1. INTRODUCTION

Eye tracking technology, which is based on an eye tracker that measures the movement and positions of the eye has played an increasingly important role in psychology, marketing, and user interfaces. Eye trackers have existed for a number of years, but, early in the development of the field of eye tracking, the use of eye trackers was largely confined to laboratory experiments to observe the nature of human eye movements, rather than to use these movements as an actual control medium within a human-computer interaction (HCI). Because the cost of eye trackers was around 30,000 a decade ago, it was too expensive to consider use in real user-computer interfaces. In recent years, with the development of better and cheaper components for gaze interaction, low-cost eye trackers have been produced by several high-profile companies, such as
Tobii's EyeX tracker, GazePoint's GP3 tracker, and the Eye Tribe Tracker. As eye tracking gear gets cheaper, new applications with the concept of using eye tracking in HCI are clearly beginning to blossom.

Traditional user interfaces provide much more bandwidth from computer to user, such as images, animations, videos, and other media which can output large amounts of information rapidly. Whereas there are hardly any means of inputting comparably large amounts of information from users. The concept of HCI is to increase the bandwidth from user to computer with more natural and more convenient communication mechanisms. The eye is one of our mainly input mediums, and about 80 to 90 percent of the outside world information is obtained from the human eye. For multimedia communication from user to computer, the eye movements can be regarded as a pivotal real-time input medium, which is especially important for people with motor disability (such as persons with Amyotrophic Lateral Sclerosis). The research of eye tracking technique in user-computer dialogue is mainly focused on incorporating eye movements into the multimedia communication with computer in a convenient and natural way.

Although the eye control systems mentioned above are of benefit for users with physical or cognitive handicaps to interact with computers appropriately, the designed eye trackers are quite complicated and expensive. Users should wear inconvenient devices and make specific actions to control the system. To lower the threshold of usability for user, MastaLomaster developed a prototype of eye control system that is based on low-cost eye trackers. The system supports most commercial low-cost eye trackers, such as Tobii EyeX and Eye Tribe. However, user should choose the desired function first and then do the real interaction with computer, which goes against user intuition and it is not natural to use.

In order to provide more natural and more convenient communication mechanisms, we present an eye tracking based control system for user-computer dialogue in this paper. The system combines both the mouse functions and keyboard functions, so that users can use our system to achieve almost all of the inputs to the computer without traditional input equipment. It was oriented towards improving the reliability, mobility, and usability for user to interact with computer by only using their eyes.

1.1 MOTIVATION

The previous proposed systems used complex algorithms. They were based on the biometric identification techniques. Some needed to mount devices on the user like Lasers which was not feasible. Hence, our aim is to devise an application that will be cost effective and not be dependent on the biometrics but on the feature classifications of the user. It should use less hardware and simpler algorithms. The objective is to use such a system that will help the upper limb disabled who cannot use the traditional mouse or keyboard**.**

The purpose of this research is to explore and improve upon existing avenues in the eye gesture tracking system. Particularly those areas which can help physically disable individuals, enabling them to use computers and programmable controlled systems. Thus, such individuals could still take on their responsibilities, improve the quality of their lives and continue with their day-to-day tasks often without the need for a helping hand.

1.2 PROBLEM DEFINITION

Eye tracking technology, which is based on an eye tracker that measures the movement and positions of the eye has played an increasingly important role in psychology, marketing, and user interfaces. Eye trackers have existed for a number of years, but, early in the development of the field of eye tracking, the use of eye trackers was largely confined to laboratory experiments to observe the nature of human eye movements, rather than to use these movements as an actual control medium within a human-computer interaction (HCI). Because the cost of eye trackers was around $30,000 a decade ago, it was too expensive to consider use in real user computer interfaces.

-Mounting devices: These systems needed a mounted device like lasers or cameras on the user which became tedious.

-Biometric identification: The system used biometric identification for which the users had to register themselves before using the system. It wasn't open for all which has been rectified by the proposed application.

-Complex algorithms: The previous systems used many complex algorithms that needed a lot of calculations to be done depending on various markers. The new proposed algorithm that is the HOG based linear SVM algorithm is easier and faster, thus increasing the response time for quicker access.

1.3 OBJECTIVE OF PROJECT

- The purpose of this work is to design a generic eye-gesture control system.
- Which can effectively track eye-movements and enable the user to perform actions mapped to specific eye movements/gestures by using computer webcam.
- It detects the pupil from the user's face and then tracks its movements.
- Provide a cheap eye-tracking system.
- To control the cursor of a computer with eyes.
- Allow physically disabled people to use computers.
- To control a computer and communicate with other systems.
- To provide a hand free mouse control system.
- To provide a complete generic eye-gesture mouse control system.
- To provide a complete wire free mouse control system.
- Easy to control cursor movement of a mouse.

1.4 LIMITATIONS

Every system cannot be perfect and have limitations to which they are confined to. Similarly, the Eyeball based cursor movement control using opencv also has few limitations.

- It is mandatory to have a built-in camera in the system.
- The irregular movement of user's head would affect the accuracy of click function.

Only a single person face is recognized and performs the operations accordingly. Might cause chaos if there are two - three people in front of the system. Because the system finds it difficult to make a decision of whose face to detect.

2. LITERATURE SURVEY

The literature was studied to address the aims, understanding of the research area, focus on the research questions, planning of the data collection approach, clarification of the meaning of the terms and proper identification of the framework. The most important task was to understand the research domain in which eyes detection and cursor movement of a mouse is involved.

Going through the literature, the focus was on how to develop a system which can fulfill the needs of physically impaired individuals and the system should be very easy to understand.

A group over at MIT has created a system titled “The sixth sense”, the system aims to enhance human-computer interaction by using gestures from the hands and eyes. The entire system is mountable on the user’s head, so that it can be projected on to smooth surfaces (like walls) and used anywhere in the world. The problem is that, it doesn’t provide enhanced assistance and accessibility to the disabled nor does it produce a system that can interact with other compatible devices.

Though Drewes, Heiko (2009), presents a comprehensive overview, it was noted that most algorithms needed further refinement as they took tedious and longwinded approaches to calibration. A no-nonsense, agile approach was defined Schmidt, Jochen, in using structure from motion algorithm for human-computer interaction. This was exploited by Kassner, Moritz Philipp, and William Rhoades Patera, in using the same sfm (structure from motion) algorithm, optimizing it and extending its use as an efficient algorithm for pupil tracking. In order to achieve this goal, they developed a framework by the name of PUPIL, to critically inquire the relationship between a human subject and space to visualize this unique spatial experience and to enable its use for gaze gesture tracking.

In 2018, an eye tracking algorithm based on Hough transform was developed. This system detects the face and eyes of a person. It uses a webcam to detect user’s face and eyes. The system is based on Matlab. The issue in this system is of real-time tracking and time-speed issue. The system is quite slow and it needs a high-quality computer system to work properly which is costly.

In 2017 a better system was introduced by the authors. This system is developed for the paralytic patients. This system uses webcam through MATLAB and moves the mouse cursor by using the pupil of a person. The issue in this system is that it takes a lot of time in detecting the pupil of a person. It uses a lot of algorithms and techniques to detect the pupil.

In 2016 a Vision-based wearable eye-gaze tracking system was introduced. This system works using a high infrared camera. It detects the eyes of the person through the infrared cam. The issue in this system was that it is slow and costly.

In 2015, a Pupil centre coordinate detection using the circular Hough transform technique was introduced. In this system, the webcam uses Hough Transform Techniques to detect the pupil of a person. The issue in this system was that it takes a lot of time and is not a real-time system. It first captures the body after that, it moves to face then eyes and finally to the pupil taking a lot of time.

In 2014, a face and eye-controlled system were developed which were based on MATLAB. It uses a webcam to control the mouse by eye and face movement. The issue in this system is that this system only works in a few centimetres radius.

In 2013, a system was developed which used eye tracking system, this system is based on the pictogram selection. It uses different eye-tracking techniques to make the system reliable. The issue in this system is that if any liquid is found in eyes, it will not work. Like female use eyeliner or mascara in their eyes, so the system stops working in those situations.

2.1 INTRODUCTION

The main purpose of the literature review work was to survey previous studies on knowledge sharing and intranets. In this, we look into the details about the existing system and try to reduce the disadvantages of the existing system. We try to improve the performance and the efficiency of the new proposed system and also learn the advantages of proposed system.

Table 2.1: Analysis of literature

Years	Papers	tools	Issues
2018	[1,2]	Webcam Matlab camera circular Hough transform	Iris tracking is slow. Web camera Connectivity issue. Slow capturing during the video.
2017	[1,2]	Optical image through the webcam. HD camera using MATLAB Viola-Jones, Kanade-Lucas-Tomasi (KLT) algorithms	Less pupil detecting (KLT) algorithms fail in this paper Need high RAM minimum 8GB.

2016	[1,2]	A webcam mounted on the glasses. Monte Carlo approach is used. Wavelets transform (WT) AdaBoost as a machine learning algorithm.	Eye-gaze tracking system with ahigh-cost webcam. Ada boost algorithm.
2015	[1,2]	Infrared video camera MATLAB Circular Hough transforms technique. Infrared camera. MATLAB	Cursor stability higher noise pupil detection algorithm error.
2014	[1,2]	Optical axis. Lab VIEW application MATLAB. Webcam camera.	Navigate the mouse pointer, noise of the EOG signals The distance between users and camera error. real-time controlling.
2013	[1,2]	Pictogram selection is performed. Eyetracking techniques, webcam, MATLAB. Iris tracking, MATLAB	Any liquid on the eyelashes affects the algorithm performance. Stop working when image quality is low. Stop working when the external light is low.

2.2 EXISTING SYSTEM

Matlab detects the iris and control cursor. Eye movement-controlled wheel chair is existing one that controls the wheel chair by monitoring eye movement**.** In MATLAB it is difficult to predict the Centroid of eye so we go for OpenCV.

Disadvantages

- Difficult to implement
- High cost
- Complex system
- It was slower than the control system that only used eye tracking and the accuracy was low.
- The image processing algorithm could not accurately detect the acquired image with low quality
- and is not robust to light intensity
- The commercial eye trackers were prohibitively expensive to use in HCI.
- All of the eye tracking control systems mentioned were proposed with self-designed hardware and
- software. These systems were hard to achieve a widespread adoption, because the software and
- hardware design was closed source.

2.3 PROPOSED SYSTEM

The Human Computer Interaction based eye-controlled mouse consist of two parts namely person detection system and cursor control system. Firstly, the person's face is detected. The position of the eye is tracked. The movement of the eye is tracked and used as the input. The system is useful for operating the computer in case of mouse failure. People with upper disability will not be able to use the computer. So, the proposed system detects the facial features and map them to cursor.

Only requirement to operate the system is, individuals having eyes with good vision and ability to control the computer. Histogram of Oriented Gradients (HOG) feature descriptor with a linear SVM (Support Vector Machine) machine learning algorithm is used for face detection.

The system proposed in this works based on the following action:

- Squinting your eyes
- Winking
- Blinking
- Gazing

The methodology is as follows:

- Since the project is based on detecting the features of the face and mapping them to the cursor, the webcam needs to be accessed first, which means that the webcam will be opened.
- The frame-rate of the video is generally around 30 frames per second, so a frame at every 1/30th of a second will be used to be processed.
- This frame undergoes a set of processes before the features of the frame are detected and mapped to the cursor. And this process continuously takes place for every frame as a part of a loop.

Once the frame is extracted, the regions of the face need to be detected. Hence, the frames will undergo a set of image-processing functions to process the frame in a suitable way, so that it is easy for the program to detect the features such as eyes, mouth, nose, etc.

Advantages of the Purposed System

- Hands-free mouse cursor control system.
- Facilitating the incapacitated to use computers.
- Mouse pointer control through eye movements.
- Quick response time
- Customized processing
- Small memory factor
- Highly secure
- High quality receiving data
- Low power consumption
- High data speed and low cost
- Faster input process

- Easy to implement and use
- Real time eye tracking and eye gaze estimation is achieved through eye based human computer interaction provide.
- Simulating mouse functions, performing different mouse functions such as left click, right click, double click and so on using their eyes.
- Instead of using complicated designed equipment, the proposed system would provide extremely lightweight devices that are more convenient for user to use.
- The proposed system is oriented towards the possibility of being used widely, which supports most of the low-cost eye trackers that are affordable for the majority of users.
- The proposed system realizes all of the functions of regular input sources, including mouse. User can efficiently interact with computer by only using their eyes.
- The proposed system provides more natural and more convenient communication mechanism between user & computer.

