology, nuclear magnetic resonance (NMR) and ultrasonic scanning. These images may be used for patient screening and monitoring or for detection of tumors' or other disease in patients.

SATELLITE IMAGING:

Images acquired by satellites are useful in tracking of earth resources; geographical mapping; prediction of agricultural crops, urban growth and weather; flood and fire control; and many other environmental applications. Space image applications include recognition and analysis of objects contained in image obtained from deep space-probe missions.

COMMUNICATION:

Image transmission and storage applications occur in broadcast television, teleconferencing, and transmission of facsimile images for office automation, communication of computer networks, closed-circuit television based security monitoring systems and in military communications.

RADAR IMAGING SYSTEMS:

Radar and sonar images are used for detection and recognition of various types of targets or in guidance and manoeuvring of aircraft or missile systems.

DOCUMENT PROCESSING:

It is used in scanning, and transmission for converting paper documents to a digital image form, compressing the image, and storing it on magnetic tape. It is also used in document reading for automatically detecting and recognizing printed characteristics.

DEFENSE/INTELLIGENCE:

It is used in reconnaissance photo-interpretation for automatic interpretation of earth satellite imagery to look for sensitive targets or military threats and target acquisition and guidance for recognizing and tracking targets in real-time smart-bomb and missile-guidance systems.

1.2 OBJECTIVE:

Once acquiring the face file of the driver's image, it is converted into consecutive frames of images. Based on the algorithm, it has been applied to detect the face portion in the image. Energy value of each frame is calculated and it is used to differentiate between the open and closed eyes. If the eyes are closed then the message box will display.

1.3 EXISTING SYSTEM

Vehicle Speed Control in school Zone and also controlling the speed of the vehicle in different zones such as bridges, highways, cities and suburbs. Alcohol detection to detect drunken driving and eye Movement Detection to detect the Sleeping of the Driver. . If our vehicle crossing the school the horn switch was activate, the horn won't make sound before crossing the school area and this same method is applicable for hospital and temple areas. In Short we are using an Eye Blink and the Alcohol detection for Driver Drowsy System finally an Accident Alert system was added by using Vibration sensor if the sensor got a Hit. It considered as accident. GSM sends a SMS to the Police and ambulance along with the GPS Location and for monitoring the image processing is used which has the stored database images.

1.3.1 EXISITING SYSTEM DISADVANTAGES:

- Cost of hardware implementation is very high.
- Only the database stored images can be monitored.

1.3.2 LITERATURE SURVEY:

1. **Drowsy Driver Warning System Using Image Processing1 Singh Himani Parmar, 2Mehul Jajal, 3Yadav Priyanka Brijbhan Electronics & Communication, GEC, Bharuch, Gujarat**

Driver in-alertness is an important cause for most accident related to the vehicles crashes. Driver fatigue resulting from sleep deprivation or sleep disorders is an important factor in the increasing number of the accidents on today's roads. Drowsy driver warning system can form the basis of the system to possibly reduce the accidents related to driver's drowsiness. The purpose of such a system is to perform detection of driver fatigue. By placing the camera inside the car, we can monitor the face of the driver and look for the eye-movements which indicate that the driver is no longer in condition to driver is no longer in condition to drive. In such a case, a warning signal should be issued. This paper describes how to find and track the eyes. We also describe a method that can determine if the eyes are open or closed. The main criterion of this system is that it must be highly non-intrusive and it should start when the ignition is turned on without having at the driver initiate the system. Nor should the driver be responsible for providing any feedback to the system. The system must also operate regardless of the texture and the color of the face. It must also be able to handle diverse condition such as changes in light, shadows, reflections etc. In given paper a drowsy driver warning system using image processing as well as accelerometer is proposed.

2. Detecting Driver Drowsiness Based on Sensors: A ReviewArun SahayadhasVels University

In recent years, driver drowsiness has been one of the major causes of road accidents and can lead to severe physical injuries, deaths and significant economic losses. Statistics indicate the need of a reliable driver drowsiness detection system which could alert the driver before a mishap happens. Researchers have attempted to determine driver drowsiness using the following measures: (1) vehicle-based measures; (2) behavioral measures and (3) physiological measures. A detailed review on these measures will provide insight on the present systems, issues associated with them and the enhancements that need to be done to make a robust system. In this paper, we review these three measures as to the sensors used and discuss the advantages and limitations of each. The various ways through which drowsiness has been experimentally manipulated is also discussed. We conclude that by designing a hybrid drowsiness detection system that combines non-intusive physiological measures with other measures one would accurately determine the drowsiness level of a driver. A number of road accidents might then be avoided if an alert is sent to a driver that is deemed drowsy.

3. Driver's eye blinking detection using novel color and texture segmentation algorithmsA. Lenskiy Jong-Soo Lee

In this paper we propose a system that measures eye blinking rate and eye closure duration. The system consists of skin-color segmentation, facial features segmentation, iris positioning and blink detection. The proposed skin-segmentation procedure is based on a neural network approximation of a RGB skin-color histogram. This method is robust and adaptive to any skin-color training set. The largest remaining skin-color region among skin-color segmentation results is further segmented into open/closed eyes, lips, nose, eyebrows, and the remaining facial regions using a novel texture segmentation algorithm. The segmentation algorithm classifies pixels according to the highest probability among the estimated facial feature class probability density functions (PDFs). The segmented eye regions are analyzed with the Circular Hough transform with the purpose of finding iris candidates. The finial iris position is selected according to the location of the maximum correlation value obtained from correlation with a predefined mask. The positions of irises and eye states are monitored through time to estimate eye blinking frequency and eye closure duration. The method of the driver drowsiness detection using these parameters is illustrated. The proposed system is tested on CCD and CMOS cameras under different environmental conditions and the experimental results show high system performance.

4.A Vision Based System for Monitoring the Loss of Attention in Automotive Drivers Anirban Dasgupta, Anjith George, S L Happy and Aurobinda Routray, Member,

On-board monitoring of the alertness level of an automotive driver has been challenging to research in transportation safety and management. In this paper, we propose a robust real-time embedded platform to monitor the loss of attention of the driver during day and night driving conditions. The percentage of eye closure has been used to indicate the alertness level. In this approach, the face is detected using Haar-like features and is tracked using a Kalman filter. The eyes are detected using principal component analysis during daytime and using the block local-binary-pattern features during night-time. Finally, the eye state is classified as open or closed using support vector machines. In-plane and off-plane rotations of the driver's face have been compensated using affine transformation and perspective transformation, respectively. Compensation in illumination variation is carried out using histogram equalization. The algorithm has been cross-validated using brain signals and, finally, has been implemented on a single-board computer that has an Intel Atom processor with a 1.66-GHz clock, a random access memory of 1 GB, ×86 architecture, and a Windows-embedded XP operating system. The system is found to be robust under actual driving conditions.

5.Drowsy Driver Identification Using MATLAB Video Processing Mrs.S. Dhanalakshmi1, J.Jasmine Rosepet2, G.Leema Rosy3, M.Philominal4 1,2,3,4Idhaya Engineering College for Women, Chinnasalem, India.

Drivers who do not take regular breaks when driving long distances run a high risk of becoming drowsy and cause accidents. It is astate which they often fail to recognize early enough according to the experts. Studies show that around one quarter of all serious motorway accidents is attributable to sleepy drivers in need of a rest, meaning that drowsiness causes more road accidents than drink-driving. Driver fatigue is a significant factor in a large number of vehicle accidents. The development of technologies for detecting drowsiness at the wheel is a major challenge in the field of accident avoidance systems. Because of the hazard that drowsiness presents on the road, methods need to be developed for counteracting its affects. The main aim of this is to develop a drowsiness detection system by monitoring the eyes and mouth; it is believed that the symptoms of driver fatigue can be detected early enough to avoid a car accident. Detection of fatigue involves the observation of eye movements, blink patterns and mouth opening for yawning. The analysis of face images is a popular research area with applications such as face recognition, and human identification security systems. This project is focused on the localization of the eyes, which involves looking at the entire image of the eye, and determining the position of the eyes, by a self-developed image-processing algorithm.

6.Drowsiness Detection system using image frequency

Drowsy Driver Detection System has been developed using a non-intrusive machine vision based concepts. The system uses a small monochrome security camera that points directly towards the driver's face and monitors the driver's eyes in order to detect fatigue. In such a case when fatigue is detected, a warning signal is issued to alert the driver. This report describes how to find the eyes, and also how to determine if the eyes are open or closed. The algorithm developed is unique to any currently published papers, which was a primary objective of the project. The system deals with using information obtained for the binary version of the image to find the edges of the face, which narrows the area of where the eyes may exist. Once the face area is found, the eyes are found by computing the horizontal averages in the area. Taking into account the knowledge that eye regions in the face present great intensity changes, the eyes are located by finding the significant intensity changes in the face. Once the eyes are located, measuring the distances between the intensity changes in the eye area determine whether the eyes are open or closed. A large distance corresponds to eye closure. If the eyes are found closed for 5 consecutive frames, the system draws the conclusion that the driver is falling asleep and issues a warning signal. The system is also able to detect when the eyes cannot be found, and works under reasonable lighting conditions.

