"Interaction to computer system based on eye and head gesture recognition using Deep Learning"

A MAJOR PROJECT REPORT SUBMITTED TO

RAJKIYA ENGINEERING COLLEGE BANDA

(An Affiliated Institute of Dr. A.P.J. Abdul Kalam Technical University, Lucknow UP)

In fulfillment of the requirements for Major Project, eighth semester

Bachelor of Engineering

In

Information Technology, REC BANDA

Submitted by

Praphull Maurya (1673413031)

Narendra Rajpoot (1673413024)

Rishabh Shukla (1673413037)

Atul Kumar Uttam (1673413009)

Under the Guidance of

Dr. Dhananjay Bisen

Assistant Professor

Dept. of IT

REC Banda - 210201

DEPARTMENT OF INFORMATION TECHNOLOGY

RAJKIYA ENGINEERING COLLEGE BANDA

Banda - 210201

2019-2020

Declaration

I hereby declare that the project report entitled **“Interaction to computer system based on eye and head gesture recognition using Deep Learning”** submitted by us to **Rajkiya Engineering College, Banda** is the requirement for the award of the degree of the B. Tech in Information Technology is a record of bonafide project work carried out by us under the guidance of **Dr. Dhananjay Bisen Sir**. I further declare that the work reported in this project has not been submitted and will not be submitted either in part or in full for the award of any other degree in this institute.

Date: Signature of the Candidates:

Rajkiya Engineering College Banda

Department of Information Technology

CERTIFICATE

This is to certify that the Minor Project entitled “**Interaction to computer system based on eye and head gesture recognition using Deep Learning**” submitted by Praphull Maurya, Narendra Rajpoot, Rishabh Shukla , Atul Kumar Uttam is a record of bonafide work carried out by them, in the fulfillment with Degree of Bachelor of Technology Information Technology, Rajkiya Engineering College, Banda. This work is done during year August’19 to March’20, under our guidance.

Date:

Internal Guide Professor:

Dr. Dhananjay Bisen
Assistant Professor

Head of Department

Dr. Vibhash Yadav
Associate Professor

Acknowledgement

We would like to express our special thanks & gratitude to our guide “Dr. Dhananjay Bisen Sir” for their able guidance and support in completing my project.

We would also like to extend our gratitude to the Head of Department of Information Technology “Dr. Vibhash Yadav Sir” and our Honorable Director “Prof. S. P Shukla Sir” for providing me with all the facility that was required.

Praphull Maurya

Narendra Rajpoot

Rishabh Shukla

Atul Kumar Uttam

Table of Contents

CONTENTS **PAGE**

List of Figures

CONTENTS **PAGE**

List of Tables

CONTENTS **PAGE**

1. Abstract

This project is centred around building up a model to control computer systems by utilizing eyes and head gestures. The face recognition system is used in combination with eyes and head gesture detection and recognition to control movements of the mouse and therefore, the computer system operates in real time. The facial detection and feature recognition are done by facial landmark detector using Dlib which uses a Histogram of Oriented Gradients based global feature detector in addition to linear classifier, a pyramid-based images, and a sliding window scheme for face detection. To recognise the feature of face we are using cascade of regressor algorithm. To capture the blinks of eyes, we are using the EAR method. The reason for this task is to develop an eyes and head motion-controlled system that can successfully utilize face parts like eyes and nose and empower the user to perform activities mapped to explicit eye blink/head motions by utilizing PC webcam. It must be consistently accurate so that users can comfortably use it like other frequent gadgets.

2. Keywords:

Human-Computer Interaction, Gesture Recognition, Facial Recognition, D-Lib, OpenCV, PyAutoGUI, EAR.

3. Introduction:

Over the previous years, the performance of Human-Computer Interaction (HCI) has impacted the interaction with PC and with the huge rise of computer system among general public, HCI has become essential a medium through which we humans can interact with computer systems, as PCs have become a significant part of our day to day lives. A lot of different techniques, such as hand gestures are very popularly used, but the people who with Locked-in syndrome (LIS), Quadriplegia or people who have lost their arm cannot interact with the computer system. The head gesture has the potential to communicate with an intelligent system. This project shows a vision based multimodal HCI utilizing eye blink followed by head movement that proposes an interface which can viably substitute the job of a current gadget, for example, a mouse.

3.1. HCI - Human-Computer Interaction

The research in the recent years has improved the design and computer technologies of the Human-computer interaction (HCI) that are used to develop the interface between humans and computers. Scientists working in the area of HCI observe human behaviour as how they interact with computer systems and develop technologies that can remove the complexities and enable the human beings to interact with the computer system in a newer way. We can interact with computers using various interfaces which are described as follows.

3.1.1. Types of Interfaces:

Four types of Interfaces,

1- Command line Interface: It is used to display a prompt, where user types a command with the help of keyboard, the computer executes that command and gives a textual output on the command prompt.

2- Menu Driven Interface: They have a list of multiple items where a user can choose and make selections by highlighting one or more items.

3- Graphical User Interface: It uses the icons, menus, windows and pointers that can be manipulated by the use of mouse. GUI are used by Desktop applications, handheld computers, internet browsers, ERP and computer kiosks.

4- Natural Language Interface: The NLI has a wide range from simple command systems to the voice activated text processing systems. The Commands should be spoken in “normal” languages

The Architecture of HCI systems are recognized by:

➢ Diversity of inputs entered and outputs given in respect to modality.

➢ Total number of inputs entered and outputs given to present in the HCI systems.

➢ The input and output should be work these diverse for interaction purpose.

On the basis of configuration and design of interface,

HCI systems is categorized into:

➢ Unimodal Human Computer interaction system.

➢ Multimodal Human Computer interaction system.

Unimodal HCI system:

Main dependency of interfaces are no. and diversity of inputs and outputs that are part of communication channels which helps user interaction with computer by the interface.

Unimodal systems are based on only 1 modality.

Base on nature of different modalities, it is divided into three types:

- ➢ Audio-Based Unimodal System
- ➢ Sensor-Based unimodal System
- ➢ Visual-Based unimodal System

Limitations of Unimodal Human Computer Interaction System:

• It is not a natural way of human interact with computer system.

•It is generally designed for the average type users.

• This modal fails to cater the need of a diverse type of peoples.

• It is difficult to use by illiterate, disabled, and untrained people.

• It does not provide universal interface for the people.

Multimodal HCI System:

- — The Combination of one or more modalities, usage of multiple independent channel signals for the interaction between user and machine is called as multimodal human computer interaction systems.
- — There can be two or more modes of input in multimodal interface that acts as facilitator between human and computer.

- MMHCI are easy to use for disabled and illiterate people.
- “Put That There” demonstration system is a famous example of a multimodal system.

Human-computer interaction considers the manners by which people utilize or don't utilize computational resources, frameworks, and foundations. Improvement in the usability of computer interfaces is an area of research.

3.2. Gesture Recognition

Gesture recognition is a computer science topic dealing with the interpretation of human gestures via mathematical algorithms. Any motion of the body can be considered as gestures, prominent sources of gestures being the face or hand.

Feature of Gesture Recognition:

- It is more accurate.
- It is Highly stable.
- It saves time to unlock a device.

Gesture recognition can be carried out with advanced technologies such as image processing and computer vision. In relation to gesture control, the touchless UI is an emerging kind of innovation. Touchless UI (TUI) is the way toward instructing the PC through body movement and motions without contacting a mouse, a keyboard, or screen.

Gesture Recognition can be seen as a way for computer system to understand the human and its body language, thus building a bridge between humans and machines rather than primitive way of interacting such as text user interface or some GUIs that limit the user to input using keyboard and mouse than interact naturally without any touch or mechanical devices. Gesture recognition can make more redundant use of fingers or hands to point at something to work and move accordingly.

There are two type of Gestures in computer interface:

- Offline gestures
- Online gestures

Dlib:

Dlib is a new C++ toolkit consists of machine learning algorithms and tools to create complex softwares in C++ to solve real world problems. It is used in industry as well as academia with different purpose such as building robotics, mobile phones, embedded system devices, and high performing computing environments. It is open source that allows, it use in any application with free of charge.

Major Features of Dlib:

1- Documentation
2- High quality portable code
3- Numerical Algorithm
4- Graphical Model Inference Algorithms
5- Image Processing
6- Machine learning Algorithms

7- Networking
8- Graphical User Interfaces
9- Threading
10- Data Compression and Integrity Algorithms
11- General Utilities
12- Testing

PyAutoGUI:

- It is a cross-platform GUI automation Python module .
- It is used to control the mouse & keyboard programmatically.
- The API of PyAutoGUI is designed to be as simple.
- It works on Windows, Linux, and macOS, and runs on Python 2, 3.

Features of PyAutoGUI:

- Clicking and moving the mouse or just typing in the windows of other applications.
- Sending keystrokes to applications (for example, to fill out forms).
- Taking screenshots of a given image and find it on the screen.
- Displaying the message box for user while the automation (GUI) scripts runs on.
- Locate an application’s window, and move, resize, maximize, minimize, or close it (Windows-only, currently)

How Does PyAutoGUI Work?

- On macOS, PyAutoGUI uses the rubicon-objc module to access the Cocoa API.
- On Linux, PyAutoGUI uses the Xlib module to access the X11 or X Window System.
- On Windows, PyAutoGUI accesses the Windows API (also called the WinAPI or win32 API) through the built-in ctypes module.

OpenCV:

- OpenCV is programmed library developed by Intel, which consist of functions mainly aimed at real-time uses such as computer vision. It is later supported by Willow Garage then Itseez.
- It is cross platform and free to use under some open-source BSD Licence.
- It supports some models from deep learning frameworks like Torch, Tensorflow, PyTorch and caffe according to defined list of supported layers.
- It promotes the Open Vision Capsules, which is compatible, portable with all o0ther formats.
- OpenCV consists a statistical machine learning library that contains, Decision tree learning ,Gradient boosting trees, Random forest ,Support vector machine , Boosting ,Expectation-maximization algorithm, Deep neural networks , k-nearest neighbor algorithm ,Naive Bayes classifier ,Artificial neural networks

Application:

- 2D and 3D feature toolkits
- Recognition and Segmentation
- Stereopsis stereo vision: depth perception from 2 cameras
- Structure from motion (SFM)
- Motion tracking
- Augmented reality
- Egomotion estimation
- Facial recognition system
- Gesture recognition
- Human computer interaction (HCI) system
- Mobile robotics
- Motion understanding
- Object identification and detection

Matplotlib:

Matplotlib is an inclusive library for creating animated, static, and interactive visualizations in Python.

It includes an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like wxPython, Tkinter, Qt, or GTK+.

