# **PROFESSIONAL TRAINING REPORT**

**at**

**Sathyabama Institute of Science and Technology (Deemed to be University)**

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

By

**CH.SAI SINDHU MEGHANA**

**REG.NO: 39110226**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SCHOOL OF COMPUTING**

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,**

**CHENNAI – 600119, TAMILNADU**

**APRIL 2022**

|  | **SATHYABAMA**  **INSTITUTE OF SCIENCE AND TECHNOLOGY** (DEEMED TO BE UNIVERSITY) **Accredited with Grade “A” by NAAC**  (Established under Section 3 of UGC Act, 1956)  JEPPIAAR NAGAR, RAJIV GANDHI SALAI  CHENNAI– 600119  [**www.sathyabama.ac.in**](http://www.sathyabama.ac.in) |  |
| --- | --- | --- |

# **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of **CH.SAI SINDHU MEGHANA (Reg. No: 39110226)** who carried out the project entitled “**Face analysis in a video call through image processing”** under my supervision from March 2022 to April 2022.

Internal Guide

**Dr.N.Srinivasan**

**Head of the Department**

**Dr. S. VIGNESHWARI, M.E., Ph.D.,**

**Dr. LAKSHMANAN L, M.E., Ph.D.,**

## Submitted for Viva voce Examination held

## Internal Examiner External Examiner

**DECLARATION**

I, **Ch.Meghana** hereby declare that the project report entitled PRIDEdone by me under the guidance of **Dr.N.Srinivasan** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering.

## DATE: CH.Meghana

**PLACE: CHENNAI SIGNATURE OF THE STUDENT**

**ACKNOWLEDGEMENT**

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D**, **Dean**, School of Computing, **Dr. S. Vigneshwari, M.E., Ph.D. and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of **Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.N.Srinivasan, M.E., Ph.D.,** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project

**ABSTRACT**

Video-based facial recognition has received significant attention in recent years due to its widespread applications. The main challenges faced during face recognition has large variability of images due to pose variations, illumination conditions, facial expressions and image occlusion.

Similarly surveillance and mobile cameras are of low cost devices which affects the video frames quality which results in severe motion blur, out-of-focus blur and large range pose variation. To achieve Face recognition from video image processing and machine learning techniques are used.

The steps for processing involves Image acquisition, Image segmentation, feature extraction, classification and face detection. The features extracted are used in training the classifiers for images that are processed. Hence most recent algorithms developed give an idea of state of art of face recognition technology in video.

TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| CHAPTER | TITLE NAME | PAGE NO |
|  | ABSTRACT | 5 |
|  | LIST OF FIGURES | 9 |
|  | LIST OF ABBREVATIONS | 8 |
| 1. | INTRODUCTION   * 1. LITERATURE SURVEY   2. PROPOSED SYSTEM | 10  12  13 |
| 2. | MODULES  2.1 MOTION BASE  2.2 LINEAR SUB SPACE METHOD  2.3 DIGITAL IMAGE PROCESSING  2.4 FACE DETECTION  2.5 FACE RECOGNITION  2.6 PROBLEM SCOP AND SYSTEM SPECIFICATION  2.7 FACE RECOGNITION DIFFICULTIES  2.8 FUNDAMENTAL IN DIGITAL IMAGE PROCESSING | 16  19  20  21  23  24  25  25  26 |
| 3. | OPEN CV  3.1 HISTORY  3.2 APPLICATIONS  3.3 PROGRAMMING LANGUAGE  3.4 HARDWARE ACCELERATION  3.5 OS SUPPORT  3.6 CONCLUSION  3.7 ALGORITHM TESTING | 30  31  32  32  32  33  33  34 |
| 4. | CODING | 35 |
| 5. | REFERENCES | 40 |

LIST OF ABBREVIATIONS

|  |  |
| --- | --- |
| ABBREVIATION | EXPANSION |
| SVM | SUPPORT VACTOR MACHINE |
| OPENCV | OPEN-SOURCE COMPUTER VISION |
| HOG | HISTOGRAM OF ORIENTED GRADIENTS |
| HCI | HUMAN-COMPUTER INTERATION |
| SFM | STRUCTURE FROM MOTION |
| WNN | WIRELESS ARTIFICAL NEURAL AREA NETWORK |

LIST OF FIGURES

FIGURE NO FIGURE NAME PAGE NO

* 1. BLOCK DIAGRAM 12
  2. UML DIAGRAM 13
  3. DATA FLOW DIAGRAM 13
  4. ARCHITECTURE DIAGRAM 13
  5. FACE RECOGNITION 21
  6. FACE RECOGNITION USING TEMPLATE MATCHING 21
  7. OBJECT RECOGNITION 25

**Face analysis in a video call through image processing**

**Objective:**

We are going to face recognition using feature matching technique. When the input face is matched with the image in the camera it will send the output image with time also.

**INTRODUCTION:**

Face detection is a computer technology that is being applied for many different applications that require the identification of human faces in digital images or video. It can be regarded as a specific case of object-class detection, where the task is to find the locations and sizes of all objects in an image that belongs to a given class. The technology is able to detect frontal or near-frontal faces in a photo, regardless of orientation, lighting conditions or skin colour.

Using a certain algorithm to detecting human faces within an Image (Detect Human Faces on Farm full with that’s when you use Face Detection. On the flip side, Face recognition describes a biometric technology that goes way beyond recognizing when a human face is present. It actually attempts to establish whose face it is. In this article, I’m not going deep into recognizing. I’ll keep that for a future blog article and for the time being, I’m going to explain how to run a simple Face Detection program using your WebCam with Python.

