Table of Contents

[Abstract: 2](#_Toc2607834)

[Introduction: 2](#_Toc2607835)

[Motivation: 2](#_Toc2607836)

[Features: 3](#_Toc2607837)

[Components: 3](#_Toc2607838)

[Hardware Components: 3](#_Toc2607839)

[Software Components: 3](#_Toc2607840)

[Work back-down among team member: 4](#_Toc2607841)

[General phases in Face Recognition: 4](#_Toc2607841)

[Image Capturing: 5](#_Toc2607842)

[Face Detection: 5](#_Toc2607842)

[Feature Extraction: 6](#_Toc2607842)

[Template Comparison: 6](#_Toc2607842)

[Declare Matches: 6](#_Toc2607842)

[Working process: 7](#_Toc2607842)

[Flowchart of Face Recognition system: 7](#_Toc2607843)

[Result: 8](#_Toc2607845)

[Future scope: 8](#_Toc2607846)

[Implementation image: 9](#_Toc2607847)

[Conclusion: 9](#_Toc2607848)

[References: 10](#_Toc2607849)

# Abstract:

Identifying a person face from an image has been popularized through the mass media. This report describes the face detection project. It reports the technologies available in the Open-Computer-Vision (OpenCV) library and methodology to implement those using Python. For face detection, Haar-Cascades were used. Next, the results are shown including screen-shots which shows detected faces from an image. The reports concluded with the authors' opinion on the project and possible applications.

# Introduction:

The following document is a report on the mini project for Robotic visual perception and autonomy. It involved building a system for face detection using several classifiers available in the open computer vision library (OpenCV). Face detection is used to identify a face from an image. This is followed by the explanation of HAAR-cascades. Next, the methodology and the results of the project are described. A discussion regarding the challenges and the resolutions are described. Finally, conclusion is provided on the pros and cons of each algorithm and possible implementations.

# Motivation:

* To identify a human face from an image is the main purpose
* To study OpenCV and implement it in this project.
* To understand how face detection method works.

# Features:

* It can detect all the faces from an image.
* It is easy to use.
* It can work on low configured computer
* It does not use too much disk space
* Its requirement is easy to get and all software is free in online.

# Components:

## Hardware Components:

* A computer
* Processor minimum Intel core i5
* 4GB ram
* Hard disk storage 1TB

## 

## Software Components:

* Operating system : Windows 10
* Python 3.7.1
* Numpy
* Matplotlib
* OpenCV 3.0

# 

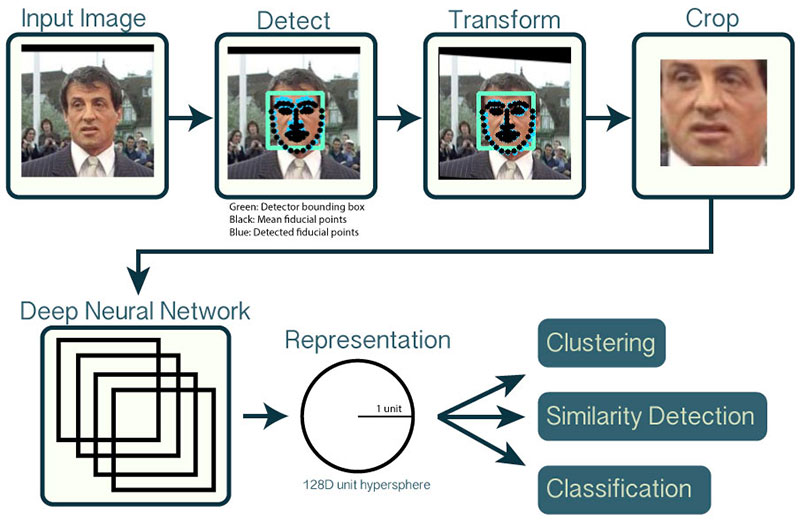
# Work back-down among team member:

|  |  |  |
| --- | --- | --- |
| **Name:** | **ID:** | **Work:** |
| 1)Sarker Adib Rahman | 153402328 | Coding, testing and project report |
| 2)Fazle Rabbi Masum | 152392317 | Implementation ,testing and project report |
| 3)Mohammad Nayeem | 153402303 | Code source and many reference collect |
| 4)Tanvir Hasan Saikat | 1534023106 | Testing and project report |

# General phases in Face Recognition:

There are five general phases in face recognition system. The system must execute all the phase before finally get into the expected result. The phases are.