It makes easy things very easy and hard things possible to create.

1- Create

- It creates quality plot for publication with just a few lines of code
- It uses interactive figures that can pan, zoom and update...

2- Customize

- It takes full control of line styles, axes properties, font properties.
- It can export and embed to no. of files format in interactive environments.

3-Extend

- It can explore many functionality provided by third party packages.
- There is a lot of external learning source available for the Matplotlib.

3.3. CNN - Convolutional Neural Network

A Convolutional Neural Network (CNN/ConvNet) is a deep learning algorithm in the class of deep neural networks, mainly used to analyse visual imagery. CNN is a classification algorithm that can take images as input and classify images, group images by similarities and accomplish object recognition within picture. For example, CNNs are used to identify faces, street signs, individuals, tumours, platypuses and other characteristics of visual data. CNNs also have applications in recommender systems, medical image analysis, and natural language processing.

3.4. SVM - Support Vector Machine

A Support Vector Machine (SVM) is a discriminative classifier formally described through a separating hyperplane. In other words, given labelled training data (supervised learning), the algorithm gives an ideal hyperplane which categorizes the new examples of testing data. In two-dimensional space this hyperplane is a line isolating a plane into two sections wherein each class lay on either side.

4. Literature Survey:

The survey is concentrated to address the points, focus on the research questions, understanding of the research area, clarification of the meaning of the terms and proper identification of the framework and planning of the data collection approach. The important thing is to understand the domain of research in which head movement and eye detection is involved. During the literature survey, the attention is on the most proficient method to build up a model which can satisfy the necessities of physically disabled people and the model should be easy to understand.

<table>
<tr><th>Author Name</th><th>Proposed Technique</th><th>Key findings</th><th>Final Results</th></tr>
<tr><td>Anush Ananthakumar (2018) [1]</td><td>Efficient Gesture and Face Recognition for Time Bound Application</td><td><b>1-</b>Face extraction is done using the Viola-Jones facial recognition system that uses Haar like attribute along with Adaboost training.
<b>2-</b>Color segmentation is used for gesture recognition, that extract the tone of skin of detected face and also detect the position of hand.
<b>3</b>-Kanade-Lucas-Tomasi (KLT) algorithm is used to obtain the gesture by traking the hand.
<b>4-</b>This research uses the background subtraction model reduce the misclassification and extract the foreground. This type of technique highly improves the facial and gesture recognition performance in cluttered and surroundings.</td><td>To check the accuracy of the system with different conditions and image quality, the proposed face recognition model has been tested on many databases and the results yielded on few of which is as shown below:<table>
<tr><th>Dataset name</th><th>Training time</th><th>Accuracy</th></tr>
<tr><td>ORL database of faces</td><td>150.2 sec</td><td>98.61%</td></tr>
<tr><td>Caltech Faces 1999 database</td><td>220.5 sec</td><td>97.15%</td></tr>
<tr><td>Faces96 dataset</td><td>376.34 sec</td><td>97.01%</td></tr>
</table>So, the result shows that with less training time the proposed method can achieve high accuracy.</td></tr>
<tr><td></td><td>CV Based Gesture</td><td><b>1-</b>Picture procurement: Image is obtained by</td><td>The fundamental objective of paper is to build up a robust and</td></tr>
</table>

S. M. Ariful Hoque MD. Sadun Haq Md. Hasanuzzaman[2]	Recognition for Object Manipulation of System. **Technique-** This research proposes a method to recognize static hand signals. Perceived motions at that point can be utilized to control work area objects.	Kinect 2.0. Kinect SDK render the obtained picture up to 30 fps. **2-**In this training, 9 unique gestures are trained. For preparing, 5 diverse individuals are utilized. This framework is assessed on an exceptional dataset from contribution of these 5 people. This framework incorporates 9 predefined motions. In the wake of running the framework, first the COP is recognized utilizing fingertip. At that point the framework holds for 2 seconds for the motion to begin. after motion begins, the framework catches the data of 30 back to back casings.	intuitive framework utilizing gestures. These motions are then used to control and control distinctive desktop objects. This framework utilizes contour detection algorithm and graham scan algorithm to detect fingertip.9 unique gestures are be utilized to control motions. A lot more can be executed as feature vector that can recognize each of the 5 fingertips with unique excellence, yet for effortlessness just 9 has been picked.
Mohd. Baqir Khan, Kavya Mishra, Mohammed Abdul Qadeer (2017) [3]	Gesture Recognition using Open-CV **Technique:** To perform a specific task first an image is captured by the camera then applied to preprocessing to minimize noise and convert image RGB to grayscale. After that feature extraction performs for finding the shape of the palm. Hidden Marcov Model, and k-curvature are used for this.	According to this paper, we can map some specific operations or tasks with a particular gesture. For example, if the Hand is detected then main window will open.	By using GRS this idea developed an efficient human interaction with machine. this will have many applications like Remote control, Aid to physically challenged, Mouse control. GRS is implemented by using glove approach and colored marker approach.

Rohit Lal, Shital Chiddarwar (2019) [5]	Real Time HCI using Facial Gesutres. **Technique-** This paper presents an algorithm that is useful for the physically challenged people. The algorithm developed in the paper find to be accurate with high facial detection accuracy which uses a mathematical equation to detect the eye whether open or not.	**1.** Facial landmark points are detected and generated by using the popular python library Dlib. **2.** Face detection is done by Haar cascade with adaboost algorithm to train classifier. **3.** Smile detection is done by suing the viola Jones algorithm. **4.** A scalar expression is developed to detect the of blink of eyes.	The proposed model work very accurately in case of small datasets. It is also tested over various functionality like surfing, playing games and doing other PC task. The model has a limitation that it is tested on RGB, in case on low lighting it is not able to detect the facial landmarks. Thus, the functionality may affect.
Rushikesh T. Bankar, Dr. Suresh S. Salankar (2015) [6]	Head Gesture Recognition System Using Gesture Cam **Technique-** This paper is mainly focused on the head movement, face tracking and detection of obstacle and using these actions to control some devices. It helps the handicapped people those who are not able to move their other body parts.	**1.** Gesture cam is used to capture the video of head movement. A smart FGPA camera is used to reduce the computation time because FGPA cam is sufficient to process the video frame by frame. **2.** Voila-Jones algorithm is used for the face detection and the image grabbed from the video are processed by this algorithm. **3.** An ultrasonic sensor is used for detection the obstacles that comes in the path to warn the user. Sensor emits the sound wave that can identify the hinderance in the path.	A microcontroller is used to handle the wheel chair equipped with FGPA camera that takes the user inputs and perform necessary action by recognizing the head gestures. The future work suggest that there is a need of more precise detection of head in all four directions and a web cam is used for more accurate results.
Hanene Elleuch, Ali Wali,	Smart Tablet Monitoring By a Real-time Head Movement And Eye	**1.** Voila-Jones algorithm is used for the face detection because it has fifteen times	The given model is tested on android tablet. It is found that the model has very high accuracy.

Adel M. Alimi (2014) [7]	Gestures Recognition System. **Technique-** This paper develops a system to monitor a tablet by using head movement and eye gaze gesture. Simple actions like opening a web browser and controlling the tablet is performed. In order to remove the system computation delay cloud computing is used. It improves the real time monitoring of the system, because tablet resources like RAM & ROM is limited.	greater accuracy than other approaches. **2.** Support vector machine (SVM) algorithm is used to detect and tract the eyes because it works well under different lighting conditions. Once the face is detected the algorithm will work on the ROI zone that is eyes. **3.** Given model is tested on the 5 people with 8 different scenarios. It gives the best accuracy but the model suffers if the user wears the glasses and also if the lighting condition is low.	The head and eye gaze gestures can have recognized the moving objects easily. Proposed model works well under different lightning condition. Cloud computing has given boost to the computation power of the system make it possible to run the model on the small platform that have low storage capability.
Jilin Tu, Thomas Huang, Hai Tao [8]	Face as Mouse Through Visual Face Tracking **Technique-** This paper presents a technique for visual tracking of the human face. The proposed 3D model is able to track the various face movement like opening and closing of eyes and mouth. This model is successfully tasted on the Windows XP platform that illustrate the accuracy and robustness of the system. Various facial expressions are captured from the video frame. This model can help the physically	**1.** It uses set of vertices to marks the 3D geometry of the face. Some action units are identified using facial articulations are being made. **2.** It uses an equation for the modeling of the face. Face is captured via video frames and detection is done using the Adaboosting algorithm. In case of worst result, user is allowed to use the GUI system to fine tune the result. **3.** Mouse control module is implemented with three modes namely, joystick	This paper successfully proposed 3D model based visual faced tracker. Various mouse operations are performed using the three different modes. Camera mouse application gives the various idea like how user can interact with the computer system using the various facial gestures. Given algorithm is not very much optimal and robust. It needs further advancement to make it more efficient. We can gain various useful insights through this paper as various useful mathematical calculations are being done in this paper.

	disabled people to run the computer system and it can also help in the development of video games.	mode, direct mode & differential mode. **4.** There are various other gestures like opening and closing of mouth, are used to run the system seamlessly. Proposed model is tested in windows XP environment that concludes the better efficiency of the system.	
Anja Jackowski , Marion Gebhard, Roland Thietje (2017) [9]	Head Motion and Head Gesture Based Robot Control: A Usability Study **Technique-** This experiment developed an assistive robot system AMiCUS for the physically disabled people suffering from tetraplegia. This robot is tested on 30 subjects in which 24 are able-bodies and 6 are tetraplegics. Overall result of this research is very useful for the mankind. AMiCUS is controlled by the head gestures and head movement of the subject. AMiCUS is successfully tested in the laboratory under different circumstances and is able to help the tetraplegics in performing their daily life task at the workplace.	**1.** To capture the head motions AMiCUS uses inertial measurement unit FSM-9. Various pick and place task has been done to ensure the proper functioning of robot. An emergency button is also provided for any uncontrolled condition. **2.** There are some sets of predefined task and complex task using the cubes has been performed. After this experiment a questionnaire is done and objective and subjective evaluation is also performed. **3.** To train the developed system lots of trials are being done in order to confirm the working of the robot. Head gestures are used to control the robot as well as gripper.	The proposed work in this paper is very useful for the various pick and place operation in the laboratory. AMiCUS is able to control 3D dimensional rotation and translation DOFs of robot. During the experiment the robot is found to be robust as it does not perform and unintended operation. The analytical algorithm used for the detection of head motions need the further optimization. Another challenge is to use the machine learning algorithm for the speech recognition or eye detection. It will give the new wings to the existing research.

