A Report On

Eye Movement Detector

Submitted to
Chhattisgarh Swami Vivekanand Technical University,
Bhilai, Bachelor of Engineering
In
Computer Science & Engineering
By

Nikhil Verma
Nitin Kumar Ramteke
Pranav Agrawal
Vishal Singh

Under the Guidance of

Dr. Abha Choubey, Associate Professor

Department of Computer Science & Engineering

Shri Shankaracharya Technical Campus
(Faculty of Engineering & Technology), Junwani, Bhilai
Session :- 2021-2022

Declaration

We the undersigned solemnly declare that the report of the project work entitled **Eye movement detector** , is based on our own work carried during the course of my study under **Dr. Abha Choubey , Associate Professor CSE department of SSGI** , Bhilai assert that the statements made are conclusions drawn are an outcome of the project work. I further declare that to the best of my knowledge and belief that the report does not contain any part of my work which has been submitted for the award of any other degree/certificate In this University or any other University.

(signature)
Nikhil Verma
Roll No: 301402218147
Enrollment No: BG0222

(signature)
Nitin Kumar Ramteke
Roll No: 301402218154
Enrollment No: BG0226

(signature)
Pranav Agrawal
Roll No: 301402218151
Enrollment No: BG0232

(signature)
Vishal Singh
Roll No: 301402218155
Enrollment No: BG0291

CERITFICATE

This is to certify that the report of the project submitted is an outcome of the project work entitled **Eye Movement Detector** carried out by **Nikhil Verma** bearing **Roll No: 301402218147, Nitin Kumar Ramteke** bearing **Roll No: 301402218154**, **Pranav Agrawal** bearing **Roll No: 301402218151, Vishal Singh** bearing **Roll No: 301402218155** and carried out under my guidance and supervision for the award of Degree of Engineering in Computer Science & Engineering (FET) of Chhattisgarh Swami Vivekanand Technical University, Bhilai (C.G.) To the best of my knowledge the report.

Embodies the work of the candidate themselves/her/himself, Has duly been completed, Fulfils the requirement of the Ordinance relating to the B.Tech degree of the University, Is up to the desired standard for the purpose of which is submitted

(Signature of the HoD with Seal)	(Signature of the Guide)
	Associate Professor ,
Dr. Siddhartha Choubey	**Dr. Abha Choubey**
Computer Science & Engineering	**Computer Science & Engineering**
SSGI, Bhilai	**SSGI, Bhilai**

The project work as mentioned above is the hereby being recommended and forwarded for examination and evaluation.

(Signature of the Head of Institution with Seal)

Certificate By The Examiners

This is to certify that the project work entitled

Eye movement detector

Submitted by

Nikhil Verma Roll No: 301402218147 Enrollment No: BG0222
Nitin Kumar Ramteke Roll No: 301402218154 Enrollment No: BG0226
Pranav Agrawal Roll No: 301402218151 Enrollment No: BG0232
Vishal Singh Roll No: 301402218155 Enrollment No: BG0291
has been examined by the undersigned as a part of the examination for the award of the Bachelor of Engineering degree in Computer Science & Engineering (FET) of Chhattisgarh Swami Vivekanand Technical University, Bhilai.

_____ _____

Internal Examiner External Examiner

Date:

Aknowledgement

Working for the project has been a great experience for us. There were moments of anxiety, when we could not solve a problem for the several days. But we have enjoyed bit of process and are thankful to all people associated with us during this period.

We convey our sincere thanks to our project guide **Dr. Abha Choubey, Associate Professor** for providing us all sorts of facilities. Her support and guidance helped us to carry out the project. We owe a great department of her gratitude for her constant advice, support, cooperation & encouragement throughout the project.

We would like to express the deep gratitude to respected **Dr. Abha Choubey, Associate Professor** for her ever helping and support. We also pay special thanks for helpful solution and comments enriched by his experience, which improved our ideas for betterment of the project. We would also like to express our deep gratitude to our college management **Shri I.P. Mishra, Chairman (Shri Gangajali Education Society, Bhilai), Mrs. Jaya Mishra, President (Shri Gangajali Education Society, Bhilai), Dr. P.B. Deshmukh, Director (SSGI) & Dr. Siddhartha Choubey HoD (CSE Department)** for providing an educational ambience. It will be our pleasure to acknowledge. Utmost cooperation and valuable suggestions from time to time given by our staff members of our department to whom we owe our entire computer knowledge and also we would like to thank all those persons who have directly or indirectly helped us by providing books and computer peripherals and other necessary amenities which helped us in the development of this project which would otherwise have not been possible.

Nikhil Verma

Nitin Kumar Ramteke

Pranav Agrawal

Vishal Singh

Table Of Contents

List of figures

1. Abstract

This paper addresses the eye gaze tracking problem using a low cost and more convenient web camera in a desktop environment, as opposed to gaze tracking techniques requiring specific hardware, e.g., infrared high-resolution camera and infrared light sources, as well as a cumbersome calibration process. In the proposed method, we first track the human face in a real-time video sequence to extract the eye regions. Then, we combine intensity energy and edge strength to obtain the iris center and utilize the piece wise eye corner detector to detect the eye corner. We adopt a sinusoidal head model to simulate the 3-D head shape, and propose an adaptive weighted facial features embedded in the pose from the orthography and scaling with iterations algorithm, whereby the head pose can be estimated. Finally, the eye gaze tracking is accomplished by integration of the eye vector and the head movement information. Experiments are performed to estimate the eye movement and head pose with the dlib, OpenCV and python. In addition, experiments for gaze tracking are performed in real-time video sequences under a desktop environment. The proposed method does not need high quality camera. This will help in detecting cheating in online exam.

1.1 PROJECT OVERVIEW

In this paper, we concentrate on visible-imaging and present an approach to the eye gaze tracking using a web camera in a desktop environment. First, we track the human face in a real time video sequence to extract the eye region. Then, we combine intensity energy and edge strength to locate the iris center and utilize the piecewise eye corner detector to detect the eye corner. Finally, eye gaze tracking is performed by the integration of the eye vector and head movement information.

Our three-phase feature-based eye gaze tracking approach uses eye features and head pose information to enhance the accuracy of the gaze point estimation.

In Phase 1, we extract the eye region that contains the eye movement information. Then, we detect the iris center and eye corner to form the eye vector.

Phase 2 obtains the parameters for the mapping function, which describes the relationship between the eye vector and the gaze point on the screen. In Phases 1 and 2, a calibration process computes the mapping from the eye vector to the coordinates of the monitor screen.

Phase 3 entails the head pose estimation and gaze point mapping. It combines the eye vector and head pose information to obtain the gaze point.

1.2 Existing System And its Drawbacks

- The video-based gaze approaches commonly use two types of imaging techniques: infrared imaging and visible imaging. The former needs infrared cameras and infrared light sources to capture the infrared images,while the latter usually utilizes high resolution cameras for images.
- Compared with the infrared-imaging approaches, visible imaging methods circumvent the aforementioned problems without the need for the specific infrared devices and infrared light sources. They are not sensitive to the utilization of glasses and the infrared sources in the environment. Visible-imaging methods should work

in a natural environment, where the ambient light is uncontrolled and usually results in lower contrast images.

