**Abstract**

With advanced technologies in this digital era, there is always scope for development in the field of computing. Hands free computing is in demand as of today it addresses the needs of quadriplegics. This paper presents a Human computer interaction (HCI) system that is of great importance to amputees and those who have issues with using their hands. The system built is an eye based interface that acts as a computer mouse to translate eye movements such as blinking, gazing and squinting towards the mouse cursor actions. The system in discussion makes use of a simple webcam and its software requirements are Python (3.6), OpenCv, numpy and a few other packages which are necessary for face recognition. The face detector can be built using the HOG (Histogram of oriented Gradients) feature along with a linear classifier, and the sliding window technique. It is hands free and no external hardware or sensors are required.

**Introduction**

The computer mouse or moving the finger has been a very common approach to move the cursor along the screen in the current technology. The system detects any movement in the mouse or the finger to map it to the movement of the cursor. Some people, who do not have their arms to be operational, called as ‘amputees’ will not be able to make use of the current technology to use the mouse. Hence, if the movement of their eyeball can be tracked and if the direction towards which the eye is looking at can be determined, the movement of the eyeball can be mapped to the cursor and the amputee will be able to move the cursor at will. An ‘eye tracking mouse’ will be of a lot of use to an amputee

**Literature Survey**

**References 1**

**Title:** Eye Tracking in Human-Computer Interaction and Usability Research

**Author Name:** Alex Poole and Linden J. Ball

**Description:** This chapter discusses the application of eye movements to user interfaces, both for analyzing interfaces (measuring usability) and as an actual control medium within a human-computer dialogue. For usability analysis, the user's eye movements are recorded during system use and later analyzed retrospectively; however, the eye movements do not affect the interface in real time. As a direct control medium, the eye movements are obtained and used in real time as an input to the user-computer dialogue

**References 2**

**Title:** Non-contact Eye Gaze Tracking System by Mapping of Corneal Reflections,

**Author Name:** D. H. Yoo, J. H. Kim, B. R. Lee, and M. J. Chung

**Description:** If the user sees the monitor, the center of a pupil is always in a polygon that is made by the glints. Consequently, the direction of the user's eye gaze can be computed without computing the geometrical relation between the eye, the camera and the monitor in 3D space. Our method is comparatively simple and fast. We introduce the method and show some experimental results.

**References 3**

**Title:** System for assisted mobility using eye movements based on electrooculography,”

**Author Name:** Rafael Barea, Luciano Boquete, Manuel Mazo, and Elena Lpez

**Description:** The system consists of a standard electric wheelchair with an on-board computer, sensors and a graphic user interface run by the computer. On the other hand, this eye-control method can be applied to handle graphical interfaces, where the eye is used as a mouse computer. Results obtained show that this control technique could be useful in multiple applications, such as mobility and communication aid for handicapped persons.

**References 4**

**Title:** International Journal of Advanced Engineering Technology

**Author Name:** H. Singh and J. Singh, “A Review on Electrooculography

**Description:** The aim and scope of the journal is to emphasize research, development and application within the fields of Scientific Research Engineering &Technology that support high-level of learning, teaching, development and research. It is an international journal that aims to contribute to the constant research and training to promote research in the relevant field.

**Existing System**

There are various methods using which this can be achieved. The camera mouse was proposed by Margrit Betke et. al. for people who are quadriplegic and nonverbal. The movements of the user are tracked using a camera and these can be mapped to the movements of the mouse pointer which is visible on the screen. Yet another method was proposed by Robert Gabriel Lupu, et. al. for human computer interaction that made use of head mounted device to track eye movement and to translate it on screen. Another technique by Prof. Prashant salunke et.al presents techniques of eye tracking using Hough transform. A lot of work is being done to improve the characteristics of HCI.A paper by Muhammad Usman Ghani, et. al suggests that the movements of the eye can be read as an input and used to help the user access the interfaces without using any other hardware device such as a mouse or a keyboard. This can be achieved by using image processing algorithms and computer vision. One way to detect the eyes is, by using the Haar cascade feature. The eyes can be detected by matching it with templates which would already be stored as suggested by Vaibhav Nangare et. al. To get an accurate image of iris an IR sensor can be used. A gyroscope can be used for the orientation of the head as suggested by Anamika Mali et. al. The click operation can be implemented by ‘gazing’ or staring at the screen. Also, by gazing at a fraction of any portion of the screen (upper or lower), the scroll function can be implemented as proposed by Zhang et. al. Along with eye movements, it becomes easier if we incorporate some subtle movements of the face and its parts as well. A real-time eye blink detection using facial landmarks as suggested by Tereza Soukupova and Jan ´ Cech brings out how the blink action can be detected using facial landmarks. This is a major aspect as blinking actions are necessary for translating it into clicking actions. Detecting eyes and other facial parts can be done using openCv and Python with dlib .Similarly even blink can be detected. The paper by Christos Sagonas et.al discusses the challenges of facial landmark localization. Akshay Chandra proposes the same by controlling the mouse cursor using facial movements.

**Proposed System**

The system proposed in this paper works based on the following actions:

* Squinting your eyes
* Winking
* Moving of head around (pitch and yaw)
* Opening the mouth

The methodology is as follows:

1) 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. Once the webcam is opened, the program needs to extract every frame from the video. 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.

2) 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.
* 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.

**Goals and Objectives**

Project undertakes to develop a system which will only use web cam to use human eyes as a pointing device for computer system and provide user friendly human-computer interaction project objectives are outlined below:

* Face & Eyes Detection
* Finding Center of Iris/Pupil
* Eye Comers Extraction
* Develop an algorithm to calculate point of gaze based on Eye features found
* Develop a GUI to show results
* Develop a simple Calibration technique

**Features**

* Simple ,easy to use
* Simple requirements
* No additional hardware
* Faster capture of frames
* Gui settings
* Multi purpose

**Benefits**

* Easier computer control
* Help disabled & handicapped people to use computer
* Helpful in commercial interactive games and advertisements

**Limitations:**

* Real time limitation
* Camera Resolution
* Frames Captured per second
* Improper Usage may lead eye problems

**Modules**

**1) Resizing:**

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.