3. SYSTEM ANALYSIS

It is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components. System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem-solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose. Analysis specifies what the system should do.

3.1 INTRODUCTION

In this phase the requirements are gathered and analysed. User's requirements are gathered in this phase. This phase is the main focus of the users and their interaction with the system.
There are few questions raised:

- Who is going to use the system?
- How will they use the system?
- What data should be input into the system?
- What data should be output by the system?

These general questions are answered during a requirement gathering phase. After requirement gathering these requirements are analysed for their validity and the possibility of incorporating the requirements in the system to be development is also studied.
Finally, a Requirement Specification document is created which serves the purpose of guideline for the next phase of the model.

3.2 SOFTWARE REQUIREMENTS

It deals with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or prerequisites are generally not included in the software installation package and need to be installed separately before the software is installed. The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product. The requirements can be obvious or hidden, known or unknown, expected or unexpected from client's point of view. We should try to understand what sort of requirements may arise in the requirement elicitation phase and what kinds of requirements are expected from the software system.

Functional Requirements

- Graphical User interface with the User.

Software Requirements

For developing the application, the following are the Software Requirements:

1. Python

Operating Systems supported

1. Windows 7
2. Windows XP
3. Windows 8

Technologies and Languages used to Develop

1. Python
2. Numpy - 1.13.3
3. OpenCV - 3.2.0
4. PyAutoGUI - 0.9.36
5. Dlib - 19.4.0
6. Imutils - 0.4.6

3.3 HARDWARE REQUIREMENTS

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, a hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application.

The hardware requirements that are required for this project are as follows:

- Processor: Pentium IV or higher
- RAM: 256 MB
- Space on Hard Disk: minimum 512MB

3.4 CONTENT DIAGRAM OF PROJECT

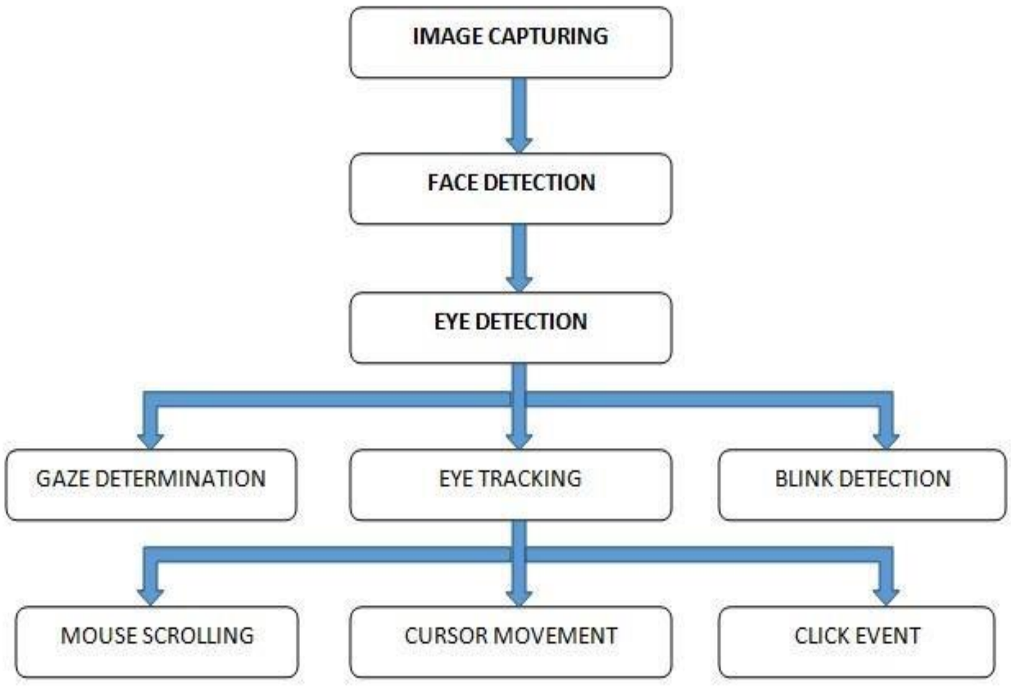


Figure 3.1: Content diagram of the project

4. SYSTEM DESIGN

The process of design involves “conceiving and planning out in mind and making a drawing, pattern or a sketch”. The system design transforms a logical representation of what a given system is required to do into the physical reality during development. Important design factors such as reliability, response time, throughput of the system, maintainability, expandability etc., should be taken into account. Design constraints like cost, hardware limitations, standard compliance etc should also be dealt with. The task of system design is to take the description and associate with it a specific set of facilities-men, machines (computing and other), accommodation, etc., to provide complete specifications of a workable system. This new system must provide for all of the essential data processing and it may also do some of those tasks identified during the work of analysis as optional extras. It must work within the imposed constraints and show improvement over the existing system. At the outset of design, a choice must be made between the main approaches.

Talks of ‘preliminary design’ concerned with identification analysis and selections of the major design options are available for development and implementation of a system. These options are most readily distinguished in terms of the physical facilities to be used for the processing who or what does the work.

4.1 INTRODUCTION

Software design is the process by which an agent creates a specification of a software artifact, intended to accomplish goals, using a set of primitive components and subject to constraints. Software design may refer to either "all the activity involved in conceptualizing, framing, implementing, commissioning, and ultimately modifying complex systems" or "the activity following requirements specification and before programming, as in a stylized software engineering process." Software design usually involves problem solving and planning a software solution. This includes both a lowlevel component design and a high level, architecture design.

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization.

Once the software requirements have been analysed and specified the software design involves four technical activities – design, coding, implementation and testing that are required to build and verify the software.

The design activities are of main importance in this phase, because in this activity, decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decisions have the final bearing upon reliability and maintainability of the system.

Design is the only way to accurately translate the customer's requirements into finished software or a system.

4.2 DFD/ER/UML DIAGRAMS

UML stands for Unified Modelling Language which is used in object-oriented software engineering. It is a standard language for specifying, visualizing, constructing, and documenting the artefacts of the software systems. UML is different from other common programming languages like C++, Java, and COBOL etc. It is pictorial language used to make software blueprints.

Although typically used in software engineering it is a rich language that can be used to model an application structure, behaviour and even business processes. There are 8 UML diagram types to help us model this behaviour.

There are two types of UML modelling:

- Structural Modelling
- Behavioural Modelling

Structural Modelling:

Structural model represents the framework for the system and this framework is the place where all other components exist. Hence, the class diagram, component diagram and deployment diagrams are part of structural modelling. They all represent the elements and the mechanism to assemble them.

Structural Modelling captures the static features of a system. They consist of the following:

i. Classes diagrams

ii. Objects diagrams

iii. Deployment diagrams

iv. Package diagrams

v. Composite structure diagram

vi. Component diagram

Behavioural Modelling:

Behavioural model describes the interaction in the system. It represents the interaction among the structural diagrams. Behavioural modelling shows the dynamic nature of the system.

They consist of the following:

i. Activity diagrams
ii. Interaction diagram
iii. Use case diagrams

All the above show the dynamic sequence of flow in a system.

4.2.1 USE CASE DIAGRAM

A use case diagram is a dynamic or behaviour diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. The "actors" are people or entities operating under defined roles within the system. Here the actors are user, web camera and the system. Use cases for user are start web camera, capture image, finding aspect ratio, eye blink detection, click event.

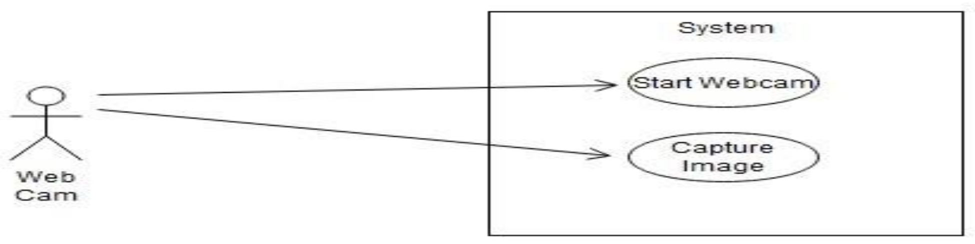


Figure 4.1*: Use case diagram for webcam*

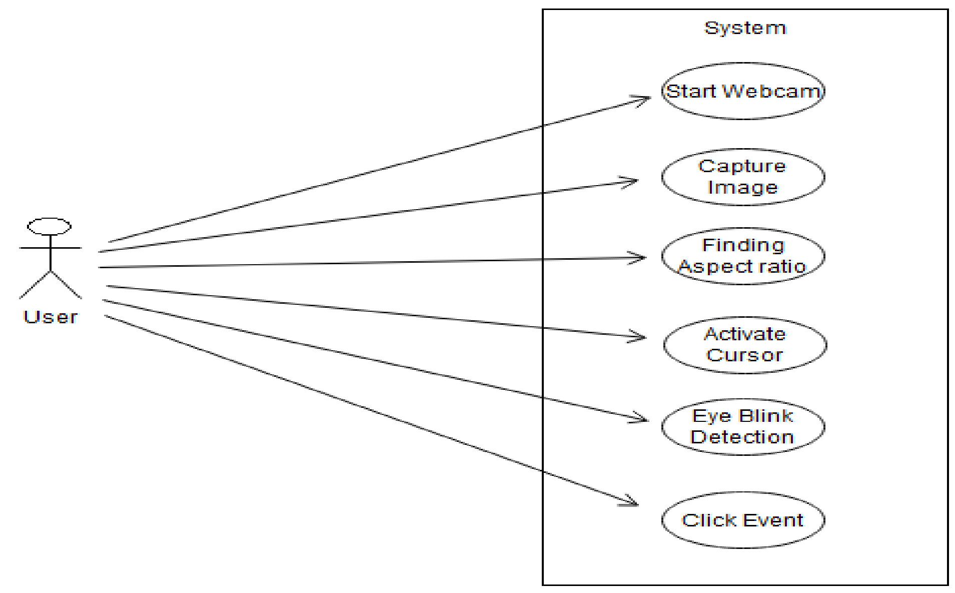


Figure 4.2: *Use case diagram for user*

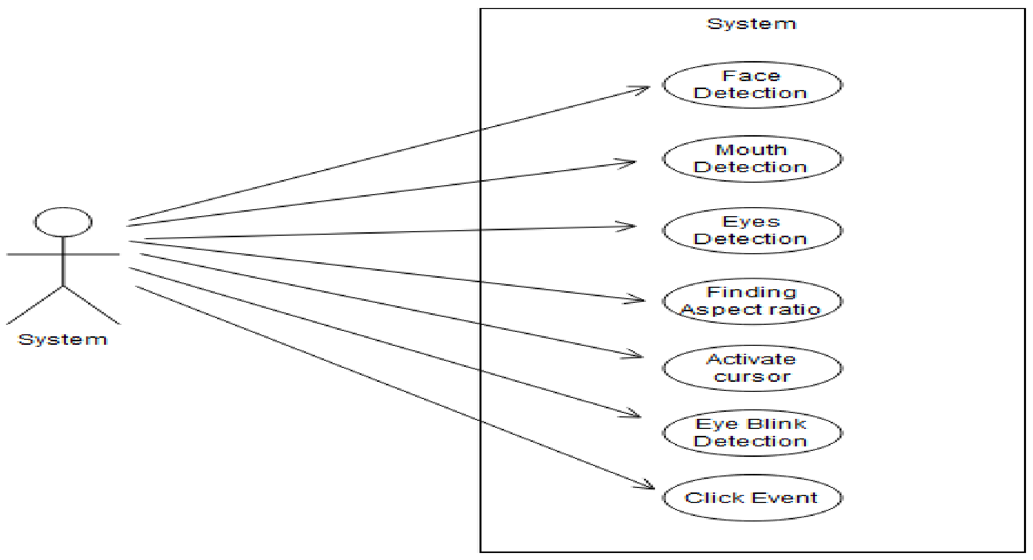


Figure 4.3: *Use case diagram for system*

4.2.2 CLASS DIAGRAM

Class diagrams are the main building blocks of every object-oriented method. It is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. The project consists of four classes named user, camera, system, cursor movements. User starts the process then camera will capture the image and the system will detect the face and facial features and map those features to cursor.