5. Proposed System:

5.1. HOG – Histogram of Oriented Gradients:

Histogram of Oriented Gradients is a Global feature descriptor which is often used to extract features from image data. It is widely used in computer vision and image processing tasks for object detection.

This technique counts the occurrence of gradient orientation in localized parts of an image. These include contour representations, texture features or shape descriptors.

To detect the edge features, if we only try to find the pixel is an edge or not. HOG can find the edge detection as well. This feature of HOG called the **gradient and orientation i.e;** magnitude and direction of the edges.

- In Addition, the orientations are calculated in the 'localized' parts. the complete image is divide into smaller portions for each region, after that gradients and orientation are calculated.

- At last, It will generate a Histogram for each of these regions in separate parts. The histograms can be created using the orientation and gradient of the pixel values, hence it is called 'Histogram of Oriented Gradients'.

In the HOG feature descriptor, the distribution (histograms) of directions of gradients (oriented gradients) are used as features. Gradients (x and y derivatives) of an image are useful because the magnitude of gradients is large around edges and corners (regions of abrupt intensity changes) and we know that edges and corners pack in a lot more information about object shape than flat regions.

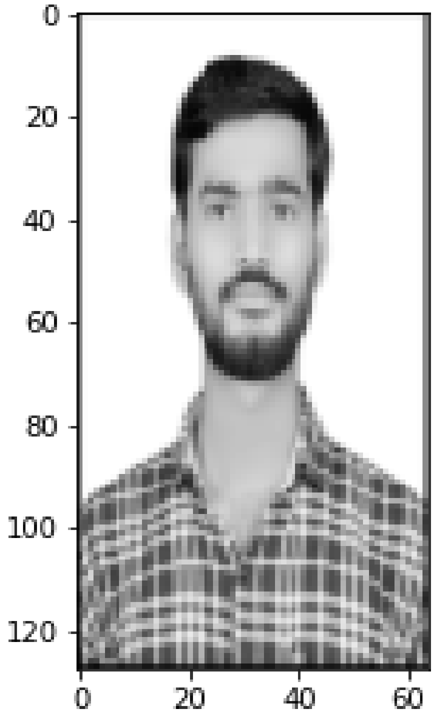
The distribution (histograms) of directions of gradients of HOG feature descriptor are used as feature. The derivative of x and y (gradients) of an image are useful because around the edges and corner the magnitude of gradients are large due to abrupt change in intensity. Edges and corners are packed in a lot more data(information) about objects shape than the flat regions.

5.1.1. Steps to create HOG:

Step 1: Pre-process the Data (64 x 128 x 3):

Original Image (398 x 512 x 3)

Figure 5.1.1.1: Original Image



Resize Image (64 x 128 x 3)

Figure 5.1.1.2: Image Resizing

Let’s understand this using an example.

In this pre-processing example firstly, image is cropped and resized into a particular size.

Step 2: Calculate the Gradients (in direction of x and y):

For calculating HOG descriptors, the first thing to calculate gradients of horizontal and vertical axis. After this, calculate the histogram of gradients.

It can be easily converted into using this kernel:

-1	0	1

-1
0
1

Figure 5.1.1.3: Kernel

These formulae are used to find magnitude and direction of gradient.

$$g = \sqrt{g_x^2 + g_y^2}$$

$$\theta = \arctan \frac{g_y}{g_x}$$

Gradient of image removes many non-essential information i.e.; constant colored background etc. and it highlights the outlines.

We can still look in the image and tell that it is image of person.

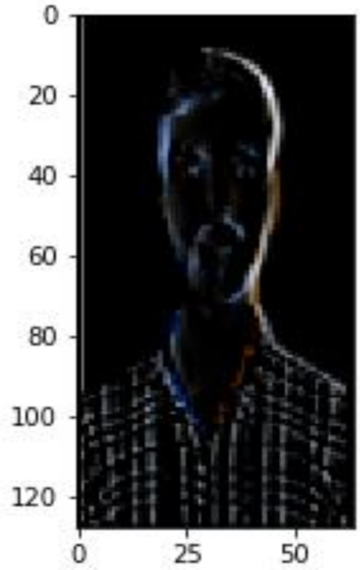


Figure 5.1.1.4: Absolute value of x axis gradient

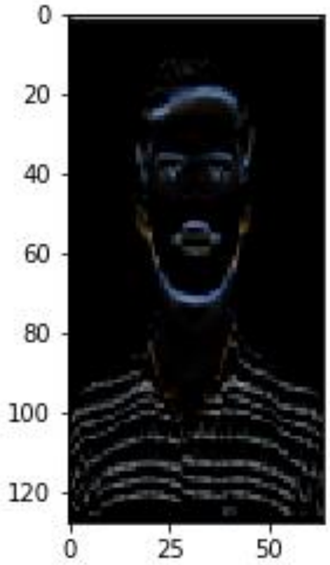


Figure 5.1.1.5: Absolute value of y axis gradient

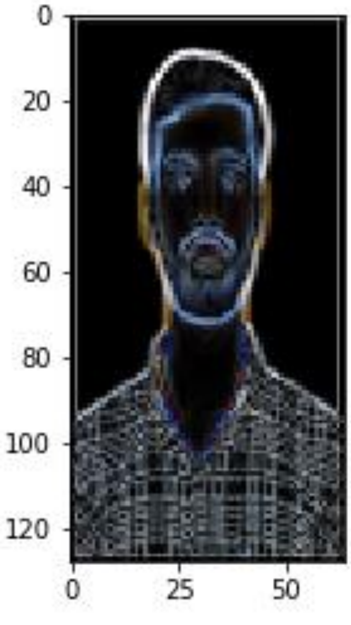


Figure 5.1.1.6: Magnitude (Total) of the gradient

Step 3: Calculating the Histogram of Gradients in 8×8 cells:

In this step, image will be divided into 8x8 cells and Hog will be calculated for each 8x8 cells.

The reason behind conversion into 8x8 cells is its precise representation, an 8×8 image patch will be equal to 8x8x3 = 192 pixel. The gradient of this portion contains 2 values only (magnitude and direction) per pixel which adds up to 8 x 8 x 2 = 128 numbers.

Patch makes it more robust to the noise and make its representation less sensitive to noise.

The histogram should contain a vector consist of 9 bins and the angels are 0,20,40,60…. 160.

Image of One 8×8 patch and its gradients look.

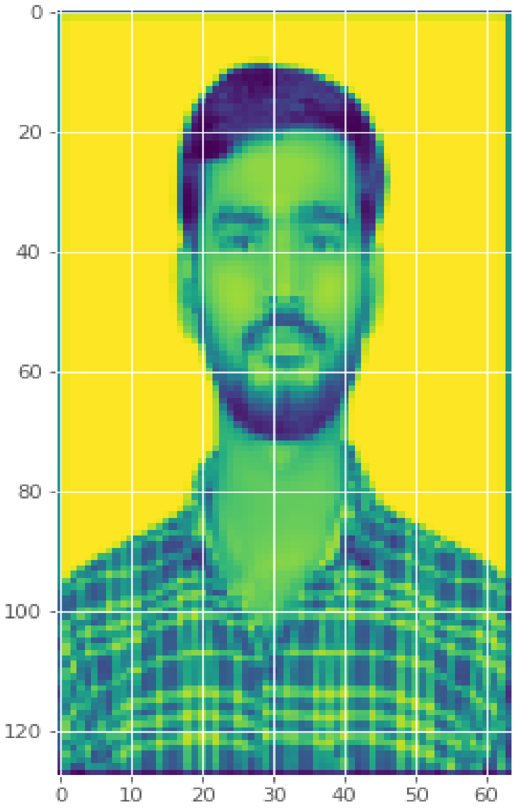


Figure 5.1.1.7: 8 x 8 cells in image

The 9-bin histogram of 8x8 cells are in below figure.

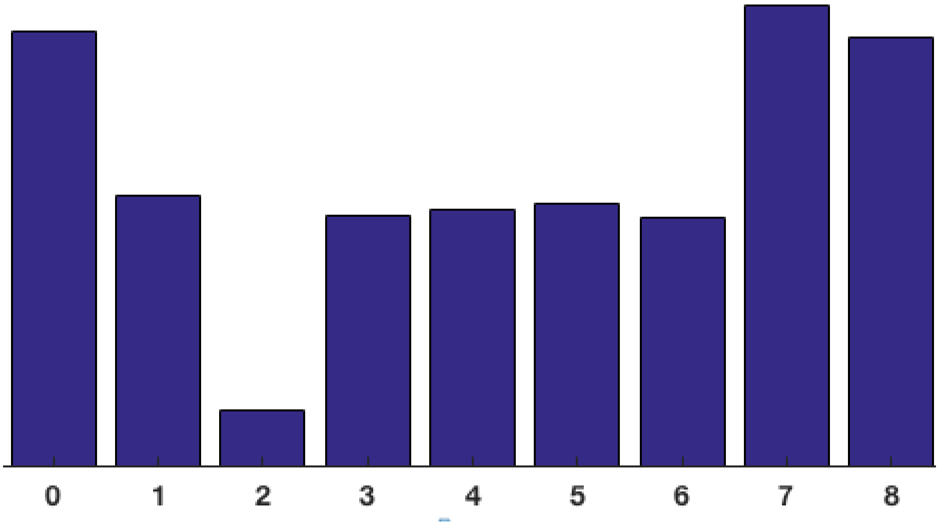


Figure 5.1.1.8: 9-bin histogram

In the given representation, the y-axis is 0 degrees. it can be seen the histogram has a lot of weight near 0 and 180 degrees, that is, In the patch the gradients are pointing either up or down.

Step 4: Calculating the HOG feature vectors

lets combine four 8×8 cells to create a 16×16 block. as we already know that each 8×8 cell has a 9×1 matrix for a histogram. So, we would have four 9×1 matrix or a single 36×1 matrix. To normalize this type of matrix, lets divide each of these values by the square root of the sum of squares of the values. Mathematically, for a given vector V:

$$V = [a1, a2, a3, \ldots.\ a36]$$

We will calculate the root of the sum of squares:

$$k = \surd(a1)2+ (a2)2+ (a3)2+ \ldots.\ (a36)2$$

And lets divide every value in the vector V with this value k:

$$\text{Normalised Vector} = \left(\frac{a_1}{k}, \frac{a_2}{k}, \frac{a_3}{k}, \dots \frac{a_{36}}{k} \right)$$

Visualizing Histogram of Oriented Gradients

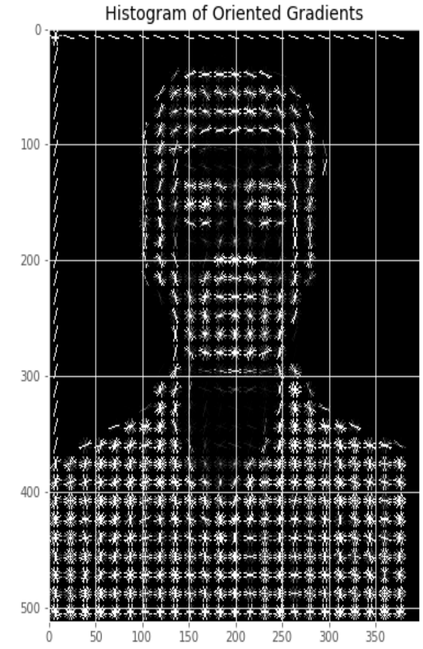


Figure 5.1.1.9: Visualization of HOG

The HOG descriptor of an image patch that can be visualized by plotting the 9×1 normalized histogram in the 8×8 cells. See image on the side. You can notice that dominant direction of the histogram captures the same shape of the person, especially around the torso and legs.

