International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 10 Issue 5 ǁ 2022 ǁ PP. 63-72

Face Detection Using Open CV

Naman Singh, Ayush Vishwakarma, Isha Lakhekar, Abhijeet Mahawar
Mrs. Nilam Parmar

Abstract
Human visual awareness is a hot topic in the machine vision world right now. In applications such as video surveillance, human computer interface, face recognition, and image management, Human face localization and recognition is possible frequently the first step. Identifying and monitoring human faces might be difficult. Although a generic face picture is frequently accessible, face recognition and/or countenance analysis are required. The issue of the study of impartial facial data by computer-based face recognition remains a relatively unexplored field of research. Face recognition is one of the many marvels that AI research has offered to the world. Many techies are interested in this topic because they wish to have a fundamental grasp of how things work. Let's go into the subject to see how it works. This study describes how deep learning, an essential aspect of the computer science discipline, may be utilized to detect the face utilizing many libraries in OpenCV and Python. This article will include a suggested technology for detecting the human face in real time. This solution may be utilized on a variety of platforms, including machines, cellphones, and software applications. In the face of any image, this strategy is efficient and effective. In addition, the article discusses popular OpenCV applications and classifiers used in these applications, such as image processing, face identification, object detection and facial recognition. Finally, we address various literary assessments applications based on OpenCV in computer vision disciplines such as face detection and recognition, recognition of facial emotions such as grief, rage, and happiness, and recognition of a person's gender and age.

Keywords: *Face Detection, Object Detection, PyCharm, OpenCV, Machine Learning*

Date of Submission: 05-05-2022 Date of acceptance: 19-05-2022

I. INTRODUCTION

Face recognition is a technique for determining a person's identification based on their particular face.

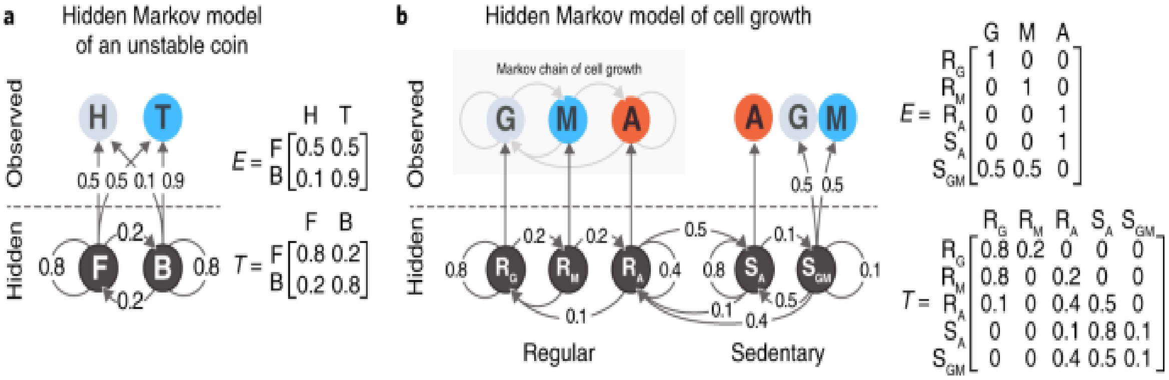
Such systems can be found in photographs, films, and real-time devices. The goal of this essay is to present a simple and straightforward way in machine technology. With the use of such a technique, one may quickly recognize a person's face by using a dataset with comparable matching look. The approach of detecting a person's face using Python and OpenCV in deep learning is the most efficient way. This strategy is beneficial in a variety of settings, including the military, security, schools, colleges and universities, aeroplanes, banking, online web applications, and gaming. It only recognises face traits and ignores everything else, including buildings, trees, and people. The perception of human faces is a hot topic in the computer vision world right now. Humans are used in applications like as video surveillance, computer interface, face recognition, and picture database management, human face localization and detection is frequently the initial step. Locating and monitoring human faces is required for facial recognition and/or emotion recognition analysis, while it is frequently believed that a normalized face picture is provided, making face detection and recognition simple and fast. One of the most exciting and difficult issues in the realm of artificial intelligence is computer vision. Computer Vision connects computer software to the images we perceive around us. It allows computer software to comprehend and learn about sights in its surroundings. As an example, the colour, shape, and size of the fruit are all determined. This task may appear straightforward to the human brain, but in the Computer Vision pipeline, we first gather data, then perform data processing operations, and then train and educate the model to learn how to distinguish between fruits based on size, shape, and colour. The major objective is to recognize and grasp the photos, as well as to provide fresh and improved photos valuable to us in the future.