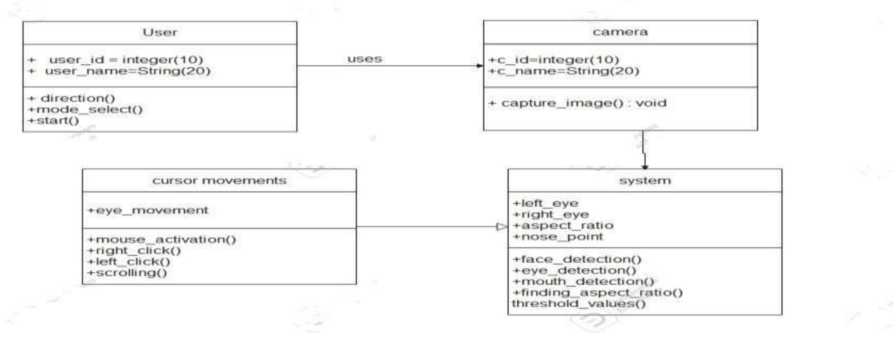


Figure 4.4: Class diagram

4.2.3 SEQUENCE DIAGRAM

A sequence diagram simply depicts interaction between objects in a sequential order i.e., the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems. nature but from a different angle.

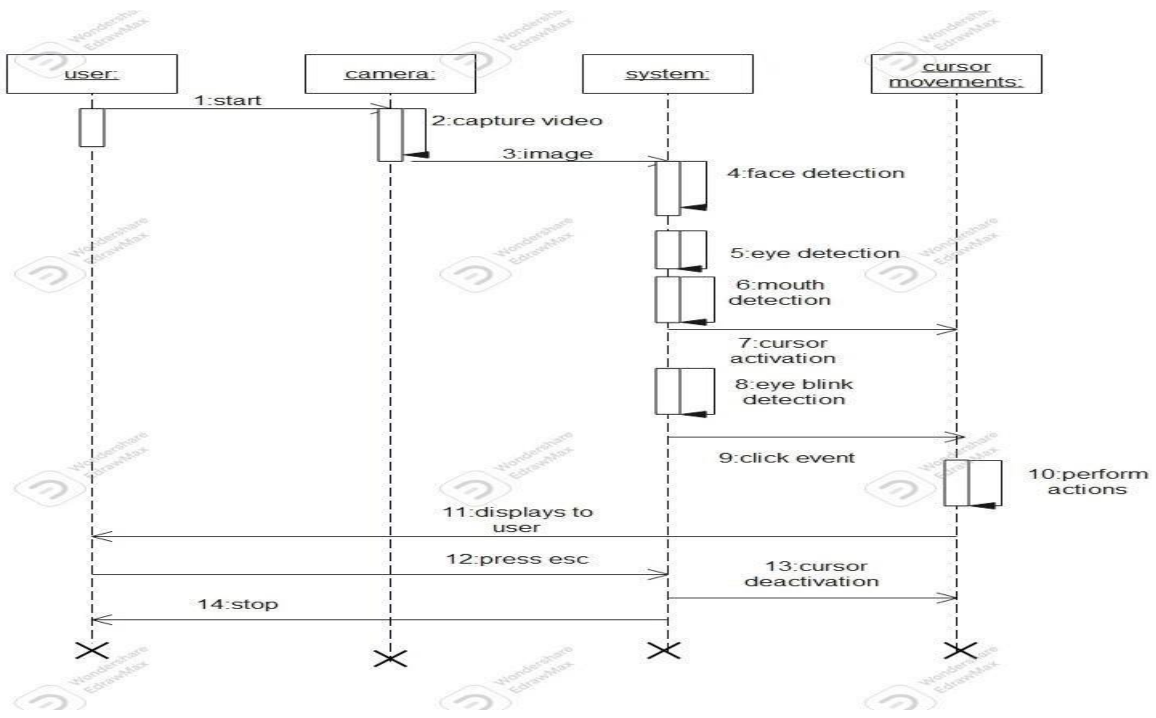


Figure 4.5: Sequence diagram

4.2.4 STATECHART DIAGRAM

State chart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. State chart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events. Initially user starts the web camera, it will capture the image and then detect the face, mouth and eyes of user. Then it will calculate the MAR and EAR and perform specific mouse movement.

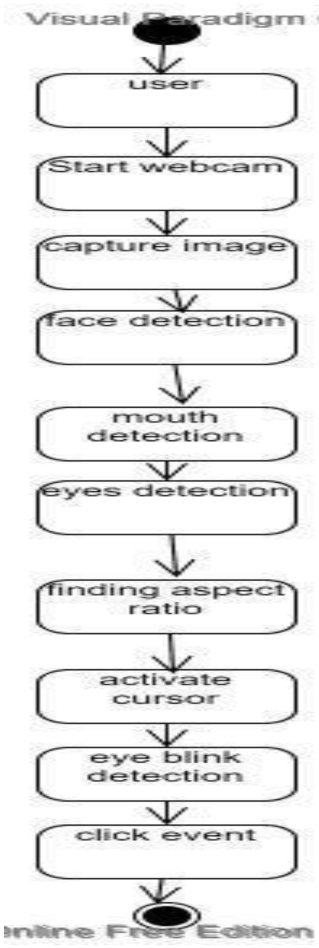


Figure 4.6: *State chart diagram*

5. IMPLEMENTATION AND RESULTS

5.1 INTRODUCTION

Functions are used for placing or storing the code which is to be repeated several times. For example, if we need same code, then we must have to write that code again and again. So, in order to remove this, we use functions.

Implementation is the stage where the theoretical design is turned into a working system. The most crucial stage in achieving a new successful system is giving confidence on the new system for the users that it will work efficiently and effectively.

The system can be implemented only after thorough testing is done and if it is found to work according to the specification.

It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the change over an evaluation of change over methods apart from planning. Two major tasks for preparing the implementation are education and training of the users and testing of the system.

5.2 METHOD OF IMPLEMENTATION

- **Resizing and converting from BGR to Gray**

The image is first flipped over the y-axis. Next, the image needs to be resized. The resize function refers to setting the new resolution of the image to any value as per the requirement. In this project, the new resolution is 640 X 480.

The data that we are using to detect the different parts of the face requires image of a grayscale format to give more accurate results. Hence, the image, i.e. the frame of the video from the webcam needs to undergo the process of converting its format from RGB to grayscale. Once the image is converted to a grayscale format, it can be used to locate the face and identify the features of the face.

- **Detection and Prediction of facial features**

To detect the face and the features, a prebuilt model is used in the project, which has the available values that can be interpreted by python to make sure that the face is located in the image. There is a function called ‘detector()’, made available by the models, which helps us to detect the face. After the face is detected, the features of the face can now be detected using the function ‘predictor’.

The function helps us to locate 68 points on any 2D image. These points correspond to different points on the face near the required parts such as eyes, mouth, etc. The values of the function that are obtained are in the form of 2D coordinates. Every one of the 68 points are essentially values of the x and y coordinates that, when connected, will roughly form an identifiable face. They are then stored as an array of values so that they can be arranged and used in the next step to connect any of the coordinates and draw a boundary to represent the required regions of the face. Four sets of arrays are taken as 4 different parts of these values which are stored in the array, to separately be stored as the coordinates to be connected to represent the required regions, those are the: Left eye, Right eye, nose and the mouth. Once the 4 arrays are prepared, boundaries, or 'contours' are drawn around the points using 3 of these arrays by connecting these points, using the 'drawcontour' function and the shape formed is around the two eyes and the mouth.

- **Mouth and Eye Aspect ratio**

Once the contours are drawn, it is necessary to have a reference for the shapes, which, when compared with, gives the program any information about any action made by these regions such as blinking, yawing, etc. These references are understood as ratios, between the 2D coordinates, and a change in the coordinates, essentially tell us that, the parts of the region of the face have moved from the regular position and an action has been performed. The system is built on predicting facial landmarks of the face. The Dib prebuilt model helps in fast and accurate face detection along with 68 2D facial landmarks as explained already. Here, Eye-Aspect Ratio (EAR) and mouth-aspect-ratio (MAR) are used to detect blinking/winking and yawing respectively. These actions are translated into mouse actions. Similarly, the MAR goes up when the mouth opens. This is used as an action to start and switch off the mouse. For example, if the ratio has increased, it can mean that the distances between the points representing the region of the face have changed and an action has been performed by the person. This action is supposed to be understood as the person trying to perform an operation using the mouse. Hence, for these functionalities to be made operational, there need to be some defined 'aspect ratios', which when cross a defined limit, interprets an action being performed.

- **Detection of Actions Performed by the face**

After the ratios are defined, the frame can now compare the ratios of the parts of the face with the ratios defined for different actions, of the current frame being processed. It is done using the 'if' statement. The actions which the program identifies are:

- **For activating the Mouse:**

The user needs to 'yaw' which is opening his mouth, vertically, in turn increasing the distance between the corresponding 2D points of the mouth. The algorithm detects the change in the distance by computing the ratio, and when this ratio crosses a specified threshold, the system is activated and the cursor can be moved. The user needs to place his nose towards, either the top, bottom, left or right of a rectangle that appears, to move the cursor in the corresponding direction. The more he is away from the rectangle, the faster is the movement of the cursor.

- **Left/Right Clicking**:

For clicking, he needs to close any one of his eye, and make sure to keep the other open. The program first checks whether the magnitude of the difference is greater than the prescribed threshold by using the difference between the ratios of the two eyes, to make sure that the user wants to perform either the left or right click, and does not want to scroll (For which both the eyes need to squint).

- **Scrolling**:

The user can scroll the mouse, either upwards or downwards. He needs to squint his eyes in such a way that the aspect ratio of both the eyes is less than the prescribed value. In this case, when the user places his nose outside the rectangle, the mouse performs scroll function, rather than moving the cursor. He can move his nose either above the rectangle to scroll upwards, or move it below the rectangle to scroll downwards.

5.2.1 TECHNOLOGIES USED

The technologies that are used in the project are:

I. Machine Learning
II. Python
III. NumPy - 1.13.3
IV. OpenCV - 3.2.0
V. PyAutoGUI - 0.9.36
VI. Dlib - 19.4.0
VII. Imutils - 0.4.6

i. Machine Learning:

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: ***The ability to learn***. Machine learning is actively being used today, perhaps in many more places than one would expect.

Classification of Machine Learning:

Machine learning implementations are classified into three major categories, depending on the nature of the learning "signal" or "response" available to a learning system which are as follows:

1. **Supervised learning:**

When an algorithm learns from example data and associated target responses that can consist of numeric values or string labels, such as classes or tags, in order to later predict the correct response when posed with new examples comes under the category of Supervised learning. This approach is indeed similar to human learning under the supervision of a teacher. The teacher provides good examples for the student to memorize, and the student then derives general rules from these specific examples.

2. **Unsupervised learning:**

When an algorithm learns from plain examples without any associated response, leaving to the algorithm to determine the data patterns on its own comes under the category of Unsupervised Learning. This type of algorithm tends to restructure the data into something else, such as new features that may represent a class or a new series of un-correlated values.
They are quite useful in providing humans with insights into the meaning of data and new useful inputs to supervised machine learning algorithms. As a kind of learning, it resembles the methods humans use to figure out that certain objects or events are from the same class, such as by observing the degree of similarity between objects. Some recommendation systems that you find on the web in the form of marketing automation are based on this type of learning.

3. **Reinforcement learning:**

When you present the algorithm with examples that lack labels, as in unsupervised learning. However, you can accompany an example with positive or negative feedback according to the solution the algorithm proposes comes under the category of Reinforcement learning, which is connected to applications for which the algorithm must make decisions (so the product is prescriptive, not just descriptive, as in unsupervised learning), and the decisions bear consequences.

4. **Semi-supervised learning:**

Where an incomplete training signal is given: a training set with some (often many) of the target outputs missing. There is a special case of this principle known as Transduction where the entire set of problem instances is known at learning time, except that part of the targets are missing.