- Sugano et al. have presented an online learning algorithm within the incremental learning framework for gaze estimation, which utilized the user's operations (i.e., mouse click) on the PC monitor.
- Nguyen first utilized a new training model to detect and track the eye, and then employed the cropped image of the eye to train Gaussian process functions for gaze estimation. In their applications, a user has to stabilize the position of his/her head in front of the camera after the training procedure.
- Williams et al. proposed a sparse and semi-supervised Gaussian process model to infer the gaze, which simplified the process of collecting training data.

1.3 Why prevention of cheating in online exam is needed?

In recent years, information and communication technologies (ICT) witnessed rapid developments and had direct impacts on human life, especially in the field of education. As a result, E-learning has become increasingly popular over the last few years and widely adopted by educational institutions. It enables to deliver information whenever students need at anytime and anywhere over the web. For this reason, it also called web-based learning or online learning. "Assessment for Learning is the process of seeking and interpreting evidence for use by learners and their teachers to decide where the learners are in their learning, where they need to go and how best to get there". Assessment is one of the main tasks of the education process. It takes an important weight during the development of any e-learning course. Exams are most widely used to assess student learning. However, exams can be classified into three types: traditional exams, online exams and distance exams (D-exams). Traditional exam defined as a set of questionnaires given in the class. They are created based on static questions per student. As a result, students must begin and end the exam within the same time limits. Online exams, sometimes referred to as e-

examination, are Internet based questionnaire. They are created randomly from questions set per student with a preset time limits by which the exam is to be completed. Furthermore, students should attend to a classroom for performing an exam.

D-exams are a way of delivering questions to students who are not physically present in a traditional setting such as a classroom. They are created randomly from questions set per student with a preset time limit by which they should be answered. Furthermore, they save or reduce time required for paper checking, as well as, they save papers, and printing, thus saving environment. D-exam present a new challenges for teachers; notably, how to prevent students from cheating. As a result, e-learning institutions depend on an examination process in which students take a face-to-face examination in a physical place located at the institution premises and under supervised conditions to ensure the student identity. However, that conflicts with the concept of E-learning, which eliminates the temporal and spatial dimensions between the students and the learning process. Each student must be physically present in the classroom in order to take the exam. Detecting and preventing cheating require a human intervention (i.e. the presence of a proctor). The proctor needs to physically authenticate students' IDs before starting the exam. However, this is not enough; we need continuous authentication all over the exam session. In addition, we need a continuous process of monitoring and controlling over all students during the exam period.

Cheating on exams has been a widespread phenomenon in the world regardless of the levels of detection development. Many studies have been conducted over the past decade about cheating activities performed by students and the means by which university could attempt to combat this problem. In the U.S., it was revealed that 80% of the higher achieving secondary school students admitted to cheat in during exams, 95% of secondary school students who admitted cheating said that they had not been caught, 51% of secondary school students did not believe cheating was wrong, 85% of college students said cheating was necessary to get ahead, 75% of college students

admitted cheating in exams, and 90% of college students did not believe cheaters would be caught.

The most common reasons that motivate students to cheat include: pressure from parents to do well, fear of failure, unclear instructional objectives, desire for a better grade, everyone else is doing it, there is no punishment if being caught, there is little chance of being caught, and no time to study and easy access to online information. Many students still use traditional cheating methods that are defined as each student can be cheating by own self or others. Cheating that appears in online exam is called online cheating. A student can be cheating via the Internet.

Moreover, there is no need for physical transfer. However, it faces the problem of cheating during examinations since there are no physical proctors invigilate and control the exam. This referred to as distance cheating. Distance cheating includes all previously mentioned kinds of cheating. In addition, there are other forms of cheating, such as: taking an examination for another student or having someone take an examination for one, using applications that help to solve the exam questions, copying test questions and sending them to an expert to send back the answers, and downloading resources from the Internet, for example, using an e-book .

2. SOFTWARE PROJECT MANAGEMENT PLAN

2.1 PROJECT ORGANIZATION

Software Process Model The Waterfall Model was first Process Model to be introduced. In a Waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. Waterfall model is the earliest SDLC approach that was used for software development. In "The Waterfall" approach, the whole process of software development is divided into separate phases. The outcome of one phase acts as the input for the next phase sequentially. This means that any phase in the development process begins only if the previous phase is complete. The waterfall model is a sequential design process in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation and Maintenance. As the Waterfall Model illustrates the software development process in a linear sequential flow; hence it is also referred to as a Linear-Sequential Life Cycle Model.

2.2 SEQUENTIAL PHASES IN WATERFALL MODEL :

1. **Requirements:** The first phase involves understanding what need to be design and what is its function, purpose etc. Here, the specifications of the input and output or the final product are studied and marked.
2. **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The software code to be written in the next stage is created now.

3. **Implementation:** With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing. Integration and Testing: All the units developed in the implementation phase are integrated into a system after testing of each unit. The software designed, needs to go through constant software testing to find out if there are any flaw or errors. Testing is done so that the client does not face any problem during the installation of the software.
4. **Deployment of System:** Once the functional and non-functional testing is done, the product is deployed in the customer environment or released into the market.
5. **Maintenance:** This step occurs after installation, and involves making modifications to the system or an individual component to alter attributes or improve performance. These modifications arise either due to change requests initiated by the customer, or defects uncovered during live use of the system. Client is provided with regular maintenance and support for the developed software. All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model".

3. HARDWARE AND SOFTWARE Requirement

Hardware components:

- A computer running either macOS or the windows operating system.
- RAM 4gb or Above
- A working web cam.

Software requirement:

- Any operating system of Mac or windows or linux with 32bit or 64bit.
- Python interpreter 3.X version to run python scripts.
- Modules to be installed
  - numpy == 1.16.1
  - opencv_python == 4.2.0.32
  - dlib == 19.16.0
- Pycharm code editor to execute python script .
- Any other code editor like VScode, sublime text can also be used.

4. Tools and programming languages Required

Python programming language: Python is an interpreted, high-level and general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library. Python was created in the late 1980s, and first released in 1991, by Guido van Rossum as a successor .

the ABC programming language. Python 2.0, released in 2000, introduced new features, such as list comprehensions, and a garbage collection system with reference counting, and was discontinued with version 2.7 in 2020. Python 3.0, released in 2008, was a major revision of the language that is not completely backward-compatible and much Python 2 code does not run unmodified on Python 3. With Python 2's end-oflife (and pip having dropped support in 2021), only Python 3.6.x and later are supported, with older versions still supporting e.g. Windows 7 (and old installers not restricted to 64-bit Windows). Python interpreters are supported for mainstream operating systems and available for a few more (and in the past supported many more). A global community of programmers develops and maintains CPython, a free and open-source reference implementation. A non-profit organization, the Python Software Foundation, manages and directs resources for Python and CPython development. As of January 2021, Python ranks third in TIOBE’s index of most popular programming languages, behind C and Java, having previously gained second place and their award for the most popularity gain for 2020

Dlib:

Dlib is principally a C++ library, however, you can use a number of its tools from python applications.