In 2008, Hong Su et. al. [15] described 'A Partial Least Squares Regression-Based Fusion Model for Predicting the Trend in Drowsiness'. They proposed a new technique of modeling driver drowsiness with multiple eyelid movement features based on an information fusion technique—partial least squares regression (PLSR), with which to cope with the problem of strong collinear relations among eyelid movement features and, thus, predicting the tendency of the drowsiness. The predictive precision and robustness of the model thus established are validated, which show that it provides a novel way of fusing multi-features together for enhancing our capability of detecting and predicting the state of drowsiness.

In June, 2010, Bin Yang et. al. described **‘Camera- based Drowsiness Reference for Driver State Classification under Real Driving Conditions’**. They proposed that measures of the driver’s eyes are capable to detect drowsiness under simulator or experiment conditions. The performance of the latest eye tracking based in-vehicle fatigue prediction measures are evaluated. These measures are assessed statistically and by a classification method based on a large dataset of 90 hours of real road drives. The results show that eye-tracking drowsiness detection works well for some drivers as long as the blinks detection works properly. Even with some proposed improvements, however, there are still problems with bad light conditions and for persons wearing glasses. As a summary, the camera based sleepiness measures provide a valuable contribution for a drows iness reference, but are not reliable enough to be the only reference.

In 2011, M.J. Flores et. al. described **‘Driver drowsiness detection system under infrared illumination for an intelligent vehicle’**. They proposed that to reduce the amount of such fatalities, a module for an advanced driver assistance system, which caters for automatic driver drowsiness detection and also driver distraction, is presented. Artificial intelligence algorithms are used to process the visual information in order to locate, track and analyze both the driver’s face and eyes to compute the drowsiness and distraction indexes. This real-time system works during nocturnal conditions as a result of a near-infrared lighting system. Finally, examples of different driver images taken in a real vehicle at nighttime are shown to validate the proposed algorithms.

In June, 2012, A. Cheng et. al. [18] described **'Driver Drowsiness Recognition Based on Computer Vision Technology’**. They presented a nonintrusive drowsiness recognition method using eye-tracking and image processing. A robust eye detection algorithm is introduced to address the problems caused by changes in illumination and driver posture. Six measures are calculated with percentage of eyelid closure, maximum closure duration , blink frequency, average opening level of the eyes, opening velocity of the eyes, and closing velocity of the eyes. These measures are combined using Fisher's linear discriminated functions using a stepwise method to reduce the correlations and extract an independent index. Results with six participants in driving simulator experiments demonstrate the feasibility of this video-based drowsiness recognition method that provided 86% accuracy.

In 2013, G. Kong et. al. described **'Visual Analysis of Eye State and Head Pose for Driver Alertness Monitoring'**. They presented visual analysis of eye state and head pose (HP) for continuous monitoring of alertness of a vehicle driver. Most existing approaches to visual detection of non-alert driving patterns rely either on eye closure or head nodding angles to determine the driver drowsiness or distraction level. The proposed scheme uses visual features such as eye index (EI), pupil activity (PA), and HP to extract critical information on non -alertness of a vehicle driver. A support vector machine (SVM) classifies a sequence of video segments into alert or non -alert driving events. Experimental results show that the proposed scheme offers high classification accuracy with acceptably low errors and false alarms for people of various ethnicity and gender in real road driving conditions.

In June, 2014, Eyosiyas et. al. described **'Driver Drowsiness Detection through \HMM based Dynamic Modeling'**. They proposed a new method of analyzing the facial expression of the driver through Hidden Markov Model (HMM) based dynamic modeling to detect drowsiness. They have implemented the algorithm using a simulated driving setup. Experimental results verified the effectiveness of the proposed method.

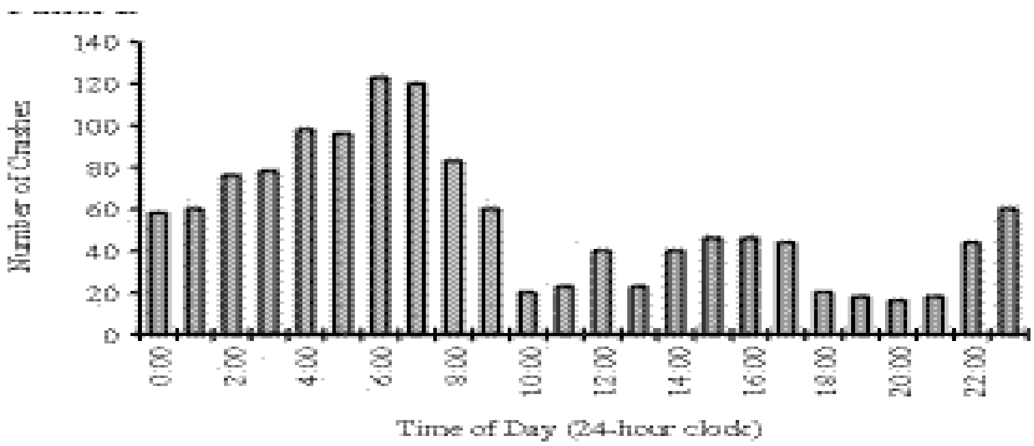
In August 2014, García et. al. described **'Driver Monitoring Based on Low-Cost 3-D Sensors'**. They proposed a solution for driver monitoring and event detection based on 3-D information from a range camera is presented. The system combines 2-D and 3-D techniques to provide head pos e estimation and regions -of-interest identification. Based on the captured cloud of 3-D points from the sensor and analyzing the 2-D projection, the points corresponding to the head are determined and extracted for further analysis. Later, head pose estimation with three degrees of freedom (Euler angles) is estimated based on the iterative closest points algorithm. Finally, relevant regions of the face are identified and used for further analysis, e.g., event detection and behavior analysis. The resulting application is a 3-D driver monitoring systembased on low-cost sensors. It represents an interesting tool for human factor research studies, allowing automatic study of specific factors and the detection of special event related to the driver, e.g., driver drowsiness, inattention, or head pose.

Motivation (1):

- A study (In the U.S) showed that 37% of drivers surveyed admitted to falling asleep at the wheel.
- An estimated 1.35 million drivers have been involved in a drowsy driving related crash in the past five years.
- Fall-asleep crashes are likely to be serious. The morbidity and mortality associated with drowsy-driving crashes are high, perhaps because of the higher speeds involved (Horne, Reyner, 1995b) combined with delayed reaction time.

Motivation (2):

Accidents study in the U.S (1990-92)

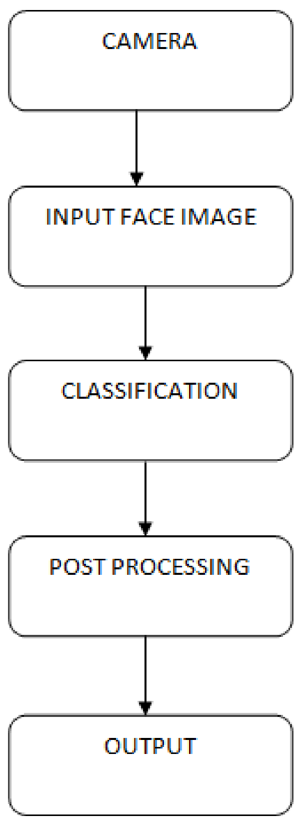


- Time of occurrence of crashes in drivers at ages 26 to 45 in which the crashes were attributed by the police to the driver being asleep (but in which alcohol was not judged to be involved).
- The X axis is the time of day and the Y axis is the number of crashes.

1.4 PROPOSED SYSTEM:

Once acquiring the video file of the driver's image, it is converted into consecutive frames of images. Based on the algorithm, it has been applied to detect the face portion in the image. Energy value of each frame is calculated and it is used to differentiate between the open and closed eyes. If the eyes are closed then the alarm will blow.