**2) BGR to gray:**

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.

**3) 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.

**4) Mouth and Eye aspect ratios:**

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 Dlib prebuilt model helps in fast and accurate face detection along with 68 2D facial landmarks as explained already. Here, Eye-AspectRatio (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.

**5) 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:

**1) 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

**2) 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)

**3) 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.

**Libraries Used**

**numpy** :

Used for Matrix Processing

**pyautogui :**

Moving the mouse and clicking or typing in the windows of other applications.)

**imutils:**

I just open sourced my personal imutils package, a series of OpenCV + convenience functions for translation, rotation, resizing, and skeletonization.)

**dlib:**

Dlib is a general purpose cross-platform software library written in the programming language C++.Its design is heavily influenced by ideas from design by contract and component-based software engineering.)

**OpenCV2:**

OpenCV is a cross-platform library using which we can develop real-time computer vision applications.

**Technology Used**

**Python:**

What is Python? Executive Summary

Python is an interpreter, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

**Anaconda IDE:**

Anaconda is the data science platform for data scientists, IT professionals and business leaders of tomorrow. It is a distribution of Python, R, etc. With more than 300 packages for data science, it becomes one of the best platforms for any project

Anaconda is an open-source distribution for python and R. It is used for data science, machine learning, deep learning, etc. With the availability of more than 300 libraries for data science, it becomes fairly optimal for any programmer to work on anaconda for data science.

Anaconda helps in simplified package management and deployment. Anaconda comes with a wide variety of tools to easily collect data from various sources using various machine learning and AI algorithms. It helps in getting an easily manageable environment setup which can deploy any project with the click of a single button.

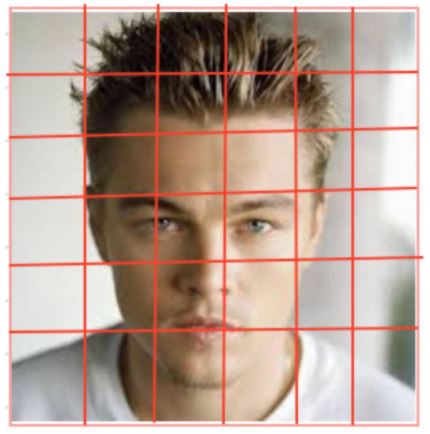
**Implementation**

Currently, for face detection, perhaps deep learning models perform the best. But face detection was there before the emergence of deep learning as well. Earlier, classical feature descriptors and linear classifiers were a really good solution for face detection. And the Dlib library provides one such classical solution for face detection. That is, HOG and Linear SVM.

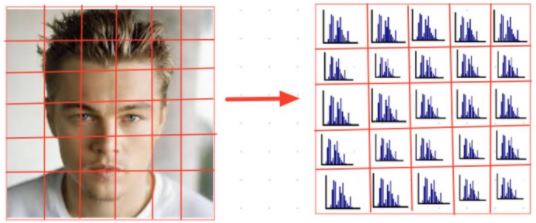
This is based on the **HOG** (Histogram of Oriented Gradients) feature descriptor with a **linear SVM** machine learning algorithm to perform face detection.

HOG is a simple and powerful feature descriptor. It is not only used for face detection but also it is widely used for object detection like cars, pets, and fruits. HOG is robust for object detection because object shape is characterized using the local intensity gradient distribution and edge direction.

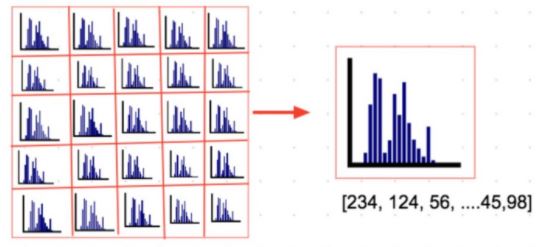
**Step1:** The basic idea of HOG is dividing the image into small connected cells



**Step2:**Computes histogram for each cell



**Step3:**Bring all histograms together to form feature vector i.e., it forms one histogram from all small histograms which is unique for each face

****

**Steps:**

* Record Video
* Convert Video into Frames
* Convert frame into Grayscale
* Find Counter and Edges in the image
* Load Facial Landmark Dataset
* Identify Eye and Mouth in the Frame
* Find Aspect Ratio of Eye and Mouth
* Eye Blink and Head Moment Detection
* Handling Mouse Operations
* Handling Mouse Operations