Categorizing on the basis of required output:

Another categorization of machine learning tasks arises when one considers the desired output of a machine-learned system:

1. **Classification:** When inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more (multi-label classification) of these classes. This is typically tackled in a supervised way. Spam filtering is an example of classification, where the inputs are email (or other) messages and the classes are “spam” and “not spam”.

2. **Regression:** Which is also a supervised problem, A case when the outputs are continuous rather than discrete.

3. **Clustering:** When a set of inputs is to be divided into groups. Unlike in classification, the groups are not known beforehand, making this typically an unsupervised task.

Machine Learning comes into the picture when problems cannot be solved by means of typical approaches.

ii. Python

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL).

Some of the key advantages of learning Python.

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

Characteristics of Python:

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and crossplatform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

iii. NumPy

NumPy is a module for Python. The name is an acronym for "Numeric Python" or "Numerical Python". It is pronounced ‘NUM-py’ or less often ‘NUM-pee’. It is an extension module for Python, mostly written in C. This makes sure that the precompiled mathematical and numerical functions and functionalities of NumPy guarantee great execution speed.

Numeric, the ancestor of NumPy, was developed by Jim Hugunin. Another package Num array was also developed, having some additional functionalities. In 2005, Travis Oliphant created NumPy package by incorporating the features of Numarray into Numeric package. There are many contributors to this open-source project.

SciPy needs Numpy, as it is based on the data structures of Numpy and furthermore its basic creation and manipulation functions. It extends the capabilities of NumPy with further useful functions for minimization, regression, Fourier-transformation and many others.Both NumPy and SciPy are not part of a basic Python installation. They have to be installed after the Python installation.

Syntax:

```
import numpy as np
```

Advantages of using NumPy:

- Open Source
- Array oriented computing

iv. Computer Vision

Computer vision is a process by which we can understand the images and videos how they are stored and how we can manipulate and retrieve data from them. Computer Vision is the base or mostly used for Artificial Intelligence. Computer-Vision is playing a major role in self-driving cars, robotics as well as in photo correction apps.

OpenCV

OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as Numpy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

Applications of OpenCV: There are lots of applications which are solved using OpenCV, some of them are listed below

- face recognition
- Automated inspection and surveillance
- number of people – count (foot traffic in a mall, etc)
- Vehicle counting on highways along with their speeds
- Interactive art installations
- Ana moly (defect) detection in the manufacturing process (the odd defective products)

OpenCV Functionality

- Image/video I/O, processing, display (core, imgproc, highgui)
- Object/feature detection (objdetect, features2d, nonfree)
- Geometry-based monocular or stereo computer vision (calib3d, stitching, videostab)
- Computational photography (photo, video, superres)

Image-Processing

Image processing is a method to perform some operations on an image, in order to get an enhanced image and or to extract some useful information from it.

If we talk about the basic definition of image processing then **"Image processing is the analysis and manipulation of a digitized image, especially in order to improve its quality". Digital-Image:**

An image may be defined as a two-dimensional function f(x, y), where x and y are spatial(plane) coordinates, and the amplitude of fat any pair of coordinates (x, y) is called the intensity or grey level of the image at that point.

Image processing basically includes the following three steps:

- Importing the image
- Analysing and manipulating the image
- Output in which result can be altered image or report that is based on image analysis

To install OpenCV, one must have Python and PIP, preinstalled on their system.

PIP is a package management system used to install and manage software packages/libraries written in Python. These files are stored in a large “on-line repository” termed as Python Package Index (PyPI).

v. PyAutoGUI

PyAutoGUI lets your Python scripts control the mouse and keyboard to automate interactions with other applications. The API is designed to be as simple. PyAutoGUI works on Windows, macOS, and Linux, and runs on Python 2 and 3.

To install with pip, run: pip install pyautogui.

PyAutoGUI has several features:

- Moving the mouse and clicking or typing in the windows of other applications.
- Sending keystrokes to applications (for example, to fill out forms).
- Take screenshots, and given an image (for example, of a button or checkbox), find it on the screen.
- Locate an application’s window, and move, resize, maximize, minimize, or close it (Windowsonly, currently)

vi. Dlib

Dlib is a general-purpose cross-platform open-source software library written in the C++ programming language. Its design is heavily influenced by ideas from design by contract and component-based software engineering. This means it is, first and foremost, a collection of independent software components, each accompanied by extensive documentation and thorough debugging modes.

Davis King has been the primary author of dib since development began in 2002. In that time dlib has grown to include a wide variety of tools. In particular, it now contains software components for dealing with networking, threads, graphical interfaces, complex data structures, linear algebra, statistical machine learning, image processing, data mining, XML and text parsing, numerical optimization, Bayesian networks, and numerous other tasks. In recent years, much of the development has been focused on creating a broad set of statistical machine learning tools. However, dlib remains a general purpose library and welcomes contributions of high quality software components useful in any domain. **kernel, extension, and abstract**

Vii. imutils

A series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, and displaying Matplotlib images easier with OpenCV and **both** Python 2.7 and Python 3.

Provided you already have NumPy, SciPy, Matplotlib, and OpenCV already installed, the imutils package is completely pip-installable: $ pip install imutils .

5.3 EXPLANATIONS OF KEY FUNCTIONS

Source Code:

```
from imutils import face_utils
from utils import *
import numpy as np
import pyautogui      as pag
import imutils import dlib
import cv2
# Thresholds and consecutive frame length for triggering the mouse action.
MOUTH_AR_THRESH = 0.6
MOUTH_AR_CONSECUTIVE_FRAMES = 15
EYE_AR_THRESH = 0.19
EYE_AR_CONSECUTIVE_FRAMES = 15
WINK_AR_DIFF_THRESH = 0.04
WINK_AR_CLOSE_THRESH = 0.19
```

```
WINK_CONSECUTIVE_FRAMES = 10

# Initialize the frame counters for each action as well as
# Booleans used to indicate if action is performed or not
MOUTH_COUNTER = 0
EYE_COUNTER = 0
WINK_COUNTER = 0
INPUT_MODE = False
EYE_CLICK = False
LEFT_WINK = False
RIGHT_WINK = False
SCROLL_MODE = False
ANCHOR_POINT = (0, 0)
WHITE_COLOR = (255, 255, 255)
YELLOW_COLOR = (0, 255, 255)
RED_COLOR = (0, 0, 255)
GREEN_COLOR = (0, 255, 0)
BLUE_COLOR = (255, 0, 0)
BLACK_COLOR = (0, 0, 0)

# Initialize Dlib's face detector (HOG-based) and then create
# the facial landmark predictor shape_predictor =
"model/shape_predictor_68_face_landmarks.dat" detector =
dlib.get_frontal_face_detector() predictor = dlib.shape_predictor(shape_predictor)

# Grab the indexes of the facial landmarks for the left and
# Right eye, nose and mouth respectively
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
(nStart, nEnd) = face_utils.FACIAL_LANDMARKS_IDXS["nose"]
(mStart, mEnd) = face_utils.FACIAL_LANDMARKS_IDXS["mouth"]

# Video capture vid =
cv2.VideoCapture(0)
```

```
resolution_w = 1366
resolution_h = 768 cam_w = 640
cam_h = 480 unit_w =
resolution_w / cam_w unit_h =
resolution_h / cam_h

while True:
    # Grab the frame from the threaded video file stream, resize
    # it, and convert it to grayscale
# channels)    _, frame = vid.read()
frame =
cv2.flip(frame, 1)      frame = imutils.resize(frame, width=cam_w,
height=cam_h)     gray = cv2.cvtColor(frame,
cv2.COLOR_BGR2GRAY)

    # Detect faces in the grayscale frame       rects =
detector (gray, 0)

    # Loop over the face detections     if
len(rects) > 0:         rect = rects[0]
    else:
        cv2.imshow("Frame",   frame)
key   =   cv2.waitKey(1)   &   0xFF
continue

    # Determine the facial landmarks for the face region, then
    # convert the facial landmark (x, y)-coordinates to a NumPy
    # array     shape = predictor(gray, rect)     shape
= face_utils.shape_to_np(shape)

    # Extract the left and right eye coordinates, then use the
# coordinates to compute the eye aspect ratio for both eyes
mouth = shape[mStart:mEnd]     leftEye = shape[lStart:lEnd]
rightEye = shape[rStart:rEnd]     nose = shape[nStart:nEnd]
```

```
    # Because I flipped the frame, left is right, right is left.
    temp = leftEye     leftEye =
rightEye     rightEye = temp

    # Average the mouth aspect ratio together for both eyes
mar = mouth_aspect_ratio(mouth)     leftEAR =
eye_aspect_ratio(leftEye)     rightEAR =
eye_aspect_ratio(rightEye) ear = (leftEAR + rightEAR) /
2.0  diff_ear = np.abs(leftEAR - rightEAR)

        nose_point = (nose[3, 0], nose[3, 1])

        # Compute the convex hull for the left and right eye, then
    #  visualize  each  of  the  eyes              mouthHull  =
cv2.convexHull(mouth)     leftEyeHull =
cv2.convexHull(leftEye)     rightEyeHull = cv2.convexHull(rightEye)
cv2.drawContours(frame, [mouthHull], -1, YELLOW_COLOR, 1)     cv2.drawContours(frame,
[leftEyeHull], -1, YELLOW_COLOR, 1)     cv2.drawContours(frame, [rightEyeHull], -1,
YELLOW_COLOR, 1)
    for (x, y) in np.concatenate((mouth, leftEye, rightEye), axis=0):
        cv2.circle(frame, (x, y), 2, GREEN_COLOR, -1)
        # Check to see if the eye aspect ratio is below the blink
    # threshold, and if so, increment the blink frame counter
    if diff_ear > WINK_AR_DIFF_THRESH:

            if leftEAR < rightEAR:              if leftEAR <
    EYE_AR_THRESH:
                WINK_COUNTER += 1

          if         WINK_COUNTER    >         WINK_CONSECUTIVE_FRAMES:
    pag.click(button='left')
                    WINK_COUNTER = 0
            elif leftEAR > rightEAR:
              if rightEAR < EYE_AR_THRESH:
                WINK_COUNTER += 1
                If WINK_COUNTER> WINK_CONSECUTIVE_FRAMES:
    pag.click(button='right')

                  WINK_COUNTER  =        0
    else:
          WINK_COUNTER = 0 else:
    if ear <= EYE_AR_THRESH:
          EYE_COUNTER += 1
          if EYE_COUNTER > EYE_AR_CONSECUTIVE_FRAMES:
            SCROLL_MODE = not SCROLL_MODE
            # INPUT_MODE = not INPUT_MODE
            EYE_COUNTER = 0
            # nose point to draw a bounding box around it
      else:
          EYE_COUNTER = 0
          WINK_COUNTER = 0

      if mar > MOUTH_AR_THRESH:
        MOUTH_COUNTER += 1

        if MOUTH_COUNTER >= MOUTH_AR_CONSECUTIVE_FRAMES:
          # if the alarm is not on, turn it on
          INPUT_MODE = not INPUT_MODE
          # SCROLL_MODE = not SCROLL_MODE
          MOUTH_COUNTER = 0
          ANCHOR_POINT = nose_point
    else:
        MOUTH_COUNTER = 0
      if INPUT_MODE:
        cv2.putText(frame, "READING INPUT!", (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7,
RED_COLOR, 2)      x, y = ANCHOR_POINT      nx, ny = nose_point      w, h = 60, 35
multiple = 1      cv2.rectangle(frame, (x - w, y - h), (x + w, y + h),
    GREEN_COLOR, 2)
    cv2.line(frame,
    ANCHOR_POINT,
    nose_point, BLUE_COLOR,
    2)     dir = direction(nose_point, ANCHOR_POINT, w, h)
               cv2.putText(frame,  dir.upper(),  (10,  90),  cv2.FONT_HERSHEY_SIMPLEX,  0.7,
RED_COLOR, 2)        drag = 18
      if dir == 'right':
    pag.moveRel(drag, 0)
    elif dir == 'left':
    pag.moveRel(-drag, 0)
    elif dir == 'up':            if
    SCROLL_MODE:
                pag.scroll(40)
    else:
                pag.moveRel(0,            -
    drag)        elif dir == 'down':
    if SCROLL_MODE:
    pag.scroll(-40)           else:
                pag.moveRel(0, drag)
      if SCROLL_MODE:
        cv2.putText(frame, 'SCROLL MODE IS ON!', (10, 60),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, RED_COLOR, 2)
      # cv2.putText(frame, "MAR: {:.2f}".format(mar), (500, 30),
      #            cv2.FONT_HERSHEY_SIMPLEX, 0.7, YELLOW_COLOR, 2)
      # cv2.putText(frame, "Right EAR: {:.2f}".format(rightEAR), (460, 80),
      #            cv2.FONT_HERSHEY_SIMPLEX, 0.7, YELLOW_COLOR, 2)
      # cv2.putText(frame, "Left EAR: {:.2f}".format(leftEAR), (460, 130),
      #            cv2.FONT_HERSHEY_SIMPLEX, 0.7, YELLOW_COLOR, 2)
      # cv2.putText(frame, "Diff EAR: {:.2f}".format(np.abs(leftEAR - rightEAR)), (460, 80),
      #            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
```

```
    #           Show   the       frame
cv2.imshow("Frame",       frame)
key = cv2.waitKey(1) & 0xFF

    # If the `Esc` key was pressed, break from the loop
if key == 27:         break

# Do a bit of cleanup cv2.destroyAllWindows(
) vid.release() import numpy
as np
# Returns EAR given eye landmarks def eye_aspect_ratio(eye):
    # Compute the euclidean distances between the two sets of
    # vertical eye landmarks (x, y)-coordinates
A = np.linalg.norm(eye[1] - eye[5])
B = np.linalg.norm(eye[2] - eye[4])

    # Compute the euclidean distance between the horizontal
    # eye landmark (x, y)-coordinates
C = np.linalg.norm(eye[0] - eye[3])
    # Compute the eye aspect ratio      ear =
(A + B) / (2.0 * C)
    # Return the eye aspect ratio     return ear
# Returns MAR given eye landmarks  def
mouth_aspect_ratio(mouth):
#    Compute    the    euclidean    distances
between the three sets
    # of vertical mouth landmarks (x, y)-coordinates
A = np.linalg.norm(mouth[13] - mouth[19])
B = np.linalg.norm(mouth[14] - mouth[18])
C = np.linalg.norm(mouth[15] - mouth[17])
    # Compute the euclidean distance between the horizontal
    # mouth landmarks (x, y)-coordinates
D = np.linalg.norm(mouth[12] - mouth[16])      # Compute the mouth aspect ratio     mar = (A + B
    + C) / (2 * D)      # Return the mouth aspect ratio     return mar
```

```
# Return direction given the nose and anchor points. def direction(nose_point,
anchor_point, w, h, multiple=1):
    nx, ny = nose_point      x, y
= anchor_point
    if nx > x + multiple * w:
return 'right'     elif nx < x - multiple *
w:          return
'left'
    if ny > y + multiple * h:
return 'down'     elif ny < y
- multiple * h:          return
'up'  return '-'
```