The growing interest in computer vision of the past decade. Fueled by the steady doubling rate of computing power every 13 months, face detection and recognition has transcended from an esoteric to a popular area of research in computer vision and one of the better and successful applications of image analysis and algorithm based understanding. Because of the intrinsic nature of the problem, computer vision is not only a computer science area of research, but also the object of neuro-scientific and psychological studies, mainly because of the general opinion that advances in computer image processing and understanding research will provide insights into how our brain work and vice versa. Because of general curiosity and interest in the matter.This application will be developed using Intel's open source computer vision project

Face recognition is an easy task for humans ,that even one to three day old babies are able to distinguish between known faces. Face recognition based on the geometric features of a face is probably the most intuitive approach to face recognition. One of the first automated face recognition systems was described in marker points (position of eyes, ears, nose,) were used to build a feature vector

**LITERATURE SURVEY:**

1. **DEVELOPMENT OF REAL TIME FACE RECOGNITION SYSTEM USING OPENCV**

A real-time, GUI based automatic Face detection and recognition system is developed in this project. It can be used as access control system by registering the staff or students of an organization with their faces, and later it will recognize the people by capturing their images with faces, when they are entering or leaving the premises. The system is implemented on a desktop with a Graphical User Interface, Initially it detects the faces in the images that are grabbed from a web camera. All the tools and operating, used to develop this system like Ubuntu, open Face, Python ..., are open source tools. This real timeGUIbasedface recognition system is developed using Open source tool Open face. Open Face is the face recognition tool developed by Carnegie Mellon University, using OpenCV. Open Face, consists in a broader Prospective, three phases: Detection, Feature extraction, and Recognition. The dimensionality of face image is reduced by the Histogram of Oriented Gradients (HOG) and this algorithm is developed to detect frontal views of faces. After detecting the face part of image, extract the 128 face features for the given image by using a Deep Neural Network algorithm and the recognition is done by the Support Vector machine (SVM) classifier.HOG is one of the most popular representation methods for a face image. It not only reducesthe dimensionality of the image, but also extracting the facial features of the given images, and retains some of the variations in the image data. So dimensionality of face image reduced by HOG using deep learning algorithm and recognition is done by SVM approach

**EXISTING SYSTEM:**

* In existing system it will not only improve classroom control during lectures, it will also possibly detect faces for student attendance purposes.
* In existing system use MATLAB to build and implement this system.

**DISADVANTAGES:**

* Time consumption high
* Accuracy level is low

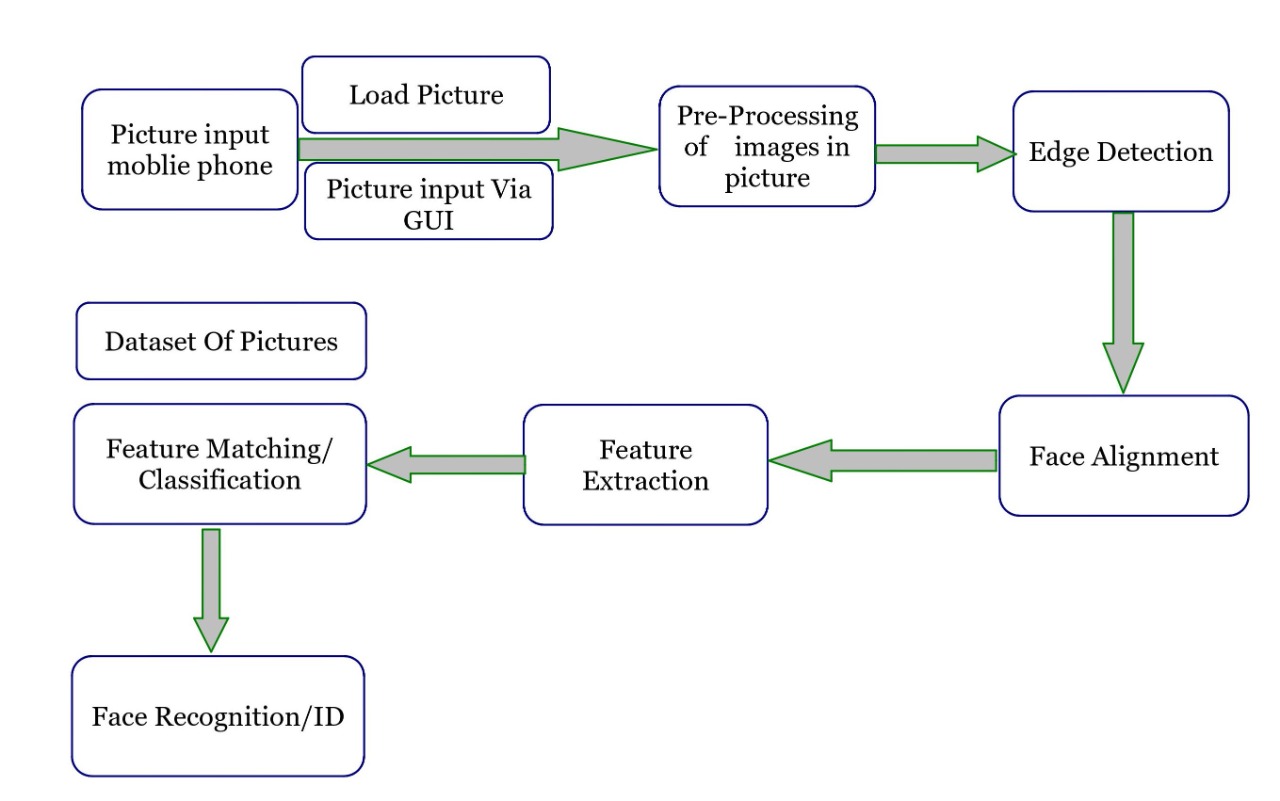
**PROPOSED SYSTEM:**

* Face recognition can be defined as the method of identifying an individual based on biometrics by way of comparing a digital captured image or video with the stored record of the person in question.
* With the most recent technology of facial recognition and detection techniques, commercial products have emerged on the markets. Despite the commercial success a few issues are still to be explored. Face Recognition in two primary tasks.
* However, Face Recognition can also be used in numerous applications such as Security, Surveillance, General Identity Verification (electoral registration, national ID cards, passports, driving licenses, student IDs), Criminal Justice systems, Image Database Investigations, Smart Card, Multi-media Environments, Video Indexing and Witness face reconstruction