1. Capture Image
2. Detect face in the image
3. Feature Extraction
4. Template Comparison
5. Declaration of matching template



# Image Capturing:

The acquisition of face images can be done by digitally scanning an existing photograph or by using an electro-optical camera. While to capture a real-time situation like in public place, the acquisition can be accomplished by using CCVT camera or any surveillance camera.

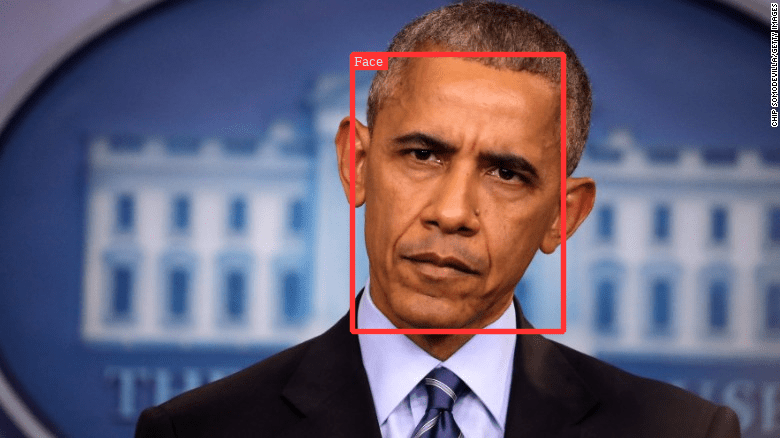


Figure : Web Camera

# Face Detection:

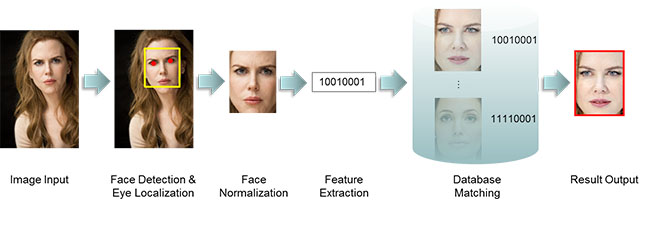
The function of this module is to determine where in an image a face is located. The face detection module works by scanning up an image at different scales and looking for some simple patterns that denote the presence of a face. After the system detected a face, it will then produce a sub-image (image chip) that is scaled such that the face appears in the centre and presented at a uniform size.

OpenCV already provide algorithm to locate faces in still images and videos. It is called the Haar-based Cascade Classifier or simply known as Haar Classifier. This algorithm scans and image and create a bounding box as a returns for each detected face.



# Feature Extraction :

Another phase in face recognition is feature extraction. This is a phase where the system does the localizing of the characteristics of face components (i.e. eyes, mouth, nose, etc) in an image. In other words, feature extraction is a step in face recognition where the system locates certain points on the face such as the corner and centre of the eyes, tip of the nose, mouth, etc. It analyze spatial geometry of differentiate feature of a face. The result of this analyzing is a set of template generated for each face. The template consists of a reduced set of data that represent the uniqueness of the enrolled face features.



**Template Comparison :**

The fourth phase of face recognition is to compare the templates generated in previous phase with the template in the database (enrolled templates). [29] There are two ways of comparing the templates based the purpose of the application itself. In *identification* application the template match all templates in the database and get the closest match (1 : N). While in *verification* application, the generated template will only be compared to one data entry in the database that is the claimed identity (1 : 1).