- **Documentation**
  - Unlike a lot of open source projects, this one provides complete and precise documentation for every class and function. There are also debugging modes that check the documented preconditions for functions. When this is enabled it will catch the vast majority of bugs caused by calling functions incorrectly or using objects in an incorrect manner.
  - Lots of example programs are provided
  - *I consider the documentation to be the most important part of the library*. So if you find anything that isn't documented, isn't clear, or has out of date documentation, tell me and I will fix it.
- **High Quality Portable Code**
  - Good unit test coverage. The ratio of unit test lines of code to library lines of code is about 1 to 4.
  - The library is tested regularly on MS Windows, Linux, and Mac OS X systems. However, it should work on any POSIX system and has been used on Solaris, HPUX, and the BSDs.
  - No other packages are required to use the library. Only APIs that are provided by an out of the box OS are needed.
  - There is no installation or configure step needed before you can use the library. See the How to compile page for details.
  - All operating system specific code is isolated inside the OS abstraction layers which are kept as small as possible. The rest of the library is either layered on top of the OS abstraction layers or is pure ISO standard C++.
- **Machine Learning Algorithms**
  - Deep Learning
  - Conventional SMO based Support Vector Machines for classification and regression
  - Reduced-rank methods for large-scale classification and regression
  - Relevance vector machines for classification and regression
  - General purpose multiclass classification tools
  - A Multiclass SVM

- A tool for solving the optimization problem associated with structural support vector machines.
- Structural SVM tools for sequence labeling
- Structural SVM tools for solving assignment problems
- Structural SVM tools for object detection in images as well as more powerful (but slower) deep learning tools for object detection.
- Structural SVM tools for labeling nodes in graphs
- A large-scale SVM-Rank implementation
- An online kernel RLS regression algorithm
- An online SVM classification algorithm
- Semidefinite Metric Learning
- An online kernelized centroid estimator/novelty detector and offline support vector one-class classification
- Clustering algorithms: linear or kernel k-means, Chinese Whispers, and Newman clustering.
- Radial Basis Function Networks
- Multi layer perceptrons

- **Numerical Algorithms**
  - A fast matrix object implemented using the expression templates technique and capable of using BLAS and LAPACK libraries when available.
  - Numerous linear algebra and mathematical operations are defined for the matrix object such as the singular value decomposition, transpose, trig functions, etc.
  - General purpose unconstrained non-linear optimization algorithms using the conjugate gradient, BFGS, and L-BFGS techniques
  - Levenberg-Marquardt for solving non-linear least squares problems
  - Box-constrained derivative-free optimization via the BOBYQA algorithm
  - An implementation of the Optimized Cutting Plane Algorithm
  - Several quadratic program solvers
  - Combinatorial optimization tools for solving optimal assignment and min cut/max flow problems as well as the CKY algorithm for finding the most probable parse tree

- A big integer object
- A random number object

- **Graphical Model Inference Algorithms**
  - Join tree algorithm for exact inference in a Bayesian network.
  - Gibbs sampler markov chain monte carlo algorithm for approximate inference in a Bayesian network.
  - Routines for performing MAP inference in chain-structured, Potts, or general factor graphs.
- **Image Processing**
  - Routines for reading and writing common image formats.
  - Automatic color space conversion between various pixel types
  - Common image operations such as edge finding and morphological operations
  - Implementations of the SURF, HOG, and FHOG feature extraction algorithms.
  - Tools for detecting objects in images including frontal face detection and object pose estimation.
  - High quality face recognition
- **Threading**
  - The library provides a portable and simple threading API
  - A message passing pipe for inter-thread and inter-process communication
  - A timer object capable of generating events that are regularly spaced in time
  - Threaded objects
  - Threaded functions
  - Parallel for loops
  - A thread_pool with support for futures
- **Networking**
  - The library provides a portable and simple TCP sockets API
  - An object to help you make TCP based servers
  - iostream and streambuf objects that enables TCP sockets to interoperate with the C++ iostreams library

- A simple HTTP server object you can use to embed a web server into your applications
- A message passing pipe for inter-thread and inter-process communication
- A tool used to implement algorithms using the Bulk Synchronous Parallel (BSP) computing model

- **Graphical User Interfaces**
  - The library provides a portable and simple core GUI API
  - Implemented on top of the core GUI API are numerous widgets
  - Unlike many other GUI toolkits, the entire dlib GUI toolkit is threadsafe
- **Data Compression and Integrity Algorithms**
  - A CRC 32 object
  - MD5 functions
  - Various abstracted objects representing parts of data compression algorithms. Many forms of the PPM algorithm are included.
- **Testing**
  - A thread safe logger object styled after the popular Java logger log4j
  - A modular unit testing framework
  - Various assert macros useful for testing preconditions
- **General Utilities**
  - A type-safe object to convert between big and little endian byte orderings
  - A command line parser with the ability to parse and validate command lines with various types of arguments and options
  - An XML parser
  - An object that can perform base64 conversions
  - Many container classes
  - Serialization support
  - Many memory manager objects that implement different memory pooling strategies

Some definitions :-

OpenCV :

OpenCV was started at Intel in 1999 by **Gary Bradsky**, and the first release came out in 2000. **Vadim Pisarevsky** joined Gary Bradsky to manage Intel's Russian software OpenCV team. In 2005, OpenCV was used on Stanley, the vehicle that won the 2005 DARPA Grand Challenge. Later, its active development continued under the support of Willow Garage with Gary Bradsky and Vadim Pisarevsky leading the project. OpenCV now supports a multitude of algorithms related to Computer Vision and Machine Learning and is expanding day by day. OpenCV supports a wide variety of programming languages such as C++, Python, Java, etc., and is available on different platforms including Windows, Linux, OS X, Android, and iOS. Interfaces for high-speed GPU operations based on CUDA and OpenCL are also under active development.
OpenCV-Python is the Python API for OpenCV, combining the best qualities of the OpenCV C++ API and the Python language.

OpenCV-Python :

OpenCV-Python is a library of Python bindings designed to solve computer vision problems.

Python is a general purpose programming language started by **Guido van Rossum** that became very popular very quickly, mainly because of its simplicity and code readability. It enables the programmer to express ideas in fewer lines of code without reducing readability.

Compared to languages like C/C++, Python is slower. That said, Python can be easily extended with C/C++, which allows us to write computationally intensive code in C/C++ and create Python wrappers that can be used as Python modules. This gives us two advantages: first, the code is as fast as the original C/C++ code (since it is the actual C++ code working in background) and second, it easier to code in Python than C/C++. OpenCV-Python is a Python wrapper for the original OpenCV C++ implementation.