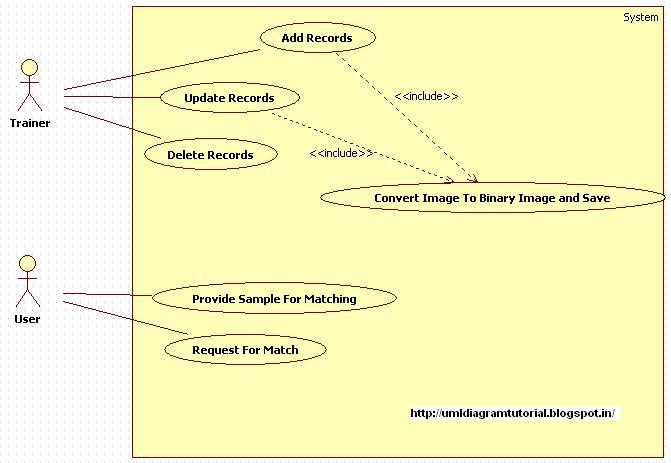
**Advantages:**

* High accuracy compared to existing system

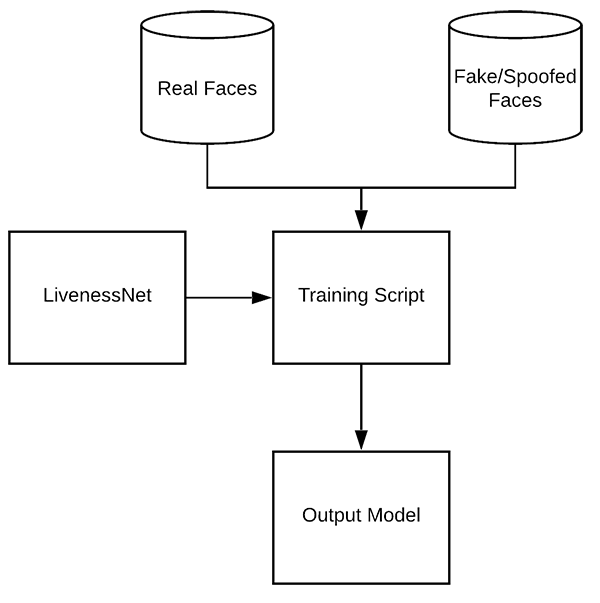
**BLOCK DIAGRAM:**



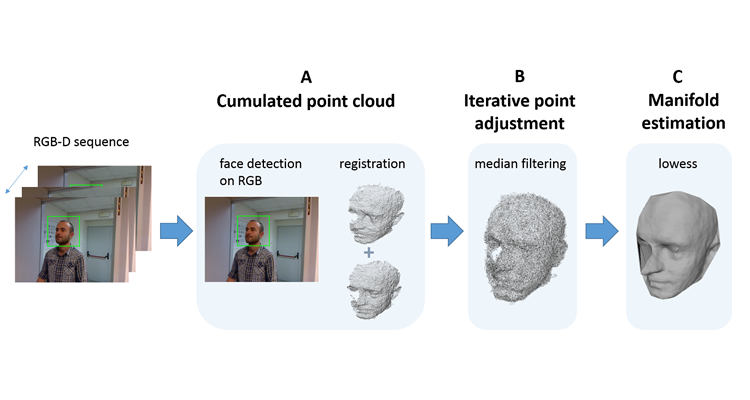
**UML DIAGRAM:**



**DATA FLOW DIAGRAM:**



**ARCHITECTURE DIAGRAM:**



**SOFTWARE MODULES:**

**DIFFERENT APPROACHES OF FACE RECOGNITION:**

There are two predominant approaches to the face recognition problem: Geometric (feature based) and photometric (view based). As researcher interest in face recognition continued, many different algorithms were developed, three of which have been well studied in face recognition literature.

**RECOGNITION ALGORITHMS CAN BE DIVIDED INTO TWO MAIN APPROACHES:**

1. **Geometric:**

Is based on geometrical relationship between facial landmarks, or in other words the spatial configuration of facial features. That means that the main geometrical features of the face such as the eyes, nose and mouth are first located and then faces are classified on the basis of various geometrical distances and angles between features.

1. **Photometric stereo:**

Used to recover the shape of an object from a number of images taken under different lighting conditions. The shape of the recovered object is defined by a gradient map, which is made up of an array of surface normal.

**FACE DETECTION:**

Face detection involves separating image windows into two classes; one containing faces (tarning the background (clutter). It is difficult because although commonalities exist between faces, they can vary considerably in terms of age, skin colour and facial expression. The problem is further complicated by differing lighting conditions, image qualities and geometries, as well as the possibility of partial occlusion and disguise. An ideal face detector would therefore be able to detect the presence of any face under any set of lighting conditions, upon any background. The face detection task can be broken down into two steps.

**THE FACE DETECTION SYSTEM CAN BE DIVIDED INTO THE FOLLOWING STEPS:-**

1. **Pre-Processing:**

To reduce the variability in the faces, the images are processed before they are fed into the network. All positive examples that is the face images are obtained by cropping images with frontal faces to include only the front view. All the cropped images are then corrected for lighting through standard algorithms.

1. **Classification:**

Neural networks are implemented to classify the images as faces or non faces by training on these examples. We use both our implementation of the neural network and the neural network toolbox for this task. Different network configurations are experimented with to optimize the results.

1. **Localization:**

The trained neural network is then used to search for faces in an image and if present localize them in a bounding box. Various Feature of Face on which the work has done on:-Position Scale Orientation Illumination

**LOW LEVEL ANALYSIS:**

there are three main steps viz.

**(1)Classify the skin region in the colour space,**

**(2) Apply threshold to mask the skin region and**

**(3) Draw bounding box to extract the face image.**

**MOTION BASE:**

When use of video sequence is available, motion information can be used to locate moving objects. Moving silhouettes like face and body parts can be extracted by simply thresholding accumulated frame differences. Besides face regions, facial features can be located by frame differences.

**GRAY SCALE BASE:**

Gray information within a face can also be treat as important features. Facial features such as eyebrows, pupils, and lips appear generally darker than their surrounding facial regions. Various recent feature extraction algorithms search for local gray minima within segmented facial regions.