II. RELATED WORK

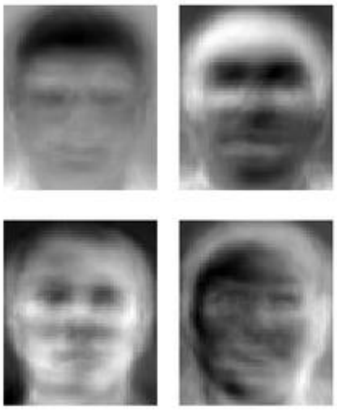
Face recognition that is both powerful and real-time is essential in many application contexts, such as biometrics, and is frequently used as part of (or in conjunction with) a facial recognition system. It's also utilized in video surveillance, HCI, and picture database management. Face detection is used for focusing in certain contemporary digital cameras. Face detection may also be used to identify places of curiosity in photo slideshows employ the Ken Burns pan-and-scale effect. Marketers are becoming interested in face detection. A camera fitted into a television can detect any passing face. The algorithm then computes the face's ethnicity, gender, and age range. The major goal or goal of this work is to build or construct a system that will utilize the computer's camera to identify and recognize a person's or individual's face using the OpenCV tool known as the Open Face and the Python programming language in the deep learning domain. The most practical use of face recognition is in biometrics, which is used for identification and makes the task simpler. Face recognition is one of the widely used technologies or systems that has the potential to perform tasks such as having records provided in by the dataset in many areas such as school and college attendance systems, it can also be useful in catching thieves or terrorists, it can be useful in the security of ordinary citizens and much-needed security zones in the country.

III. LITERATURE SURVEY

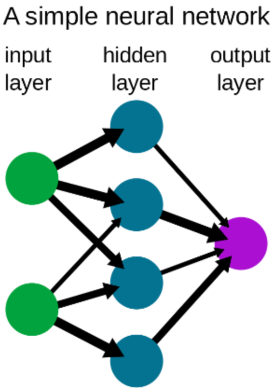
This section provides a general overview of the key approaches utilized in the face recognition system, which are largely applicable to the human's front face. Neural networks, hidden Markov models, geometric face matching, and template matching are among the approaches used. A Markov model is a stochastic approach for randomly changing systems in which future states are expected to be independent of past states. These models depict all potential states as well as transitions, transition rates, and the chances of their colliding. Markov models are frequently used to describe the probability of various states as well as the speeds of transitions between them. In general, the approach is used to model systems. Markov models may also be used to identify patterns, forecast outcomes, and discover the statistics of serial data


a
Hidden Markov model of an unstable coin
Observed
Hidden
H
T
0.5 0.5 0.1 0.9
F
B
0.2
0.2
0.8
0.8
E = F [0.5 0.5] B [0.1 0.9]
H T
F B
T = F [0.8 0.2] B [0.2 0.8]
b
Hidden Markov model of cell growth
Markov chain of cell growth
G
M
A
A
G
M
0.5 0.5
Observed
Hidden
R_G
R_M
R_A
S_A
S_GM
0.8
0.2
0.2
0.5
0.1
0.4
0.8
0.1
0.1
0.1
0.4
0.5
Regular
Sedentary
G M A
E = R_G [1 0 0] R_M [0 1 0] R_A [0 0 1] S_A [0 0 1] S_GM [0.5 0.5 0]
R_G R_M R_A S_A S_GM
T = R_G [0.8 0.2 0 0 0] R_M [0.8 0 0.2 0 0] R_A [0.1 0 0.4 0.5 0] S_A [0 0 0.1 0.8 0.1] S_GM [0 0 0.4 0.5 0.1]

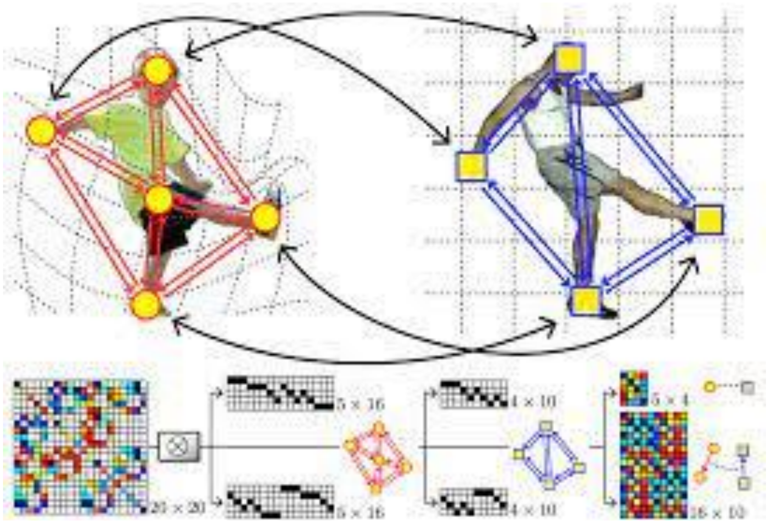
Eigenface is one of the most extensively used approaches for face identification and detection, often known as the principal components in mathematics. The eigenvectors are arranged to represent various quantities of the differences in the faces. When applied in the computer vision issue of human face identification, an eigenface is the term given to a set of eigenvectors. The eigenvectors are obtained from the probability distribution's covariance face's high-dimensional vector space of face photos. The eigenfaces comprise a basis set of all pictures required to build the covariance matrix. Dimension reduction is achieved by enabling the reduced collection of foundation photos to reflect the original training images. Classification may be accomplished by comparing how the base set represents faces. For instance, one's face may be made up of the average face plus 10% from eigenface 1, 55% from eigenface 2, and even 3% from eigenface 3. Surprisingly, a reasonable approximation of most faces may be obtained by combining a small number of eigenfaces. Furthermore, because a person's face is not recorded as a digital image, but as a list of values (one value for each eigenface in the database utilized, each person's face takes up far less space.