**Software &Hardware requirements**

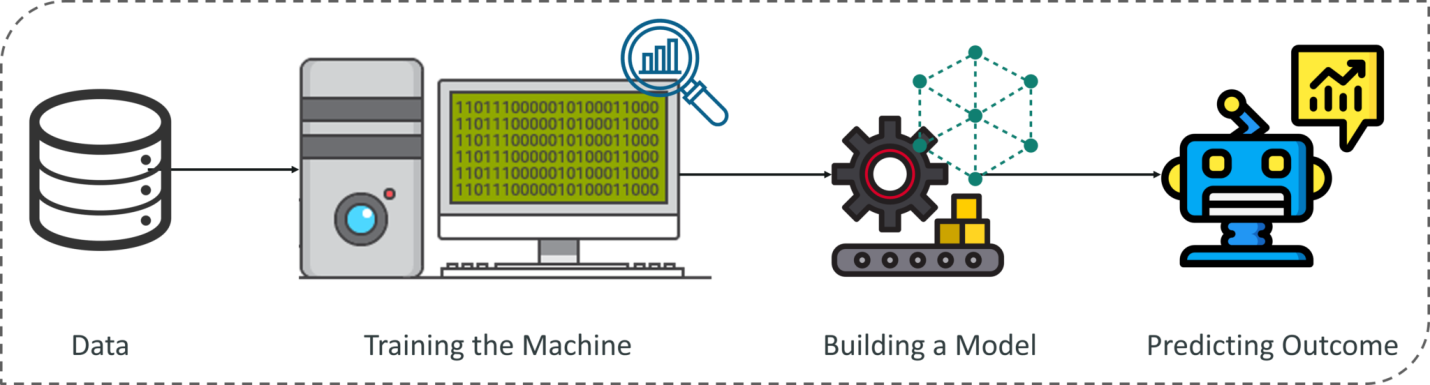
**HARDWARE REQUIREMENTS**

* Processor : Intel i3 and above
* RAM : 4GB and Higher
* Hard Disk : 500GB: Minimum

**SOFTWARE REQUIREMENTS**

* Programming Language / Platform : Python
* IDE : pycharm/jupyter

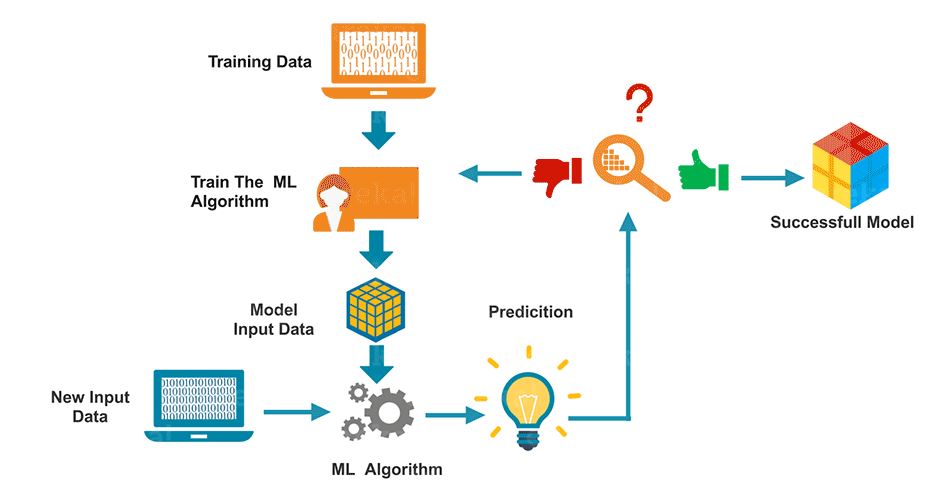
**Machine Learning Process**

****

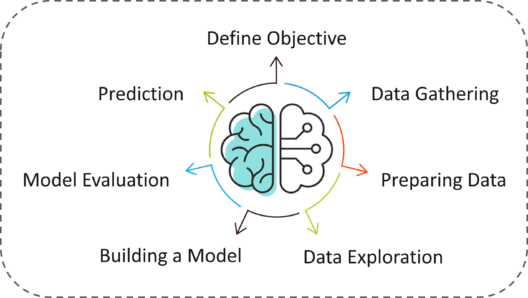
How does Machine Learning Work?

Machine Learning algorithm is trained using a training data set to create a model. When new input data is introduced to the ML algorithm, it makes a prediction on the basis of the model.

The prediction is evaluated for accuracy and if the accuracy is acceptable, the Machine Learning algorithm is deployed. If the accuracy is not acceptable, the Machine Learning algorithm is trained again and again with an augmented training data set.



The Machine Learning process involves building a Predictive model that can be used to find a solution for a Problem Statement. To understand the Machine Learning process let’s assume that you have been given a problem that needs to be solved by using Machine Learning.



The below steps are followed in a Machine Learning process:

**Step 1:** Define the objective of the Problem Statement

At this step, we must understand what exactly needs to be predicted. In our case, the objective is to predict the possibility of rain by studying weather conditions. At this stage, it is also essential to take mental notes on what kind of data can be used to solve this problem or the type of approach you must follow to get to the solution.

**Step 2:**Data Gathering

At this stage, you must be asking questions such as,

* What kind of data is needed to solve this problem?
* Is the data available?
* How can I get the data?

Once you know the types of data that is required, you must understand how you can derive this data. Data collection can be done manually or by web scraping. However, if you’re a beginner and you’re just looking to learn Machine Learning you don’t have to worry about getting the data. There are 1000s of data resources on the web, you can just download the data set and get going.

Coming back to the problem at hand, the data needed for weather forecasting includes measures such as humidity level, temperature, pressure, locality, whether or not you live in a hill station, etc. Such data must be collected and stored for analysis.

**Step 3**: Data Preparation

The data you collected is almost never in the right format. You will encounter a lot of inconsistencies in the data set such as missing values, redundant variables, duplicate values, etc. Removing such inconsistencies is very essential because they might lead to wrongful computations and predictions. Therefore, at this stage, you scan the data set for any inconsistencies and you fix them then and there.

**Step 4:** Exploratory Data Analysis

Grab your detective glasses because this stage is all about diving deep into data and finding all the hidden data mysteries. EDA or Exploratory Data Analysis is the brainstorming stage of Machine Learning. Data Exploration involves understanding the patterns and trends in the data. At this stage, all the useful insights are drawn and correlations between the variables are understood.

For example, in the case of predicting rainfall, we know that there is a strong possibility of rain if the temperature has fallen low. Such correlations must be understood and mapped at this stage.

**Step 5:** Building a Machine Learning Model

All the insights and patterns derived during Data Exploration are used to build the Machine Learning Model. This stage always begins by splitting the data set into two parts, training data, and testing data. The training data will be used to build and analyze the model. The logic of the model is based on the Machine Learning Algorithm that is being implemented.