BLOCK DIAGRAM FOR GENERAL IMAGE PROCESSING:



BLOCK DIAGRAM FOR PROPOSED SYSTEM:

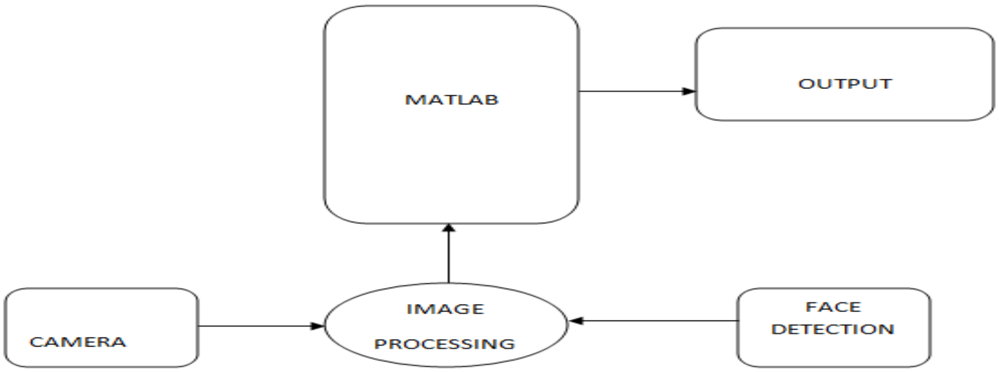


Fig1.4: Block Diagram of the proposed system

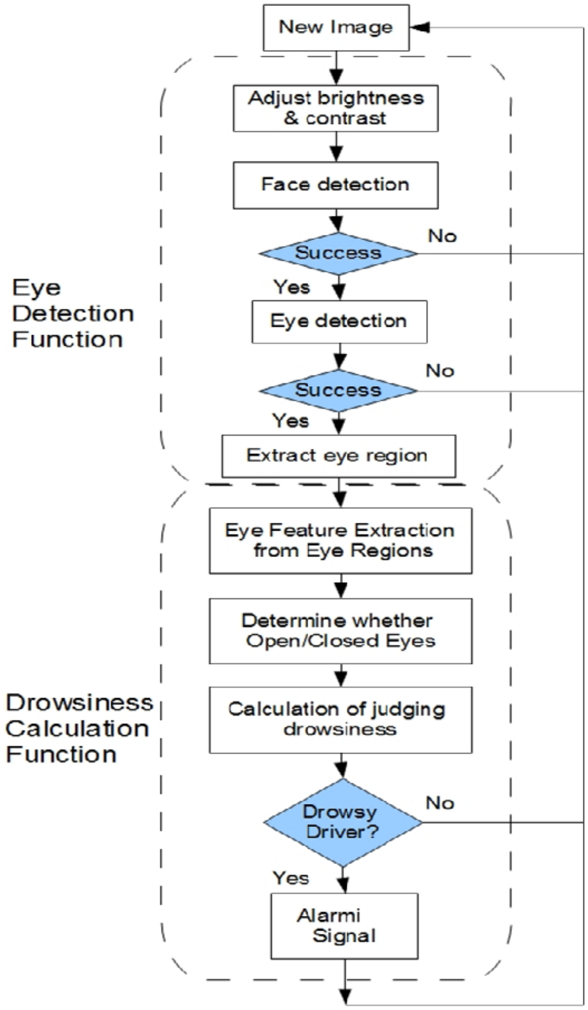


Fig 1.5 Flowchart for the Proposed System

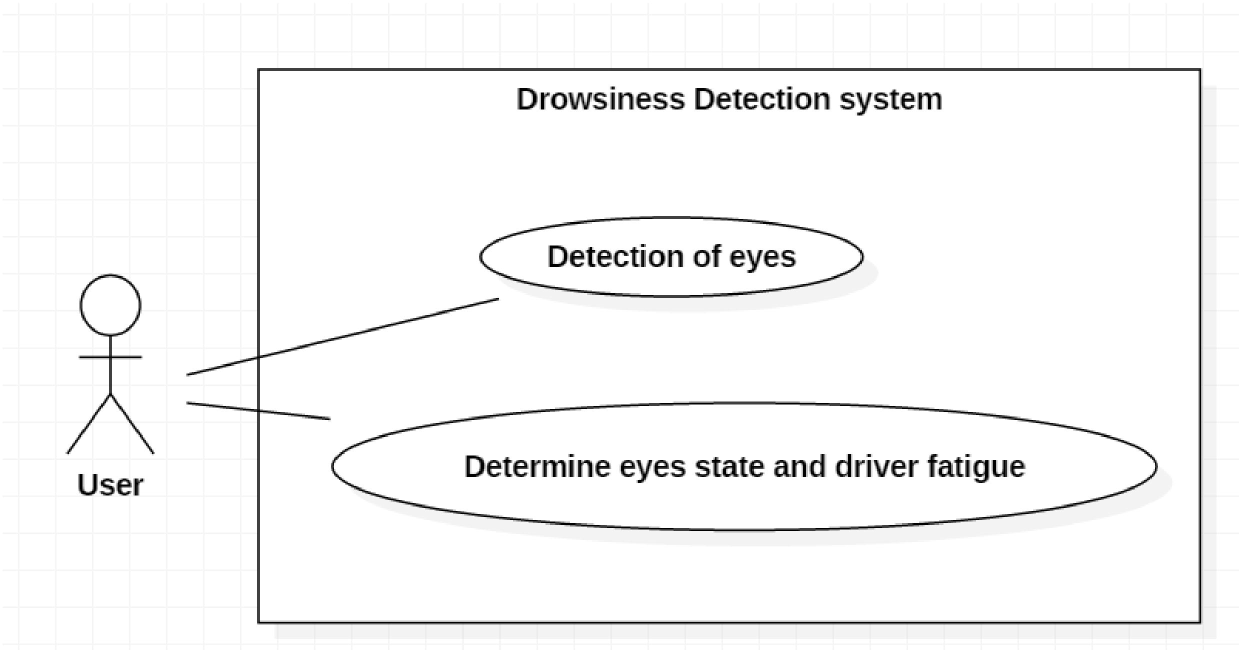


Fig 1.6 Use case for proposed system

1.4.1 ADVANTAGES

- The concept will be done in real time whereas camera is used to detect the face of the person.
- Provide high accuracy.

2. PROJECT DESCRIPTION

2.1 INTRODUCTION

Majority of the accidents caused today by cars are mainly due to the driver fatigue. Driving for a long period of time causes excessive fatigue and tiredness which in turn makes the driver sleepy or loose awareness. With the rapid increase in the number of accidents seems to be increasing day to day. Therefore a need arises to design a system that keeps the driver focused on the road. The aim of this paper is to develop a prototype of drowsy driver warning system. Our whole focus and concentration will be placed on designing the system that will accurately monitor the open and closed state of the driver's eye in real time. By constantly monitoring the eyes, it can be seen that the symptoms of driver fatigue can be detected early enough to avoid an accident. This detection can be done using a sequence of images of eyes as well as face and head movement. The observation of eye movements and its edges for the detection will be used. Devices to detect when drivers are falling asleep and to provide warnings to alert them of the risk, or even control the vehicle's movement, have been the subject to much research and development. Driver fatigue is a serious problem resulting in many thousands of road accidents each year. It is not currently possible to calculate the exact number of sleep related accidents because of the difficulties in detecting whether fatigue was a factor and in assessing the level of fatigue. However research suggests that up to 25% of accidents on monotonous roads in India are fatigue related. Research in other countries also indicates that driver fatigue is a serious problem.. These techniques are based on the model of retina and visual cortex V1. We then introduce a new feature extraction method to estimate the state of eye, mouth and head in a sequence of images. This technique uses the energy of static information in an image. To improve this process, a sharpening filter is added to the system. Finally, a robust decision algorithm is proposed. This algorithm detects the driver's drowsiness based on various possible combinations of extracted features.

MODULE 1:

2.2 INPUT FACE ACQUISITION

We track the eye by looking for the darkest pixel in the predicted region. In order to recover from tracking errors, we make sure that none of the geometrical constraints are violated. If they are, we relocalize the eyes in the next frame. To find the best match for the eye template, we initially centre it at the darkest pixel, and then perform a gradient descent in order to find a local minimum.

MODULE 2:

2.3 PREPROCESSING

The preprocessing operations include the binarization of a facial image to increase the processing speed and conserve memory capacity and noise removal. The image processor developed for this drowsiness warning system performs the expansion and contraction operation on the white pixels and processing for noise removal is performed on the small black pixels of the facial images. After the binarization, the noise removal procedure involves an expansion processing method combined with the use of a median filter. These preprocessing operations are sufficient to support detection of the vertical positions of the eyes. However, following identification of the eye positions, the size of the eyes must be converted back to the original image format at the time the degree of eye openness is output. To facilitate that, data contraction is performed in the latter stage of preprocessing

MODULE 3:

2.4 CLASSIFICATION:

The maximum width of the driver’s face must be detected in order to determine the lateral positions of the areas in which the eyes are present. Face width is detected by judging the continuity of white pixels and the pattern of change in pixel number. On that basis, the outer edges of the face are recognized and determined.Each vertical eye position is detected independently within an area demarcated by the center line of the face, which is found from the face width, and straight lines running through the right and left outer edges of the face. In a binary image, the eye becomes collection of black pixels, along with the eyebrows, nostrils, mouth and other facial features. These collections of black pixels are recognized on the basis of a labeling operation, and the position of each eye is extracted by judging the area of each label along with its aspect ratio and relative coordinate positions in the facial image. Through this process of detecting each vertical eye position, the central coordinates of each eye are recognized. The coordinates serve as references for defining the areas of eye presence.