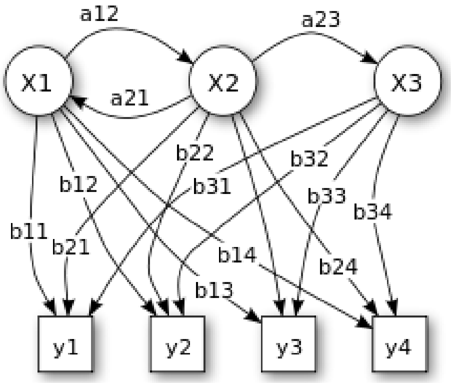
Face recognition and detection systems make extensive use of neural networks. In face recognition, an ANN (artificial neural network) with a single layer was deployed, demonstrating adaptability in critical face recognition systems. Face recognition is accomplished by the use of a double layer of WISARD in neural networks. WISARD is a Neural network model built on RAM model created in 1984 by Igor Aleksander at Brunel University. RAM–based neural networks, which rely on search tables to store the function generated by each neuron, are simply implemented in digital hardware and offer efficient training techniques. A neural network is a set of algorithms that attempts to recognize underlying links in a batch of data using a technique that mimics how the human brain works. In this way, neural networks interact with neuronal systems that might be biological or artificial in origin. An ANN is built from a network of linked units or nodes known as artificial neurons, which are roughly modelled after the neurons in the human brain.



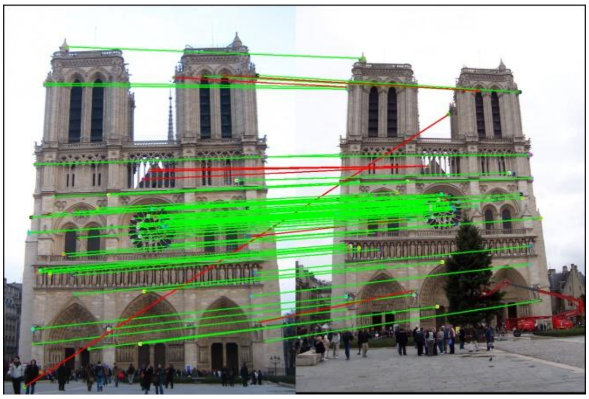
Graph matching is other option for face recognition. The object as well as the face recognition can be formulated using graph matching performed by optimization of a matching function. The task of discovering similarities between graphs is known as graph matching. Many applications, like computer vision and pattern recognition, employ graphs to encode structural information, and graph matching is a significant tool in these fields. It is often believed in these domains that the contrast is between the data and model graphs. The graph isomorphism issue is the situation of precise graph matching. The precise matching of a graph to a section of another graph is known as the subgraph isomorphism issue. The subgraph isomorphism issue is a difficult computer problem in theoretical computer science in which two graphs G and H are supplied as input, and one must determine if G includes a subgraph that is isomorphic to H. In image recognition applications, for example, the outputs of image segmentation in image processing frequently create data graphs with many more vertices than the model graphs data is anticipated to match against. Even though the number of vertices and edges in an attributed graph is the same, the matching may be merely approximate.