So, this is how a vector of feature descriptor is created for the images used for the training and the testing of in the model.

The training of dataset used to train the model is an xml file that lists the images in the training dataset and also contains the positions of the face boxes. To create XML files, we can use the imglab tool. It is a simple graphical tool used for labelling the objects in images with boxes.

Once, We will done with the generation of feature vector for a given image. An SVM is trained to detect whether this feature vector is of a face or not. A simple 1/0 binary classification will suffice.

Predicting the face:

We are using shape predictor function of Dlib to find the important details:

“This object is a tool that takes in an image region containing some object and outputs a set of point locations that define the pose of the object. The classic example of this is human face pose prediction, where you take an image of a human face as input and are expected to identify the locations of important facial landmarks such as the corners of the mouth and eyes, tip of the nose, and so forth.”

The shape predictor functions are using an ensemble of regression trees can be used to regress the location of facial landmarks from a sparse subset of intensity values extracted from an input image.

5.1.2 Drawing Function:

Drawing methods work with frameworks/pictures of subjective profundity. The limits of the shapes can be rendered only for 8-bit images. All the methods accept the parameter as color that uses an RGB value (and these color value may be obtained with CV_RGB or the constructor of scalar) for brightness of grayscale images and for color images for color images, color ordering in Blue, Green, Red. In open CV library functions imshow(), imread() and imwrite() expect these color values as parameters. All scalar function has following sign:

Scalar(blue_component, green_component, red_component[, alpha_component])

If the figure that drawing function want to draw is partially outside or completely outside, the drawing method or function need to handle the coordinates of pixel given with sub-pixel accuracy. The real pixel coordinates calculated as:

$$\texttt{Point}(x, y) \rightarrow \texttt{Point2f}(x * 2^{-\text{shift}}, y * 2^{-\text{shift}}).$$

This feature is mainly effective when rendering anti-aliased shapes.

RGB to Grayscale Conversion:

1- luminosity method /Weighted method
2- Average Method

1- Average method

This method is one of the simplest methods to convert an RGB image to grayscale image. We just need to take the average of the value of three colors.

For Example:

We have an RGB image with three color value r for red, g for green and b for blue. Then we just need to calculate the sum of all three value and divided by three.

It’s done in following way:

$$grayscale\ image = (R * G \ * \ B)/3$$

Let we have a color image as given below we calculate the average of each pixel of given image and replace each pixel with average to get grayscale image:

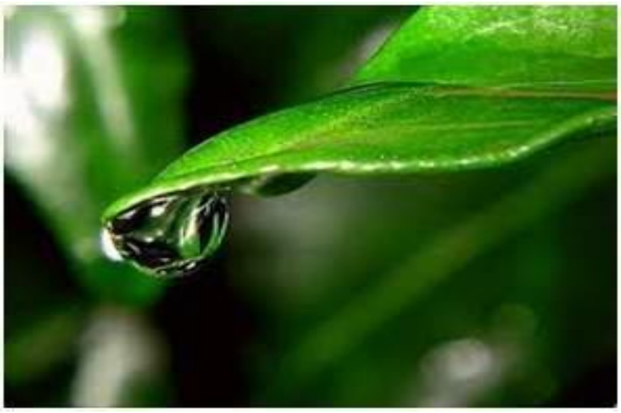


Figure 5.1.2.1: Average Method (1)

After replacing each pixel value we'll get following grayscale image:

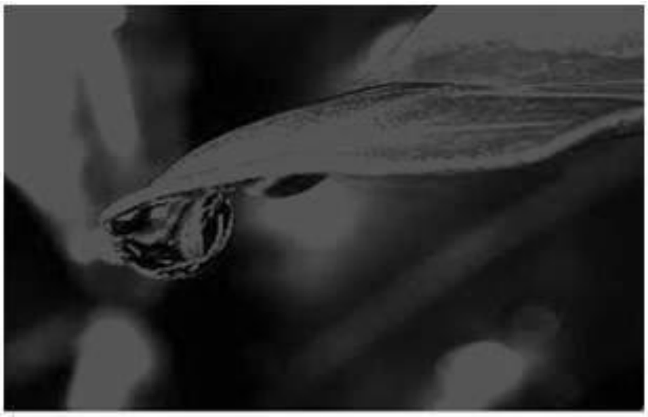


Figure 5.1.2.2: Average Method (2)

But in average method we detect that the brightness of image is not good enough. This problem is caused by the average value of three color. Three colors red, green and blue have three different value of their wavelength and have their own different contribution in the image. So, we need to take average in respect of their contribution instead of taking equal contribution. In average method we take contribution same as following

$$33\%\ of\ Red, 33\%\ of\ Green\ and\ 33\%\ of\ Blue$$

2- Luminosity method /Weighted method:

We have seen that the problem in average method due to their different wavelength. This method gives the solution of previous method problem. We know that the wavelength of red colour is higher in all three colours, and the wavelength of green colour is lesser than red colour and green colour also gives soothing effect of our eyes.

Due to soothing effect and lesser wavelength we need to increase the contribution of green colour and due to higher wavelength, we need to decrease the contribution of red colour and blue colour. According to above consideration the new formula for grayscale image is;

$$grayscale\ image = (0.3\ *\ R) + (0.59 * G) + (0.11\ *\ B))$$

According to above equation color all three colors Red, Green and Blue have contributed respectively 30%, 59% and 11%.

On applying above equation, we get more cleared grayscale image of an RGB image.

Let we have following coloured image:

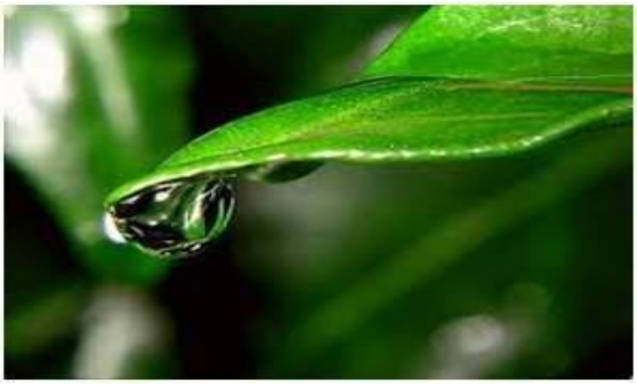


Figure 5.1.2.3: Luminosity method (1)

After applying luminosity method or weighted method we get following grayscale images:

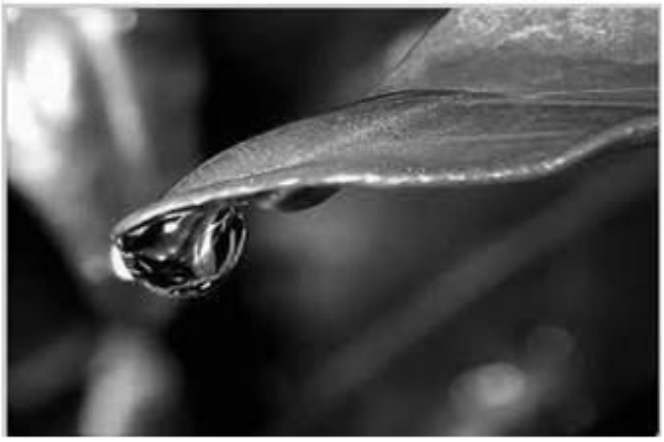
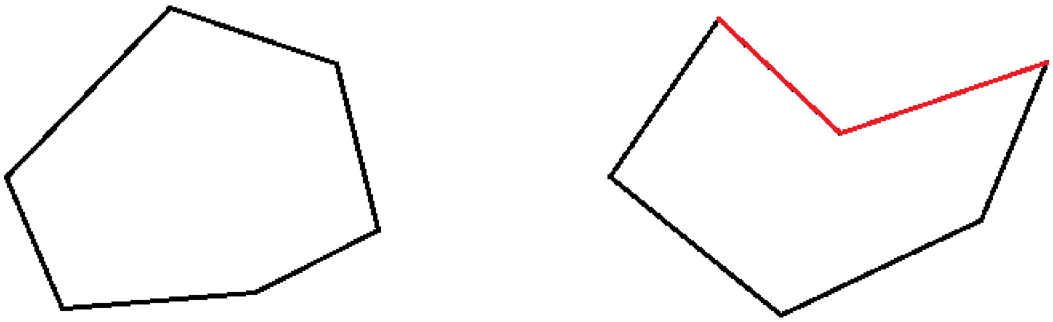


Figure 5.1.2.4: Luminosity method (2)

Convex hull:

The convex hull is a set of points of smallest polygon, this polygon enclosed all points those are present in set with minimum use of points. Convex word means that polygon has zero corners twisted inwards.



Convex Hull polygon

Non-Convex Hull polygon

Figure 5.1.2.5: Convex Hull

The red edge shows that the shape is concave instead of convex, concave is the opposite shape of convex.

Finding the convex hull:

Jarvis's Algorithm:

This is the simple approach to find convex points. First, we select left most point (or a point that has minimum x coordinate value). After selecting left most point, select a right most point (or a point that has highest value on x coordinate). Connect these two points and divide sets of points into two parts, lower parts and above parts.

Next, select a point that has highest perpendicular distance from divider line (a line draw between left most and right most point).

This algorithm takes $O(n\ h) time$ in worst case, where h is equal to the number of vertices in convex hull.

Algorithm:

- Find two points first 'left most' and second 'right most'.
- Divide points into two parts above and below the line joining left most and right most points.

- In the bottom half, select another left most point and add the point with lowest angle to the y-axis from the current point.
- Repeat scanning in same way in the above half.