5.4 OUTPUT SCREENS

Step 1: initial frame for Face detection

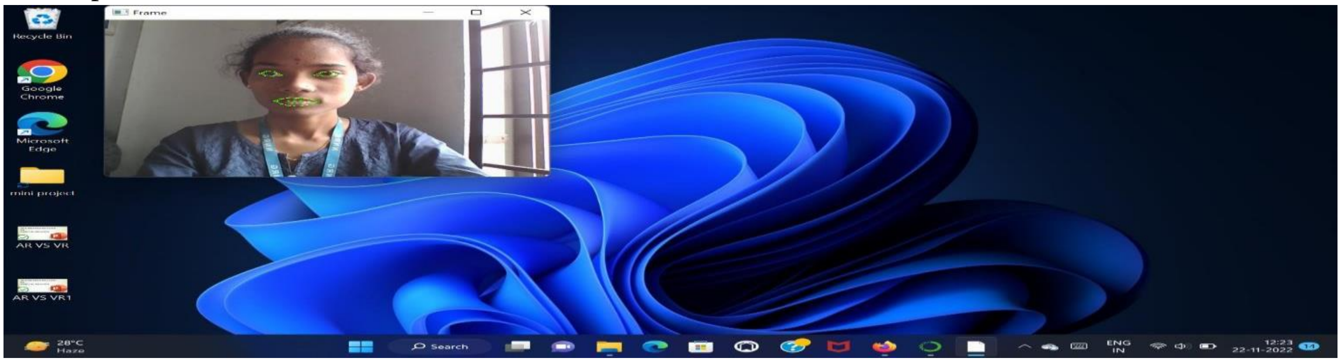


Figure 5.1: Face Detection

Step 2: open mouth for cursor activation

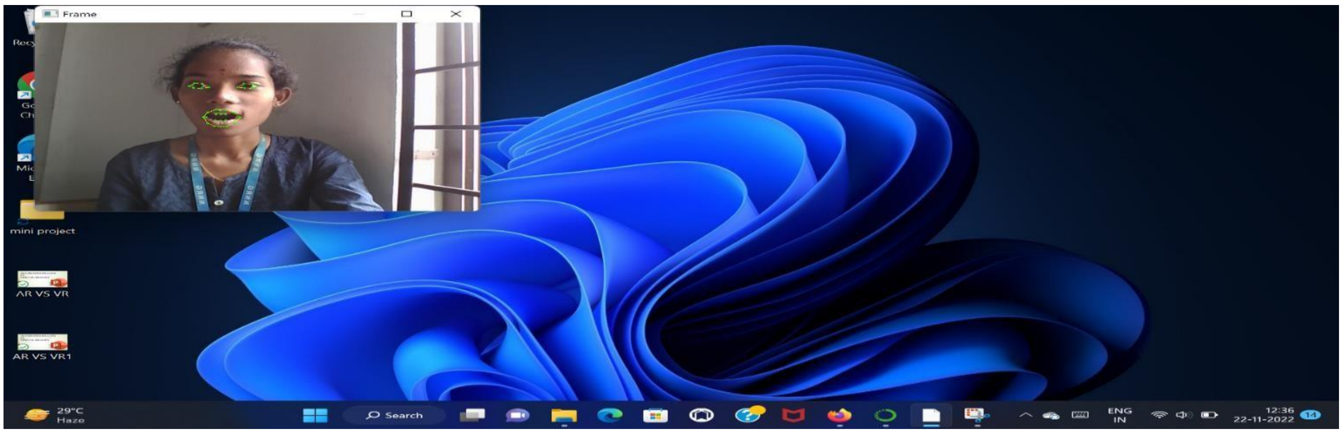


Figure 5.2: Cursor Activation

Step 3: In order to give the input, open the mouth for cursor activation Then the message READING INPUT! appears on the top left on the frame

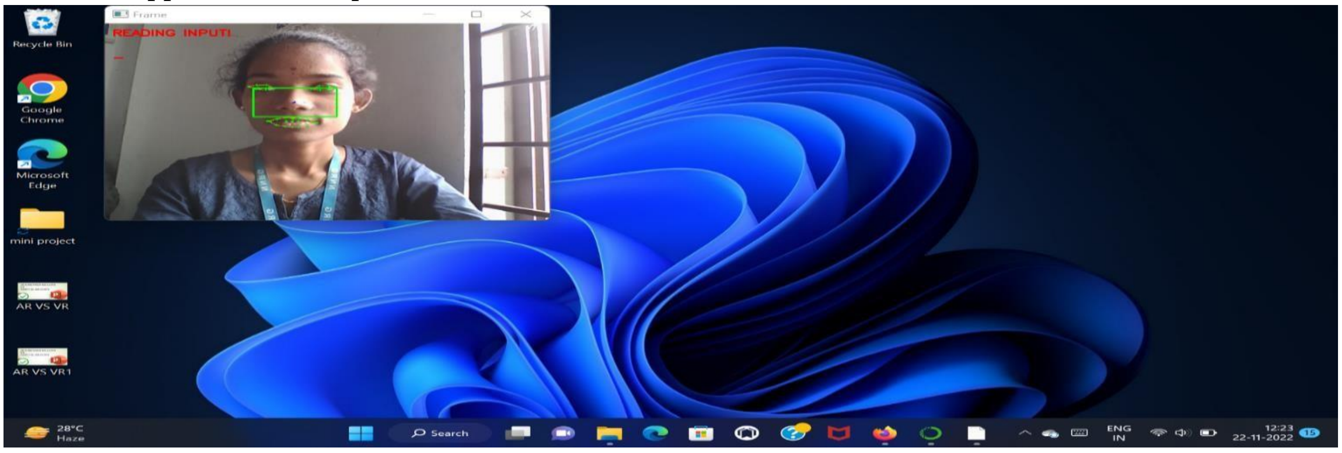


Figure 5.3: Cursor Activated

Step 4: Then you can perform the mouse operations

In order to move cursor to right/left

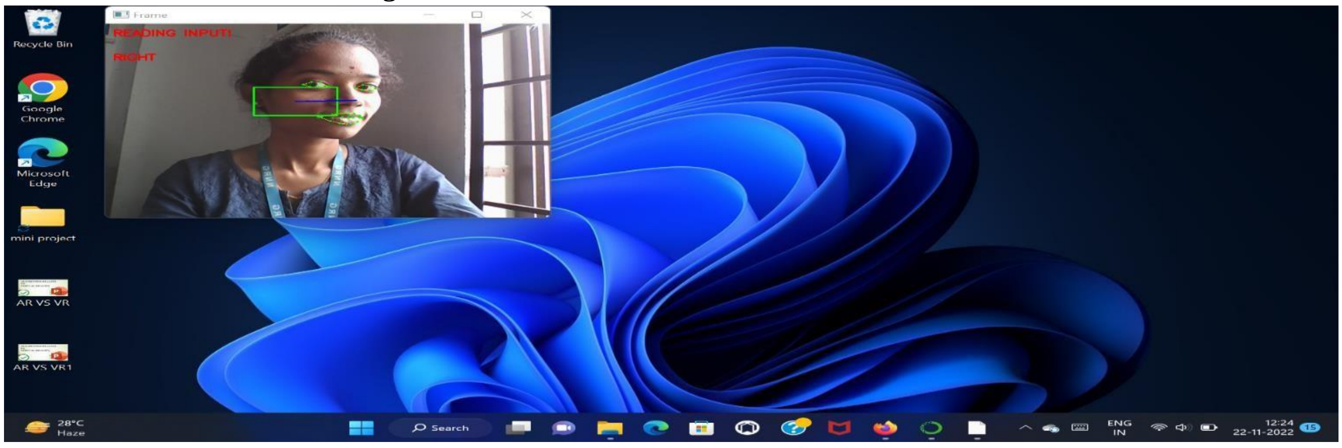


Figure 5.4: Cursor right movement

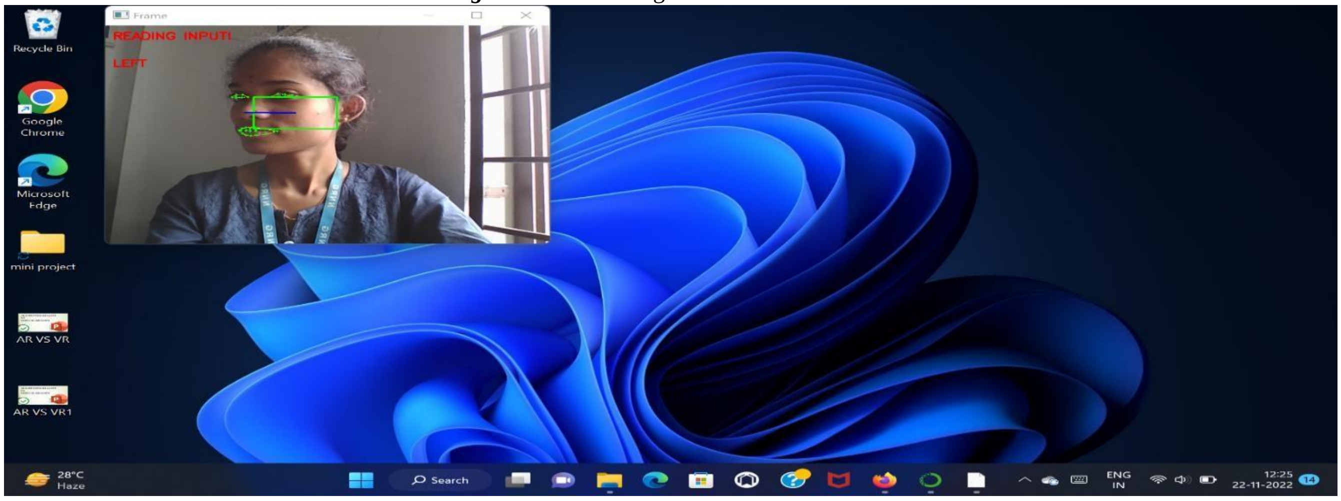


Figure 5.5: Cursor left movement

Step 5: Then you can perform the mouse operations

In order to drag the mouse up and down.

