ATTENDANCE MONITORING SYSYTEM USING FACE RECOGNITION

Pushkar Kumar

Student, Dept of CSE

Galatia's University Greater

Noida UP

ABSTRACT

Face recognition has been a very active research area in the past two decades. Many attempts have been made to understand the process of how human beings recognize human faces. It is widely accepted that face recognition may depend on both componential information (such as eyes, mouth and nose) and non-componential/holistic information (the spatial relations between these features), though how these cues should be optimally integrated remains unclear. This gives an ideal way of detecting and recognizing human face using Open-CV, and python which is part of deep learning.

Key Words: Python, Open-CV, Deep Learning, Face detection

INTRODUCTION

Face recognition is the technique in which the identity of a human being can be identified using one's individual face. Such kind of systems can be used in photos, videos, or in real time machines. The objective of this is to provide a simpler and easy method in machine technology. With the help of such a technology one can easily detect the face by the help of dataset in similar matching appearance of a person. The method in which with the help of python and Open-CV in deep learning is the most efficient way to detect the face of the person. This method is useful in many fields such as the military, for security, schools, colleges and universities, airlines, banking, online web applications, gaming etc. this system uses powerful python algorithm through which the detection and recognition of face is very easy and efficient. A facial recognition system is a technology capable of matching a human face from a digital image or a video frame against a database of faces, typically employed to authenticate users through ID verification services, works by pinpointing and measuring facial features from a given image While initially a form of computer application, facial recognition systems have seen wider uses in recent times on smartphones and in other forms of technology, such as robotics. Because computerized facial recognition involves the measurement of a human's physiological characteristics facial recognition systems are categorised as biometrics. Although the accuracy of facial recognition systems as a biometric technology is lower than iris recognition and fingerprint recognition, it is widely adopted due to its contactless process. Facial recognition systems have been deployed in advanced human-computer interaction, video surveillance and automatic indexing of images. They are also used widely by law enforcement agencies.

LITERATURE REVIEW

This section is a basic overview of the major techniques used in the face recognition system that apply mostly to the front face of the human being. The methods include neural networks, hidden Markov model, face matching done geometrically and template matching. Eigen-face is one of the most widely used methods in face recognition and detection which are broadly called as the principle components in mathematical terms. The eigenvectors are ordered to represent different amounts of the variations in the faces. Neural networks are highly used in the face recognition and detection systems. An ANN (artificial neural network) Was used in face recognition which contained a single layer Which shows adaptiveness in crucial face recognition systems. The face verification is done using a double layer of WISARD in neural networks. 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. Hidden Markov Models is the way by which stochastic modeling of non- stationary vector time series based on HMM model applied to the human face recognition wherein the faces get divided into parts such as the eyes, nose, ears, etc. The face recognition and correct matching is 87% correct as it always gives out the best and right choice of face detection through stored dataset. Or else the relevant model reveals the identity of the face. The geometrical feature matching is the technique which is based on the geometrical shapes of the face. This is one of the commonly used method of the face recognition and detection. This system apparently gives satisfactory results. Template matching is one of the techniques through which the test image is represented as a two- dimensional array of values which can be compared using

Euclidean distance with single template representing the whole face. This method can also use more than one face template from different points of view to represent an individual face.

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 5 | P a g e 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 eyes, nose and mouth are first located and then faces are classified on the basis of various geometrical distances and angles between features.

2. 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 normals . Popular recognition algorithms include:

1. Principal Component Analysis using Eigenfaces, (PCA)

2. Linear Discriminate Analysis,

3. Elastic Bunch Graph Matching using the Fisherface algorithm

FACE DETECTION:

Face detection involves separating image windows into two classes; one containing faces (tarningthe 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 first step is a classification task that takes some arbitrary image as input and outputs a binary value of yes or no, indicating whether there are any faces present in the image. The second step is the face localization task that aims to take an image as input and output the location of any face or faces within that image as some bounding box with (x, y, width, height). 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.

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

3. 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.

Methodology

The concept of Open CV was put forth by Gary Bradski which had the ability to perform on multi-level framework. Open CV has a number of significant abilities as well as utilities which appears from the outset. The OpenCV helps in recognizing the frontal face of the person and also creates XML documents for several areas such as the parts of the body. Deep learning evolved lately in the process of the recognition systems. Hence deep learning along with the face recognition together work as the deep metric learning systems. In short deep learning in face detection and recognition will broadly work on two areas the first one being accepting the solidary input image or any other relevant picture and the second being giving the best outputs or the results of the image of the picture. We would be using dlib facial recognition framework that would be the easy way to organize the face evaluation. The two main significant libraries used in the system are dlib and face_recognition. Python being a very powerful programming languages and one of the programming languages that are being used all over the world has proven to give best results in the face recognition and detection systems. Together face recognition and detection becomes very easy and fruitful with the help of the python programming language and Open CV.