**EDGE BASE:**

Face detection based on edges was introduced by Sakai et al. This work was based on analysing line drawings of the faces from photographs, aiming to locate facial features. Initially the images are enhanced by applying median filter for noise removal and histogram equalization for contrast adjustment. In the second step the edge images constructed from the enhanced image by applying sober operator

**FEATURE ANALYSIS**

These algorithms aim to find structural features that exist even when the pose, viewpoint, or lighting conditions vary, and then use these to locate faces. These methods are designed mainly for face localization.

**Neural Network**

Neural networks gaining much more attention in many pattern recognition problems, such as OCR, object recognition, and autonomous robot driving. Since face detection can be treated as a two class pattern recognition problem, various neural network algorithms have been proposed. The advantage of using neural networks for face detection is the feasibility of training a system to capture the complex class conditional density of face patterns. However, one demerit is that the network architecture has to be extensively tuned (number of layers, number of nodes, learning rates, etc.) to get exceptional performance. In early days most hierarchical neural network.The first stage having two parallel sub networks in which the inputs are filtered intensity values from an original image. The inputs to the second stage network consist of the outputs from the sub networks and extracted feature values. An output at the second stage shows the presence of a face in the inputregion.Propp and Samal developed one of the earliest neural networks for face detection . Their network consists of four layers with 1,024 input units, 256 units in the first hidden layer, eight units in the second hidden layer, and two output units

**LINEAR SUB SPACE METHOD**

**Eigen faces Method:**

An early example of employing eigen vectors in face recognition was done by Kohonen in which a simple neural network is demonstrated to perform face recognition for aligned and normalized face images. Kirby and Sirovich suggested that images of faces can be linearly encoded using a modest number of basis images. Given a collection of n by m pixel training.

Images represented as a vector of size m X n, basis vectors spanning an optimal subspace are determined such that the mean square error between the projection of the training images onto this subspace and the original images is minimized.They call the set of optimal basis vectors Eigen pictures since these are simply the eigen vectors of the covariance matrix computed from the vectorized face images in the training set.Experiments with a set of 100 images show that a face image of 91 X 50 pixels can be effectively encoded using only50 Eigen pictures.

**STATISTICAL APPROCH**

**Support Vector Machine (SVM):**

SVMs were first introduced Osuna et al. for face detection. SVMs work as a new paradigm to train polynomial function, neural networks, or radial basis function (RBF) classifiers.SVMs works on induction principle, called structural risk minimization, which targets to minimize an upper bound on the expected generalization error. developed an efficient method to train an SVM for large scale problems, and applied it to face detection. Based on two test sets of 10,000,000 test patterns of 19 X 19 pixels, their system has slightly lower error rates and runs approximately30 times faster than the system by Sung and Poggio . SVMs have also been used to detect faces and pedestrians in the wavelet domain.

**DIGITAL IMAGE PROCESSING**

**INTEREST IN DIGITAL IMAGE PROCESSING METHODS STEMS FROM TWO PRINCIPAL APPLICATION AREAS:**

1. Improvement of pictorial information for human interpretation

2. Processing of scene data for autonomous machine perception

In this second application area, interest focuses on procedures for extracting image information in a form suitable for computer processing.

**IMAGE:**

Am image refers a 2D light intensity function f(x, y), where(x, y) denotes spatial coordinates and the value of f at any point (x, y) is proportional to the brightness or gray levels of the image at that point. A digital image is an image f(x, y) that has been discretized both in spatial coordinates and brightness. The elements of such a digital array are called image elements or pixels.

**FUNDAMENTAL STEPS IN IMAGE PROCESSING**

**Fundamental steps in image processing are**

1. **Image acquisition**: to acquire a digital image

2. **Image pre-processing**: to improve the image in ways that increases the chances for success of the other processes.

3. **Image segmentation**: to partitions an input image into its constituent parts of objects.

4. **Image segmentation**: to convert the input data to a from suitable for computer processing.

5. **Image description**: to extract the features that result in some quantitative information of interest of features that are basic for differentiating one class of objects from another.

6. **Image recognition**: to assign a label to an object based on the information provided by its description.

**ELEMENTS OF DIGITAL IMAGE PROCESSING SYSTEMS**

The basic operations performed in a digital image processing system include

1. Acquisition

2. Storage

3. Processing

4. Communication

5. Display

**FACE DETECTION**

The problem of face recognition is all about face detection. This is a fact that seems quite bizarre to new researchers in this area. This is essentially a segmentation problem and in practical systems, most of the effort goes into solving this task. In fact the actual recognition based on features extracted from these facial landmarks is only a minor last step.

There are two types of face detection problems:

1) Face detection in images and

2) Real-time face detection

**FACE DETECTION IN IMAGES**

Most face detection systems attempt to extract a fraction of the whole face, thereby eliminating most of the background and other areas of an individual's head such as hair that are not necessary for the face recognition task. With static images, this is often done by running a across the image. Unfortunately, with static images there is a very large search space of possible locations of a face in an image.

**REAL-TIME FACE DETECTION**

Real-time face detection involves detection of a face from a series of frames from a video-capturing device. While the hardware requirements for such a system are far more stringent, from a computer vision stand point, real-time face detection is actually a far simpler process thandetecting a face in a static image.

**FACE DETECTION PROCESS**

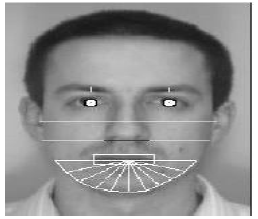
It is process of identifying different parts of human faces like eyes, nose, mouth, etc… .To do this it would be useful to study the grey-scale intensity distribution of an average human face. The following 'average human face' was constructed from a sample of 30 frontal view human faces, of which 12 were from females and 18 from males. The above facial area performs well as a basis for a face template, probably because of the clear divisions of the bright intensity invariant area by the dark intensity invariant regions

**FACE RECOGNITION**

Over the last few decades many techniques have been proposed for face recognition. Many of the techniques proposed during the early stages of computer vision cannot be considered successful, but almost all of the recent approaches to the face recognition problem have been creditable all approaches to human face recognition can be divided into two strategies:

(1) Geometrical features and (2) Template matching.