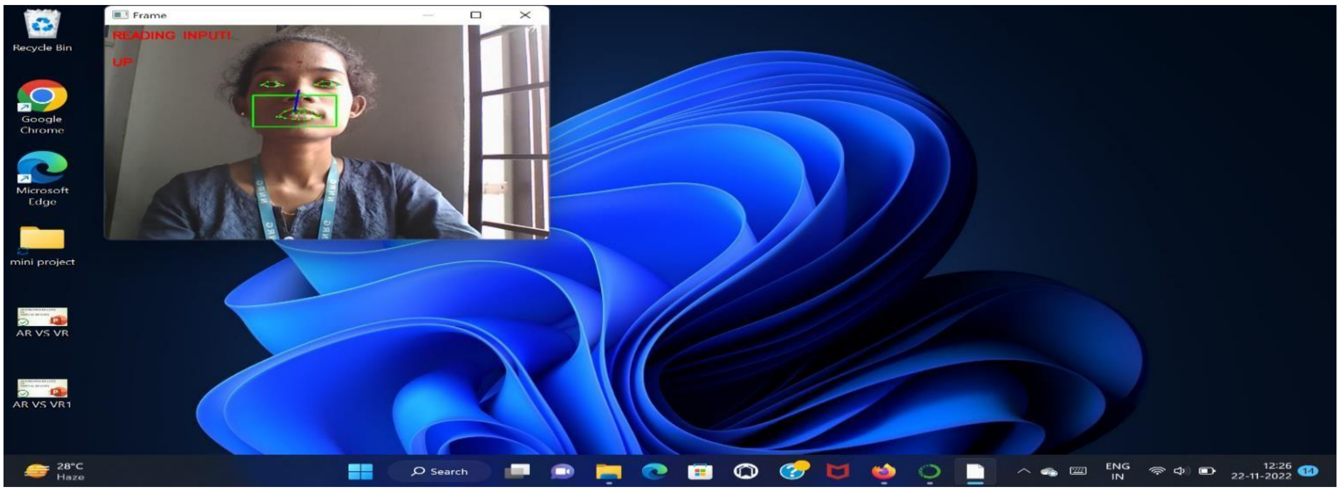


Figure 5.6: Cursor up movement

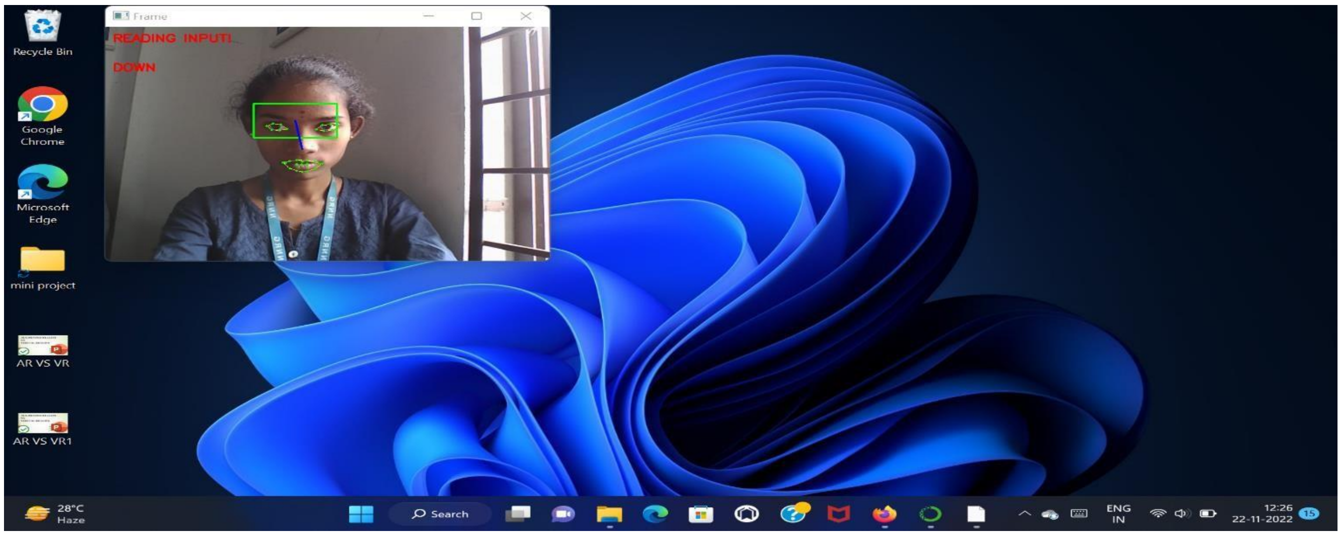


Figure 5.7: Cursor down movement

Step 6: To perform the left click and right click

Wink left eye for left click and wink right eye for right click

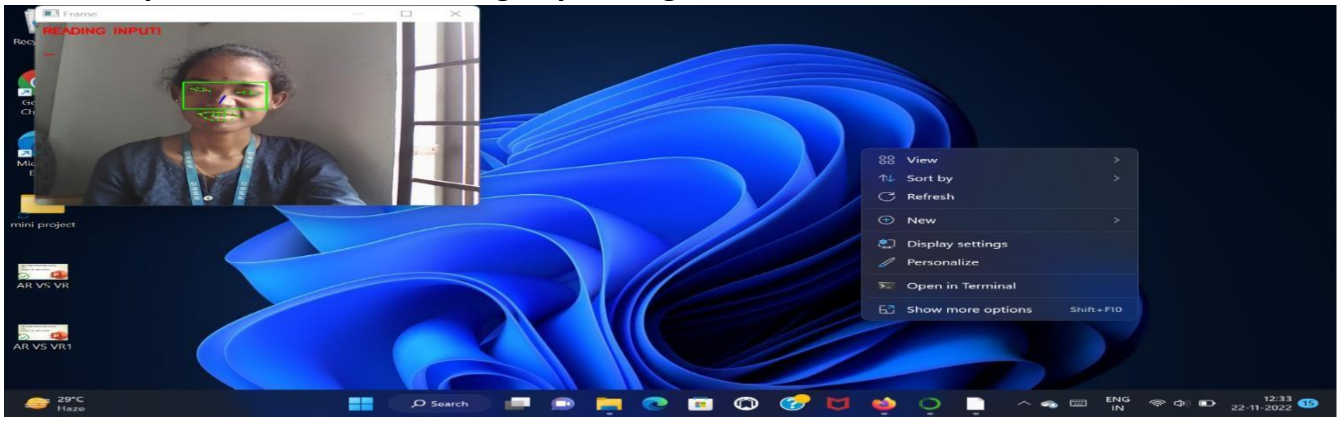


Figure 5.8: Right Click

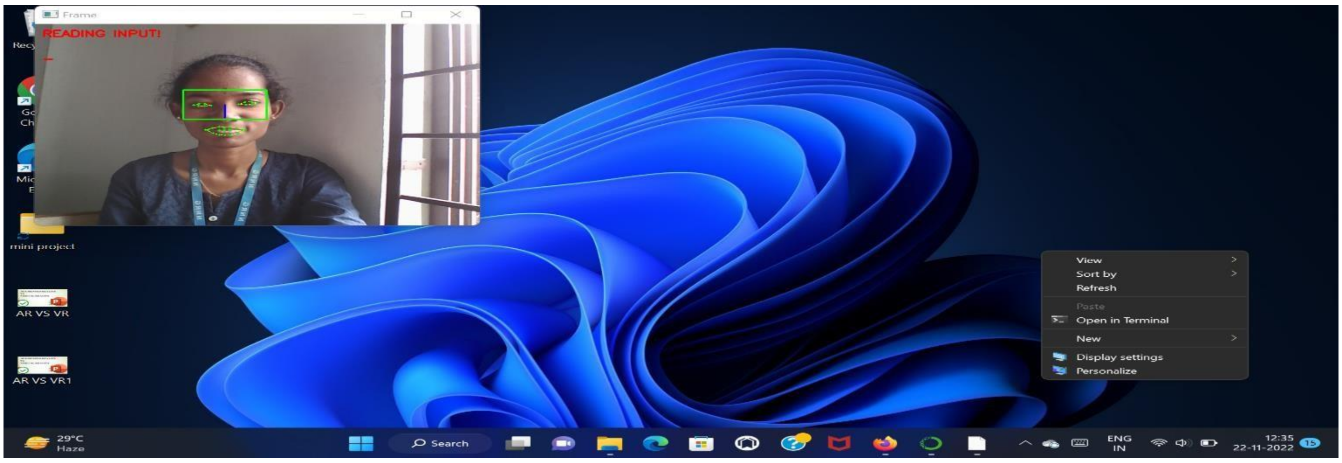


Figure 5.9: Left Click

Step 7: squint eyes for scrolling mode

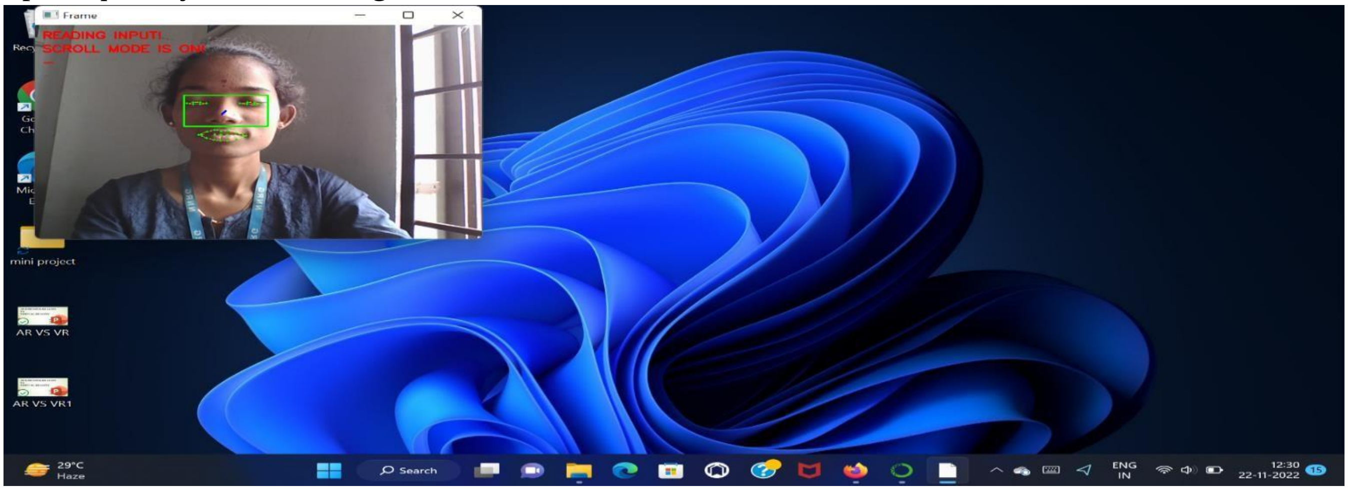
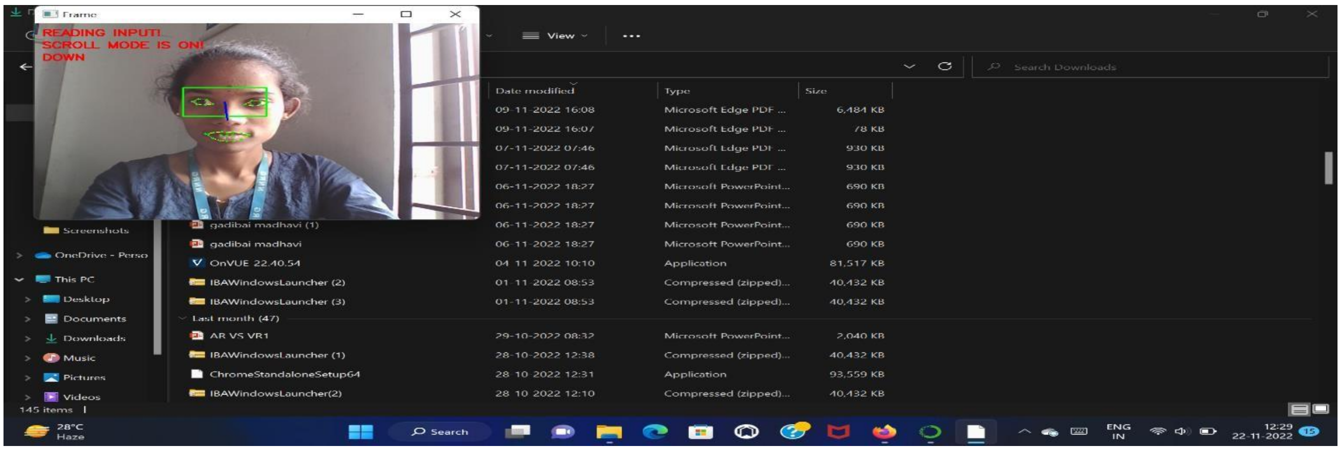
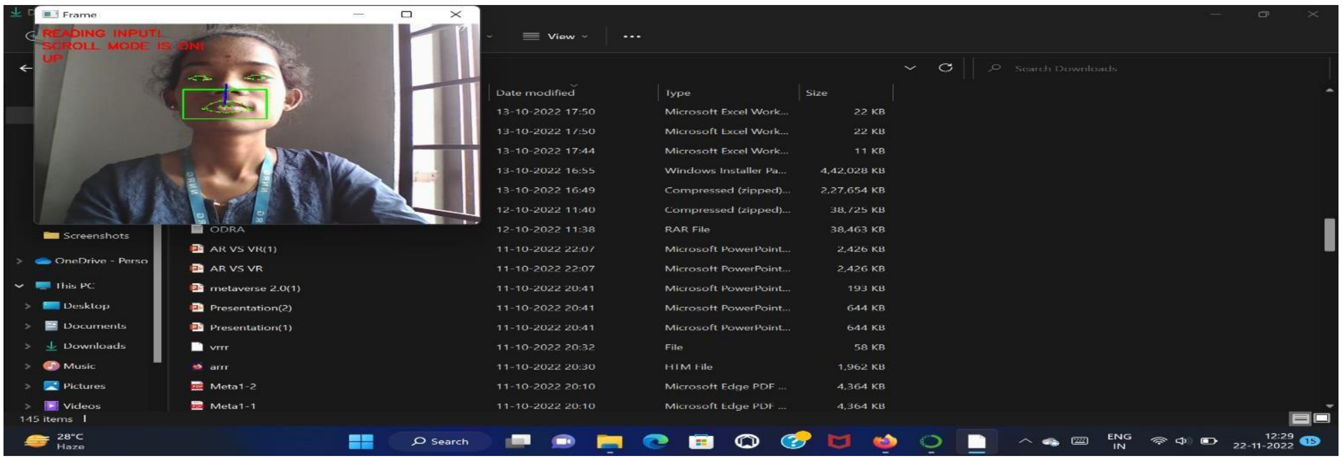


Figure 5.10: Scrolling mode on



	Date modified	Type	Size
	09-11-2022 16:08	Microsoft Edge PDF ...	6,484 KB
	09-11-2022 16:07	Microsoft Edge PDF ...	78 KB
	07-11-2022 07:46	Microsoft Edge PDF ...	930 KB
	07-11-2022 07:46	Microsoft Edge PDF ...	930 KB
	06-11-2022 18:27	Microsoft PowerPoint...	690 KB
	06-11-2022 18:27	Microsoft PowerPoint...	690 KB
gadibai madhavi (1)	06-11-2022 18:27	Microsoft PowerPoint...	690 KB
gadibai madhavi	06-11-2022 18:27	Microsoft PowerPoint...	690 KB
OnVUE 22.40.54	04-11-2022 10:10	Application	81,517 KB
IBAWindowsLauncher (2)	01-11-2022 08:53	Compressed (zipped)...	40,432 KB
IBAWindowsLauncher (3)	01-11-2022 08:53	Compressed (zipped)...	40,432 KB
Last month (47)			
AR VS VR1	29-10-2022 08:32	Microsoft PowerPoint...	2,040 KB
IBAWindowsLauncher (1)	28-10-2022 12:38	Compressed (zipped)...	40,432 KB
ChromeStandaloneSetup64	28-10-2022 12:31	Application	93,559 KB
IBAWindowsLauncher(2)	28-10-2022 12:10	Compressed (zipped)...	40,432 KB

Figure 5.11: Scrolling down



READING INPUT!
SCROLL MODE IS ON!
UP

Name	Date modified	Type	Size
	13-10-2022 17:50	Microsoft Excel Work...	22 KB
	13-10-2022 17:50	Microsoft Excel Work...	22 KB
	13-10-2022 17:44	Microsoft Excel Work...	11 KB
	13-10-2022 16:55	Windows Installer Pa...	4,42,028 KB
	13-10-2022 16:49	Compressed (zipped)...	2,27,654 KB
	12-10-2022 11:40	Compressed (zipped)...	38,725 KB
ODRA	12-10-2022 11:38	RAR File	38,463 KB
AR VS VR(1)	11-10-2022 22:07	Microsoft PowerPoint...	2,426 KB
AR VS VR	11-10-2022 22:07	Microsoft PowerPoint...	2,426 KB
metaverse 2.0(1)	11-10-2022 20:41	Microsoft PowerPoint...	193 KB
Presentation(2)	11-10-2022 20:41	Microsoft PowerPoint...	644 KB
Presentation(1)	11-10-2022 20:41	Microsoft PowerPoint...	644 KB
vrrr	11-10-2022 20:32	File	58 KB
arrr	11-10-2022 20:30	HTM File	1,962 KB
Meta1-2	11-10-2022 20:10	Microsoft Edge PDF ...	4,364 KB
Meta1-1	11-10-2022 20:10	Microsoft Edge PDF ...	4,364 KB

Figure 5.12: Scrolling up

Step 8: open the mouth to stop feeding the input

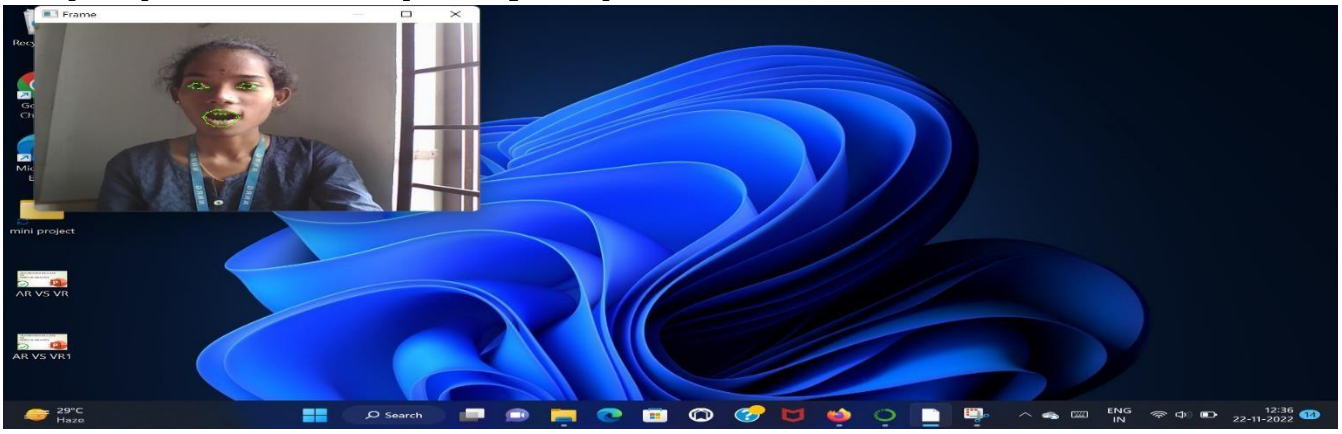


Figure 5.13: cursor deactivation

Step 9: press escape to close the frame and written to command prompt

6. TESTING

6.1 INTRODUCTION

Software Testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves execution of a software component or system component to evaluate one or more properties of interest. Software testing also helps to identify errors, gaps or missing requirements in contrary to the actual requirements. It can be either done manually or using automated tools. Some prefer saying Software testing as a White Box and Black Box Testing.

The process of software testing aims not only at finding faults in the existing software but also at finding measures to improve the software in terms of efficiency, accuracy and usability. It mainly aims at measuring specification, functionality and performance of a software program or application.

Software Testing can be done in two ways:

1. **Verification:** It refers to the set of tasks that ensure that software correctly implements a specific function.
2. **Validation:** It refers to a different set of tasks that ensure that the software that has been built is traceable to customer requirements.

Importance of Software Testing:

The importance of software testing is imperative. Software Testing is important because of the following reasons:

1. Software Testing points out the defects and errors that were made during the development phases. It looks for any mistake made by the programmer during the implementation phase of the software.