MODULE 4:

2.5 POST PROCESSING

A function for tracking the positions of the eye is an important capability for achieving high-speed processing because it eliminates the need to process every frame in order to detect each eye position from the entire facial image. This function consists of a subordinate for updating the areas of eye presence and recognizing when tracking becomes impossible. The basic concept of eye tracking is to update the area of eye presence, in which an eye search is made in the following frame, according to the central coordinates of the eye in the previous frame. The updating process involves defining an arc of eye presence on the basis at the point of intersection of center lines running through the Feret's diameter of the detected eye. The area thus becomes the area of eye presence in which the system searches for the eye in the image data of the next frame. This process of using information on eye position to define the eye position for obtaining the next facial image data makes it possible to track the position of the eye. As it is clear from this description, the size of the area of eye position changes. If the eyes are tracked correctly, their degree of openness will always vary within certain specified range for each individual driver. Consequently, if the value found by the system falls outside the range, it judges that the eyes are not being tracked correctly. The process of detecting the position of each eye from the entire facial image is then executed once more and the Judgment is done whether the eye are open/closed.

3. SOFTWARE SPECIFICATION

3.1 GENERAL

This paper proposes a novel nonridged inter-subject multichannel image registration method which combines information from different modalities/channels to produce a unified joint registration. Multichannel images are created using co-registered multimodality images of the same subject to utilize information across modalities comprehensively. Contrary to the existing methods which combine the information at the image/intensity level, the proposed method uses feature-level information fusion method to spatio-adaptively combine the complementary information from different modalities that characterize different tissue types, through Gabor wavelets transformation and Independent Component Analysis (ICA), to produce a robust inter-subject registration.

3.2 SOFTWARE REQUIREMENTS

- MATLAB 7.14 Version

MATLAB

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation.

Typical uses include:

- Math and computation
- Algorithm development
- Modelling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics.

- Application development, including Graphical User Interface building

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar non-interactive language such as C or FORTRAN

3.3 INTRODUCTION

MATLAB (**mat**rix **lab**oratory) is a numerical computing environment and fourth-generation programming language. Developed by Math Works, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, and Fortran.

In 2004, MATLAB had around one million users across industry and academia. MATLAB users come from various backgrounds of engineering, science, and economics. MATLAB is widely used in academic and research institutions as well as industrial enterprises.

MATLAB was first adopted by researchers and practitioners in control engineering, Little's specialty, but quickly spread to many other domains. It is now also used in education, in particular the teaching of linear algebra and numerical analysis, and is popular amongst scientists involved in image processing. The MATLAB application is built around the MATLAB language.

MATLAB provides a number of features for documenting and sharing your work. You can integrate your MATLAB code with other languages and applications, and distribute your MATLAB algorithms and applications.

3.4 FEATURES OF MATLAB

- High-level language for technical computing.
- Development environment for managing code, files, and data.
- Interactive tools for iterative exploration, design, and problem solving.
- Mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, and numerical integration.
- 2-D and 3-D graphics functions for visualizing data.
- Tools for building custom graphical user interfaces.
- Functions for integrating MATLAB based algorithms with external applications and languages, such as C, C++, Fortran, Java™, COM, and Microsoft Excel.

3.5 THE MATLAB SYSTEM

The MATLAB system consists of five main parts:

- **Development Environment**.

This is the set of tools and facilities that help you use MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and Command Window, a command history, and browsers for viewing help, the workspace, files, and the search path.

- **The MATLAB Mathematical Function Library**.

This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, Bessel functions, and fast Fourier transforms.

- **The MATLAB Language**.

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create complete large and complex application programs.

Handle Graphics.

This is the MATLAB graphics system. It includes high-level commands for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level commands that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.

The MATLAB Application Program Interface (API).

This is a library that allows you to write C and FORTRAN programs that interact with MATLAB. It include facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

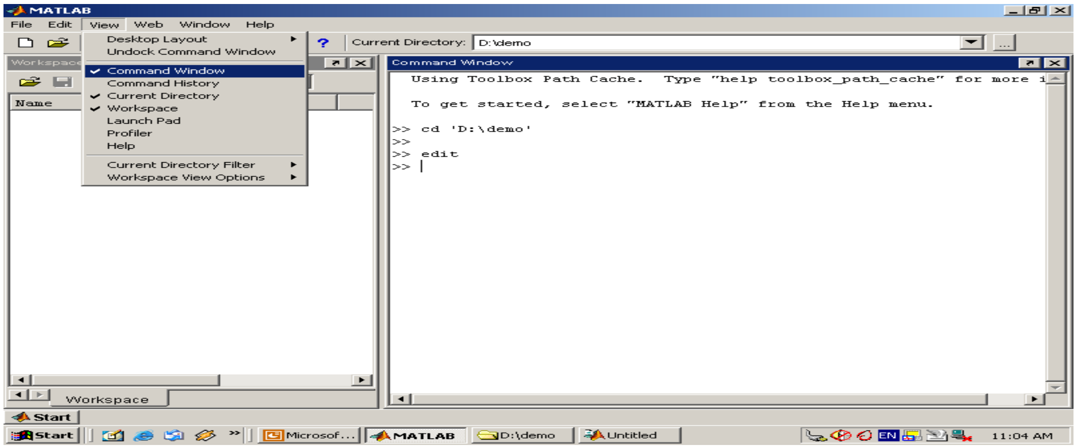
3.5.1 DESKTOP TOOLS

This section provides an introduction to MATLAB's desktop tools. You can also use MATLAB functions to perform most of the features found in the desktop tools. The tools are:

- Current Directory Browser
- Workspace Browser

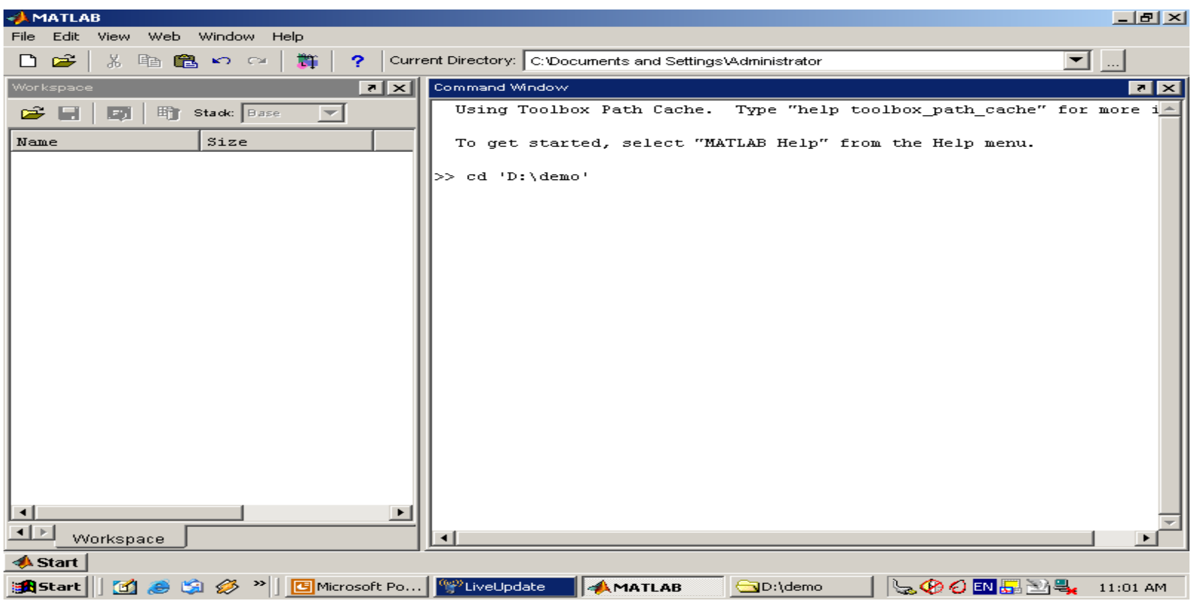
- Array Editor
- Editor/Debugger
- Command Window
- Command History
- Launch Pad
- Help Browser

Command Window



Use the Command Window to enter variables and run functions and M-files.

Current Directory Browser



MATLAB file operations use the current directory and the search path as reference points. Any file you want to run must either be in the current directory or on the search path.

Search Path

To determine how to execute functions you call, MATLAB uses a search path to find M-files and other MATLAB-related files, which are organized in directories on your file system. Any file you want to run in MATLAB must reside in the current directory or in a directory that is on the search path. By default, the files supplied with MATLAB and Math Works toolboxes are included in the search path.

- ### Workspace Browser

The MATLAB workspace consists of the set of variables (named arrays) built up during a MATLAB session and stored in memory. You add variables to the workspace by using functions, running M-files, and loading saved workspaces.

To view the workspace and information about each variable, use the Workspace browser, or use the functions who and whos.

To delete variables from the workspace, select the variable and select Delete from the Edit menu. Alternatively, use the clear function.

The workspace is not maintained after you end the MATLAB session. To save the workspace to a file that can be read during a later MATLAB session, select Save Workspace As from the File menu, or use the save function. This saves the workspace to a binary file called a MAT-file, which has a .mat extension. There are options for saving to different formats. To read in a MAT-file, select Import Data from the File menu, or use the load function.

- ### Array Editor

Double-click on a variable in the Workspace browser to see it in the Array Editor. Use the Array Editor to view and edit a visual representation of one- or two-dimensional numeric arrays, strings, and cell arrays of strings that are in the workspace.