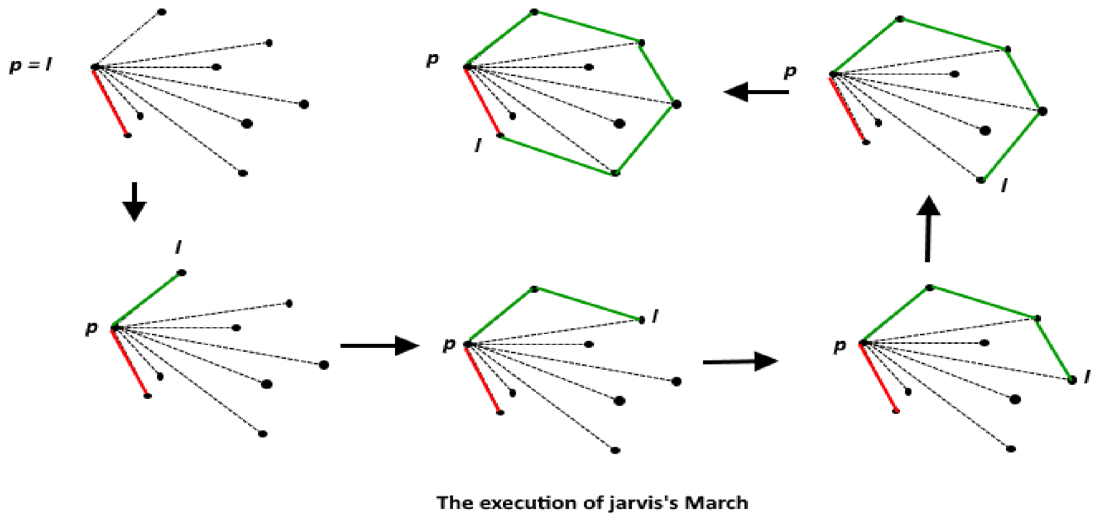


Figure 5.1.2.6: The execution of Jarvi’s March

Graham Scan:

This algorithm, selecting a point, that have minimum value on y-axis and also lie surly on convex hull. If multiple point has lowest y coordinate, then we select a point that has highest x coordinate.

After that we sort all the points according to their angle at which they lie. This angle also known ad polar angle. If we have two points with same angle, then we select one with minimum distance from starting point.

We use stack data structure to store these points on memory. During the traversing all sorted points, if we found convex shape then we push that point to stack and if we found concave shape we pop point on top of the stack.

This method has worst case complexity $O(n \log n)$ time.

Algorithm:

1. find a point P_0, that has minimum value on Y-axis.
2. Calculate polar angle of each points.
3. Sort all points with respect to their polar angle.
4. Initialize a stack, S
5. Push three starting points in stack
6. Push(S,P_0)

```
7.  Push(S,$P_1$)
8.  Push(S,$P_2$)
9.  For i in range (3,n):
10. While (angle( topnext(S), top(S) and $P_I$) == right turn)
11. Pop(S)
12. Push(S,$P_i$)
13. Return S
```

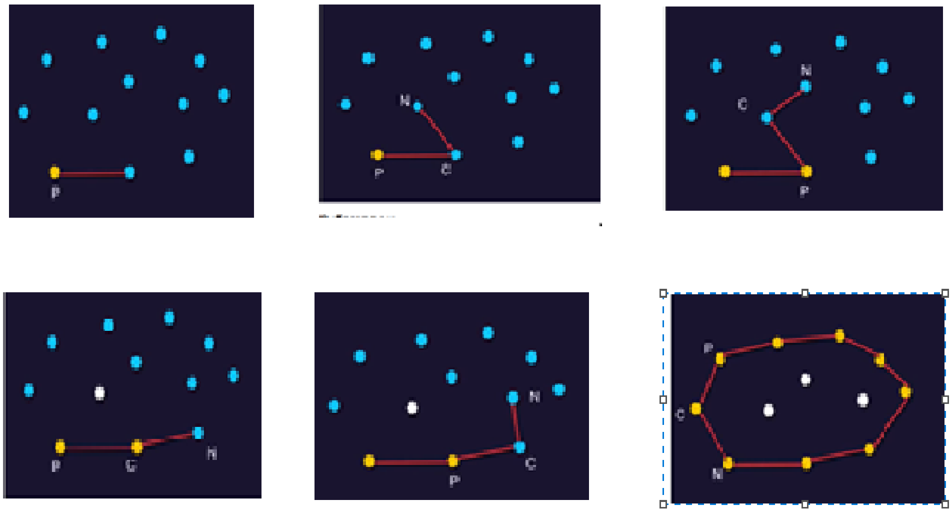


Figure 5.1.2.7: Graham Scan

5.2. Flow Chart:

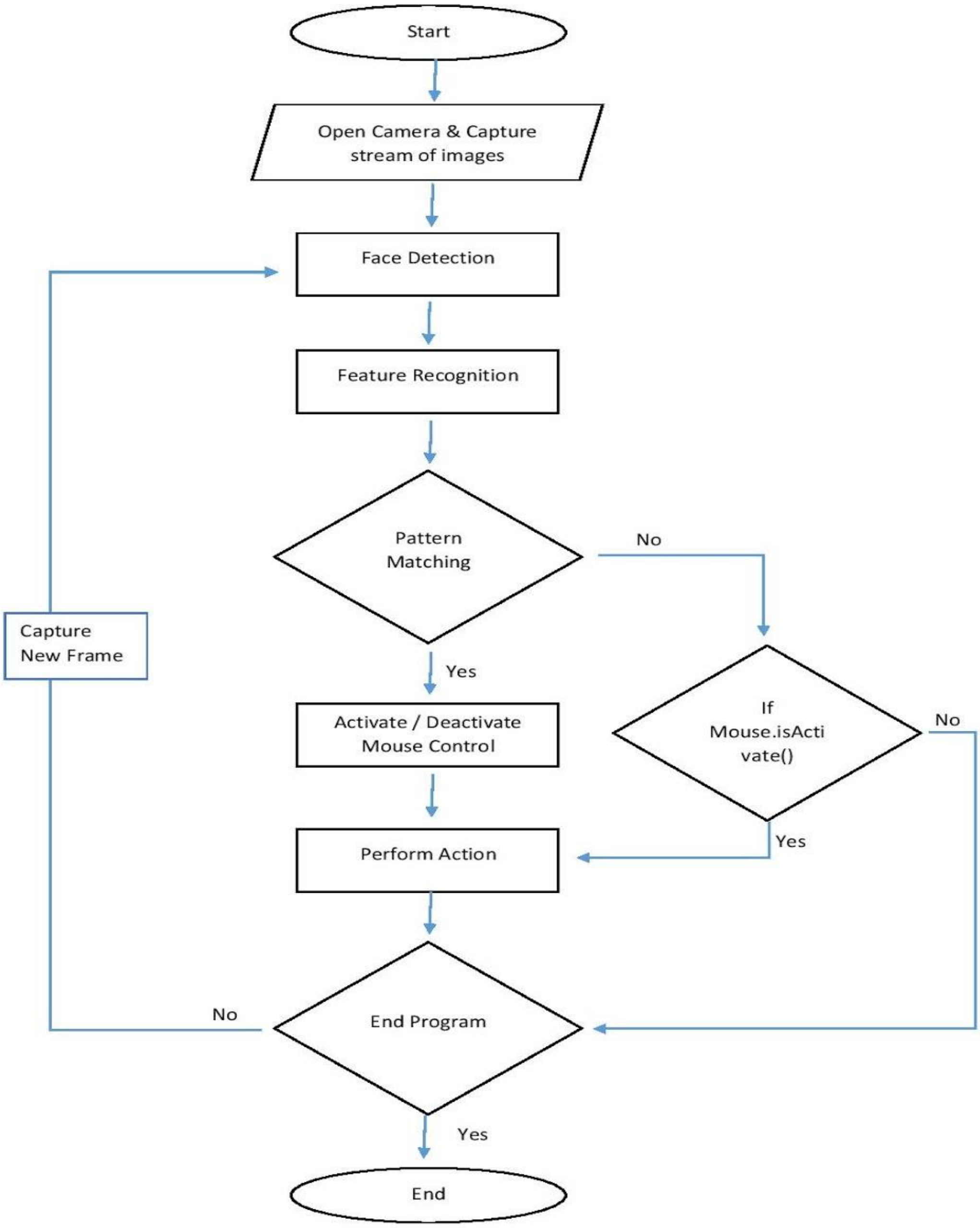


Figure 5.2: Flow Chart

(Above Flow Chart shows the sequence of routines that are running inside the system)

Above Flow Chart has following steps-:

1. Start program
2. Turning on the webcam and webcam continuously captures frames of image.
3. The system performs an action to detect face and recognize its features like eyes, nose.
4. The system finds the pattern and match the pattern.

5. If pattern has been matched, then system take an action of activation or deactivation of mouse.
6. Perform Action:

    a) Mouse control by moving of head.
    b) Scroll vertically, scroll horizontally.
    c) Right click and left click on eye blink.

5.3. Mouse Events:

Mouse events like (double click, left click and right click) would be perform according to blink count. We point noise tip to move mouse cursor if nose tip move left from its centre position then mouse cursor move to left and tip move to right then mouse cursor move to right same as for up and down. We capture each frame from webcam and detect facial landmark like (eye, nose).

This system will count blinks of eye but we detect that normally a person blinks their eyes in specific time duration. So, we need avoid those normal blinks or single blink in a fix time duration. To avoid this problem, we capture following graph between EAR (Eye Aspect Ratio) and time.