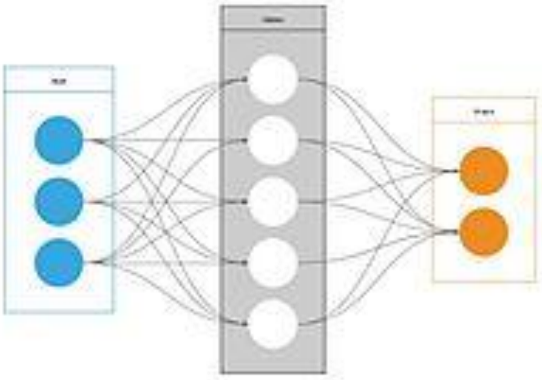
Hidden Markov Models are a method of applying stochastic modelling based on nonstationary vector time series HMM model to human face recognition, in which the faces are separated into sections such as the eyes, nose, ears, and so on. Face recognition and proper matching are 87 percent accurate because it always provides the best and correct choice of face detection through stored dataset. Otherwise, the relevant model shows the face's identity. A hidden Markov process may be seen as an extension of the urn problem with replacement in its discrete form (where each urn item before being restored to the original urn before proceeding to the next step). Part-of-speech tagging is one example, in which the hidden states indicate the fundamental elements of speech that correspond to a word sequence seen. In this scenario, the complete sequence of parts of speech is of relevance, rather than the segment of speech for a particular word, as computed by filtering or smoothing. It may be worthwhile to inquire about statistical significance for any of the difficulties listed above.



The technique of geometrical feature matching is based on the geometrical forms of the face. The geometrical face configuration has a large enough dataset to support a face detection and recognition system. This is a widely used method of face recognition and detection. This system appears to produce good outcomes. A feature is a piece of information that is useful for performing a computational job associated with a certain application. Features in a picture might be distinct structures such as points, edges, or objects. Features in an image can also be the outcome of a broad neighbourhood operation or feature detection. Mountain peaks, building corners, doors, or unusually shaped patches of snow are examples of characteristics found in specific areas in the photographs. These types of localized features are sometimes referred to as keypoint features (or even corners), and they are distinguished by the emergence of pixel patches around the spot position. Edges are characteristics in an image sequence that may be matched according to their orientation and physical appearance in the nearby area (edge profiles). They can also be useful markers of object boundaries and occlusion occurrences. The local appearance surrounding each feature point is defined in some form that is (hopefully) invariant when illumination, translation, scale, and in-plane rotation are changed. For each feature point, we usually end up with a descriptor vector. The interest point is the point at which the direction of the object's border rapidly changes or the junction point of two or more edge segments.



Template matching is one of the ways used to represent the test picture as a two-dimensional array of values that can be compared using Euclidean distance. A single template represents the entire face. This technique may potentially utilize many face templates from various sources angles of view to portray a single face. A template-based method may be successful for templates lacking significant characteristics or when the majority of the template picture forms the matched image. As previously stated, because template-based matching may necessitate the sample of a high number of points, the number of sampling points can be reduced by lowering the resolution of the search and template pictures by the same factor and executing the operation on the search and template images resulting downsized images (multiresolution, or pyramid), providing a search window of data points within the search image so that the template does not have to search every viable data point.



III. TOOLS USED

1. OPEN CV

OpenCV is a free-to-use library of open-source machine learning and computer vision software. The platform provides broad infrastructure for computer vision applications in order to accelerate the adoption of machine perception in commercially available inventions. Furthermore, OpenCV is licenced under the BSD licence, which has little restrictions on the library's use. Commercial enterprises can then modify the code to fit their needs. Over 2,500 algorithms are available through OpenCV for use in the implementation of various machine learning and computer vision capabilities such as item identification and facial recognition. It supports Windows, Linux, Android, and Mac OS and offers C++, Python, Java, and MATLAB interfaces. OpenCV is primarily geared at real-time vision applications, and it makes use of MMX and SSE instructions where they are available. Right present, full-featured CUDA and OpenCL interfaces are being actively developed. There are over 500 algorithms and around ten times that number of functions that build or support those algorithms. OpenCV is developed in C++ and provides a templated interface that integrates with STL containers. Hardware makers with a strong emphasis on certain platforms can improve implementations. This enables developers to design code that is portable between platforms many vendors, platforms, and hardware types. Computer Vision, on the other hand, enables computers to recognize things by analyzing digital photos or videos.

Benefits:

1. Vast Algorithms

OpenCV provides access to over 2,500 cutting-edge and traditional algorithms. Users may use this library to conduct a variety of activities such as erasing red eyes, extracting 3D models of objects, tracking eye movements, and so on.

2. Extensive Use

OpenCV is used for a variety of tasks by large corporations like as IBM, Google, and Toyota, as well as startups such as Zeitera and Applied Minds. This ensures that people have access to a library that is used by government agencies and businesses. Users in the huge OpenCV community may seek and offer assistance to other developers. This provides developers with access to people's knowledge of libraries and codes.

3. Efficient Solution

OpenCV's algorithmic efficiency is mostly used to process real-time programmers. Furthermore, it has been developed in such a manner that it can take use of hardware acceleration and multi-core systems when deployed.