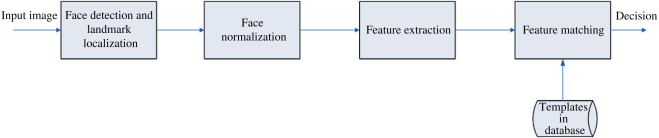
**Declare Matches:**

The final phase of face recognition is to declare the highest matching score resulted in the previous step. [29] The ground rules applied to the application to declare the strictness of the application are based on the configuration set up by the end users. The configuration will determine how the application should behave based on the desired security and operational consideration.

# Working process:

* First we do the downloaded Software and setup PC using Command Prompt.
* Then we write the code in a file and save it as dataset.py
* Then we launch python from our computer.
* We already wrote the code on the dataset.py file.
* Then we work the trainer we generate the code and check our process.
* Last we generate the code detector file and this is main part of our project because this part run and our face detected with our run.

# Flowchart of Face Recognition system:



# Result:

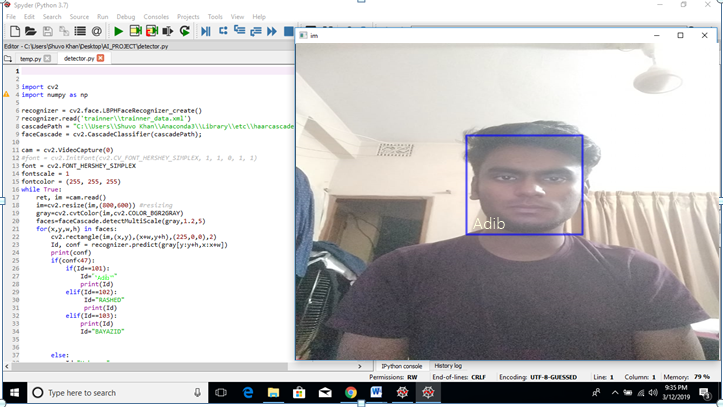
Our project was able to detect the faces from Matching then show name and do not match than display show in unknown successfully.

# Future scope:

The use of spherical canonical images allows us to perform matching in the spherical harmonic transform domain, which does not require preliminary alignment of the images. The errors introduced by embedding into an expressional space with some predefined geometry are avoided. In this facial expression recognition setup, end-to-end processing comprises the face surface acquisition and reconstruction, smoothening, sub sampling to approximately 2500 points. Facial surface cropping measurement of large positions of distances between all the points using a parallelized parametric version is utilized.

The general experimental evaluation of the face expressional system guarantees better face recognition rates. Having examined techniques to cope with expression variation, in future it may be investigated in more depth about the face classification problem and optimal fusion of color and depth information. Further study can be laid down in the direction of allele of gene matching to the geometric factors of the facial expressions. The genetic property evolution framework for facial expressional system can be studied to suit the requirement of different security models such as criminal detection, governmental confidential security breaches etc.

# Implementation image:



# Conclusion:

This paper describes the project for visual perception and autonomy module. Next, it explains the technologies used in the project and the methodology used. Finally, it shows the results, discuss the challenges and how they were resolved followed by a discussion. Using Haar-cascades for face detection worked extremely well even when subjects wore spectacles. Considering all factors combined with Haar-cascades can be implemented as a cost effective face detection platform.

# References:

[1] Takeo Kanade. Computer recognition of human faces, volume 47. Birkhauser Basel, 1977.

[2] Lawrence Sirovich and Michael Kirby. Low-dimensional procedure for the characterization of human faces. Josa a, 4(3):519{524, 1987.

[3] X. Wang, T. X. Han, and S. Yan. An hog-lbp human detector with partial occlusion handling. In

2009 IEEE 12th International Conference on Computer Vision, pages 32{39, Sept 2009.

[4] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In Proceed-ings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.

CVPR 2001, volume 1, pages I{511{I{518 vol.1, 2001.

[5] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection.

In Image Processing. 2002. Proceedings. 2002 International Conference on, volume 1, pages I{I.

IEEE, 2002.

[6] John P Lewis. Fast template matching. In Vision interface, volume 95, pages 15{19, 1995.