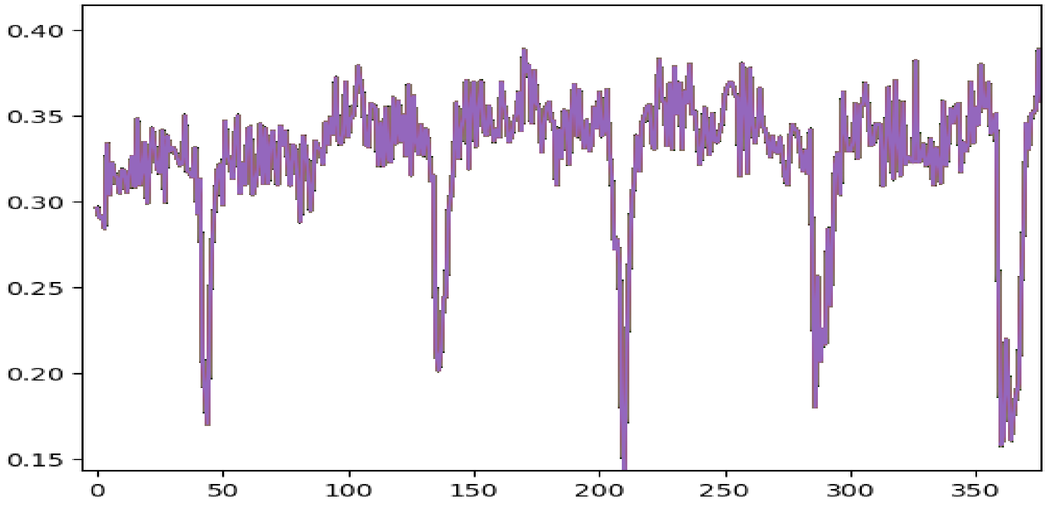


Figure 5.3: EAR (Eye Aspect Ratio)

We absorb above graph and find that a normal person performs single blinks in 2-2.5 seconds. So, we take 2.5 second as specific time to avoid routine single blinks made by user. To perform any action like Double Click, Left Click, Right Click user need to blink his eyes more than one time in time duration of 2.5 second. This system start time counting as the user perform single blink and after 2.5 second system check how many times eye blink by user in each 2.5 second. If user blink his eyes only one time in 2.5 second, system will avoid this one blink.

Double click -:

If blink counts in 2.5 second is two, then system will perform double click action.

We define following function to perform double click:

```
def doubleClick():
        mPoint = pyautogui.position();
        pyautogui.doubleClick(mPoint.x,mPoint.y)
        return
```

Right click -:

if blink count in 2.5 second is three, then system will perform right click action.

Following function is defined to perform Right Click:

```
def rightClick():
        mPoint = pyautogui.position();
        pyautogui.rightClick(mPoint.x,mPoint.y)
        return
```

Left click -:

if blink count in 2.5 second is four, then system will perform left click action and following function is used to perform left click:

```
def leftClick():
        mPoint = pyautogui.position();
        pyautogui.leftClick(mPoint.x,mPoint.y)
        return
```

5.3.1 Algorithm

```
1. Initialize video streams as vs;
2. load face predictor;
3. BLINT_COUNT: = 0;
4. TIME_COUNTER: = 0;
5. while(true)
    a. if fame is not available in vs:
        i. break;
    b. frame is captured frame from video_stream vs;
```

```
c.  convert frame RGB to grey scale;
d.  Detect Face in frame;
e.  Detect Eye after face detection;
f.  Calculate EAR (Eye Aspect Ratio) for each eye;
g.  If EAR < EAR_THRESHOLD:
        i.   start_time_counter ();
        ii.  BLINK_COUNT++;
h.  if TIME_COUNTER > THRESHOLD_TIME_COUNTER:
        i.   stop_time_counter ();
        ii.  TIME_COUNTER: =0;
        iii. BLINK_COUNT: = 0;
        iv.  IF BLINK_COUNT == 2:
                 Perform_double_click ();
        v.   ELSE IF BLINK_COUNT == 3:
                 Perform_left_click ();
        vi.  ELSE IF BLINK_COUNT == 4:
                 Perform_right_click ();
```

5.3.2 Facial Landmark detection and eye blink detection -:

Eye blink detected with useful facial landmark detection. Many libraries use to detect facial landmark. But this system uses DLIB pre-trained library with 68 index which help to find facial landmarks.

For eye blink detection we need to focus on index 37 to 40 for right eye and 43 to 46 for left eye.

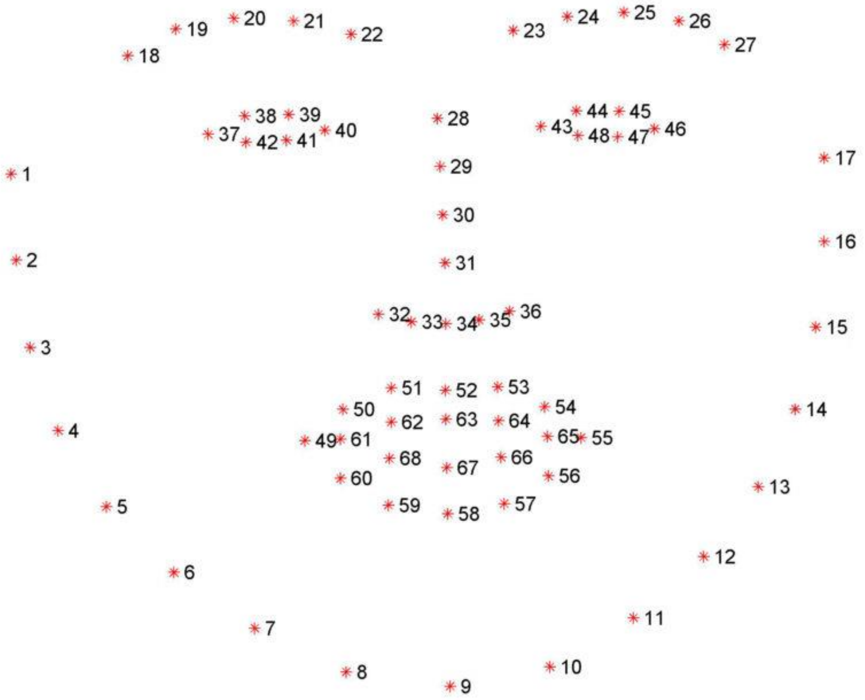


Figure 5.3.2: Dlib Facial Landmark points

5.3.3. Calculation of EAR (Eye Aspect Ratio)

EAR is the ratio Value of EAR are tending to zero when eye is closed and value of EAR is constant for open eye. We need to find threshold EAR value that will decide eye is open or close.

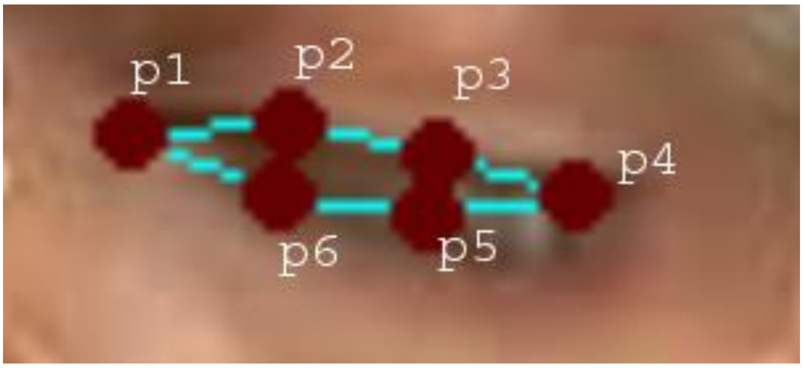


Figure 5.3.3.1: EAR (1)

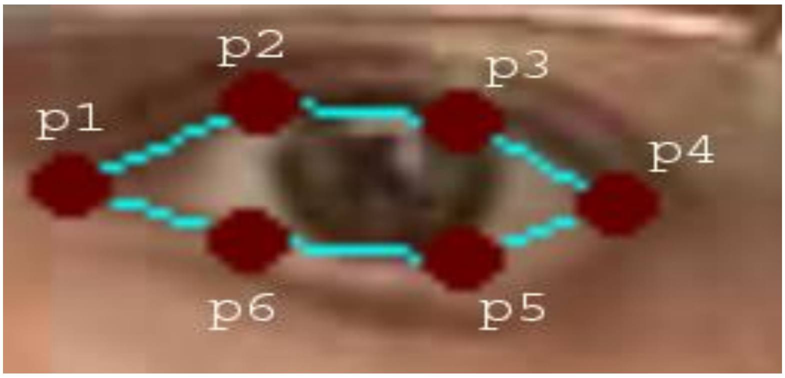


Figure 5.3.3.2: EAR (2)

EAR given by following formula-:

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

Following graph shows the fall in value of EAR when eye is closed and grow in value of EAR when eye is open.

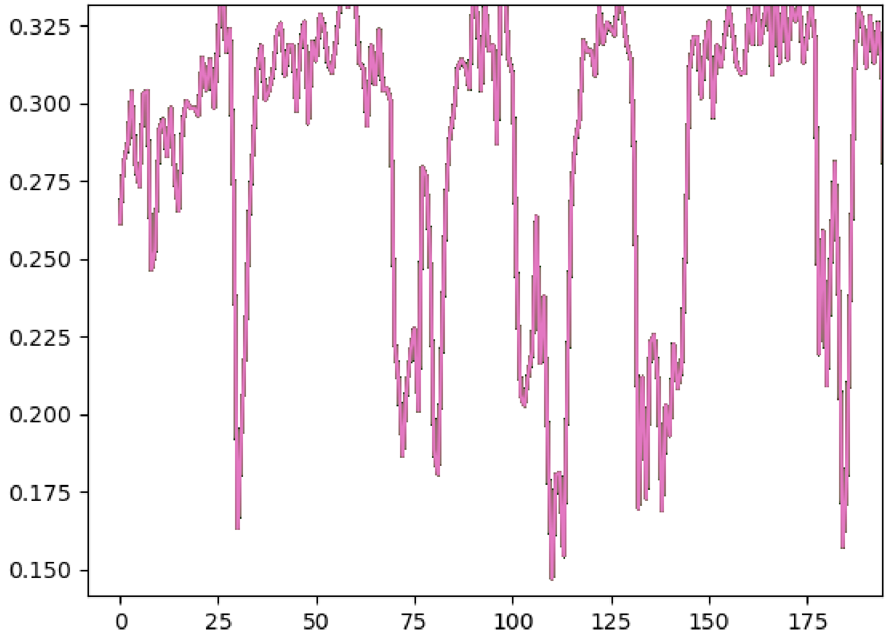
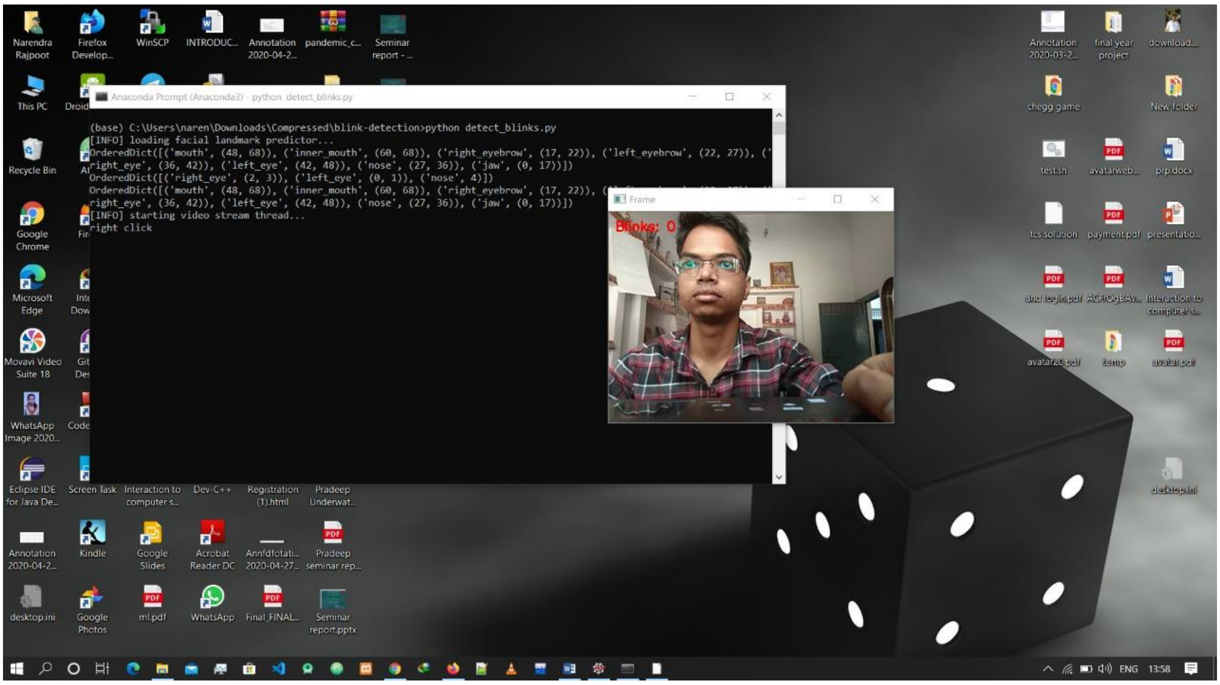