2. PYTHON

Python is a high-level programming language that is interpreted, interactive, and object-oriented. Python is the most popular programming language today. Python is a popular programming language because it provides more code reliability, clean syntax, advanced language features, scalability of code, portability of code, support for object oriented programming, a broad standard library, is easy to learn and read, supports GUI mode, interactive, versatile, and interpreted, and interfaces to all major commercial databases, among other benefits. Python widely used in both industry and academia because of its simple, concise and extensive support of libraries.

Benefits:

1. Easy to learn

Python is simple to learn and apply. It is a high-level programming language that is user-friendly for developers. Python has minimal keywords, a straightforward structure, and a well-defined syntax, making it simple to learn. Python is more expressive, which means it is easier to grasp and comprehend for programmers. Python programming programmes are simple to design and run because they lack some of the bulky, difficult-to-understand, and perplexing elements found in other programming languages such as C++ and Java.

2. Interpreted Language

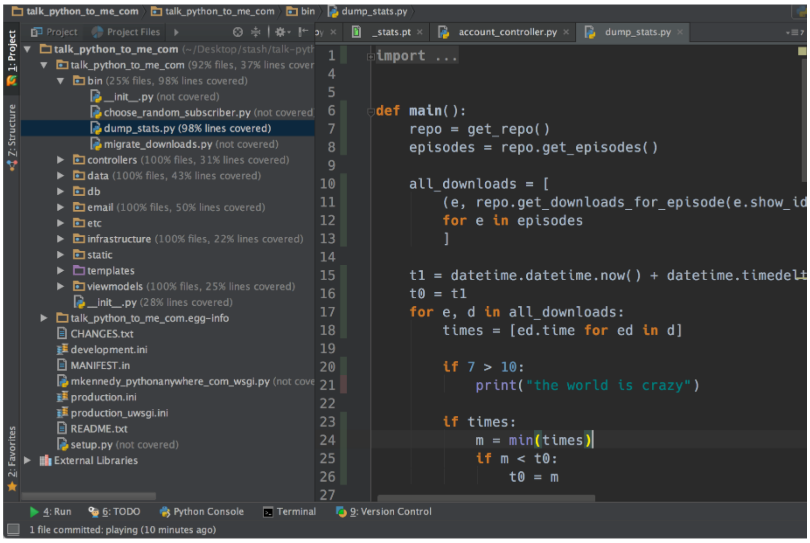
Python is an interpreted language, which means that the interpreter runs the code line by line. This simplifies debugging and makes it appropriate for novices. An good, simple tool for working with Python code is an interactive interpreter. To get started with Python, we don't need to learn a build system, an IDE, a particular text editor, or anything else. We simply require a command prompt and the interactive editor. Python includes a Python Shell (also known as the Python Interactive Shell) that may be used to run a single Python command and obtain the output seen below.

3. Object Oriented Language

An object-oriented programming language is one that can mimic the actual world. It is object-oriented and integrates data and functions. A procedure-oriented language, on the other hand, focuses around functions, which are reusable pieces of code. Python, as one of its core characteristics, enables both procedure-oriented and object-oriented programming. Unlike Java, it also permits multiple inheritance. Python offers a powerful yet simple approach to OOP, particularly when contrasted to C++ and Java. Python allows object oriented programming, and the ideas of classes and objects emerge. Python features a huge and diverse library, as well as a comprehensive set of modules and functions enabling quick application development. Python languages bulk library is portable and cross platform compatible with Unix, Windows etc. Python software is freely distributable, and anybody can use and read its source code, make changes/modifications to it, and combine the components in new free applications.

3. PyCHARM

An Integrated Development Environment (IDE) includes a code editor and a compiler for writing and compiling programmes in one or more programming languages. An integrated development environment (IDE) contains a multitude of features that facilitate software development.



```
talk_python_to_me_com (~/Desktop/stash/talk-pyth
  talk_python_to_me_com (92% files, 37% lines cover
    bin (25% files, 98% lines covered)
      __init__.py (not covered)
      choose_random_subscriber.py (not covered
      dump_stats.py (98% lines covered)
      migrate_downloads.py (not covered)
    controllers (100% files, 31% lines covered)
    data (100% files, 43% lines covered)
    db
    email (100% files, 50% lines covered)
    etc
    infrastructure (100% files, 22% lines covered)
    static
    templates
    viewmodels (100% files, 25% lines covered)
    __init__.py (28% lines covered)
  talk_python_to_me_com.egg-info
  CHANGES.txt
  development.ini
  MANIFEST.in
  mkennedy_pythonanywhere_com_wsgi.py (not cove
  production.ini
  production_uwsgi.ini
  README.txt
  setup.py (not covered)
External Libraries
```