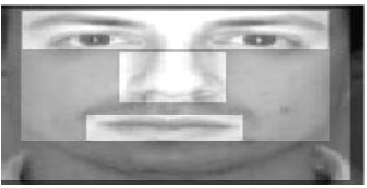
**FACE RECOGNITION USING GEOMETRICAL FEATURES**

This technique involves computation of a set of geometrical features such as

nose width and length, mouth position and chin shape, etc. from the picture of the face we want to recognize. This set of features is then matched with the features of known individuals..

**FACE RECOGNITION USING TEMPLATE MATCHIN**

This is similar the template matching technique used in face detection, except here we are not trying to classify an image as a 'face' or 'non-face' but are trying to recognize a face.



Whole face, eyes, nose and mouth regions which could be used in a template matching strategy.The basis of the template matching strategy is to extract whole facial regions (matrix of pixels) and compare these with the stored images of known individuals.

**PROBLEM SCOP AND SYSTEM SPECIFICATION**

The following problem scope for this project was arrived at after reviewing the literature on face detection and face recognition, and determining possible real-world situations where such systems would be of use. The following system(s) requirements were identified

1 A system to detect frontal view faces in static images

2 A system to recognize a given frontal view face

3 Only expressionless, frontal view faces will be presented to the face detection recognition

4 All implemented systems must display a high degree of lighting invariance.

5 All systems must possess near real-time performance.

6 Both fully automated and manual face detection must be supported

7 Frontal view face recognition will be realised using only a single known image

8 Automated face detection and recognition systems should be combined into a fully automated face detection and recognition system.

**FACE RECOGNITION DIFFICULTIES**

1. Identify similar faces (inter-class similarity)

2. Accommodate intra-class variability due to

2.1 head pose

2.2 illumination conditions

2.3 expressions

2.4 facial accessories

2.5 aging effects

3. Cartoon faces

**INTER – CLASS VARIABILITY**

Faces with intra-subject variations in pose, illumination, expression, accessories, color, occlusions, and brightness

**ADVANTAGES OF IMAGE PROCESSING:**

The purpose of image processing is divided into 5 groups.

They are:

**VISUALIZATION** - Observe the objects that are not visible.

**IMAGE SHARPENING AND RESTORATION** - To create a better image.

**IMAGE RETRIEVAL** - Seek for the image of interest.

**MEASUREMENT OF PATTERN** – Measures various objects in an image.

**IMAGE RECOGNITION** – Distinguish the objects in an image.

**FUNDAMENTAL STEPS IN DIGITAL IMAGE PROCESSING:**

**1. IMAGE ACQUISITION:**

This is the first step or process of the fundamental steps of digital image processing. Image acquisition could be as simple as being given an image that is already in digital form. Generally, the image acquisition stage involves pre-processing, such as scaling etc.

**2. IMAGE ENHANCEMENT:**

Image enhancement is among the simplest and most appealing areas of digital image processing. Basically, the idea behind enhancement techniques is to bring out detail that is obscured, or simply to highlight certain features of interest in an image. Such as, changing brightness & contrast etc.

**3. IMAGE RESTORATION:**

Image restoration is an area that also deals with improving the appearance of an image. However, unlike enhancement, which is subjective, image restoration is objective, in the sense that restoration techniques tend to be based on mathematical or probabilistic models of image degradation.

**4. COLOR IMAGE PROCESSING**

Color image processing is an area that has been gaining its importance because of the significant increase in the use of digital images over the Internet. This may include color modelling and processing in a digital domain etc.

**5. WAVELETS AND MULTIRESOLUTION PROCESSING**

Wavelets are the foundation for representing images in various degrees of resolution. Images subdivision successively into smaller regions for data compression and for pyramidal representation.

**6. COMPRESSION**

Compression deals with techniques for reducing the storage required to save an image or the bandwidth to transmit it. Particularly in the uses of internet it is very much necessary to compress data.

**7. MORPHOLOGICAL PROCESSING**

Morphological processing deals with tools for extracting image components that are useful in the representation and description of shape.

**8. SEGMENTATION**

Segmentation procedures partition an image into its constituent parts or objects. In general, autonomous segmentation is one of the most difficult tasks in digital image processing. A rugged segmentation procedure brings the process a long way toward successful solution of imaging problems that require objects to be identified individually.

**9. REPRESENTATION AND DESCRIPTION**

Representation and description almost always follow the output of a segmentation stage, which usually is raw pixel data, constituting either the boundary of a region or all the points in the region itself. Choosing a representation is only part of the solution for transforming raw data into a form suitable for subsequent computer processing.

**10. OBJECT RECOGNITION**

Both class and pose label are in consideration in the triplet loss. The loss score will be smaller when features from the same class and same pose is more similar and features from different classes or different poses will lead to a much larger loss score.



Depending on the kind of sensor used, theres more or less steps required to actually get the depth map. The Kinect camera for example uses infrared sensors combined with RGB cameras and as such you get a depth map right away (because it is the information processed by the infrared sensor).

But what if you don’t have anything else but your phone camera?. In this case you need to do stereo reconstruction. Stereo reconstruction uses the same principle your brain and eyes use to actually understand depth.

The gist of it consists in looking at the same picture from two different angles, look for the same thing in both pictures and infer depth from the difference in position. This is called stereo matching.

why stereo 3D reconstruction requires the following steps:

1. **Camera calibration**: Use a bunch of images to infer the focal length and optical centers of your camera
2. **Undistort images**: Get rid of lens distortion in the pictures used for reconstruction
3. **Feature matching**: Look for similar features between both pictures and build a depth map
4. **Reproject points**: Use depth map to reproject pixels into 3D space.
5. **Build point cloud**: Generate a new file that contains points in 3D space for visualization.
6. Build mesh to get an actual 3D model (outside of the scope of this tutorial, but coming soon in different tutorial)

Step 1 only needs to be executed once unless you change cameras. Steps 2–5 are required every time you take a new pair of pictures…and that is pretty much it.

The actual mathematical theory (the why) is much more complicated but it will be easier to tackle after this tutorial since you will have a working example that you can experiment with by the end of it.