Figure 5.3.3.3: Threshold EAR

After observing above fig., we take threshold EAR is 0.22. When EAR is less than 0.22 eye will consider as close eye and EAR is greater than 0.22 EYE will consider as open eye.

5.4. Screenshots:

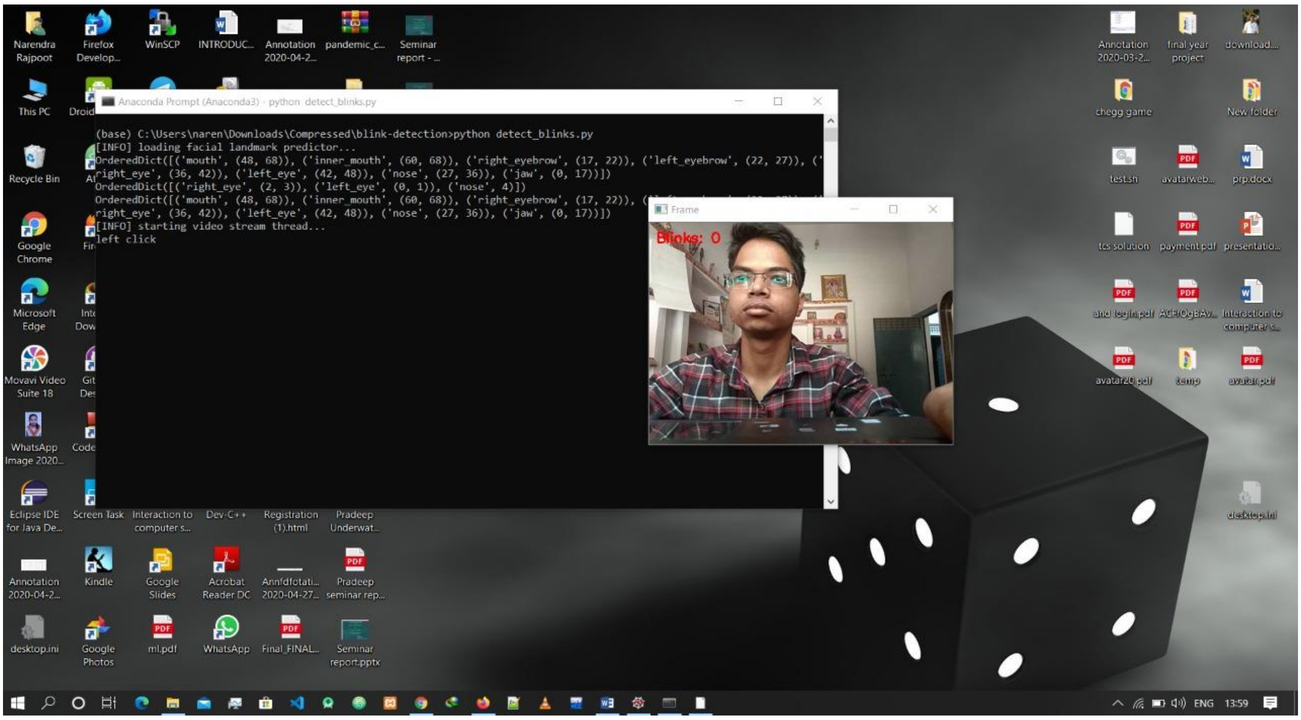
1: Right Click



```
(base) C:\Users\naren\Downloads\Compressed\blink-detection>python detect_blinks.py
[INFO] loading facial landmark predictor...
OrderedDict([('mouth', (48, 68)), ('inner_mouth', (60, 68)), ('right_eyebrow', (17, 22)), ('left_eyebrow', (22, 27)), ('
right_eye', (36, 42)), ('left_eye', (42, 48)), ('nose', (27, 36)), ('jaw', (0, 17))])
OrderedDict([('right_eye', (2, 3)), ('left_eye', (0, 1)), ('nose', 4)])
OrderedDict([('mouth', (48, 68)), ('inner_mouth', (60, 68)), ('right_eyebrow', (17, 22)), (
right_eye', (36, 42)), ('left_eye', (42, 48)), ('nose', (27, 36)), ('jaw', (0, 17))])
[INFO] starting video stream thread...
right click
```

Figure 5.4.1: Right Click

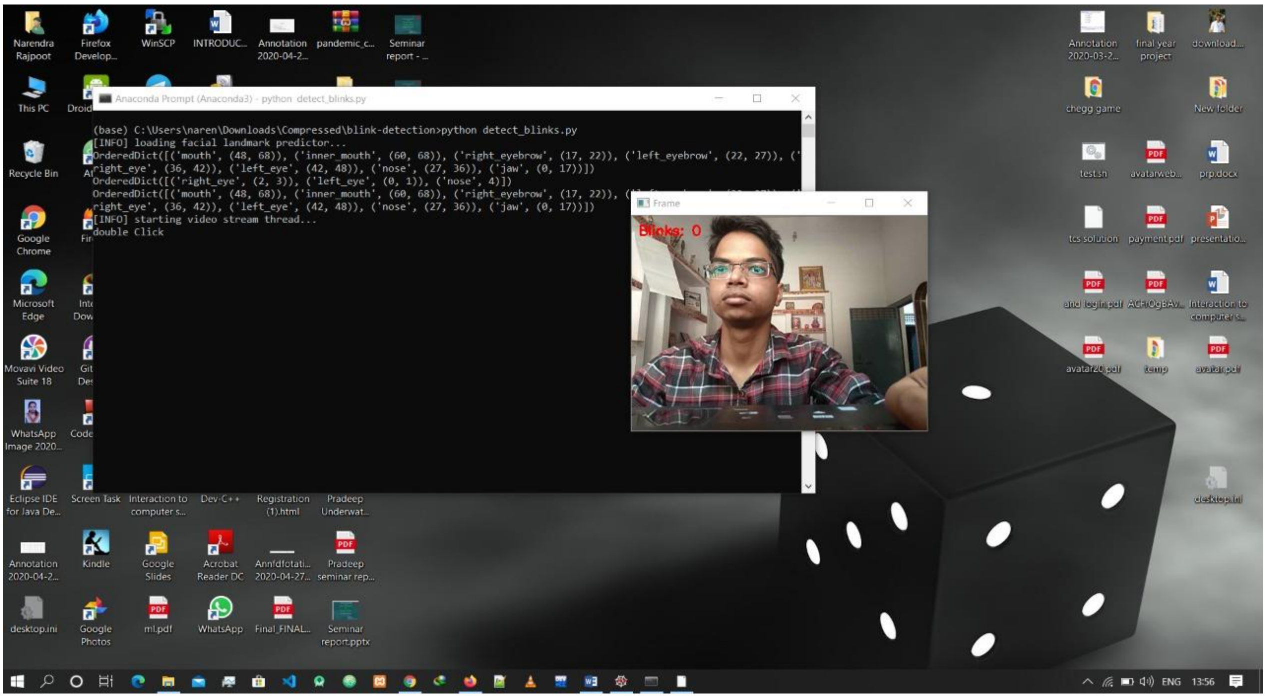
2: Left Click



```
(base) C:\Users\naren\Downloads\Compressed\blink-detection>python detect_blinks.py
[INFO] loading facial landmark predictor...
OrderedDict([('mouth', (48, 68)), ('inner_mouth', (60, 68)), ('right_eyebrow', (17, 22)), ('left_eyebrow', (22, 27)), ('
right_eye', (36, 42)), ('left_eye', (42, 48)), ('nose', (27, 36)), ('jaw', (0, 17))])
OrderedDict([('right_eye', (2, 3)), ('left_eye', (0, 1)), ('nose', 4)])
OrderedDict([('mouth', (48, 68)), ('inner_mouth', (60, 68)), ('right_eyebrow', (17, 22)), (
right_eye', (36, 42)), ('left_eye', (42, 48)), ('nose', (27, 36)), ('jaw', (0, 17))])
[INFO] starting video stream thread...
left click
```

Figure 5.4.3: Left Click

3 – Double Click



```
(base) C:\Users\naren\Downloads\Compressed\blink-detection>python detect_blinks.py
[INFO] loading facial landmark predictor...
OrderedDict([('mouth', (48, 68)), ('inner_mouth', (60, 68)), ('right_eyebrow', (17, 22)), ('left_eyebrow', (22, 27)), ('
right_eye', (36, 42)), ('left_eye', (42, 48)), ('nose', (27, 36)), ('jaw', (0, 17))])
OrderedDict([('right_eye', (2, 3)), ('left_eye', (0, 1)), ('nose', 4)])
OrderedDict([('mouth', (48, 68)), ('inner_mouth', (60, 68)), ('right_eyebrow', (17, 22)), (
right_eye', (36, 42)), ('left_eye', (42, 48)), ('nose', (27, 36)), ('jaw', (0, 17))])
[INFO] starting video stream thread...
double click
```

Figure 5.4.3: Double Click

6. Evaluation of Proposed System/Result:

This model is tested on Laptop having following configuration:

1. Operating System : Windows 10 x64,
2. CPU : i5 core, 2.30GHz,
3. Webcam : 2MP

And we tested this system in various light intensity like high, moderate and low light. Also, we add glass as measurement. We tested this system four times in same light intensity and we conclude the following four tables. In each condition we check

TABLE - I

Test - I

Light Intensity	Having Glasses	Eye Detected	Blink Detected
High	Yes	Yes	Yes
	No	Yes	Yes
Moderate	Yes	Yes	No
	No	Yes	Yes
Low	Yes	No	No
	No	No	No

Table 6.1: Test - I

TABLE – II

Test - II

Light Intensity	Having Glasses	Eye Detected	Blink Detected
High	Yes	Yes	Yes
	No	Yes	Yes
Moderate	Yes	No	No
	No	Yes	No
Low	Yes	No	No
	No	Yes	No

Table 6.2: Test - II

TABLE – III

Test - III

Light Intensity	**Having Glasses**	**Eye Detected**	**Blink Detected**
High	Yes	Yes	Yes
	No	Yes	Yes
Moderate	Yes	No	No
	No	Yes	No
Low	Yes	No	No
	No	Yes	No

Table 6.3: Test - III

TABLE – IV

Test - IV

Light Intensity	**Having Glasses**	**Eye Detected**	**Blink Detected**
High	Yes	Yes	Yes
	No	Yes	Yes
Moderate	Yes	Yes	No
	No	Yes	Yes
Low	Yes	No	No
	No	Yes	No

Table 6.4: Test – IV

TABLE – V

Test - V

Light Intensity	**Having Glasses**	**Eye Detected**	**Blink Detected**
High	Yes	Yes	Yes
	No	Yes	Yes
Moderate	Yes	No	No
	No	Yes	No
Low	Yes	No	No
	No	Yes	No

Table 6.5: Test – V

According to the above data we conclude that our system gives better performance in High light intensity or in sun light than moderate light intensity or low light intensity and when the light intensity is moderate then system performance is better than performance in low light intensity and worse than performance in high light intensity or sun light. This system is not appropriate in low light intensity it will work fine in proper light intensity or in day light.

We tested performance of this system by taking distance (between user and webcam (d)) as a measurement parameter and get following result.

TABLE – VI

Test - VI

Distance(d) between user and webcam(cm)	**Eye Detected**	**Blink Detected**
c<15cm	No	No
15cm<d<30cm	Yes	Yes
30cm<d<60cm	Yes	Yes
60cm<d<100cm	Yes	No
100cm<d	Yes	No

Table 6.6: Test - VI

On the basis of above parameter, we conclude that this system has high accuracy when the distance value is in range 15 to 60 cm.

6.1 Confusion matrix in different scenario:

1. High Intensity Light with Glass

------------ Eye Detection

	Predicted		
Actual	**N=10**	**1**	**0**
	1	4	1
	0	1	4

$$\text{Accuracy} = \frac{TN+TP}{TP+TN+FP+FN}$$

$$= \frac{4+4}{4+4+1+1} = \frac{8}{10}$$

Accuracy = **80%**

2. High Intensity Light with Glass

------------ Blink Detection

	Predicted		
Actual	**N=10**	**1**	**0**
	1	3	0
	0	2	5

$$\text{Accuracy} = \frac{TN+TP}{TP+TN+FP+FN}$$

$$= \frac{3+5}{3+5+0+2} = \frac{8}{10}$$

Accuracy = **80%**

3. High Intensity Light Without Glass

------------ Eye Detection

	Predicted		
	N=10	**1**	**0**

Actual	**1**	4	0
	0	1	5

Accuracy = $\frac{TN+TP}{TP\ +TN\ +\ FP\ +FN}$

$= \frac{4+5}{4+5+0+1} \quad = \frac{9}{10}$

Accuracy = **90%**

4. High Intensity Light Without Glass

------------ Blink Detection

	Predicted		
Actual	**N=10**	**1**	**0**
	1	4	0
	0	1	5

Accuracy = $\frac{TN+TP}{TP\ +TN\ +\ FP\ +FN}$

$= \frac{4+5}{4+5+0+1} \ = \ \frac{9}{10}$

Accuracy = **90%**

5. Moderate Intensity Light with Glass

------------ Eye Detection

	Predicted		
Actual	**N=10**	**1**	**0**
	1	3	1
	0	2	4

Accuracy = $\frac{TN+TP}{TP\ +TN\ +\ FP\ +FN}$

$= \frac{3+4}{3+4+2+1} \ = \frac{7}{10}$

Accuracy = **70%**

6. Moderate Intensity Light with Glass

------------ Blink Detection

	Predicted		
Actual	**N=10**	**1**	**0**
	1	3	1
	0	2	4

Accuracy = $\frac{TN+TP}{TP\ +TN\ +\ FP\ +FN}$

$= \frac{3+4}{3+4+2+1} \quad = \frac{7}{10}$

Accuracy = **70%**

7. Moderate Intensity Light Without Glass

------------ Eye Detection

	Predicted		
Actual	**N=10**	**1**	**0**
	1	4	1
	0	1	4

Accuracy = $\frac{TN+TP}{TP\ +TN\ +\ FP\ +FN}$

$= \frac{4+4}{4+4+0+1} \quad = \frac{8}{10}$

Accuracy = **80%**

8. Moderate Intensity Light Without Glass

------------ Blink Detection

	Predicted		
Actual	**N=10**	**1**	**0**
	1	3	1
	0	2	4

Accuracy = $\frac{TN+TP}{TP\ +TN\ +\ FP\ +FN}$

$= \frac{3+4}{3+4+2+1} \quad = \frac{7}{10}$

Accuracy = **70%**

9. Low Intensity Light with Glass

------------ Eye Detection

Actual	**Predicted**		
	N=10	**1**	**0**
	1	2	3
	0	3	2

Accuracy = $\frac{TN+TP}{TP+TN+FP+FN}$

$$= \frac{2+2}{2+2+3+2} \quad = \frac{4}{10}$$

Accuracy = **40%**

10. Low Intensity Light with Glass

------------ Blink Detection

Actual	**Predicted**		
	N=10	**1**	**0**
	1	1	3
	0	4	2

Accuracy = $\frac{TN+TP}{TP+TN+FP+FN}$

$$= \frac{1+2}{1+2+3+4} \quad = \frac{3}{10}$$

Accuracy = **30%**

11. Low Intensity Light Without Glass

------------ Eye Detection

Actual	**Predicted**		
	N=10	**1**	**0**
	1	3	3
	0	2	2

Accuracy = $\frac{TN+TP}{TP+TN+FP+FN}$

$$= \frac{3+2}{3+2+3+2} \quad = \frac{5}{10}$$

Accuracy = **50%**

12. Low Intensity Light without Glass

------------ Blink Detection

	Predicted		
Actual	**N=10**	**1**	**0**
	1	2	3
	0	3	2

Accuracy = $\frac{TN+TP}{TP+TN+FP+FN}$

$$= \frac{2+2}{2+2+3+2} \quad = \frac{4}{10}$$

Accuracy = **40%**

6.2 Summary of above confusion matrix

Intensity of light		**Eye Detected**	**Blink Detected**
High	With Glass	80	80
	Without Glass	90	90
Moderate	With Glass	70	70
	Without Glass	80	70
Low	With Glass	40	30
	Without Glass	50	40

Table 6.3: Summary of above confusion matrix

7. Conclusion and Future Scope:

In this project, we proposed a model that interact to computer system based on eye and head gesture recognition using Deep Learning. This project is based on a model with recognition accuracy greater than the real-time models. Proposed model is able to recognize blink of eyes and gestures of the head. We are using eye aspect ratio in order to achieve the maximum accuracy and will perform the left, right and double click of mouse operations on respective left, right and both blink of eyes. The principle objective of this project is to build up a powerful and intuitive model utilizing eye flicker and head gesture. These motions are then used to control the system. This system uses HOG algorithm for feature descriptor and SVM algorithm for training the model.

In this project we are using HOG as a feature descriptor and Ensemble of Regression Trees for face alignment, HOG is an old technique to extract feature, Using CNN as feature extractor can be more efficient than HOG, how it works in real time is still on paper. We can implement some other feature descriptor and face alignment algorithm. In this project we are not using any activator to activate the movement of cursor. In future we can implement this part to in the project.

8. References:

[1] Anush Ananthakuma, Georgia Institute of Technology Atlanta, GA, 30332, “Efficient Face And Gesture Recognition For Time-Sensitive Application ”, in IEEE 2018.

[2] S. M. Ariful Hoque, MD. Sadun Haq, Md. Hasanuzzaman , “ Computer Vision Based Gesture Recognition for Desktop Object Manipulation”. International Conference on Innovation in Engineering and Technology (ICIET) 27-29 December, 2018

[3] Mohd. Baqir Khan, Kavya Mishra, Mohammed Abdul Qadeer, “ Gesture Recognition using Open-CV” 7th International Conference on Communication Systems and Network Technologies 2017

[4] Saransh Sharma, Samyak Jain, Khushboo.” A Static Hand Gesture and Face Recognition System For Blind People”. 6th International Conference on Signal Processing and Integrated Networks (SPIN) 2019

[5] Rohit Lal, Shital Chiddarwar.” Real Time Human Computer Intretaction Using Facial Gestures”. 10th ICCCNT 2019 July 6-8, 2019, IIT - Kanpur, Kanpur, India

[6] Rushikesh T. Bankar, Dr. Suresh S. Salankar.”Head Gesture Recognition System Using Gesture Cam “. 5th International Conference on Communication Systems and Network Technologies 2015

[7] Hanene Elleuch, Ali Wal, Adel M. Alimi. “Smart Tablet Monitoring By a Real-time Head Movement And Eye Gestures Recognition System” International Conference on Future Internet of Things and Cloud 2014

[8] Jilin Tu, Thomas Huang, Hai Tao. “Face as Mouse Through Visual Face Tracking” Proceedings of the Second Canadian Conference on Computer and Robot Vision (CRV’05)

[9] Anja Jackowski, Marion Gebhard, Roland Thietje. “Head Motion and Head Gesture Based Robot Control: A Usability Study” IEEE transactions on neural systems and rehabilitation engineering 2017

[10] Navneet Dalal, Bill Triggs. “Histograms of Oriented Gradients for Human Detection” INRIA Rhone-Alps, ˆ 655 avenue de l’Europe, Montbonnot 38334, France

[11] Vahid Kazemi, Josephine Sullivan. “One Millisecond Face Alignment with an Ensemble of Regression Trees” IEEE Explore 2014, Sweden