2. It ensures that the customer finds the organization reliable and their satisfaction in the application is maintained. Sometimes contracts include monetary penalties with respect to the timeline and quality of the product and software testing prevent monetary losses.
3. It also ensures the Quality of the product. Quality product delivered to the customers helps in gaining their confidence. It makes sure that the software application requires lower maintenance cost and results in more accurate, consistent and reliable results.

Applications of Software Testing:

- **Cost Effective Development** - Early testing saves both time and cost in many aspects, however reducing the cost without testing may result in improper design of a software application rendering the product useless.
- **Product Improvement** - During the SDLC phases, testing is never a time-consuming process. However, diagnosing and fixing the errors identified during proper testing is a timeconsuming but productive activity.
- **Test Automation** - Test Automation reduces the testing time, but it is not possible to start test automation at any time during software development. Test automaton should be started when the software has been manually tested and is stable to some extent. Moreover, test automation can never be used if requirements keep changing.
- **Quality Check** - Software testing helps in determining following set of properties of any software such as ○ Functionality ○ Reliability ○ Usability ○ Efficiency ○ Maintainability ○ Portability

6.1.1 TYPES OF SOFTWARE TESTING:

Software Testing can be broadly classified into two types:

i. Manual Testing:

Manual testing is a software testing process in which test cases are executed manually without using any automated tool. All test cases executed by the tester manually according to the end user's perspective. It ensures whether the application is working, as mentioned in the requirement document or not. Test cases are planned and implemented to complete almost 100 percent of the software application. Test case reports are also generated manually.

Manual Testing is one of the most fundamental testing processes as it can find both visible and hidden defects of the software. The difference between expected output and output, given by the software, is defined as a defect. The developer fixed the defects and handed it to the tester for retesting.

Types of Manual Testing:

There are various methods used for manual testing. Each technique is used according to its testing criteria. Types of manual testing are given below:

- White Box Testing

- Black Box Testing

Advantages of Manual Testing:

- It does not require programming knowledge while using the Black box method It is used to test dynamically changing GUI design.
- Tester interacts with software as a real user so that they are able to discover usability and user interface issues.

Disadvantages of Manual Testing:

- It requires a large number of human resources.
- It is very time-consuming.
- Tester develops test cases based on their skills and experience. There is no evidence that they have covered all functions or not.

ii. Automation Testing:

Automation testing, which is also known as Test Automation, is when the tester writes scripts and uses another software to test the product. This process involves automation of a manual process. Automation Testing is used to re-run the test scenarios that were performed manually, quickly, and repeatedly.

Advantages of Automation Testing:

- Automation testing takes less time than manual testing.
- A tester can test the response of the software if the execution of the same operation is repeated several times.
- Automation Testing provides re-usability of test cases on testing of different versions of the same software.
- Automation testing is reliable as it eliminates hidden errors by executing test cases again in the same way.