Development of the technology in the area of the cameras, computers and algorithms for 3D the reconstruction of the objects from the images resulted in the increased popularity of the photogrammetry. Algorithms for the 3D model reconstruction are so advanced that almost anyone can make a 3D model of photographed object. The main goal of this paper is to examine the possibility of obtaining 3D data for the purposes of the close-range photogrammetry applications, based on the open source technologies. All steps of obtaining 3D point cloud are covered in this paper. Special attention is given to the camera calibration, for which two-step process of calibration is used. Both, presented algorithm and accuracy of the point cloud are tested by calculating the spatial difference between referent and produced point clouds.

**OpenCV** (*Open source computer vision*) is a [library of programming functions](https://en.wikipedia.org/wiki/Library_(computing)) mainly aimed at real-time [computer vision](https://en.wikipedia.org/wiki/Computer_vision). Originally developed by [Intel](https://en.wikipedia.org/wiki/Intel_Corporation), it was later supported by [Willow Garage](https://en.wikipedia.org/wiki/Willow_Garage) then Itseez. The library is [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) and free for use under the [open-source](https://en.wikipedia.org/wiki/Open-source_software)[BSD license](https://en.wikipedia.org/wiki/BSD_license).

Open CV supports the [deep learning](https://en.wikipedia.org/wiki/Deep_learning) frameworks [Tensor Flow](https://en.wikipedia.org/wiki/TensorFlow), [Torch](https://en.wikipedia.org/wiki/Torch_(machine_learning))/[Py Torch](https://en.wikipedia.org/wiki/PyTorch%22) and [Caffe](https://en.wikipedia.org/wiki/Caffe_(software))

**HISTORY:**

Officially launched in 1999 the Open CV project was initially an [Intel Research](https://en.wikipedia.org/wiki/Intel_Research_Lablets) initiative to advance [CPU](https://en.wikipedia.org/wiki/Central_processing_unit)-intensive applications, part of a series of projects including [real-time](https://en.wikipedia.org/wiki/Real-time_computing)[r tracing](https://en.wikipedia.org/wiki/Ray_tracing_(graphics)) and [3D display](https://en.wikipedia.org/wiki/3D_Display) walls. The main contributors to the project included a number of optimization experts in Intel Russia, as well as Intel's Performance Library Team. In the early days of Open CV, the goals of the project were described as:

* Advance vision research by providing not only open but also [optimized code](https://en.wikipedia.org/wiki/Code_optimization) for basic vision infrastructure. No more [reinventing the wheel](https://en.wikipedia.org/wiki/Reinventing_the_wheel).
* Disseminate vision knowledge by providing a common infrastructure that developers could build on, so that code would be more readily readable and transferable.
* Advance vision-based commercial applications by making [portable](https://en.wikipedia.org/wiki/Portability_(computer_science)), performance-optimized code available for free – with a license that did not require code to be open or free itself.

The first alpha version of OpenCV was released to the public at the [IEEE Conference on Computer Vision and Pattern Recognition](https://en.wikipedia.org/wiki/Conference_on_Computer_Vision_and_Pattern_Recognition) in 2000, and five betas were released between 2001 and 2005. The first 1.0 version was released in 2006. A version 1.1 "pre-release" was released in October 2008.

The second major release of the OpenCV was in October 2009. OpenCV 2 includes major changes to the [C++](https://en.wikipedia.org/wiki/C++) interface, aiming at easier, more type-safe patterns, new functions, and better implementations for existing ones in terms of performance (especially on multi-core systems). Official releases now occur every six months[]](https://en.wikipedia.org/wiki/OpenCV#cite_note-6) and development is now done by an independent Russian team supported by commercial corporations.

**APPLICATIONS:**

OpenCV's application areas include:

* [Egomotion](https://en.wikipedia.org/wiki/Egomotion) estimation
* [Facial recognition system](https://en.wikipedia.org/wiki/Facial_recognition_system)
* [Gesture recognition](https://en.wikipedia.org/wiki/Gesture_recognition)
* [Human–computer interaction](https://en.wikipedia.org/wiki/Human–computer_interaction) (HCI)
* [Mobile robotics](https://en.wikipedia.org/wiki/Mobile_robotics)
* Motion understanding
* Object identification
* [Segmentation](https://en.wikipedia.org/wiki/Segmentation_(image_processing)) and recognition
* [Stereopsis](https://en.wikipedia.org/wiki/Stereopsis) stereo vision: depth perception from 2 cameras
* [Structure from motion](https://en.wikipedia.org/wiki/Structure_from_motion) (SFM)
* [Motion tracking](https://en.wikipedia.org/wiki/Video_tracking)
* [Augmented reality](https://en.wikipedia.org/wiki/Augmented_reality)

## PROGRAMMING LANGUAGE

OpenCV is written in [C++](https://en.wikipedia.org/wiki/C++) and its primary interface is in C++, but it still retains a less comprehensive though extensive older [C interface](https://en.wikipedia.org/wiki/C_(programming_language)). There are bindings in [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) and [MATLAB](https://en.wikipedia.org/wiki/MATLAB)/[OCTAVE](https://en.wikipedia.org/wiki/GNU_Octave). The API for these interfaces can be found in the online documentation. Wrappers in other languages such as [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)), [Perl](https://en.wikipedia.org/wiki/Perl), [Ch](https://en.wikipedia.org/wiki/Ch_(computer_programming)), [Haskell](https://en.wikipedia.org/wiki/Haskell_(programming_language)), and [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)) have been developed to encourage adoption by a wider audience.

All of the new developments and algorithms in OpenCV are now developed in the C++ interface

## HARDWARE ACCELERATION

If the library finds Intel's [Integrated Performance Primitives](https://en.wikipedia.org/wiki/Integrated_Performance_Primitives) on the system, it will use these proprietary optimized routines to accelerate itself.

A [CUDA](https://en.wikipedia.org/wiki/CUDA)-based [GPU](https://en.wikipedia.org/wiki/Graphics_processing_unit) interface has been in progress since September 2010.

An [OpenCL](https://en.wikipedia.org/wiki/OpenCL)-based [GPU](https://en.wikipedia.org/wiki/Graphics_processing_unit) interface has been in progress since October 2012, documentation for version 2.4.13.3 can be found at docs.opencv.org.