OpenCV-Python makes use of **Numpy**, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

Pycharm :

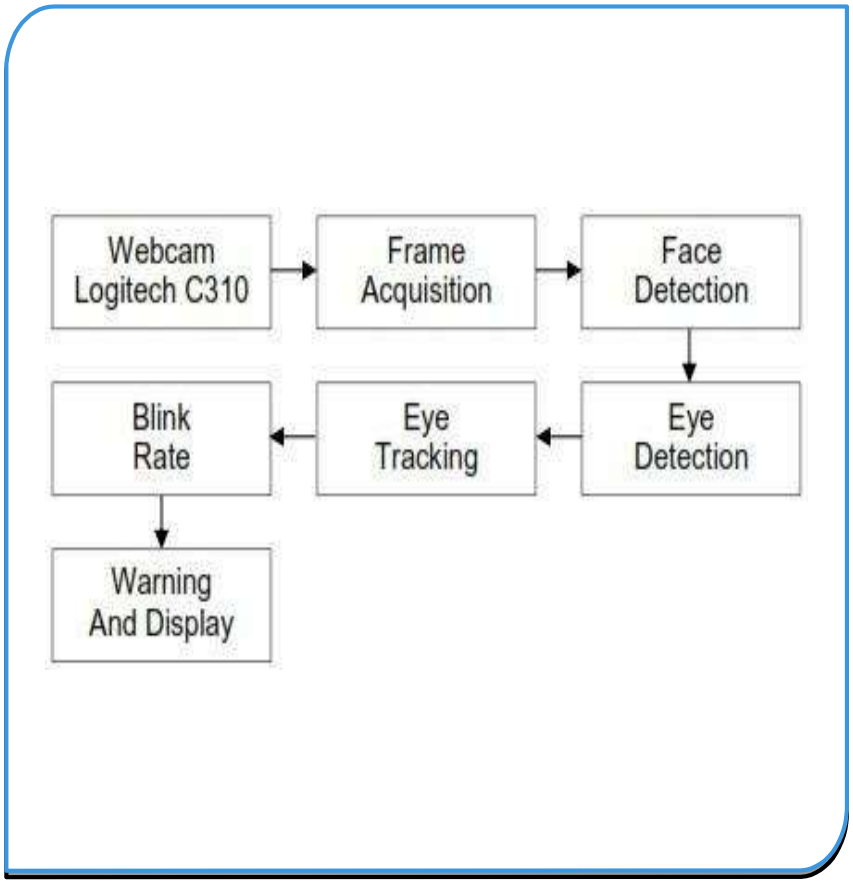
PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as data science with Anaconda. PyCharm is cross-platform, with
Windows, macOS and Linux versions. The Community Edition is released under the Apache License, and there is also Professional Edition with extra features – released under a proprietary license.

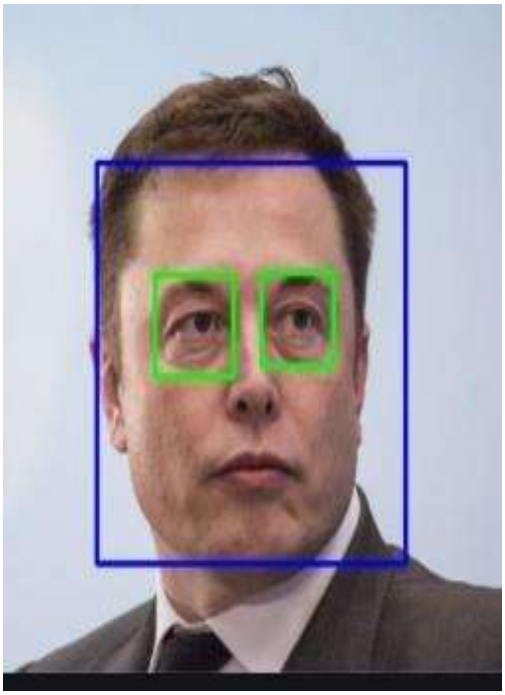
Face Encoding:

The face_recognition API generates face encodings for the face found in the images. A face encoding is basically a way to represent the face using a set of 128 computer-generated measurements. Two different pictures of the same person would have similar encoding and two different people would have totally different encoding.

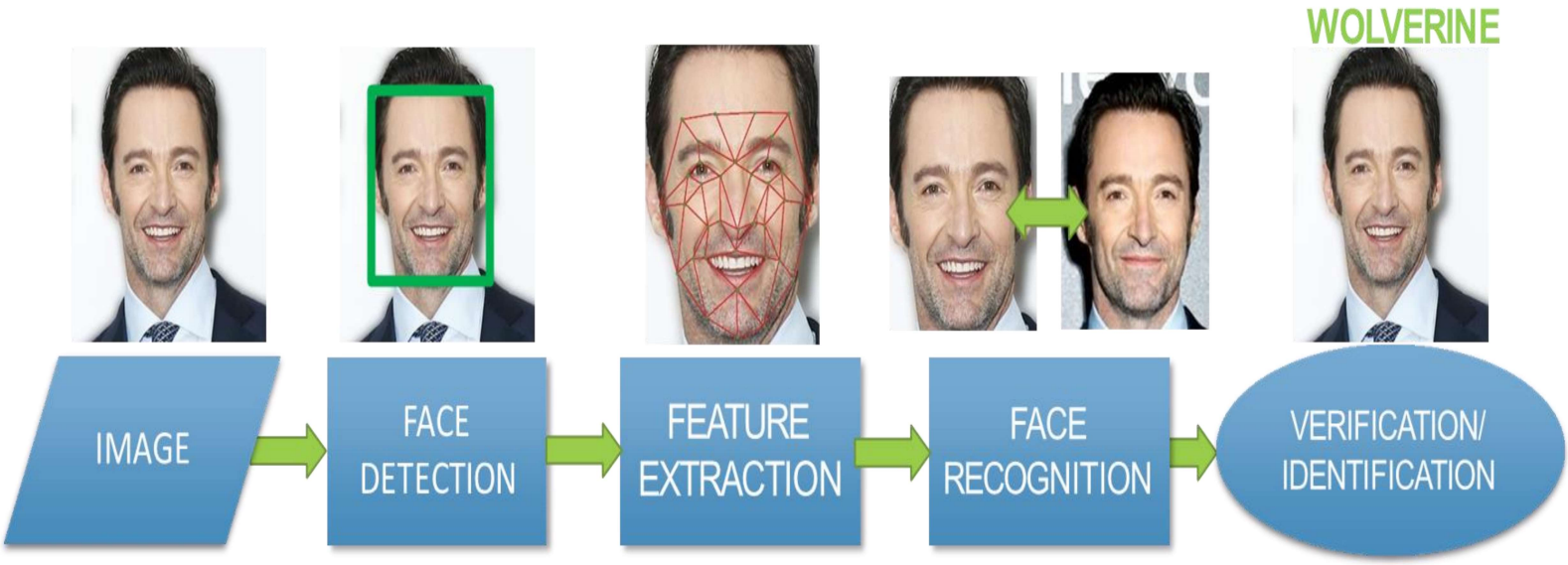
After all the face encodings are generated, Support Vector Classifier (SVC) with scikit-learn is trained on the face encodings along with their labels from all the known faces in the training directory. Finally, the API detects all the faces in the test image you provide and the trained SVC predicts all the known faces in the test image.

5 . Working and use case models





(Fig 5.1 Block diagram of eye movement detection)



(Fig 5.2 Block diagram of face recognition)

5.1 Working Phases :

5.1.1 Eye movement tracking

- Our three-phase feature-based eye gaze tracking approach uses eye features and head pose information to enhance the accuracy of the gaze point estimation.

- In Phase 1, we extract the eye region that contains the eye movement information. Then, we detect the iris center and eye corner to find the eye coordinates.