- **Editor/Debugger**

Use the Editor/Debugger to create and debug M-files, which are programs you write to run MATLAB functions. The Editor/Debugger provides a graphical user interface for basic text editing, as well as for M-file debugging.

You can use any text editor to create M-files, such as Emacs, and can use preferences (accessible from the desktop File menu) to specify that editor as the default. If you use another editor, you can still use the MATLAB Editor/Debugger for debugging, or you can use debugging functions, such as dbstop, which sets a breakpoint.

If you just need to view the contents of an M-file, you can display it in the Command Window by using the type function.

3.5.2 ANALYZING AND ACCESSING DATA

MATLAB supports the entire data analysis process, from acquiring data from external devices and databases, through preprocessing, visualization, and numerical analysis, to producing presentation-quality output.

DATA ANALYSIS

MATLAB provides interactive tools and command-line functions for data analysis operations, including:

- Interpolating and decimating
- Extracting sections of data, scaling, and averaging
- Thresholding and smoothing
- Correlation, Fourier analysis, and filtering
- 1-D peak, valley, and zero finding
- Basic statistics and curve fitting
- Matrix analysis

DATA ACCESS

MATLAB is an efficient platform for accessing data from files, other applications, databases, and external devices. You can read data from popular file formats, such as Microsoft Excel; ASCII text or binary files; image, sound, and video files; and scientific files, such as HDF and HDF5. Low-level binary file I/O functions let you work with data files in any format. Additional functions let you read data from Web pages and XML.

VISUALIZING DATA

All the graphics features that are required to visualize engineering and scientific data are available in MATLAB. These include 2-D and 3-D plotting functions, 3-D volume visualization functions, tools for interactively creating plots, and the ability to export results to all popular graphics formats. You can customize plots by adding multiple axes; changing line colors and markers; adding annotation, Latex equations, and legends; and drawing shapes.

2-D PLOTTING

Visualizing vectors of data with 2-D plotting functions that create:

- Line, area, bar, and pie charts.
- Direction and velocity plots.
- Histograms.
- Polygons and surfaces.
- Scatter/bubble plots.
- Animations.

3-D PLOTTING AND VOLUME VISUALIZATION

MATLAB provides functions for visualizing 2-D matrices, 3-D scalar, and 3-D vector data. You can use these functions to visualize and understand large, often complex, multidimensional data. Specifying plot characteristics, such as camera viewing angle, perspective, lighting effect, light source locations, and transparency.

3-D plotting functions include:

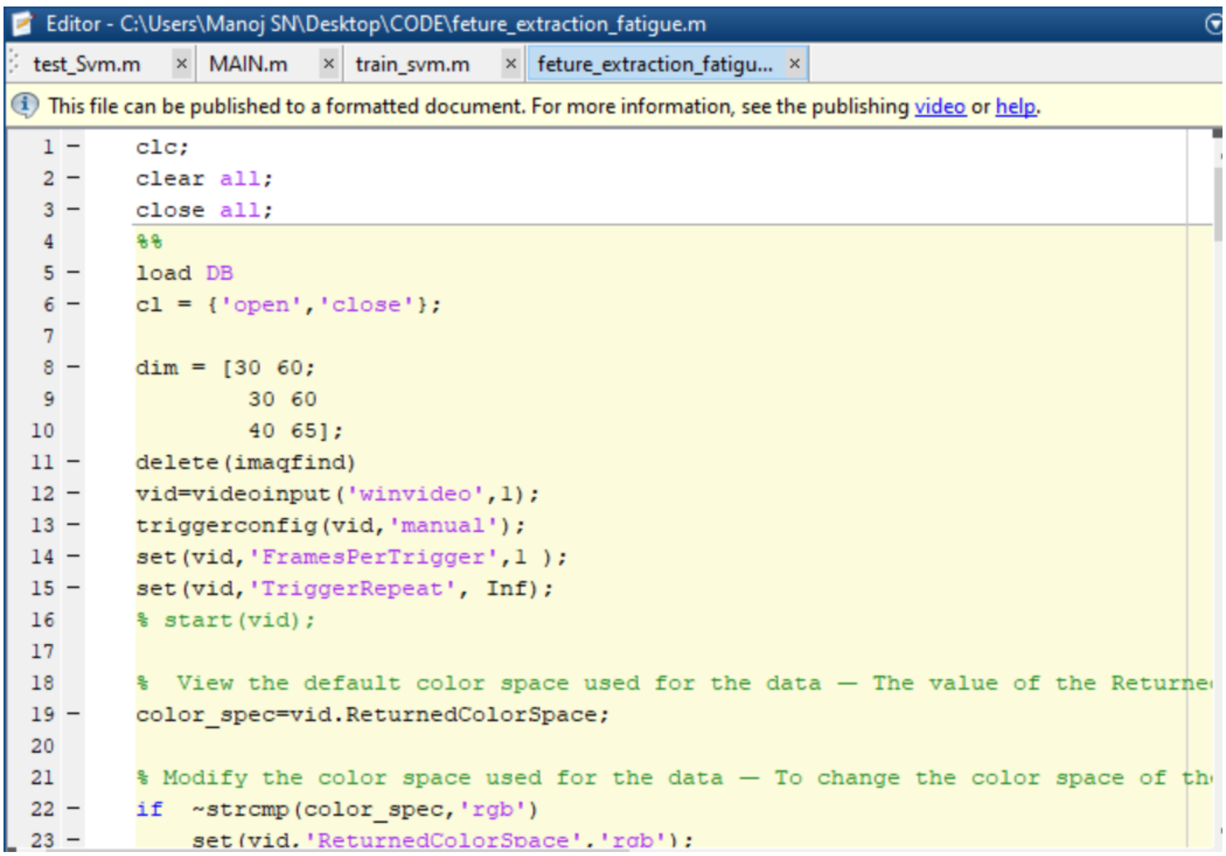
- Surface, contour, and mesh.
- Image plots.
- Cone, slice, stream, and iso surface.

4. IMPLEMENTATION

GENERAL

Matlab is a program that was originally designed to simplify the implementation of numerical linear algebra routines. It has since grown into something much bigger, and it is used to implement numerical algorithms for a wide range of applications. The basic language used is very similar to standard linear algebra notation, but there are a few extensions that will likely cause you some problems at first.

SCREENSHOTS:



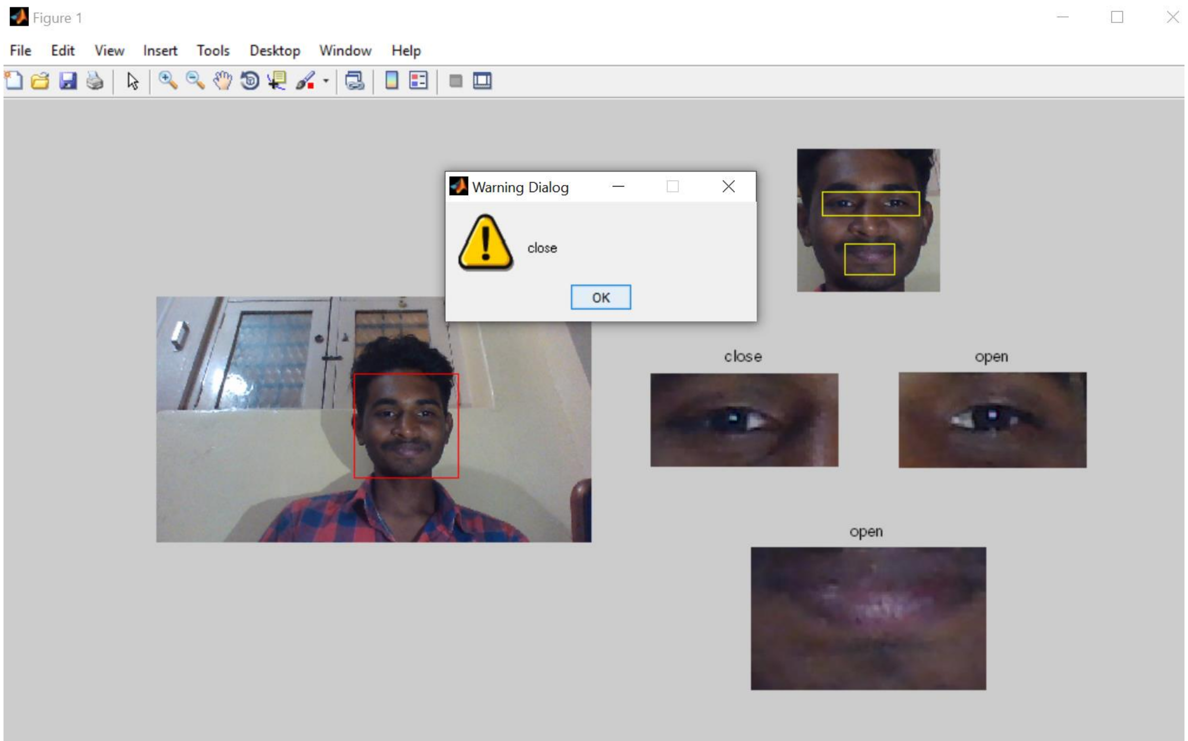


Figure 1
File Edit View Insert Tools Desktop Window Help
Warning Dialog
close
OK
close
open
open