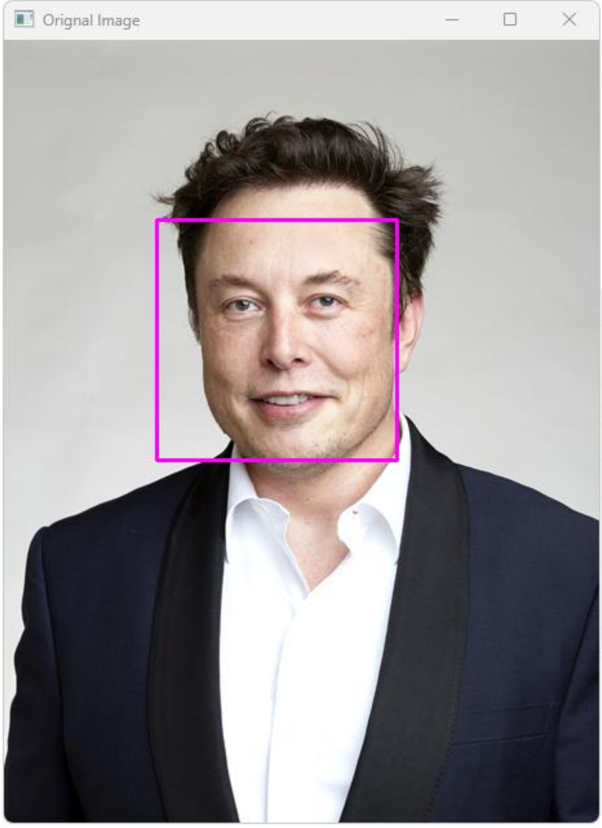
```python
import ...


def main():
    repo = get_repo()
    episodes = repo.get_episodes()

    all_downloads = [
        (e, repo.get_downloads_for_episode(e.show_id
        for e in episodes
        ]

    t1 = datetime.datetime.now() + datetime.timedelt
    t0 = t1
    for e, d in all_downloads:
        times = [ed.time for ed in d]

        if 7 > 10:
            print("the world is crazy")

        if times:
            m = min(times)
            if m < t0:
                t0 = m

```

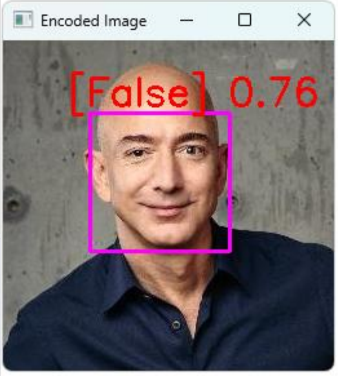
1 file committed: playing (10 minutes ago)