Choosing the right algorithm depends on the type of problem you’re trying to solve, the data set and the level of complexity of the problem. In the upcoming sections, we will discuss the different types of problems that can be solved by using Machine Learning.

**Step 6:** Model Evaluation & Optimization

After building a model by using the training data set, it is finally time to put the model to a test. The testing data set is used to check the efficiency of the model and how accurately it can predict the outcome. Once the accuracy is calculated, any further improvements in the model can be implemented at this stage. Methods like parameter tuning and cross-validation can be used to improve the performance of the model.

**Step 7:** Predictions

Once the model is evaluated and improved, it is finally used to make predictions. The final output can be a Categorical variable (eg. True or False) or it can be a Continuous Quantity (eg. the predicted value of a stock).

In our case, for predicting the occurrence of rainfall, the output will be a categorical variable.

**Functional requirements**

Functional requirement should include function performed by a specific screen outline work-flows performed by the system and other business or compliance requirement the system must meet.

Functional requirements specify which output file should be produced from the given file they describe the relationship between the input and output of the system, for each functional requirement a detailed description of all data inputs and their source and the range of valid inputs must be specified.

The functional specification describes what the system must do, how the system does it is described in the design specification.

If a user requirement specification was written, all requirements outlined in the user requirements specifications should be addressed in the functional requirements.

* Eye detection
* Eye Moment Detection
* Mouth Moment Detection
* Head Moment Detection
* Mouse Cursor Controlling

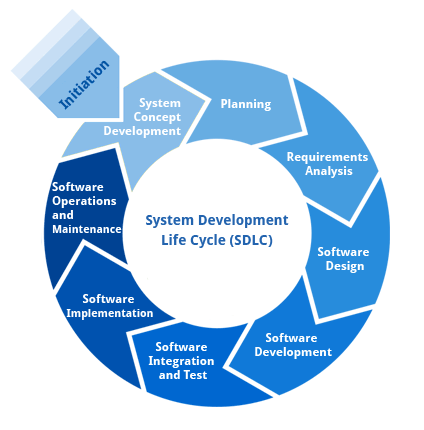
**Nonfunctional requirement**

Describe user-visible aspects of the system that are not directly related with the functional behavior of the system. Non-Functional requirements include quantitative constraints, such as response time (i.e. how fast the system reacts to user commands.) or accuracy (.e. how precise are the systems numerical answers.).

* Portability
* Reliability
* Usability
* Time Constraints
* Error messages
* Actions which cannot be undone should ask for confirmation
* Responsive design should be implemented
* Space Constraints
* Performance
* Standards
* Ethics
* Interoperability
* Security
* Privacy
* Scalabilit

**Software Development Life Cycle:**

The Systems Development Life Cycle (SDLC), or Software Development Life Cycle in systems engineering, information systems and software engineering, is the process of creating or altering systems, and the models and methodologies use to develop these systems.



**Requirement Analysis and Design**

Analysis gathers the requirements for the system. This stage includes a detailed study of the business needs of the organization. Options for changing the business process may be considered. Design focuses on high level design like, what programs are needed and how are they going to interact, low-level design (how the individual programs are going to work), interface design (what are the interfaces going to look like) and data design (what data will be required). During these phases, the software's overall structure is defined. Analysis and Design are very crucial in the whole development cycle. Any glitch in the design phase could be very expensive to solve in the later stage of the software development. Much care is taken during this phase. The logical system of the product is developed in this phase.

**Implementation**

In this phase the designs are translated into code. Computer programs are written using a conventional programming language or an application generator. Programming tools like Compilers, Interpreters, and Debuggers are used to generate the code. Different high level programming languages like PYTHON 3.6, Anaconda Cloud are used for coding. With respect to the type of application, the right programming language is chosen.

**Testing**

In this phase the system is tested. Normally programs are written as a series of individual modules, this subject to separate and detailed test. The system is then tested as a whole. The separate modules are brought together and tested as a complete system. The system is tested to ensure that interfaces between modules work (integration testing), the system works on the intended platform and with the expected volume of data (volume testing) and that the system does what the user requires (acceptance/beta testing).

**Maintenance**

Inevitably the system will need maintenance. Software will definitely undergo change once it is delivered to the customer. There are many reasons for the change. Change could happen because of some unexpected input values into the system. In addition, the changes in the system could directly affect the software operations. The software should be developed to accommodate changes that could happen during the post implementation period.

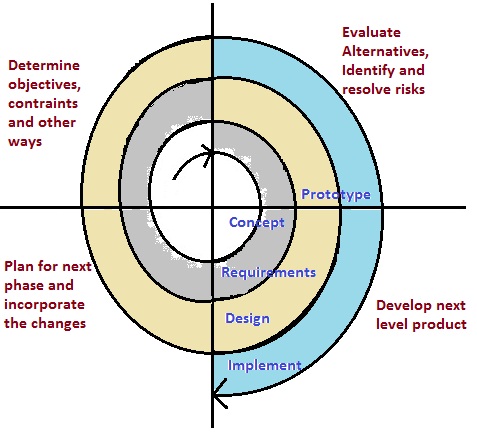
SDLC METHDOLOGIES

This document play a vital role in the development of life cycle (SDLC) as it describes the complete requirement of the system. It means for use by developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

SPIRAL MODEL was defined by Barry Boehm in his 1988 article, “A spiral Model of Software Development and Enhancement. This model was not the first model to discuss iterative development, but it was the first model to explain why the iteration models.

As originally envisioned, the iterations were typically 6 months to 2 years long. Each phase starts with a design goal and ends with a client reviewing the progress thus far. Analysis and engineering efforts are applied at each phase of the project, with an eye toward the end goal of the project.