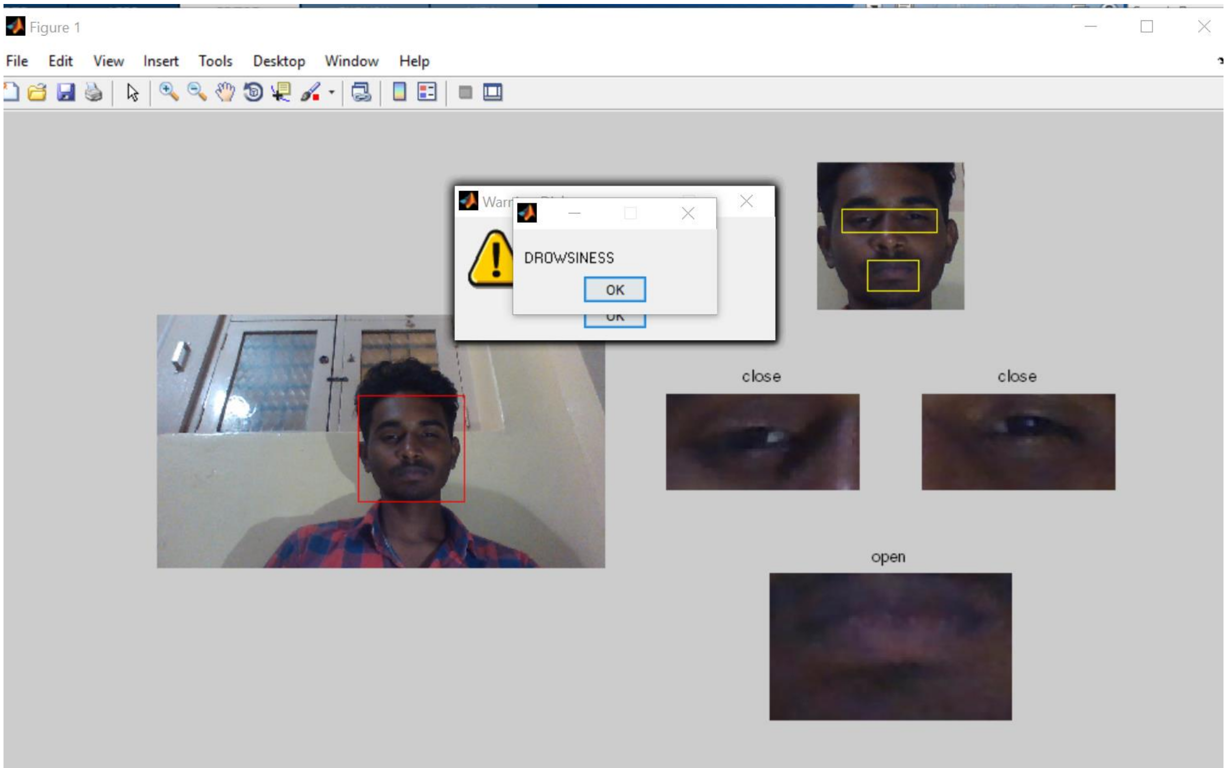
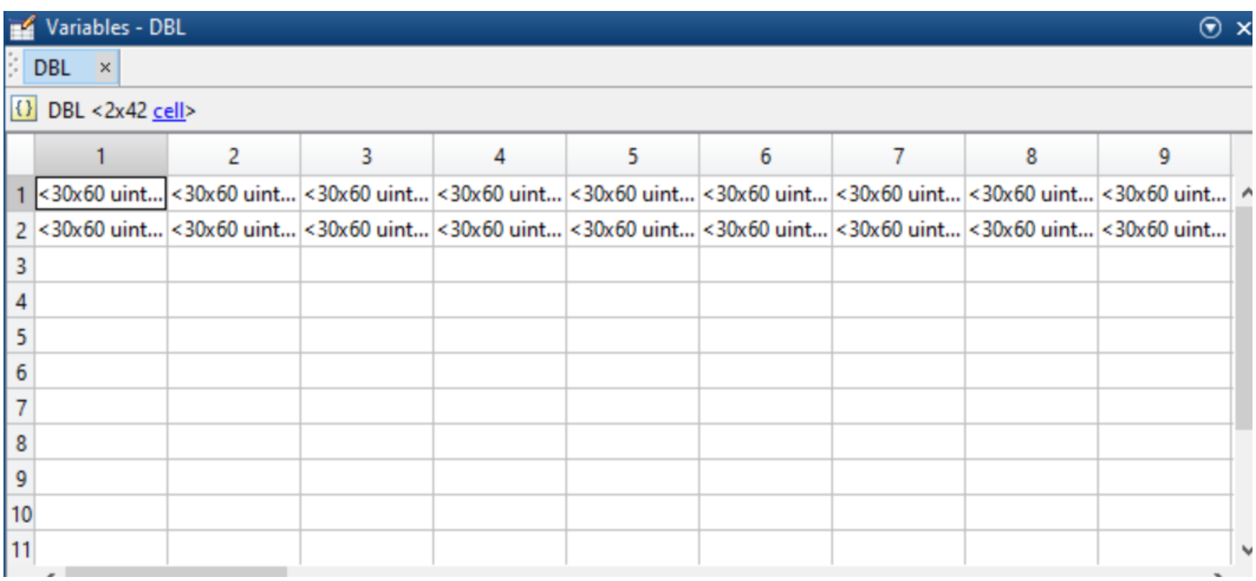


Figure 1
File Edit View Insert Tools Desktop Window Help
DROWSINESS
OK
close
close
open

```
s =

b


s =

b


Feature =

    1.0000         0    1.0000    1.0000
    0.5000    0.5000    1.0000         0
    1.3333    1.6667    0.3333    1.0000
```



Variables - Eeye

DBL × DBM × DBR × Eeye ×

Eeye <48x193x3 uint8>

```
val(:,:,1) =

  Columns 1 through 16
```

45	44	41	40	40	40	38	36	33	33	32	33	31	31	31	30
43	43	41	39	38	38	36	34	33	33	33	34	33	32	31	30
41	42	41	39	38	38	36	34	33	33	34	35	33	33	32	30
43	42	40	38	37	37	36	35	34	35	35	35	35	34	33	33
42	41	40	39	38	37	36	36	35	35	35	35	35	34	34	33
41	40	40	39	39	39	37	36	35	35	35	35	35	34	34	34
40	39	39	39	40	39	38	37	35	35	35	35	35	34	34	34
39	39	39	39	39	39	38	37	36	36	35	35	35	35	34	34
39	39	38	38	39	38	38	37	36	36	36	35	35	35	35	35
41	40	39	38	38	38	38	37	37	36	36	35	35	35	35	35
42	40	39	38	38	38	38	37	37	36	36	35	35	35	35	35
43	42	41	40	39	39	39	38	38	37	37	36	35	34	34	34
43	42	41	40	39	39	39	39	38	38	37	36	35	34	34	34
43	42	41	39	38	38	39	39	38	38	37	36	35	34	34	34
44	43	41	39	38	38	39	40	38	38	37	36	35	34	34	34
44	43	41	39	38	38	39	40	38	38	37	36	35	34	33	33
44	43	42	40	39	39	40	40	39	38	37	36	35	34	34	34
44	43	42	41	40	40	40	40	39	38	37	36	35	34	33	33
44	43	42	41	40	40	40	39	38	38	37	35	34	33	33	33

Emouth ×

Emouth <40x65 uint8>

	1	2	3	4	5	6
1	38	37	39	41	43	47
2	39	40	40	41	45	48
3	40	42	42	44	45	47
4	44	44	45	48	48	50
5	49	49	48	49	51	53
6	52	51	50	49	51	54
7	50	48	48	49	49	51
8	49	46	45	46	47	49
9	49	46	44	44	46	48
10	48	46	44	43	43	44
11	50	47	43	41	39	38
12	51	47	42	39	37	35
13	54	50	46	43	41	38
14	56	53	50	48	49	49
15	52	51	51	51	52	55
16	50	50	50	51	53	54
17	51	49	50	50	52	54
18	51	49	50	50	50	51

Inspector: videoinput

BayerSensorAlignment	grbg
DeviceID	1
DiskLogger	[0x0 double array]
DiskLoggerFrameCount	0
EventLog	[1x35 struct array]
FrameGrabInterval	1
FramesAcquired	34
FramesAcquiredFcnCount	0
FramesAvailable	1
FramesPerTrigger	1.0
InitialTriggerTime	[1x6 double array]
Logging	off
LoggingMode	memory
Name	MJPG_1280x720-winvideo-1
NumberOfBands	3
Previewing	off
ROIPosition	[1x4 double array]
ReturnedColorSpace	rgb
Running	on
SelectedSourceName	input1
Source	videosource
Tag	null
Timeout	10.0
TimerPeriod	1.0
TriggerCondition	none
TriggerFrameDelay	0
TriggerRepeat	Infinity
TriggerSource	none
TriggerType	manual
TriggersExecuted	34

5. CONCLUSION

We developed a system that localizes and track the eyes movements of the driver in order to detect drowsiness. The system uses a combination of template – based matching and feature based matching in order to localize the eyes. During tracking, system will be able to decide if the eyes are open or closed and whether the driver is looking in front. When the eyes will be closed for too long, a warning signal will be given in the form of buzzer or alarm author-kit message.