- Phase 2 obtains the parameters for the mapping function, which describes the relationship between the eye coordinates and the gaze point on the screen. In Phases 1 and 2, a calibration process computes the mapping from the eye coordinates to the coordinates of the monitor screen.

- Phase 3 entails the head pose estimation and gaze point mapping. It combines the eye coordinates and head pose information to obtain the gaze point.

5.1.2 Face Recognition

- Phase 1 Detect the face and crop the image to reflect only face .

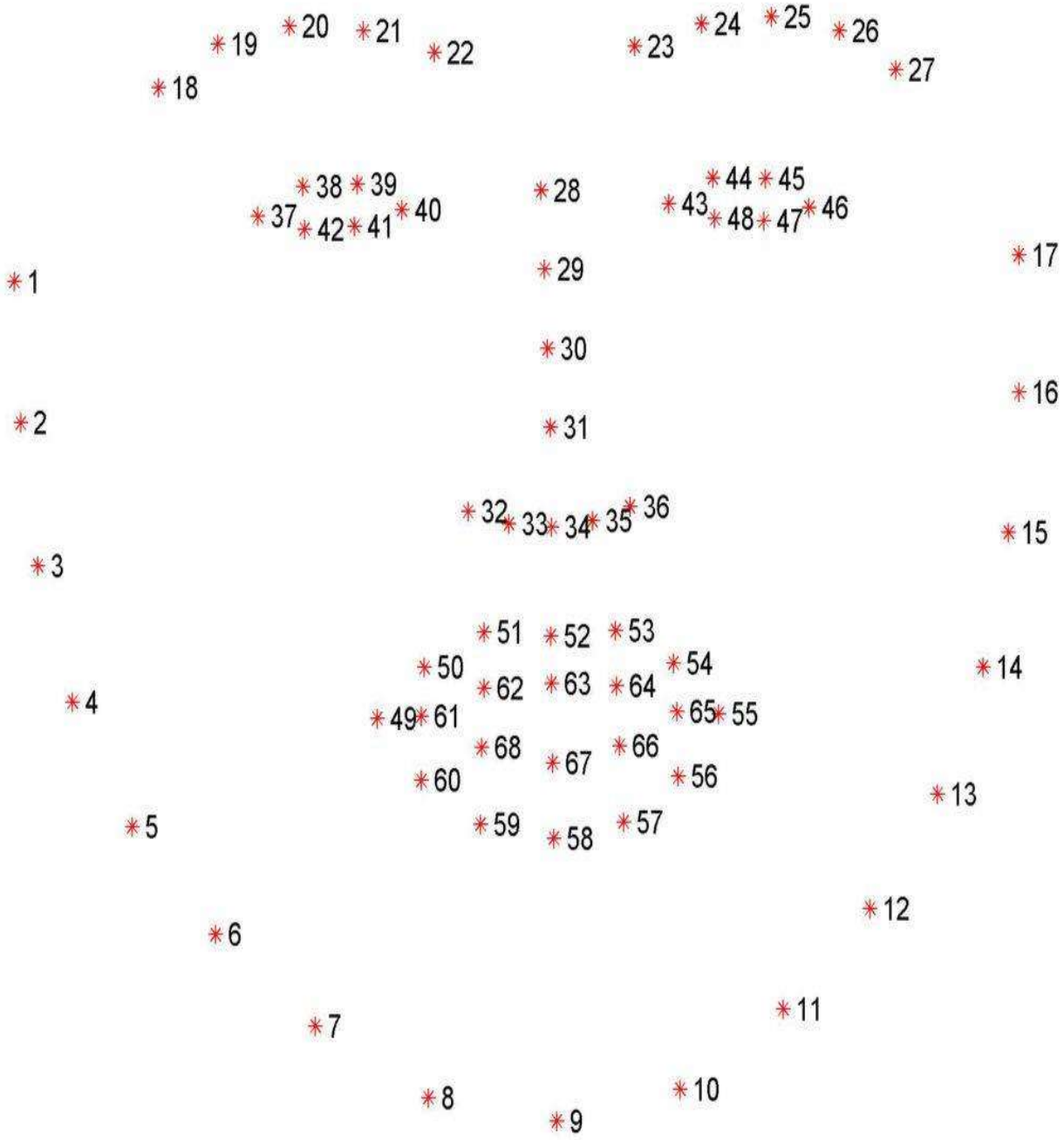
- In Phase 2 Convert physical feature of face into mathematical vectors using FACE RECOGNITION algorithms.

- Phase 3 Compare the feature of test image to Face database with the help of FACE ENCODING.

- Phase 4 If the person matches with images in database it display the name of recognized person otherwise displays unknown.

5.2 dlib's facial landmark detector :

The pre-trained facial landmark detector inside the dlib library is used to estimate the location of ***68 (x, y)-coordinates*** that map to facial structures on the face. The indexes of the 68 coordinates can be visualized on the image below:



(Fig 5.3 the 68 facial landmark coordinates)

5.3 **Classes Used :**

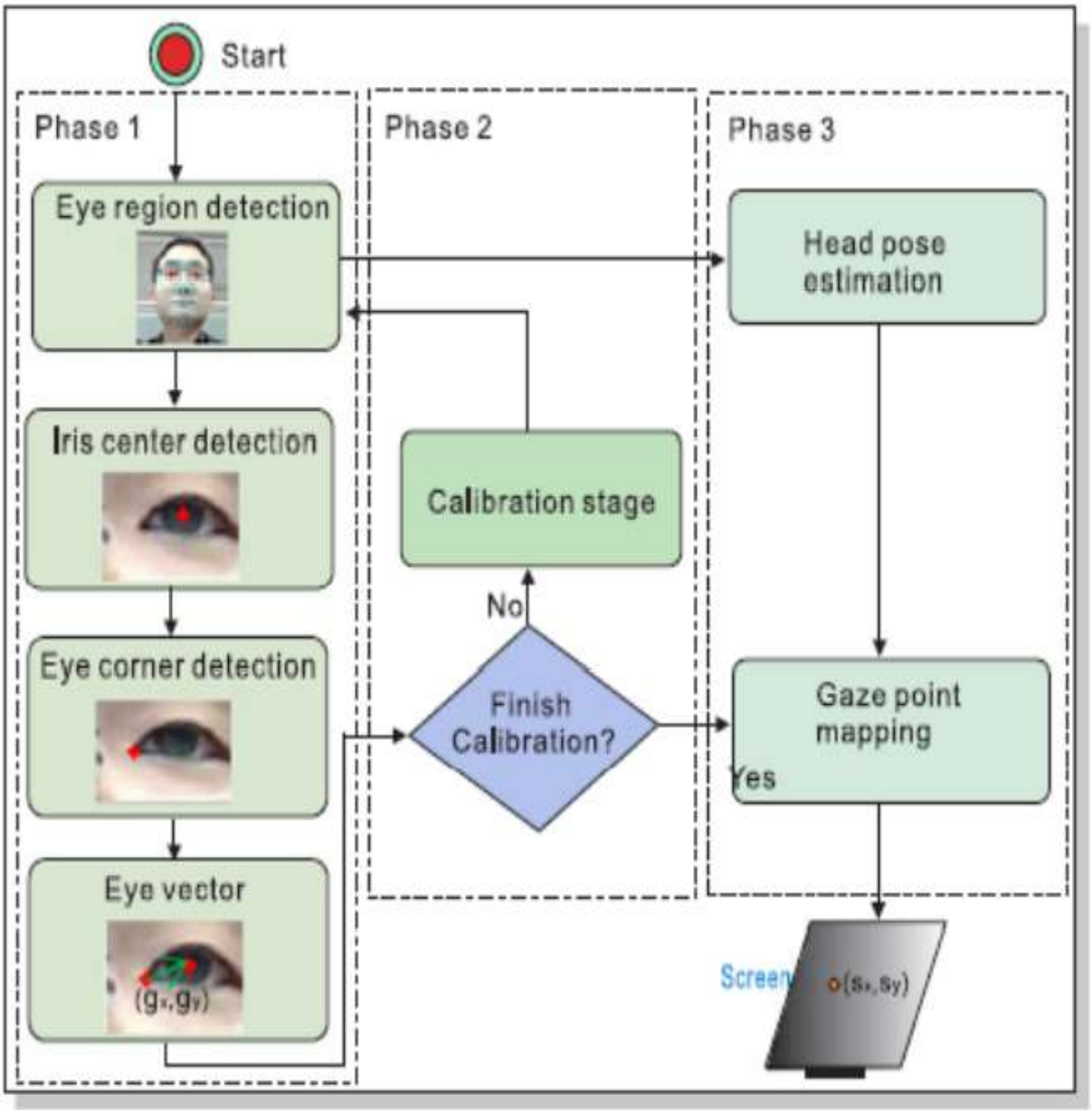
- **Gaze tracking :** This class tracks the user's gaze. It provides useful information like the position of the eyes and pupils and allows to know if the eyes are open or closed.