The following diagram shows how a spiral model acts like:



The steps for Spiral Model can be generalized as follows:

* The new system requirements are defined in as much details as possible.
* This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.
* A preliminary design is created for the new system.
* A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.
* A second prototype is evolved by a fourfold procedure:

1. Evaluating the first prototype in terms of its strengths, weakness, and risks.
2. Defining the requirements of the second prototype.
3. Planning a designing the second prototype.
4. Constructing and testing the second prototype.

* At the customer option, the entire project can be aborted if the risk is deemed too great. Risk factors might involve development cost overruns, operating-cost miscalculation, or any other factor that could, in the customer’s judgment, result in a less-than-satisfactory final product.
* The existing prototype is evaluated in the same manner as was the previous prototype, and if necessary, another prototype is developed from it according to the fourfold procedure outlined above.
* The preceding steps are iterated until the customer is satisfied that the refined prototype represents the final product desired.
* The final system is constructed, based on the refined prototype.
* The final system is thoroughly evaluated and tested. Routine maintenance is carried on a continuing basis to prevent large scale failures and to minimize down time.

**SYSTEM DESIGN**

System design is transition from a user oriented document to programmers or data base personnel. The design is a solution, how to approach to the creation of a new system. This is composed of several steps. It provides the understanding and procedural details necessary for implementing the system recommended in the feasibility study. Designing goes through logical and physical stages of development, logical design reviews the present physical system, prepare input and output specification, details of implementation plan and prepare a logical design walkthrough.

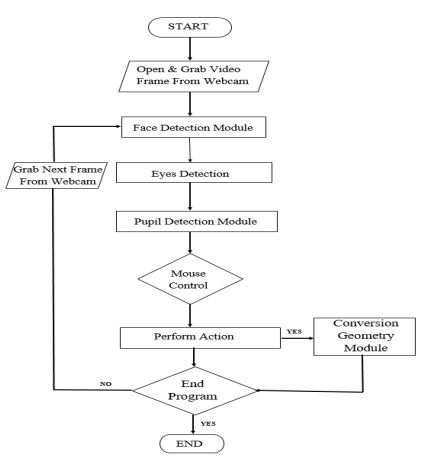
The database tables are designed by analyzing functions involved in the system and format of the fields is also designed. The fields in the database tables should define their role in the system. The unnecessary fields should be avoided because it affects the storage areas of the system. Then in the input and output screen design, the design should be made user friendly. The menu should be precise and compact.

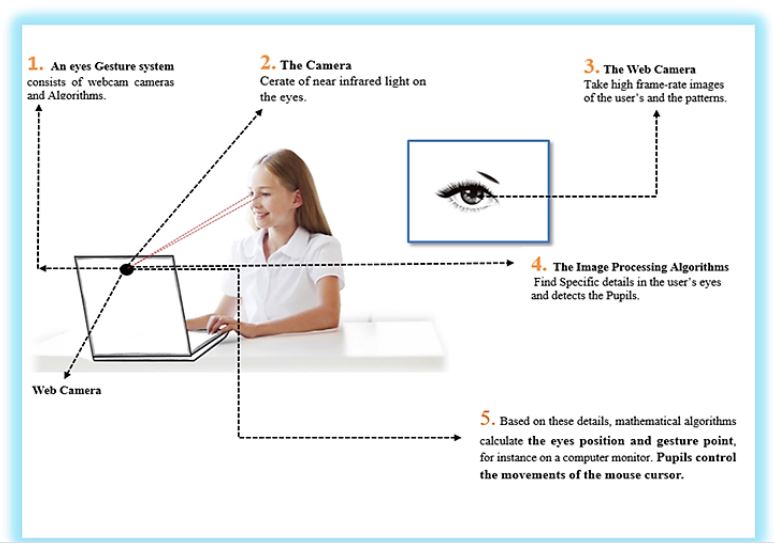
**DATA FLOW DIAGRAM:**

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional details

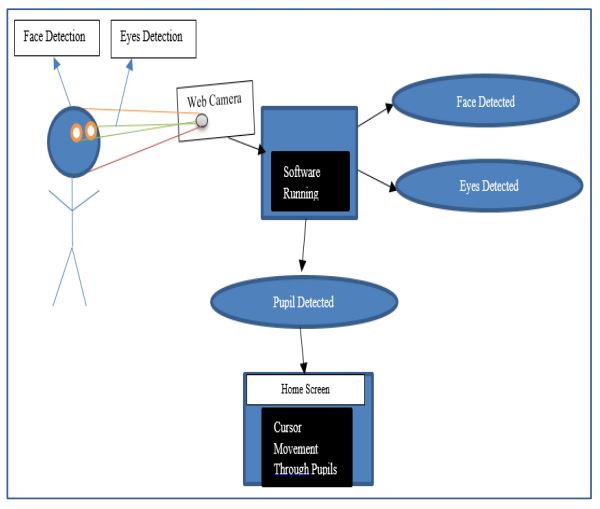
Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation.

Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow.



****

**System Architecture:**

****

**UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.

Provide extendibility and specialization mechanisms to extend the core concepts.

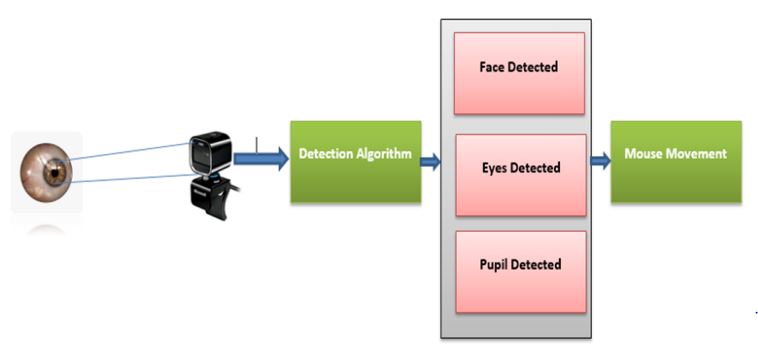
Be independent of particular programming languages and development process.