Disadvantages of Automation Testing:

- Automation Testing requires high-level skilled testers.
- It requires high-quality testing tools.
- When it encounters an unsuccessful test case, the analysis of the whole event is complicated.
- Test maintenance is expensive because high fee license testing equipment is necessary.

6.1.2 TESTING ACTIVITIES:

Software level testing can be majorly classified into 4 levels:

1. Unit Testing
2. Integration Testing
3. System Testing
4. Acceptance Testing

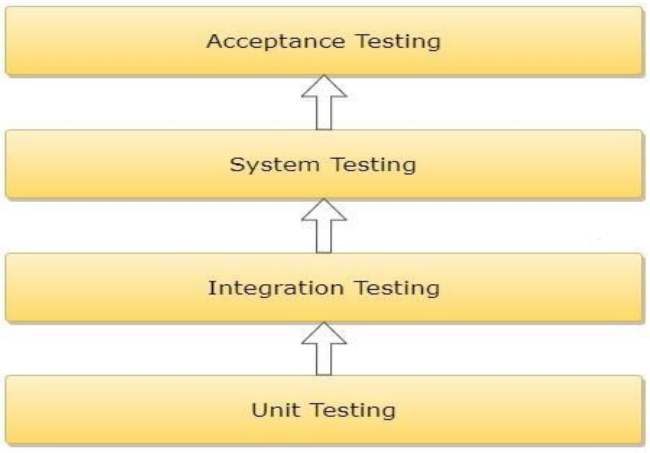


Figure 6.1: Levels of Testing

i. Unit Testing:

Unit Testing is a software testing technique by means of which individual units of software i.e., group of computer program modules, usage procedures and operating procedures are tested to determine whether they are suitable for use or not. It is a testing method using which every independent module is tested to determine if there are any issue by the developer himself. It is correlated with functional correctness of the independent modules.

Unit Testing is defined as a type of software testing where individual components of a software are tested. Unit Testing of software product is carried out during the development of an application. An individual component may be either an individual function or a procedure. Unit Testing is typically performed by the developer.

Objective of Unit testing:

The objective of Unit Testing is:

1. To isolate a section of code.
2. To verify the correctness of code.

3. To test every function and procedure.
4. To fix bug early in development cycle and to save costs.

Advantages:

- Reduces Cost of Testing as defects are captured in very early phase.
- Unit Tests, when integrated with build gives the quality of the build as well
- Unit Testing allows developers to learn what functionality is provided by a unit and how to use it to gain a basic understanding of the unit API.

ii. Integration Testing:

Integration testing is the second level of the software testing process comes after unit testing. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units.

Unit testing uses modules for testing purpose, and these modules are combined and tested in integration testing. The Software is developed with a number of software modules that are coded by different coders or programmers. The goal of integration testing is to check the correctness of communication among all the modules.

Objectives of Integration Testing:

Integration testing reduces the risk of finding the defects in integrated components in the System testing phase. Integration defects can be complex to fix and they can be time-consuming as well. Finding them early in the cycle eliminates the risk of making too many changes at the System testing phase. As each of the integrating components has been tested in the integration phase, the System testing can focus on end-to-end journeys and user-specific flows.

- Reducing risk by testing integrating components as they become available.
- Verify whether the functional and non-functional behaviors of the interfaces are designed as per the specification.
- To build confidence in the quality of the interfaces.

Guidelines for Integration Testing:

- First, determine the test case strategy through which executable test cases can be prepared according to test data.
- Examine the structure and architecture of the application and identify the crucial modules to test them first.

- Design test cases to verify each interface in detail.
- Choose input data for test case execution. Input data plays a significant role in testing.

iii. System Testing:

System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements.

In system testing, integration testing passed components are taken as input. The goal of integration testing is to detect any irregularity between the units that are integrated together. System testing detects within both the integrated units and the whole system. The result of system testing is the observed behaviour of a component or a system when it is tested.

Guidelines for System Testing:

- The very first step is to create a Test Plan.
- Create System Test Cases and test scripts.
- Prepare the test data required for this testing.
- Execute the system test cases and script.
- Report the bugs. Re-testing the bugs once fixed.
- Regression testing to verify the impact of the change in the code.
- Repetition of the testing cycle until the system is ready to be deployed.
- Sign off from the testing team.

iv. Acceptance Testing:

Acceptance Testing is a method of software testing where a system is tested for acceptability. The major aim of this test is to evaluate the compliance of the system with the business requirements and assess whether it is acceptable for delivery or not.

The standard definition of Acceptance testing is given as, “It is a formal testing according to user needs, requirements and business processes conducted to determine whether a system satisfies the acceptance criteria or not and to enable the users, customers or other authorized entities to determine whether to accept the system or not.”

Objectives of Acceptance Testing:

Following are the three major objectives of Acceptance Testing:

- Confirm that the system meets the agreed-upon criteria.

- Identify and resolve discrepancies, if there are any.
- Determine the readiness of the system for cut-over to live operations. The final acceptance of a system for deployment is conditioned upon the outcome of the acceptance testing. The acceptance test team produces an acceptance test report which outlines the acceptance conditions.

6.2 DESIGN OF TEST CASES AND SCENARIOS

6.2.1 TEST CASE DESIGN:

The design of tests for software and other engineering products can be as challenging as the initial design of the product. Test case methods provide the developer with a systematic approach to testing.

Any Engineered product can be tested in either of the two ways:

1. Knowing the specified function that a product has been designed to perform, tests can be conducted. These tests demonstrate whether each function is fully operational and at the same time searches for errors in each function.
2. Knowing the internal workings of a product, tests can be conducted to ensure that internal operations are performed according to specifications and all internal components hence been adequately exercised.

Test case design methods are divided into two types:

1. White-box testing
2. Black-box testing

1. White-Box Testing

White –box testing, sometimes called glass-box testing is a test, case designed method that uses the control structure of the procedural design to derive test cases. Using white-box testing methods, the s/w engineer can derive test cases that guarantee that all independent paths within a module have been exercised at least once. Exercise all logical decisions on their true and false sides. Execute all loops at their boundaries and within their operational bounds. Exercise internal data structures to ensure their validity.

The following table lists the advantages and disadvantages of white-box testing.

Table 6.2: Advantages and Disadvantages of White box testing

Advantages	**Disadvantages**
As the tester has knowledge of the source code, it becomes very easy to find out which type of data can help in testing the application effectively.	Due to the fact that a skilled tester is needed to perform white-box testing, the costs are increased.
It helps in optimizing the code.	Sometimes it is impossible to look into every nook and corner to find out hidden errors that may create problems, as many paths will go untested.
Extra lines of code can be removed which can bring in hidden defects.	It is difficult to maintain white-box testing, as it requires specialized tools like code analysers and debugging tools.

2. Black-Box Testing

Black-box testing, also called behavioural testing, focuses on the functional requirements of the s/w. Black-box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements of a program. It is a complementary approach that is likely to uncover a different class of errors that white-box methods could not.

Black-box testing attempted to find errors in the following categories.

- Incorrect or missing functions.
- Interface errors.
- Errors in data structures or external data base access.
- Behaviour or performance errors.
- Initialization and termination errors.

Black-box testing purposely disregards control structure; attention is focused on information domain. By applying black-box techniques, we derive a set of cases that satisfies the criteria test cases that reduce, by a count that is greater than one, the number of additional test cases that must be designed to achieve reasonable testing. Test cases that tell us something about the presence or absence of classes of errors, rather than an error associated only with the specified. The following table lists the advantages and disadvantages of black-box testing.

Table 6.2: Advantages and Disadvantages of Black box testing

Advantages	**Disadvantages**
Well suited and efficient for large code segments.	Limited coverage, since only a selected number of test scenarios is actually performed.
Code access is not required.	Inefficient testing, due to the fact that the tester only has limited knowledge about an application.
Clearly separates user's perspective from the developer's perspective through visibly defined roles.	Blind coverage, since the tester cannot target specific code segments or error-prone areas.
Large numbers of moderately skilled testers can test the application with no knowledge of implementation, programming language, or operating systems.	The test cases are difficult to design.

6.2.2 SCENARIOS

The following are the test cases in our project:

Table 6.3: Test cases

Test Case Id	Test Case Description	Inputs	Expected Output	Actual Output	Status
1	Capture frame from camera	Video stream	Need to read frame from camera	Reading frame is done	success
2	Detect facial features	Face bounding box region	Need to extract eye, mouth and nose contours	contours generated	success

3	Eyes detected	Colour of eyes are not black	Eyes need to be detected	Eyes are properly detected	success
4	Cursor activation	Mouth opened	Cursor need to be activated	Cursor activated successfully	success
5	Right click	Blink right eye	Need to perform right click	right click performed successfully	success
6	Left click	Blink left eye	Need to perform left click	left click performed successfully	success
7	Cursor movement	Nose movement	Cursor need to be moved	Cursor is moved successfully	success
8	Activating Scrolling	Squint the eyes	Cursor need to be activated for scrolling	Cursor activated for scrolling	success
9	Scrolling up	Moving nose to the top of the rectangle	Need to scroll up	Scrolled up successfully	success
10	Scrolling down	Moving nose to the down of the rectangle	Need to scroll down	Scrolled down successfully	success
11	Eyes not detected, if user is sitting more than 2 feet away	Face bounding box region	User eyes should not be detected	User eyes not detected	success

7. CONCLUSION AND FUTURE ENHANCEMENT

7.1 PROJECT CONCLUSION

We have implemented a system to access the mouse pointer on the computer screen using only Head and Eye movements. This system is especially useful for the upper limb disabled.
Presently, this system can be useful for the overall operational behaviour by interacting with the computer system without the use of mouse.

We have implemented a system to access the mouse pointer on the computer screen using only Head and Eye movements. With the use of a camera, python, OpenCV technology, the system architecture is prepared. User is able to view head and eye movements captured through the camera which is displayed on the screen; accordingly, the user can move the mouse pointer as needed and also perform various mouse actions. This system is especially useful for the upper limb disabled. The experimental results show that we can control the functions of cursor efficiently without the use of the mouse. The operations performed using this system are easy in terms of controlling the cursor. This system is a possible solution to all the problems that are faced due to the existing manual of controlling the cursor with the help of the mouse which is not possible in case of people with disability. This system offers users with new means to control the computer system.

Presently, this system can be useful for the overall operational behaviour by interacting with the computer system without the use of mouse. Through Eyeball movement-based cursor control system, it can be concluded that there can be considerable development in the field of human computer interface with the use of IoT.

7.2 FUTURE ENHANCEMENT

In future, we can extend our implementation to support keyboard press technology for the ease of the User to use the Keyboard hands free along with the already existing mouse movements provided by the system. This would then enable the User to access the computer owing to only facial features and movements without the use of traditional mouse and keyboard i.e Hands free system.
In the future, we can also add new functions which can be operable in useful circumstances to control the cursor by the user and implement this system on platforms like mobile phones, tablet etc.
In the future, we can also develop a series of operational units so that we can attain a fully operating experience for the handlers from turning on to turning off the computer system.

8. REFERENCES

8.1 PAPER REFERENCES

[1]. Tereza Soukupova´ and Jan Cˇ ech. Real-Time Eye Blink Detection using Facial Landmarks. In 21st Computer Vision Winter Workshop, February 2016.

[2]. Adrian Rosebrock. Detect eyes, nose, lips, and jaw with dlib, OpenCV, and Python.

[3]. Adrian Rosebrock. Eye blink detection with OpenCV, Python, and dlib.

[4]. Vahid Kazemi, Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In CVPR, 2014.

[5]. S. Zafeiriou, G. Tzimiropoulos, and M. Pantic. The 300 videos in the wild (300-VW) facial landmark tracking in-the-wild challenge. In ICCV Workshop, 2015.

8.2 TEXT BOOKS

- Software Engineering, A practitioner's Approach- Roger S. Pressman, 6th edition, Mc Graw Hill International Edition.
- The unified modeling language user guide Grady Booch, James Rambaugh, Ivar Jacobson, Pearson Education.
- Software Testing techniques - Baris Beizer, Dreamtech, second edition.

8.3 WEBSITES

- www.geeksforgeeks.org
- en.wikipedia.org/wiki/Machine_learning
- www.tutorialspoint.com
- www.slideshare.net