- **Eye detection :** This class creates a new frame to isolate the eye and initiates the pupil detection.

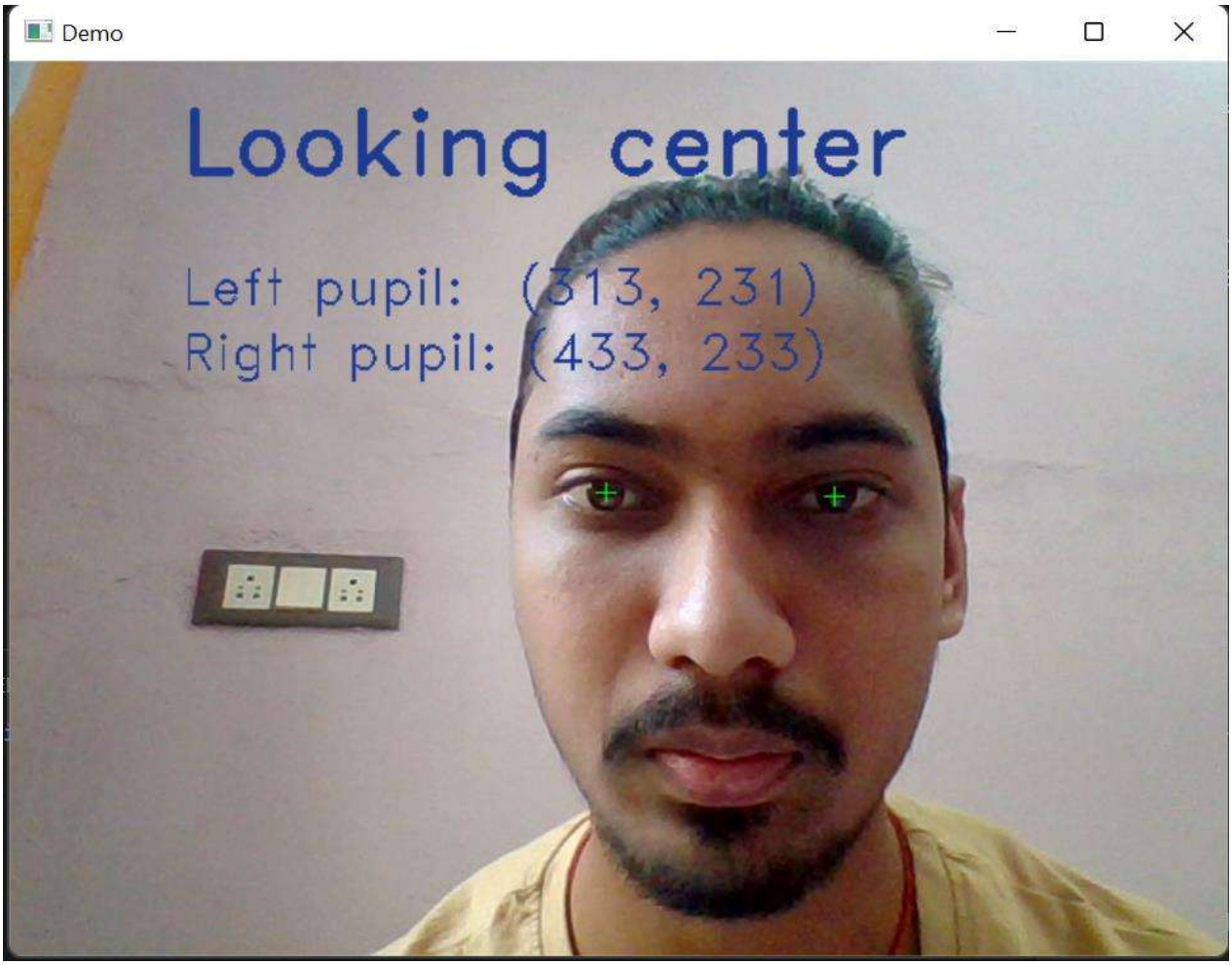
- **Pupil detection :** This class detects the iris of an eye and estimates the position of the pupil.

- **Calibration :** This class calibrates the pupil detection algorithm by finding the best binarization threshold value for the person and the webcam.