Provide a formal basis for understanding the modeling language.

Encourage the growth of OO tools market.

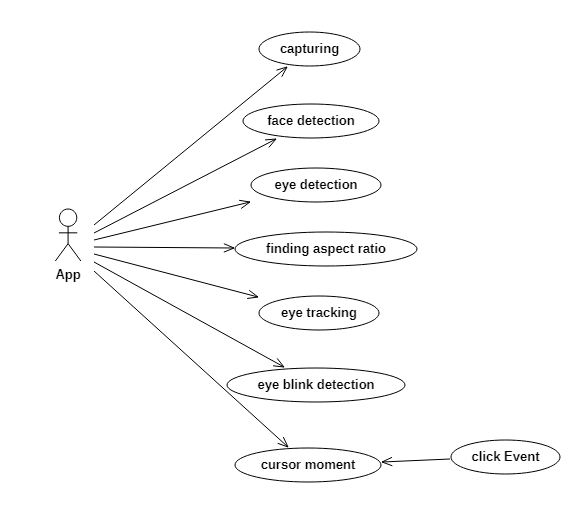
Support higher level development concepts such as collaborations, frameworks, patterns and components.

Integrate best practices

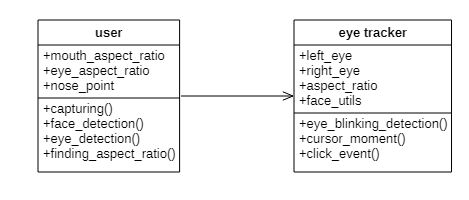
****

**USE CASE DIAGRAM:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

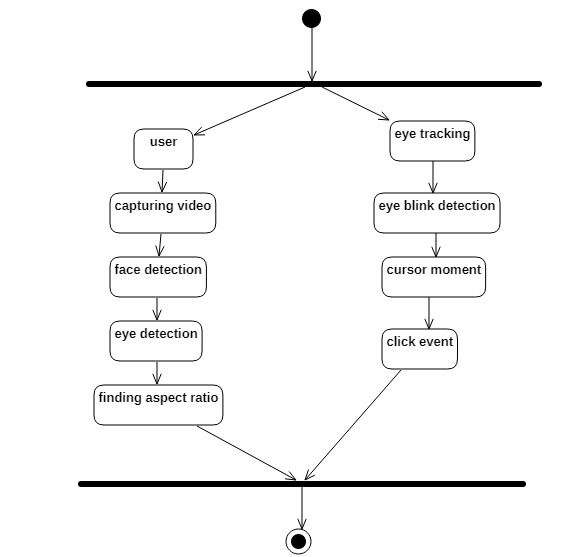


**Class Diagram:**

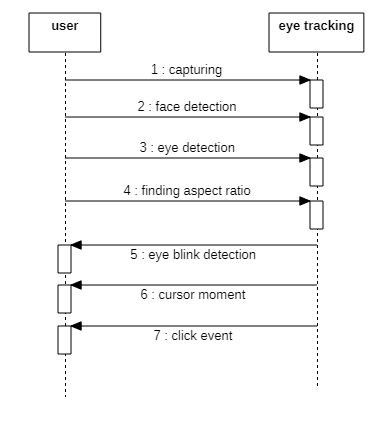


In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

**Acticity Diagram:**



Activity diagrams are graphical representations of Workflows of stepwise activities and actions with support for choice, iteration and concurrency.In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

**Sequence:**Sequence Diagrams Represent the objects participating the interaction horizontally and time vertically. A Use Case is a kind of behavioral classifier that represents a declaration of an offered behavior. Each use case specifies some behavior, possibly including variants that the subject can perform in collaboration with one or more actors. Use cases define the offered behavior of the subject without reference to its internal structure. These behaviors, involving interactions between the actor and the subject, may result in changes to the state of the subject and communications with its environment. A use case can include possible variations of its basic behavior, including exceptional behavior and error handling.

**State Chart Diagram:**

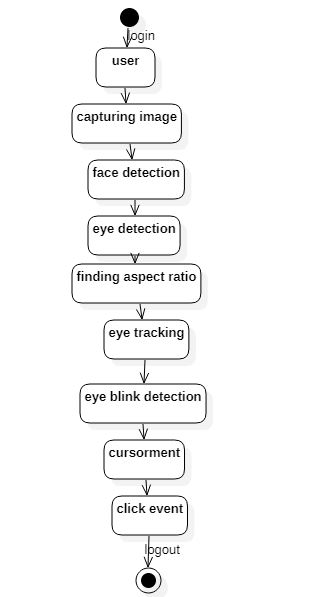
Statechart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. Statechart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

Statechart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of Statechart diagram is to model lifetime of an object from creation to termination.

Statechart diagrams are also used for forward and reverse engineering of a system. However, the main purpose is to model the reactive system.

Following are the main purposes of using Statechart diagrams −

* To model the dynamic aspect of a system.
* To model the life time of a reactive system.
* To describe different states of an object during its life time.
* Define a state machine to model the states of an object.