6. APPENDENCES

Main:

```
clc;
clear all;
close all;
%%
load DB
cl = {'open','close'};

dim = [30 60;
        30 60
        40 65];
delete(imaqfind)
vid=videoinput('winvideo',1);
triggerconfig(vid,'manual');
set(vid,'FramesPerTrigger',1 );
set(vid,'TriggerRepeat', Inf);
% start(vid);

%  View the default color space used for the data — The value of
the ReturnedColorSpace property indicates the color space of the
image data.
color_spec=vid.ReturnedColorSpace;

% Modify the color space used for the data — To change the color
space of the returned image data, set the value of the
ReturnedColorSpace property.
if  ~strcmp(color_spec,'rgb')
    set(vid,'ReturnedColorSpace','rgb');
end

start(vid)


% Create a detector object
faceDetector = vision.CascadeObjectDetector;
faceDetectorLeye = vision.CascadeObjectDetector('EyePairBig');
faceDetectorM = vision.CascadeObjectDetector('Mouth');
tic
% Initialise vector
```

```
LC = 0; % Left eye closer
RC = 0; % Right eye closer
MC = 0; % Mouth closer
TF = 0; % Total frames
TC = 0; % Total closure
Feature = [];
c1p = 1;
for ii = 1:100

    trigger(vid);
    im=getdata(vid,1); % Get the frame in im
    imshow(im)

    subplot(3,4,[1 2 5 6 9 10]);
    imshow(im)

    % Detect faces
    bbox = step(faceDetector, im);

    if ~isempty(bbox);
        bbox = bbox(1,:);

        % Plot box
        rectangle('Position',bbox,'edgecolor','r');

        Ic = imcrop(im,bbox);
        subplot(3,4,[3 4]);
        imshow(Ic)

        bboxeye = step(faceDetectorLeye, Ic);

        if ~isempty(bboxeye);
            bboxeye = bboxeye(1,:);

            Eeye = imcrop(Ic,bboxeye);
            % Plot box
            rectangle('Position',bboxeye,'edgecolor','y');
        else
            disp('Eyes not detected')
        end

        if isempty(bboxeye)
            continue;
        end
       Ic(1:bboxeye(2)+2*bboxeye(4),:,:) = 0;
```

```
% Detect Mouth
bboxM = step(faceDetectorM, Ic);


if ~isempty(bboxM);
    bboxMtemp = bboxM;

    if ~isempty(bboxMtemp)

        bboxM = bboxMtemp(1,:);
        Emouth =  imcrop(Ic,bboxM);

        % Plot box
        rectangle('Position',bboxM,'edgecolor','y');
    else
        disp('Mouth  not detected')
        continue;
    end
else
    disp('Mouth not detected')
    continue;
end

[nre nce k ] = size(Eeye);

% Divide into two parts
Leye = Eeye(:,1:round(nce/2),:);
Reye = Eeye(:,round(nce/2+1):end,:);

subplot(3,4,7)
imshow(Leye);
subplot(3,4,8)
imshow(Reye);
subplot(3,4,[11,12]);
imshow(Emouth);

Leye = rgb2gray(Leye);
Reye = rgb2gray(Reye);
Emouth = rgb2gray(Emouth);

% Template matching using correlation coefficient
% Left eye
% Resize to standard size
Leye =  imresize(Leye,[dim(1,1) dim(1,2)]);
```

```
  c1 =match_DB(Leye,DBL);
  subplot(3,4,7)
  title(cl{c1})


  % Right eye
  % Resize to standard size
  Reye =  imresize(Reye,[dim(2,1) dim(2,2)]);
  c2 = match_DB(Reye,DBR);
  subplot(3,4,8)
  title(cl{c2})

  % Mouth
  % Resize to standard size
  Emouth =  imresize(Emouth,[dim(3,1) dim(3,2)]);
  c3 = match_DB(Emouth,DBM);
  subplot(3,4,[11,12]);
  title(cl{c3})


  if c1 == 2
      LC = LC+1;
      if c1p == 1
          TC = TC+1;
      end
                  s='a';
              warndlg('close');



  end
  if c2==2
      RC = RC+1;
                              s='a';
                           warndlg('close');


  end

if c1==2 & c2==2
    LC = LC+1;
    RC = RC+1;
    s = 'b'
    msgbox ('DROWSINESS' );
end
```

```
        if c1==1 & c2==1
            msgbox ( 'NO DROWSINESS' );
        end


          if c3 == 1
              MC = MC + 1;
          end

          TF = TF + 1; % Total frames

          if toc>10
              Feature = [Feature;LC/TF RC/TF MC/TF TC]
              tic
              % Initialise vector
              LC = 0; % Left eye closer
              RC = 0; % Right eye closer
              MC = 0; % Mouth closer
              TF = 0; % Total frames
              TC = 0; % Total closure
          end
          c1p = c1;
          pause(0.00005)
      end
  end

  save FA Feature
```

Training SVM Dataset Code:

```
clc;
clear all;
close all;
%%
load FA
F1 = Feature;
load NF
F2 = Feature;
xdata = [F1;F2];
```

```
group = cell(1,1);

for ii = 1:size(F1,1)
    group{ii,1} = 'Fatigue';
end

for ii = 1:size(F2,1)
    group{ii+size(F1,1),1} = 'Non-Fatigue';
end

svmStruct = svmtrain(xdata,group,'showplot',true);
% Testing
save svm svmStruct
load svm
for ii = 1:size(F1,1)
    species = svmclassify(svmStruct,F1(ii,:));
    disp([ group{ii,1} ' = ' species]);
end

for ii = 1:size(F2,1)
    species = svmclassify(svmStruct,F2(ii,:));
    disp([ group{ii+size(F1,1),1} ' = ' species]);
end
```

Testing SVM Dataset Code:

```
clc;
clear all;
close all;
%%
load fisheriris
xdata = meas(51:end,3:4);
%Find a line separating the Fisher iris data on versicolor and
virginica species, according to the petal length and petal width
measurements.
%These two species are in rows 51 and higher of the data set,
and the petal length and width are the third and fourth columns.

group = species(51:end);
svmStruct = svmtrain(xdata,group,'showplot',true);
%svmtrain(Training,Group,Name,Value) returns a structure with
additional options
%specified by one or more Name,Value pair arguments.

species = svmclassify(svmStruct,[5 2]);
```

Match DB:

```
function c1 =match_DB(Leye,DBL)


MS1 = [];
for tt = 1:length(DBL)
    % Obtain the image
    It = DBL{1,tt};
    if ~isempty(It)
        % Calculate correlation coefficient
        MS1 = [MS1 corr2(It,Leye)];
    else
        break;
    end
end

MS2 = [];
for tt = 1:length(DBL)
    % Obtain the image
    It = DBL{2,tt};

    if ~isempty(It)
        % Calculate correlation coefficient
        MS2 = [MS2 corr2(It,Leye)];
    else
        break;
    end
end

% Take mean value
M(1) = mean(MS1);
M(2) = mean(MS2);

% Decide about the condition
c1 = find(M == max(M));
```

Skin Segment:

```
function segment = skin_seg2(I1)
% Convert image to double precision
I=double(I1);
[hue,s,v]=rgb2hsv(I);
```

```
%  Ycbcr = rgb2ycbcr(I);
% % cb=Ycbcr(:,:,2)+128;
% % cr=Ycbcr(:,:,3)+128;
cb =  0.148* I(:,:,1) - 0.291* I(:,:,2) + 0.439 * I(:,:,3) +
128;
cr =  0.439 * I(:,:,1) - 0.368 * I(:,:,2) -0.071 * I(:,:,3) +
128;
[w h]=size(I(:,:,1));

segment = 140<=cr & cr<=165 & 140<=cb & cb<=195 & 0.01<=hue &
hue<=0.1;

segment=imfill(segment,'holes');
segment=bwmorph(segment,'dilate');
segment=bwmorph(segment,'majority');
```

Sobel:

```
A=imread('peppers.png');
B=rgb2gray(A);

C=double(B);


for i=1:size(C,1)-2
    for j=1:size(C,2)-2
        %Sobel mask for x-direction:
        Gx=((2*C(i+2,j+1)+C(i+2,j)+C(i+2,j+2))-
(2*C(i,j+1)+C(i,j)+C(i,j+2)));
        %Sobel mask for y-direction:
        Gy=((2*C(i+1,j+2)+C(i,j+2)+C(i+2,j+2))-
(2*C(i+1,j)+C(i,j)+C(i+2,j)));

        %The gradient of the image
        %B(i,j)=abs(Gx)+abs(Gy);
        B(i,j)=sqrt(Gx.^2+Gy.^2);

    end
end
figure,imshow(B); title('Sobel gradient');
Thresh=100;
B=max(B,Thresh);
B(B==round(Thresh))=0;

B=uint8(B);
figure,imshow(~B);title('Edge detected Image');
```