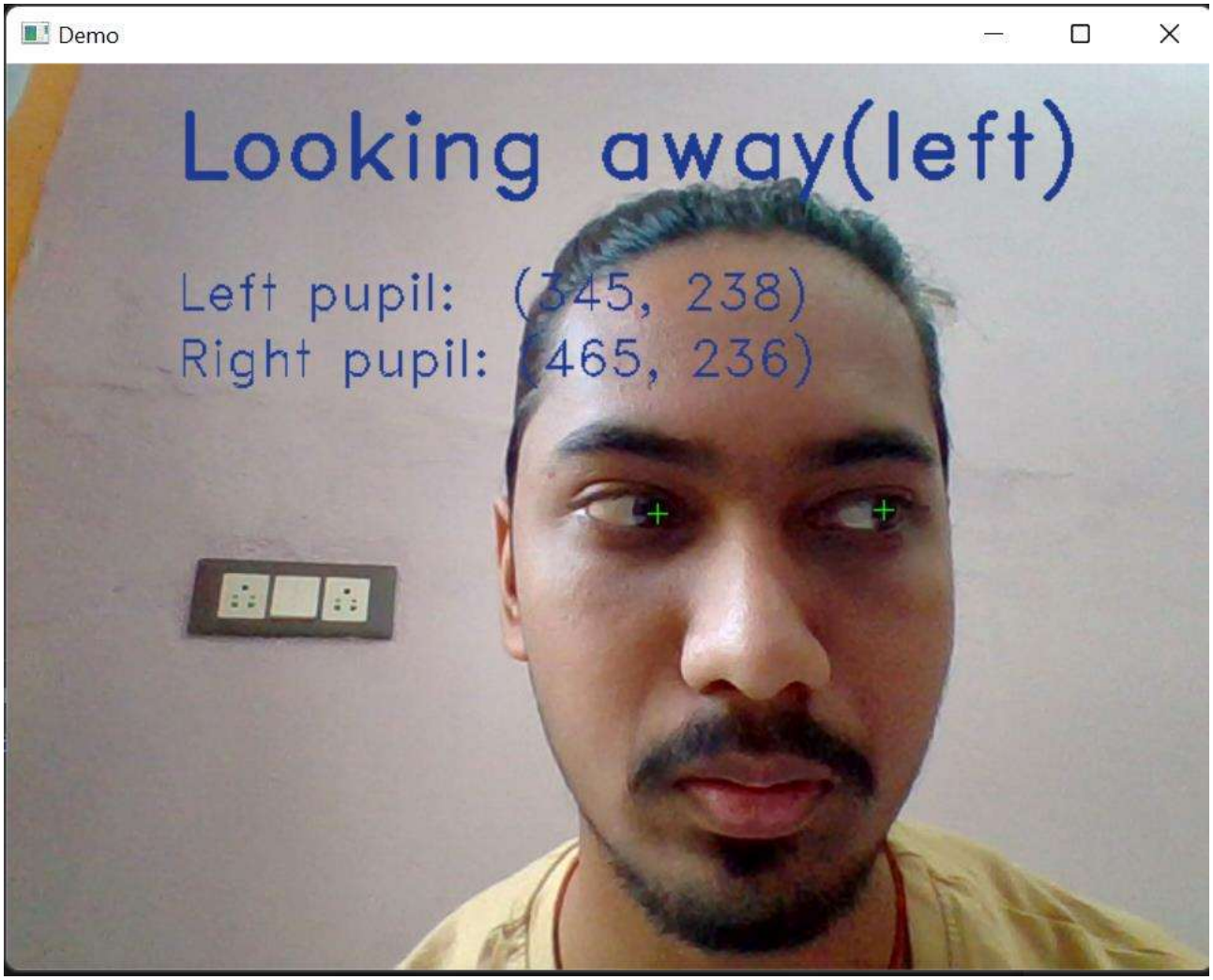
- **Face Encoding :** This The face_recognition API generates face encodings for the face found in the images. A face encoding is basically a way to represent the face using a set of *128* computer-generated measurements. Two different pictures of the same person would have similar encoding and two different people would have totally different encoding.



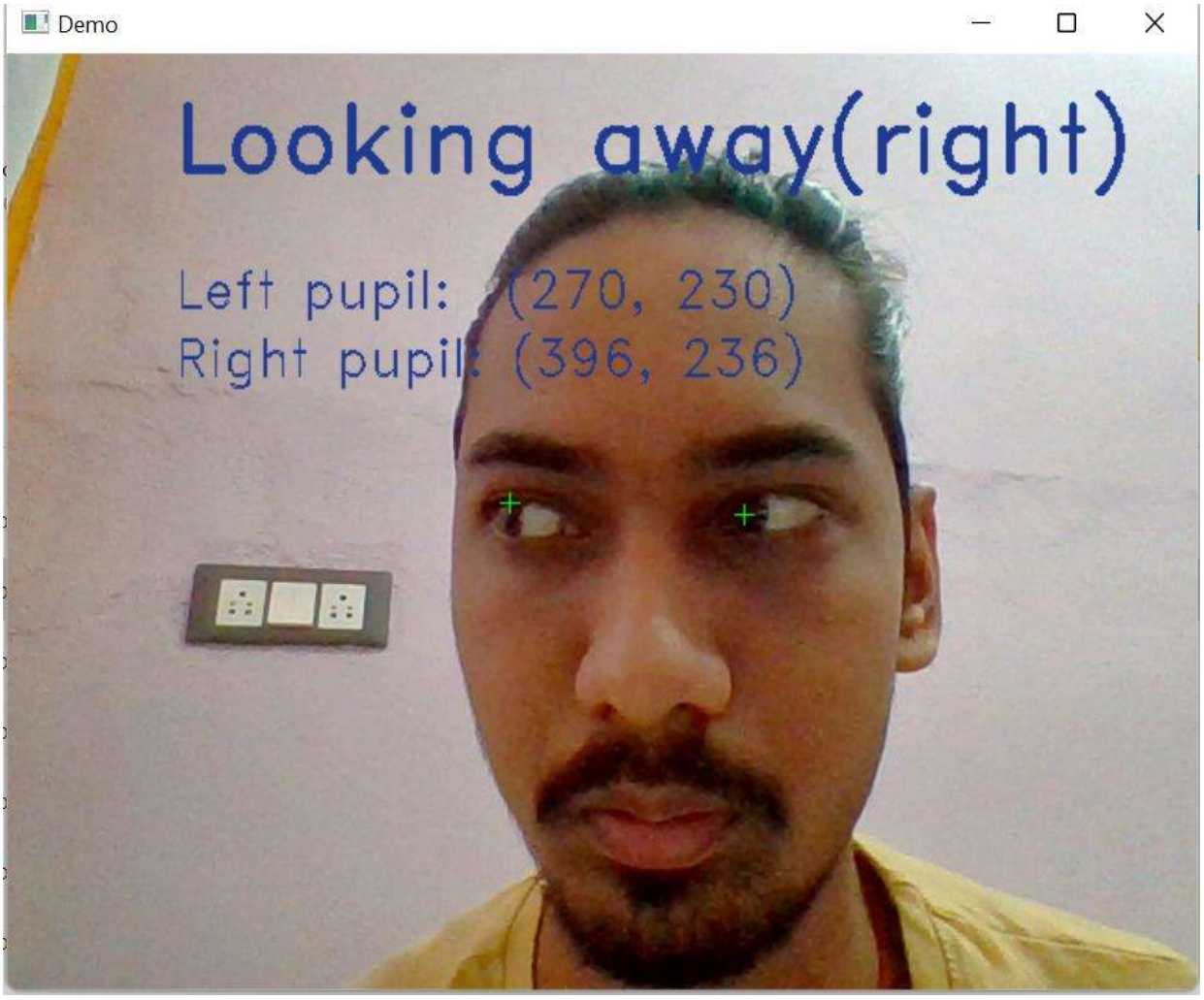
(fig 5.4 Flow chart)