**Sample Code:**

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 '-'

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:

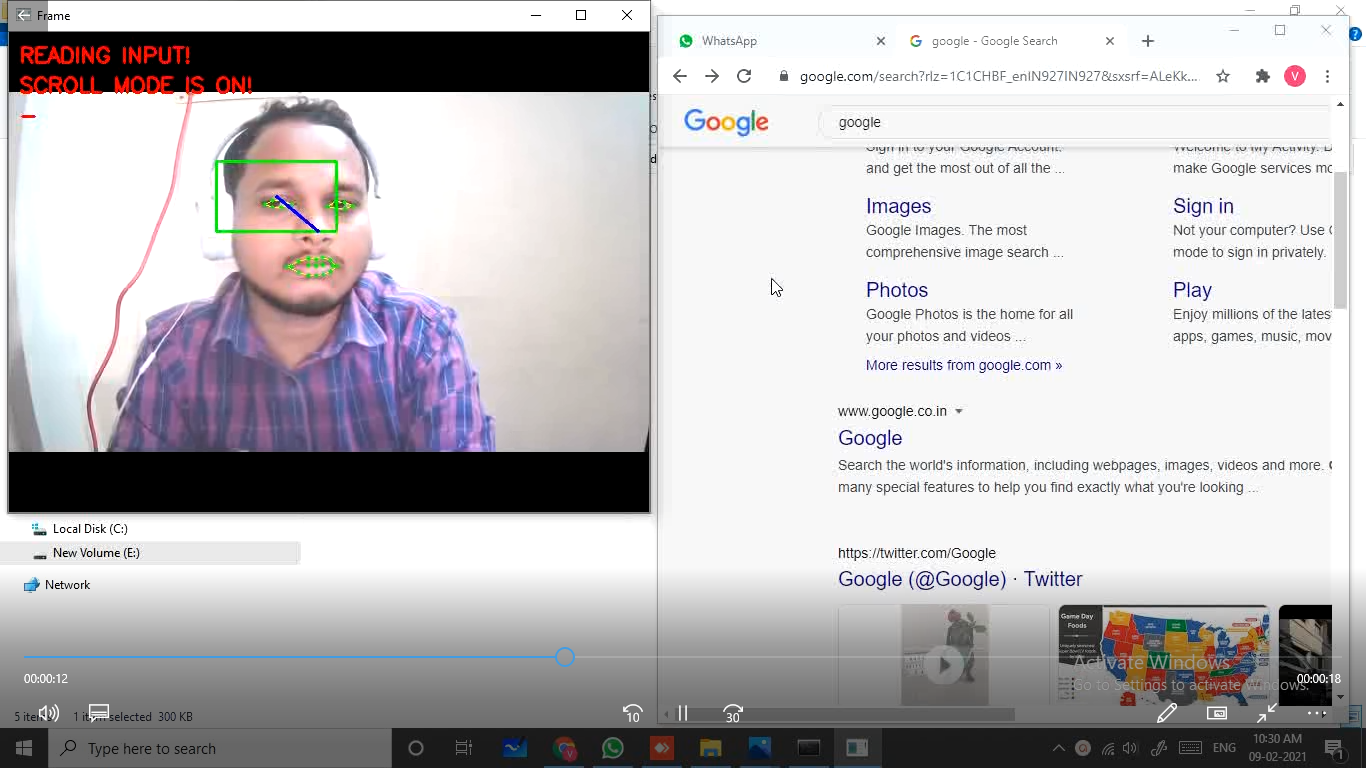
break

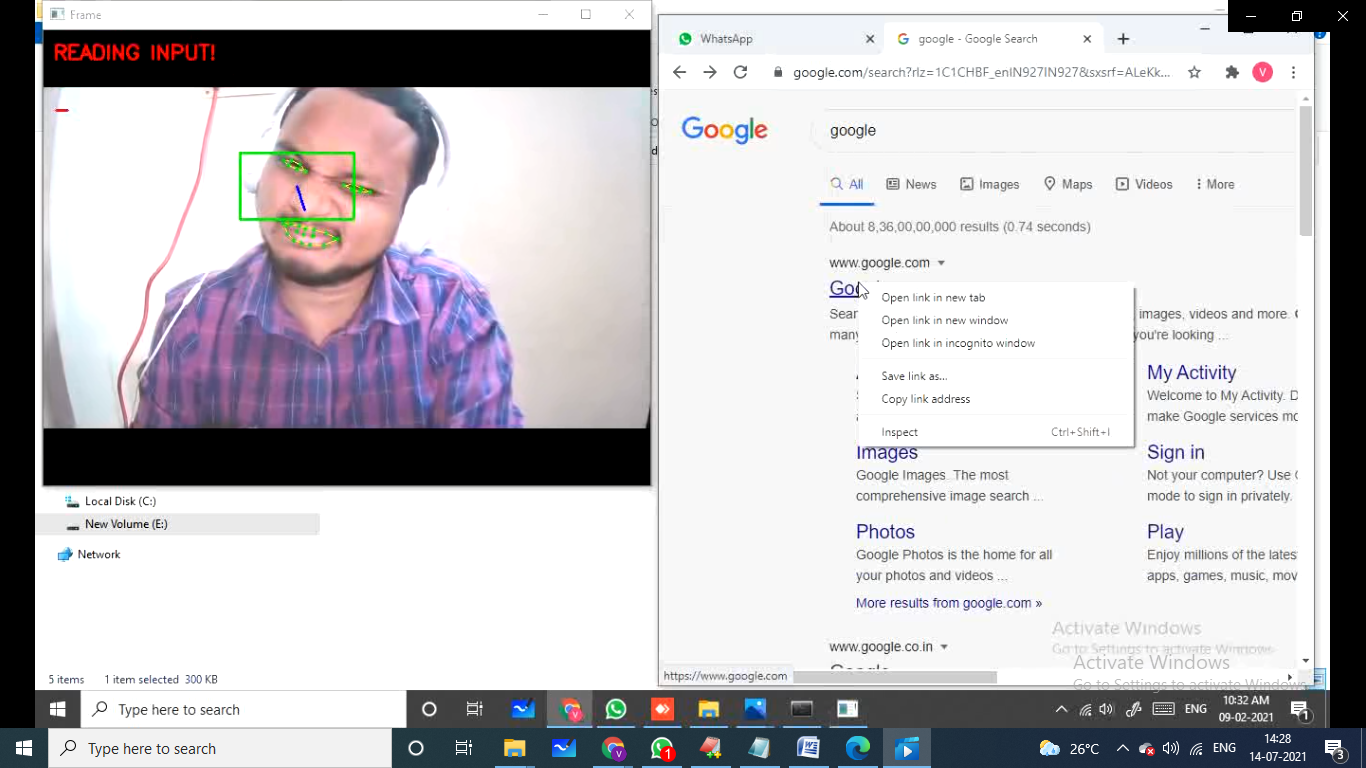
# Do a bit of cleanup

cv2.destroyAllWindows()

vid.release()