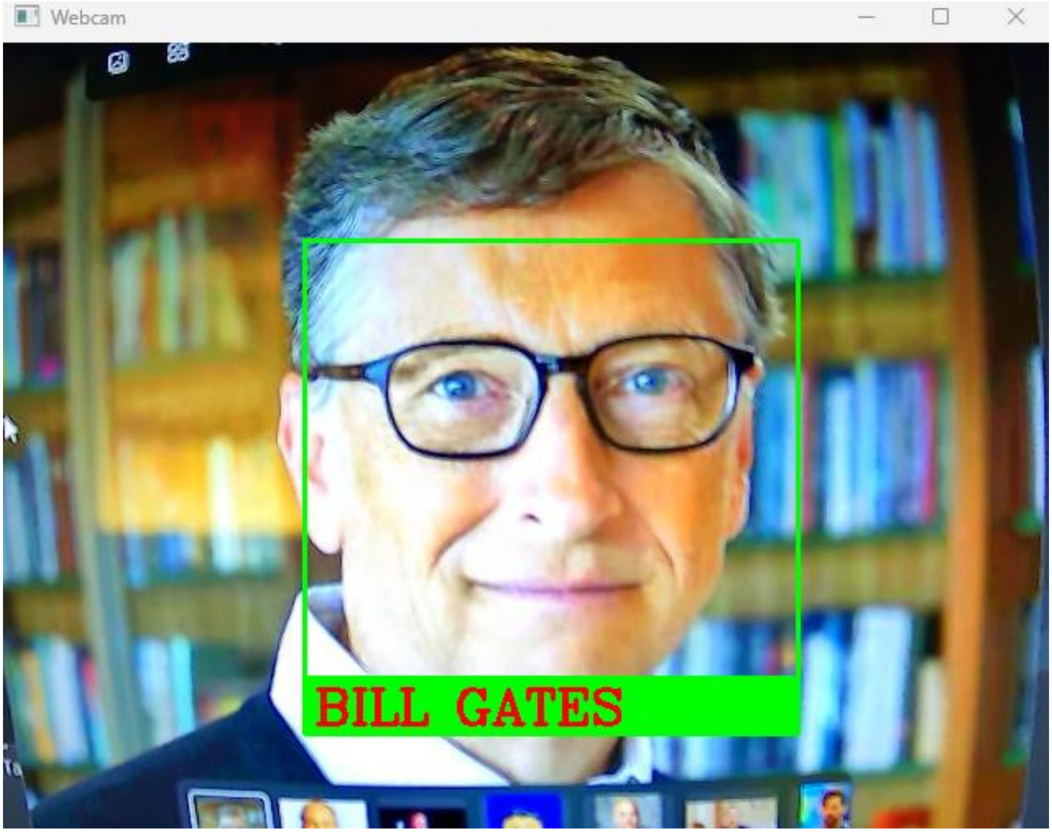
PyCharm has a number of interpreters, and the Run/Debug configuration allows you to choose which interpreter to use for your code. Each configuration setting includes options for Environment variables, Python interpreter, and working directory. This option allows you to modify how PyCharm executes your application. PyCharm offers faster code completion, whether for built-in or external packages. For SQL Language code, you can create a breakpoint, stop in the debugger, and examine the SQL representation of the user expression. Queries are common for a developer when working in Python. In PyCharm, you can quickly verify the last commit since it contains blue portions that define the difference between the last commit and the current one. All of the installed packages are properly represented visually. This contains a list of installed packages as well as the option to search for and add new ones. Local History is always keeping track of changes in a way that compliments Git. One of the cool features of PyCharm is the Structure tap in the bottom-left corner, which allows you to view how a file is structured with all kinds of classes, methods, and packages in your Python code. More advanced capabilities of PyCharm include the ability to split the screen horizontally and vertically and to work on many files in a project at the same time.

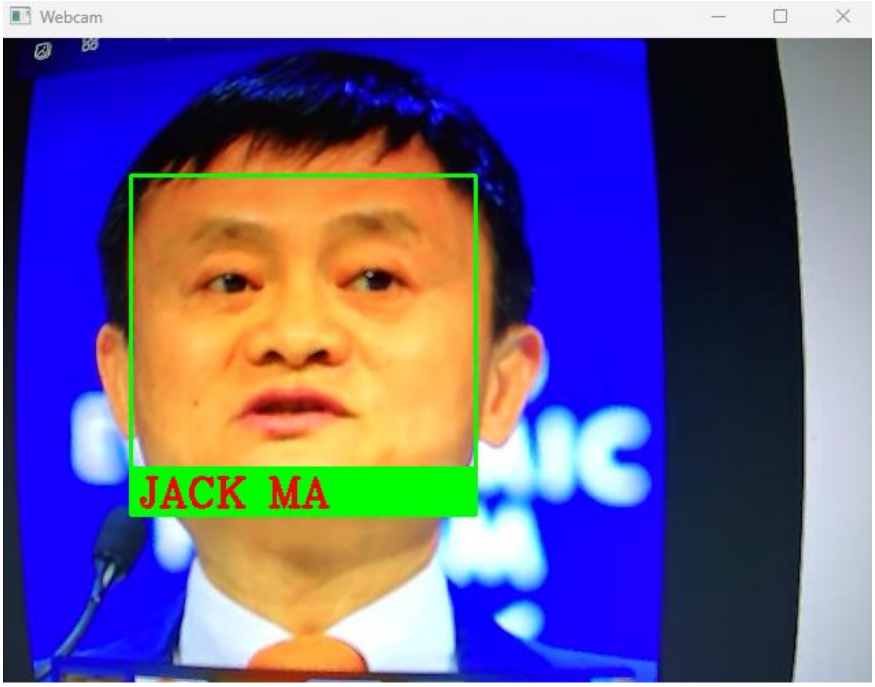
- PyCharm is simple to use and install.
- **Syntax and Error Maintenance**: It can assist in the analysis of syntax errors before compiling your code, reducing any cost.
- **Language Injections**: It enables you to utilise the result of a function, variable values, or any object within a template string that may be injected into an HTML tag for further use.
- **Import Assistance**: This allows missing libraries from another portion of the project to be imported.
- **Code Completion**: This IDE (Integrated Environment Development) based on the circumstances, allows you to finish the names of classes, methods, arguments, and other variables
- PyCharm comes with many plug-ins that can be used to enhance your project.
- It has a huge community that can help you in case you find any difficulty.