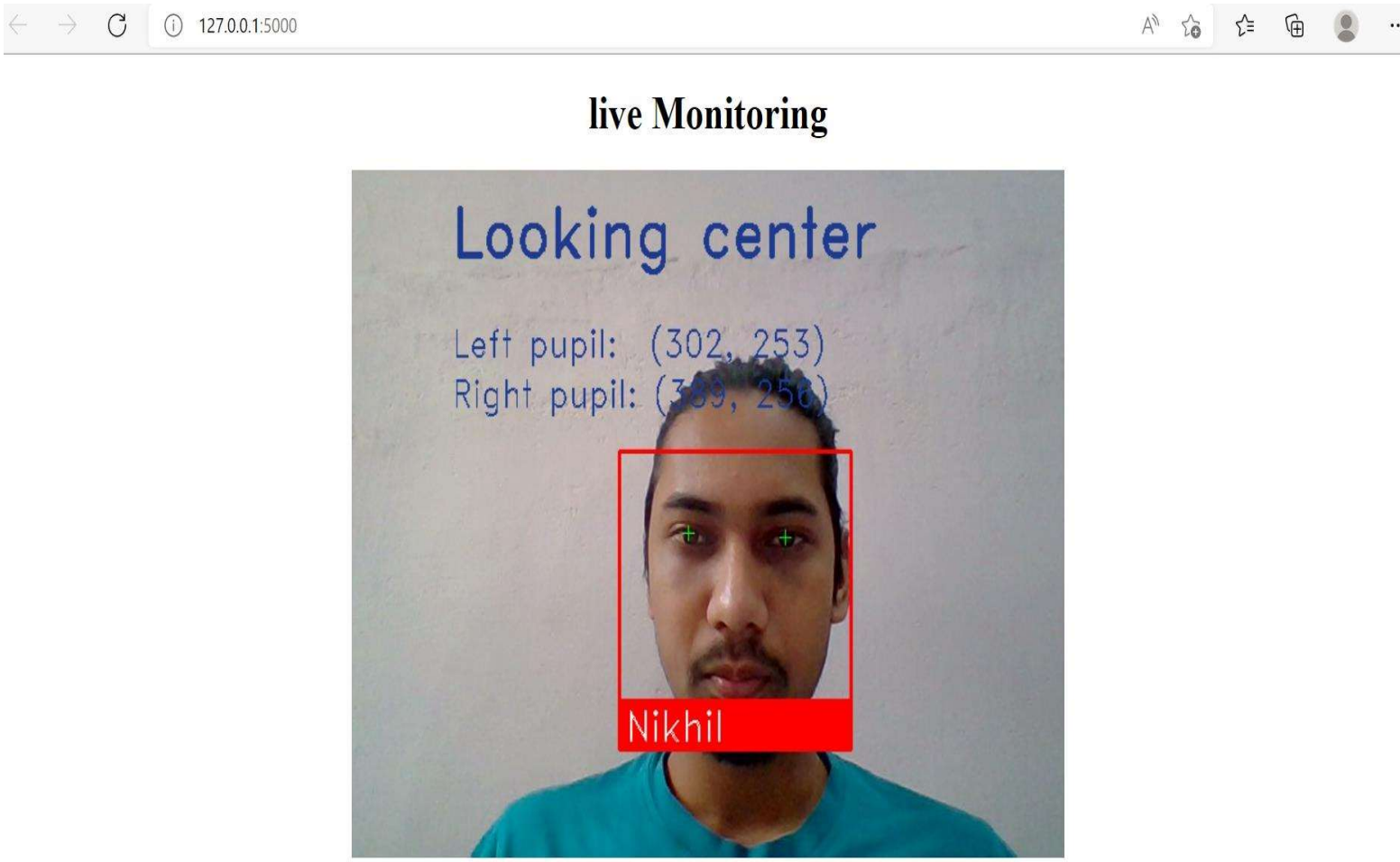
(fig 5.5 person looking center)



(fig 5.6 person looking away - left)



(fig 5.7 person looking away - right)



(fig 5.8 Face recognition)

6. ADVANTAGES OF PROPOSED SYSTEM

- The proposed approach can tolerate illumination changes and robustly extract the eye region, and provides an accurate method for the detection of the iris center and eye corner.
- A novel weighted adaptive algorithm for pose estimation is proposed to address pose estimation error; thus, improving the accuracy of gaze tracking.
- It does not require high resolution images for face recognition.
- It does not require HD camera and work completely fine with a laptops webcam.

7. Risks Involved

- The iris center detection will become more difficult than the pupil center detection because the iris is usually partially occluded by the upper eyelid.

- The construction of the classifier needs a large number of training samples, which consist of the eye images from subjects looking at different positions on the screen under the different conditions.

- They are sensitive to head motion and light changes, as well as the number of training samples.
- They are not tolerant to head movement.

- Operational Risks: Risks of loss due to improper process implementation Failed system or some external events risks.

- Causes of Operational risks: Failure to address priority conflicts. Failure to resolve the responsibilities .

8. Conclusion

Today, the human eye-gaze can be recorded by relatively unremarkable techniques. This thesis argues that it is possible to use the eye-gaze of a computer user in the interface to prevent the cheating in online exam.

we concentrate on visible-imaging and present an approach to the eye gaze tracking using a web camera in a desktop environment. First, we track the human face in a real time video sequence to extract the eye region. Then, we combine intensity energy and edge strength to locate the iris center and utilize the piecewise eye corner detector to detect the eye corner. Finally, eye gaze tracking is performed by the integration of the eye vector and head movement information

9. References

- https://docs.python.org/3/
- https://docs.opencv.org/
- http://dlib.net
- https://www.tutorialspoint.com/python3/python_gui_programming.html
- https://www.pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/
- https://www.geeksforgeeks.org/software-engineering-software-project management-spm/
- LAURENCE R. YOUNG AND DAVID SHEENA, Survey of Eye Movement Recording Methods, Behavior Research Methods and Instrumentation, Vol. 7, No. 5, pp. 397-429, 1975.
- ARIE E. KAUFMAN, AMIT BANDOPADHAY, BERNARD D. SHAVIV, An Eye Tracking Computer User Interface , Research Frontier in Virtual Reality Workshop Proceedings, IEEE Computer Society Press, pp. 78-84. October 1993,
- GLENN A. MYERS, KEITH R. SHERMAN, AND LAWRENCE STARK, Eye Mornitor, IEEE Computer Magazine, Vol. March, pp. 14-21, 1991. YOSHINOBU EBISAWA, Improved Video-Based Eye- Gaze Detection Method, IEEE IMTC '94, Hamamatsu, May, 1998.
- THOMAS E. HUTCHINSON, K. PRESTON WHITE, JR., WORTHY N. MARTIN, KELLY C. REICHERT, AND LISA A. FREY, Human-Computer Interaction Using Eye- Gaze Input, IEEE Trans. on Systems, Man, and Cybernetics, Vol. 19, No. 6, pp. 1527-1534, 1998.

- C. COLOMBO, S. ANDRONICO, AND P. DARIO, Prototype of a Vision-Based Gaze-Driven Man-Machine Interface, Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems, August, 1995.

- CHRISTOPHE COLLET, ALAIN FINKEL, AND RACHID GHERBI, CapRe: A Gaze Tracking System in Man- Machine Interaction, Proceedings of IEEE International Conference on Intelligent Engineering Systems, September, 1997.

- BAOSHEN Hu AND MINGHUA QIU, A New Method for Human-Computer Interaction by using Eye Gaze, Proceedings of IEEE International Conference on Systems, Man and Cybernetics, October, 1994. (8)RAINER STIEFELHAGEN, Gaze Tracking for Multimodal Human-Computer Interaction, Diplomarbeit, Universitiit Karlsruhe, September, 1996.