Database Training:

```
clc;
clear all;
close all;
%%
cl = {'open','close'};

% Initialise empty cell structure
DBL = cell(1,1);
DBR = cell(1,1);
DBM = cell(1,1);

dim = [30 60;
        30 60
        40 65];
% Left eye
disp('Training left eye');
for ii = 1:2
    disp(cl{ii})
    fpath = ['Database/LE/' cl{ii} '/*.bmp'];
    D = dir(fpath);
    for kk = 1:length(D)
        impath = ['Database/LE/' cl{ii} '/' D(kk).name];

        I = imread(impath);

        % Resize to standard size
        Is = imresize(I,[dim(1,1) dim(1,2)]);
%          [nr nc] = size(I);

        % Save to database
        DBL{ii,kk} = Is;
        imshow(Is);
        pause(0.01)
    end
end

% Right Eye
disp('Training Right eye');
for ii = 1:2
    disp(cl{ii})
    fpath = ['Database/RE/' cl{ii} '/*.bmp'];
    D = dir(fpath);
```

```
    for kk = 1:length(D)
        impath = ['Database/RE/' cl{ii} '/' D(kk).name];

        I = imread(impath);

        % Resize to standard size
        Is = imresize(I,[dim(2,1) dim(2,2)]);
%         [nr nc] = size(I);

        % Save to database
        DBR{ii,kk} = Is;
        imshow(Is);
        pause(0.01)
    end
end

% Mouth
disp('Training Mouth');
for ii = 1:2
    disp(cl{ii})
    fpath = ['Database/M/' cl{ii} '/*.bmp'];
    D = dir(fpath);
    for kk = 1:length(D)
        impath = ['Database/M/' cl{ii} '/' D(kk).name];

        I = imread(impath);

        % Resize to standard size
        Is = imresize(I,[dim(3,1) dim(3,2)]);

        % Save to database
        DBM{ii,kk} = Is;
        imshow(Is);
        pause(0.01)
    end
end

% Save the database
save DB DBL DBR DBM
```

Database Creation:

```
clc;  % clc clears all input and output from the Command Window
display, giving you a "clean screen."
clear all;  %  removes all variables from the current workspace,
releasing them from system memory.
close all;  % deletes the current figure or the specified
figure(s). It optionally returns the status of the close
operation.
%%

delete(imaqfind)  % imaqfind: Find image acquisition objects
% returns an array containing all the video input objects that
exist in memory.
% If only a single video input object exists in memory, imaqfind
displays a detailed summary of that object.
% Nest a call to the imaqfind function within the delete
function to delete all these objects from memory.

vid=videoinput('winvideo',1);  % OS Generic Video Interface
% creates a video input object for a webcam image acquisition
device.
% MATLAB files to use Windows Video, Macintosh Video, or Linux
Video cameras with the toolbox.
% The correct OS files will be installed, depending on your
system.
% Kinect for Windows (kinect)
% Linux Video (linuxvideo)

triggerconfig(vid,'manual');
% Configure video input object trigger properties.
% configures the value of the TriggerType property of the video
input object "vid"
% to the value specified by the text string type that here is
"manual".
% TriggerType Value are immediate,manual,hardware.
% immediate: The trigger occurs automatically, immediately after
the start function is issued. This is the default trigger type.
% manual: The trigger occurs when you issue the trigger
function. A manual trigger can provide more control over image
acquisition.
% For example, you can monitor the video stream being acquired,
using the preview function,
% and manually execute the trigger when you observe a particular
condition in the scene.
```

```
% hardware: Hardware triggers are external signals that are
processed directly by the hardware.


set(vid,'FramesPerTrigger',1 );  % The default is 10 frames per
trigger
set(vid,'TriggerRepeat', Inf);
% Specify number of additional times trigger executes
% If TriggerRepeat is set to its default value of zero, then the
trigger executes once.
% If TriggerRepeat is set to inf then the trigger executes
continuously until a stop function is issued or an error occurs.

% start(vid);

%  View the default color space used for the data — The value of
the ReturnedColorSpace property indicates the color space of the
image data.
color_spec=vid.ReturnedColorSpace;
% specifies the color space you want the toolbox to use when it
returns image data to the MATLAB workspace.
% grayscale,rgb,YCbCr

% Modify the color space used for the data —
% To change the color space of the returned image data, set the
value of the ReturnedColorSpace property.
if  ~strcmp(color_spec,'rgb')
    set(vid,'ReturnedColorSpace','rgb');
end

start(vid)  % to start the image acquisition object.


% Create a detector object
faceDetector = vision.CascadeObjectDetector;
faceDetectorLeye = vision.CascadeObjectDetector('EyePairBig');
faceDetectorM = vision.CascadeObjectDetector('Mouth');
load ix
for ii = 1:100
    trigger(vid);  % as we are calling it manually
    im=getdata(vid,1); % Get the frame in im
    imshow(im)  % Display image

    subplot(3,4,[1 2 5 6 9 10]);  % Create and control multiple
axes
```

```
% subplot('Position',[left bottom width height])
imshow(im)

% Detect faces
bbox = step(faceDetector, im);

if ~isempty(bbox);
    bbox = bbox(1,:);

    % Plot box
    rectangle('Position',bbox,'edgecolor','r');

    Ic = imcrop(im,bbox);
    subplot(3,4,[3 4]);
    imshow(Ic)

    bboxeye = step(faceDetectorLeye, Ic);

    if ~isempty(bboxeye);
        bboxeye = bboxeye(1,:);

        Eeye = imcrop(Ic,bboxeye);
        % Plot box
        rectangle('Position',bboxeye,'edgecolor','y');
    else
        disp('Eyes not detected')
    end

    if isempty(bboxeye)
        continue;
    end
   Ic(1:bboxeye(2)+2*bboxeye(4),:,:) = 0;

    % Detect Mouth
    bboxM = step(faceDetectorM, Ic);


    if ~isempty(bboxM);
        bboxMtemp = bboxM;

        if ~isempty(bboxMtemp)

            bboxM = bboxMtemp(1,:);
            Emouth =  imcrop(Ic,bboxM);
```

```
                % Plot box
                rectangle('Position',bboxM,'edgecolor','y');
            else
                disp('Mouth  not detected')
            end
        else
            disp('Mouth not detected')
        end

        [nre nce k ] = size(Eeye);

        % Divide into two parts
        Leye = Eeye(:,1:round(nce/2),:);
        Reye = Eeye(:,round(nce/2+1):end,:);

        subplot(3,4,7)




imshow(Leye);
        subplot(3,4,8)
        imshow(Reye);
        subplot(3,4,[11,12]);
        imshow(Emouth);

        Leye = rgb2gray(Leye);
        Reye = rgb2gray(Reye);
        Emouth = rgb2gray(Emouth);

        imwrite(Leye,['Database/LE/' num2str(ix) '.bmp']);
        imwrite(Reye,['Database/RE/' num2str(ix) '.bmp']);
        imwrite(Emouth,['Database/M/' num2str(ix) '.bmp']);
       ix  = ix+1;
       save ix ix
        pause(0.00005)
    end
end
```

6. REFERENCES:

[1] Drowsy Driver Warning System Using Image Processing 1Singh Himani Parmar, 2Mehul Jajal, 3Yadav Priyanka Brijbhan Electronics & Communication, GEC, Bharuch, Gujarat.

[2] Detecting Driver Drowsiness Based on Sensors: A Review Arun Sahayadhas , Kenneth Sundaraj and Murugappan Murugappan

[3] Driver's Eye Blinking Detection Using Novel Color and Texture Segmentation Algorithms Artem A. Lenskiy and Jong-Soo Lee*

[4] A Vision Based System for Monitoring the Loss of Attention in Automotive Drivers Anirban Dasgupta, Anjith George, S L Happy and Aurobinda Routray, Member, IEEE

[5] Drowsy Driver Identification Using MATLAB Video Processing Mrs.S. Dhanalakshmi1, J.Jasmine Rosepet2, G.Leema Rosy3, M.Philominal4 1,2,3,4Idhaya Engineering College for Women, Chinnasalem, India

[6] Qiang Ji, Zhiwei Zhu and Peilin Lan ―IEEE transactions on Vehicular Technology Real Time Non-intrusive Monitoring and Prediction of Driver Fatigue, vol. 53, no. 4, July 2004.

[7] N.G. Narole, and G.H. Raisoni., ―IJCSNS A Neuro-genetic System Design for Monitoring Driver's Fatigue. vol. 9. No. 3, March 2009.

[8] Wei-niin Huang & Robert Mariani, ―Face Detecion and precise Eyes Location ―, Proceeding of the International Conference on Patteni Recognization(ICPP‖OO),Vol.4,2000

[9] Gonzalez, Rafel C. and Woods, Richard E. ―Digital Image Processing‖, Prentice Hall: Upper Saddle River, N.J., 2002