IV. OUTPUT









V. ADVANTAGES AND DISADVANTAGES

Face recognition systems have many advantages, including faster processing, identity automation, breach of privacy, massive data storage, best results, enhanced security, real-time face recognition of students in schools and colleges, employees in corporate offices, smartphone unlock, and many more in everyday life. A few disadvantages of this system include the costing or funding, the requirement for very good high definition cameras, poor image quality may limit the effectiveness of this system, the size of the image will matter because it becomes difficult to recognise the face in small images, face angles may limit the reliability of face recognition, and massive storage is required for this system to work effectively. Face recognition has also been used to target those participating in protected speech. Face recognition technology will most likely become more common in the near future.

VI. CONCLUSION

We underline the importance of OpenCV in face detection and identification in this research paper. Face recognition technologies are now linked to many major technical firms and sectors, making face recognition work easier. The use of Python Programming with OpenCV makes it a better and more useful tool or system that anybody may create based on their needs. The suggested system presented during this project would be beneficial to many people since it is user-friendly and cost-effective. As a result, the face recognition system may be created for a variety of uses using Python and OpenCV. Furthermore, such technology can be effective in tracking down a misplaced object in a dynamic environment. This technique may be improved further by including stereo depth analysis of face identification utilizing two image sensors interfaced with a high-speed processor. The usage of Python programming with OpenCV makes it a simpler and more useful tool or system that anybody may create based on their needs. The suggested system mentioned in this project would be beneficial to many people since it is user-friendly and cost-effective. As a result, the face recognition system may be created for a variety of uses using Python and OpenCV. The improvement of real-time face identification in remote monitoring aids in the development of numerous efficient industrial and commercial applications.

REFERENCES

[1]. Jagtap, A. M., Kangale, V., Unune, K., & Gosavi, P. (2019, February). A Study of LBPH, Eigenface, Fisherface and Haar-like features for Face recognition using OpenCV. In 2019 International Conference on Intelligent Sustainable Systems (ICISS) (pp. 219-224). IEEE

[2]. Sigut, J., Castro, M., Arnay, R., & Sigut, M. (2020). OpenCV basics: a mobile application to support the teaching of computer vision concepts. IEEE Transactions on Education, 63(4), 328-335

[3]. Mostafa, S. A., Mustapha, A., Gunasekaran, S. S., Ahmad, M. S., Mohammed, M. A., Parwekar, P., & Kadry, S. (2021). An agent architecture for autonomous UAV flight control in object classification and recognition missions. Soft Computing, 1-14

[4]. Hoque, M. A., Islam, T., Ahmed, T., & Amin, A. (2020, March). Autonomous face detection system from realtime video streaming for ensuring the intelligence security system. In 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS) (pp. 261-265). IEEE

[5]. Soomro, Z. A., Memon, T. D., Naz, F., & Ali, A. (2020, January). FPGA Based Real-Time Face Authorization System for Electronic Voting System. In 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET) (pp. 1-6).

[6]. Gupta, S. (2018, January). Facial emotion recognition in real-time and static images. In 2018 2nd international conference on inventive systems and control (ICISC) (pp. 553-560). IEEE

[7]. Kashinath, S. A., Mostafa, S. A., Mustapha, A., Mahdin, H., Lim, D., Mahmoud, M. A., & Yang, T. J. (2021). Review of Data Fusion Methods for Real-Time and Multi-Sensor Traffic Flow Analysis. IEEE Access

[8]. Abdulrazaq, M. B., Mahmood, M. R., Zeebaree, S. R., Abdulwahab, M. H., Zebari, R. R., & Sallow, A. B. (2021, February). An analytical appraisal for supervised classifiers' performance on facial expression recognition based on relief-F feature selection. In Journal of Physics: Conference Series (Vol. 1804, No. 1, p. 012055). IOP Publishing

[9]. Sriratana, W., Mukma, S., Tammarugwattana, N., & Sirisantisamrid, K. (2018, July). Application of the OpenCVPython for Personal Identifier Statement. In 2018 International Conference on Engineering, Applied Sciences, and Technology (ICEAST) (pp. 1-4). IEEE

[10]. James, C., & Nettikadan, D. (2019, April). Student monitoring system for school bus using facial recognition. In 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI) (pp. 659-663). IEEE

[11]. Balachandran, B., Saad, K. F., Patel, K., & Mekhiel, N. (2019, December). Parallel Computer for Face Recognition Using Artificial Intelligence. In 2019 14th International Conference on Computer Engineering and Systems (ICCES) (pp. 158-162). IEEE

[12]. Gupta, N., Sharma, P., Deep, V., & Shukla, V. K. (2020, June). Automated attendance system using OpenCV. In 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO) (pp. 1226-1230). IEEE