## OS SUPPORT

OpenCV runs on the following desktop operating systems: [Windows](https://en.wikipedia.org/wiki/Microsoft_Windows), [Linux](https://en.wikipedia.org/wiki/Linux), [macOS](https://en.wikipedia.org/wiki/MacOS), [FreeBSD](https://en.wikipedia.org/wiki/FreeBSD), [NetBSD](https://en.wikipedia.org/wiki/NetBSD), [OpenBSD](https://en.wikipedia.org/wiki/OpenBSD). OpenCV runs on the following mobile operating systems: [Android](https://en.wikipedia.org/wiki/Android_(operating_system)), [iOS](https://en.wikipedia.org/wiki/IOS), [Maemo](https://en.wikipedia.org/wiki/Maemo), [BlackBerry 10](https://en.wikipedia.org/wiki/BlackBerry_10). The user can get official releases from [SourceForge](https://en.wikipedia.org/wiki/SourceForge) or take the latest sources from [GitHub](https://en.wikipedia.org/wiki/GitHub). OpenCV uses [CMake](https://en.wikipedia.org/wiki/CMake).

**CONCLUSION:**

The purpose of reducing the errors that occur in the traditional attendance taking system has been achieved by implementing this automated attendance system. In this paper, face recognition system have been presented using deep learning which exhibits robustness towards recognition of the users with accuracy of 98.3% . The result shows the capability of the system to cope with the change in posing and projection of faces. From face recognition with deep learning, it has been determined that during face detection, the problem of illumination is solved as the original image is turned into a HOG representation that captures the major features of the image regardless of image brightness. In the face recognition method, local facial landmarks are considered for further processing. After which faces are encoded which generates 128 measurements of the captured face and the optimal face recognition is done by finding the person’s name from the encoding. The result is then used to generate an excel sheet, the pdf of which is sent to the students and professors on weekly interval. This system is convenient to the user and it gives better security.

**Algorithm testing:**

Algorithm was tested by comparison of the 3D point cloud acquired with an open source solution, OpenCV and point cloud acquired with the proven software AgisoftPhotoScan. RO in OpenCV was calculated based on the 152 image points in each pair of the images used for the orientation i.e. stereo calibration. Totally 10 pairs of the images were used for the orientation. On the other hand, Agisoft’s algorithm detected 2247 tie points visible on the both images, creating that way a strong geometry. Ground sample distance was 0.22mm. Precalibrated camera was used for the point cloud generation and the further optimization of camera is not done. Agisoft point cloud was created with the maximal offered quality, from alignment to point cloud generation, and with minimal further processing i.e. depth filtering and interpolation, to fit OpenCV point cloud that is minimally processed.

Accuracy of the point cloud Accuracy of the point cloud was tested by the comparison of produced point cloud to the model acquired with better camera and in larger scale, which is considered as an reference in this paper. Reference i.e. etalon model is produced based on the 73 images of the resolution 4272x2848 with the ground sample distance of 0.07mm. In order to examine accuracy of the stereo point cloud, point cloud created with the OpenCV and point cloud created in the AgisoftPhotoScan with the same images were compared with the reference model.