**Sample Screens:**

****

****

**Test Cases**

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct pag

The actual purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. TYPES OF TESTING

There are many types of testing methods are available in that mainly used testing methods are as follows

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined

**System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**Test cases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tested** | **Test name** | **Inputs** | **Expected output** | **Actual Output** | **status** |
| 1 | Load Dataset | Facial Land Mark Dataset | Read dataset | Load dataset | success |
| 2 | Capture Frame from Camera | Video Stream | Need to read frame from Camera | Reading Frame is done | success |
|  | Detect Face | Gray Scale Image | Need to Extract Face Region from Given Gray Scale Image | Face Detected Successfully | success |
| 4 | Detect Facial Features | Face Bounding Box Region | Need to Extract Eye Mouse Nose Contours | Contours Generated | success |
| 5 | Calculate Aspect Ratio | Starting and ending point | Need to calculate the aspect ratio of given contour | Aspect ratio calculated | success |

**Result Analysis**

The mouse control program will be operational, and the user can move the cursor, scroll, or click at his will. The amount of change of the position of the cursor along any axis can be changed as per the needs of the user. The mouse control is activated by opening the mouth when the MAR value crosses a certain threshold.

The scroll mode is activated by squinting. The scrolling can be done by moving the head up-down which is called as pitching and by sideways called as yawing. Scroll mode is deactivated by squinting again. The clicking action takes place by winking the eye. Right wink corresponds to right wink and left click corresponds to left wink

The sensitivity of the mouse can be changed accordingly as per the needs of the user. Overall, the project works as required. Though the comfort is not the same as in case of hands controlled mouse, this project can be used with some ease with some practice.

**CONCLUSION**

This work can be extended to improve the speed of the system by using better trained models. Also, the system can be made more dynamic by making the change in the position of the cursor, proportional to the amount of rotation of the user’s head, i.e., the user can decide, at what rate he wants the position of the cursor to change. Also, future research work can be done on making the ratio more accurate, since the range of the values are the result of the aspect ratios, which is usually small. Hence, to make the algorithm detect the actions more accurately, there can be some modification in the formulae for the aspect ratios used. Also, to make the process of detection of the face more easy, some image processing techniques can be used before the model detects the face and features of the face.

**Feature Enhancement**

In future, many people who are unable to operate a standard computer mouse or keyboard because of disabilities of their hands or arms, can get possible alternative in multimodal system, which allows controlling a computer without using standard mouse and keyboard. Using head movements to control the cursor across the computer screen and by using the speech for giving the control commands. Automatic speech recognition and head tracking in joint multimodal action are combined to operate the system.

**REFERENCES**

[1] Alex Poole and Linden J. Ball, “Eye Tracking in Human-Computer Interaction and Usability Research: Current Status and Future Prospects,” in Encyclopedia of Human Computer Interaction (30 December 2005) Key: citeulike:3431568, 2006, pp. 211-219.

[2] D. H. Yoo, J. H. Kim, B. R. Lee, and M. J. Chung, “Non-contact Eye Gaze Tracking System by Mapping of Corneal Reflections,” in Fifth IEEE International Conference on Automatic Face and Gesture Recognition (FGR02), 2002, pp. 94-99.

[3] Rafael Barea, Luciano Boquete, Manuel Mazo, and Elena Lpez, “System for assisted mobility using eye movements based on electrooculography,” IEEE TRANSACTIONS ON NEURAL SYSTEMS AND REHABILITATION ENGINEERING, vol. 10, no. 4, pp. 209-217, DECEMBER 2002.

[4] H. Singh and J. Singh, “A Review on Electrooculography,” International Journal of Advanced Engineering Technology, vol. III, no. IV, 2012.

[5] K. Irie, B. A. Wilson, and R. D. Jones, “A laser-based eye-tracking system,” Behavior Research Methods, Instruments, & Computers, vol. 34, no. 4, pp. 561-572, 2002.

[6] P Ballard and George C. Stockman, “Computer operation via face orientation,” in Pattern Recognition, 1992. Vol.I. Conference A: Computer Vision and Applications, Proceedings., 11th IAPR International Conference on, 1992, pp. 407-410.

[7] T. Horprasert, Y. Yacoob, and L.S. Davis, “Computing 3-D head orientation from a monocular image sequence,” in Second International Conference on Automatic Face and Gesture Recognition, 1996, pp. 242- 247.

[8] K. Arai and M. Yamaura, “Computer Input with Human Eyes-Only Using Two Purkinje Images Which Works in a Real-Time Basis without Calibration,” CSC Journals, vol. 1, no. 3, pp. 71-82, 2010.

[9] D. Back, “Neural Network Gaze Tracking using Web Camera.,” Linkping University, MS Thesis 2005. [10] R. Gonzalez and R. Woods, Digital Image Processing, 3rd ed.: Pearson Education, 2009.

[11] P. Viola and M. Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features,” in COMPUTER VISION AND PATTERN RECOGNITION, 2001.