**CODING:**

**import face\_recognition**

**import cv2**

**import numpy as np**

**video\_capture = cv2.VideoCapture(0)**

**# Load a second sample picture and learn how to recognize it.**

**obama\_image = face\_recognition.load\_image\_file("obama.jpg")**

**obama\_face\_encoding = face\_recognition.face\_encodings(obama\_image)[0]**

**meghana\_image = face\_recognition.load\_image\_file("meghana.jpeg")**

**meghana\_face\_encoding = face\_recognition.face\_encodings(meghana\_image)[0]**

**priya\_image = face\_recognition.load\_image\_file("priya.jpeg")**

**priya\_face\_encoding = face\_recognition.face\_encodings(priya\_image)[0]**

**tarun\_image = face\_recognition.load\_image\_file("tarun.jpeg")**

**tarun\_face\_encoding = face\_recognition.face\_encodings(tarun\_image)[0]**

**# Create arrays of known face encodings and their names**

**known\_face\_encodings = [**

**meghana\_face\_encoding,**

**obama\_face\_encoding,**

**priya\_face\_encoding,**

**tarun\_face\_encoding**

**]**

**known\_face\_names = [**

**"meghana",**

**"obama",**

**"priya",**

**"tarun"**

**]**

**while True:**

**# Grab a single frame of video**

**ret, frame = video\_capture.read()**

**# Convert the image from BGR color (which OpenCV uses) to RGB color (which face\_recognition uses)**

**rgb\_frame = frame[:, :, ::-1]**

**# Find all the faces and face enqcodings in the frame of video**

**face\_locations = face\_recognition.face\_locations(rgb\_frame)**

**face\_encodings = face\_recognition.face\_encodings(rgb\_frame, face\_locations)**

**# Loop through each face in this frame of video**

**for (top, right, bottom, left), face\_encoding in zip(face\_locations, face\_encodings):**

**# See if the face is a match for the known face(s)**

**matches = face\_recognition.compare\_faces(known\_face\_encodings, face\_encoding)**

**name = "Unknown"**

**face\_distances = face\_recognition.face\_distance(known\_face\_encodings, face\_encoding)**

**best\_match\_index = np.argmin(face\_distances)**

**if matches[best\_match\_index]:**

**name = known\_face\_names[best\_match\_index]**

**# Draw a box around the face**

**cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)**

**# Draw a label with a name below the face**

**cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0, 255), cv2.FILLED)**

**font = cv2.FONT\_HERSHEY\_DUPLEX**

**cv2.putText(frame, name, (left + 6, bottom - 6), font, 1.0, (255, 255, 255), 1)**

**# Display the resulting image**

**cv2.imshow('Video', frame)**

**# Hit 'q' on the keyboard to quit!**

**if cv2.waitKey(1) & 0xFF == ord('q'):**

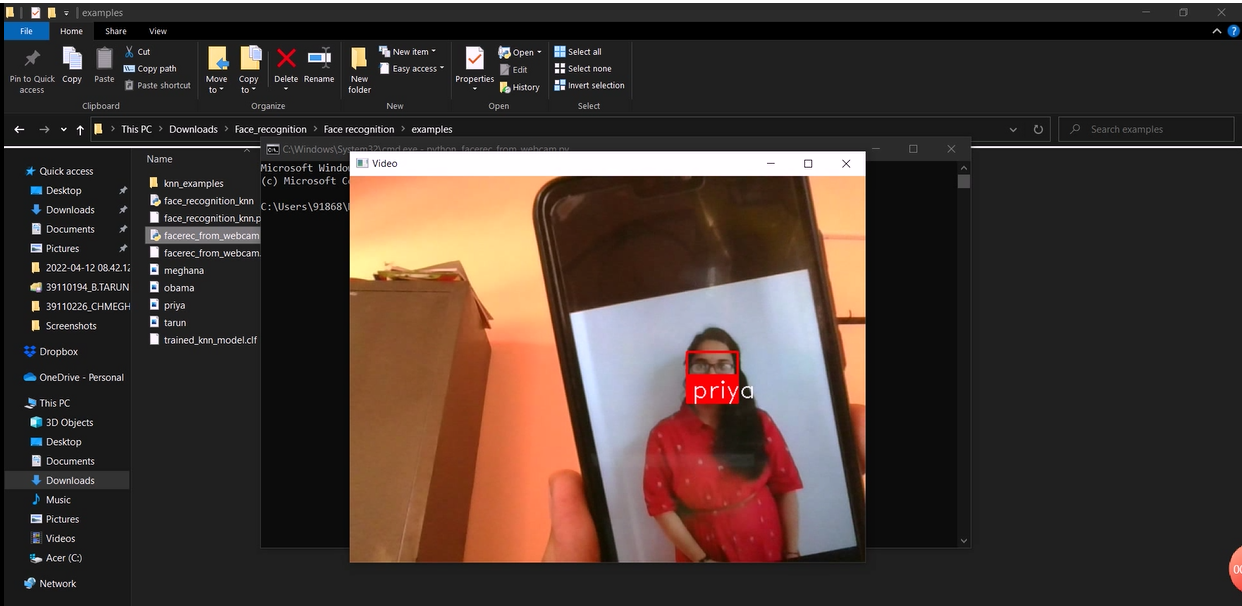
**break**

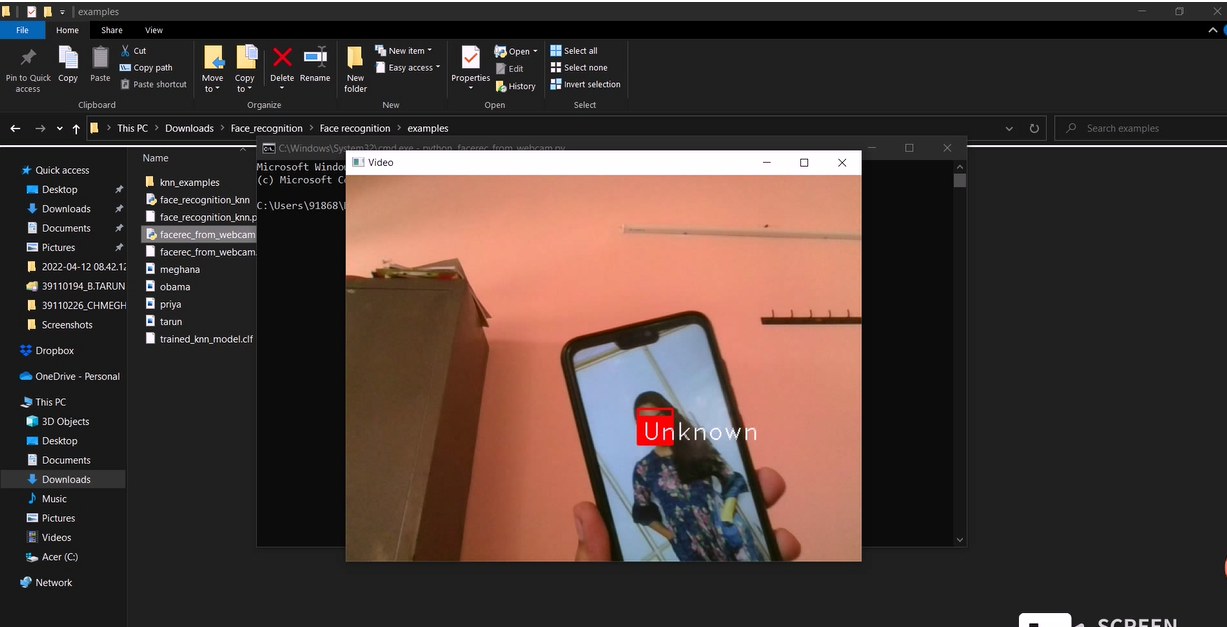
**# Release handle to the webcam**

**video\_capture.release()**

**cv2.destroyAllWindows()**

**OUTPUT SCREENSHOT:**





**REFERENCES:**

[1] V. Blanz and T. Vetter. Face recognition based on fitting a 3dmorphable model.IEEE Transactions on Pattern AnalysisandMachine Intelligence, 25(9):1063–1074, 2003.2

[2] P. Breuer, K.-I. Kim, W. Kienzle, B. Scholkopf, and V. Blanz.Automatic 3d face reconstruction from single images orvideo. InProc. of the 8th IEEE International Conferenceon Automatic Face Gesture Recognition (FG ’08), pages 1–8, 2008.2

[3] Chaoyang, L. Wang, Y. Wang, F. Matsushita, K., and Soong.Binocular photometric stereo acquisition and reconstructionfor 3d talking head applications. InInterspeech 2013 sub-mission, 2013.3

[4] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using depth cameras for dense 2d modelingofindoorenvironments.Proc. of International Symposium onExperimental Robotics, 2010.1

[5] M. Hernandez, J. Choi, and G. Medioni. Laser scan qual-ity 3-d face modeling using a low-cost depth camera.Proc.of the 20th European Signal Processing Conference (EU-SIPCO), pages 1995–1999, 2012.1,2,3,4,5,7