Need of an automated system

Due to the rising need for the systems which can help in the areas such as surveillance as well as security this kind of individual authentication can no longer be done using simple handmade methods hence there is a rising need of the automated systems that can easily rectify the faults and process the human face recognition. When the work is done by machines it can perform tasks efficiently in very less duration of time and cuts off the major mistakes occurred by humans. A real time GUI based face recognition system built can ease this work of face detection and can be achieved in various ways.

Graphical User Interface

The graphical user interface (GUI) is the platform that will allow the inputs from the user ends a kind of interaction with the system. GUI's are used in mobiles, media players, games and many others. We can design visual composition and the temporal behaviour of the GUI in any of the software application as well as programming in the areas of the human computer interaction. The GUI for this project will be widely based on the training and the testing phase which in turn will allow the capture and train of the image. The minimum requirements for the software would be python along with Open CV and the required dataset. The minimum requirements for the hardware would be intel i3 or any processor above it and 4 core CPU. Operating systems of windows 10 will be sufficient and random access memory 8GB required. From the user end a computer or laptop active internet connection and a scanner optional.

System Design

Face recognition is really a series of several related problems. Here are the steps which we have to follow for making our model. 1. First, look at a picture and find all the faces in it

2. Second, focus on each face and be able to understand that even if a face is turned in a weird direction or in bad lighting, it is still the same person.

3. Third, be able to pick out unique features of the face that you can use to tell it apart from other people— like how big the eyes are, how long the face is, etc.

4. Finally, compare the unique features of that face to all the people you already know to determine the person’s name.

In order to create this system first we will have to make the datasets. When the image quality becomes favourable different procedures will take place in the face recognition system the tasks are performed using the python queries “python encode_faces.py”. The input will be taken from the dataset which will be received in the “encodings.py”. There will be precision formatting in the system wherein face embedding for each face will occur. Secondly a file “recognize_faces_images.py” will contain all the required methods and the techniques for the process of identification of the face of the person from the given image of the dataset. The given file will be executed by the python command “python recognize_faces_image.py-encodings”. We can resize or turn the image for a proximity with the goal for getting the desired output. The present classifier along with Open CV libraries will enhance the outcome or results in the face recognition system.

Face Recognition — Step by Step

So for the above mention steps, Let’s tackle this problem one step at a time. We are going to describe all the steps in details for better understanding of the project and how face recognition works in python.

Step 1: Finding all the Faces

The first step in our pipeline is face detection. Obviously we need to locate the faces in a photograph before we can try to tell them apart!

Face detection went mainstream in the early 2000's when Paul Viola and Michael Jones invented a way to detect face that was fast enough to run on cheap cameras. However, much more reliable solutions exist now. We're going to use a method invented in 2005 called Histogram of Oriented Gradients — or just HOG for short. To find faces in an image, we'll start by making our image black and white because we don't need color data to find faces: This might seem like a random thing to do, but there's a really good reason for replacing the pixels with gradients. If we analyze pixels directly, really dark images and really light images of the same person will have totally different pixel values. But by only considering the direction that brightness changes, both really dark images and really bright images will end up with the same exact representation. That makes the problem a lot easier to solve! But

saving the gradient for every single pixel gives us way too much detail. We end up missing the forest for the trees. It would be better if we could just see the basic flow of lightness/darkness at a higher level so we could see the basic pattern of the image. To do this, we'll break up the image into small squares of 16x16 pixels each. In each square, we'll count up how many gradients point in each major direction (how many point up, point up-right, point right, etc…). Then we'll replace that square in the image with the arrow directions that were the strongest. The end result is we turn the original image into a very simple representation that captures the basic structure of a face in a simple way: To find faces in this HOG image, all we have to do is find the part of our image that looks the most similar to a known HOG pattern that was extracted from a bunch of other training faces. With the help of this technique, we can now easily find faces in any image.

Step 2: Posing and Projecting Faces

Now, we isolated the faces in our image. But now we have to deal with the problem that faces turned different directions look totally different to a computer: To account for this, we will try to wrap each picture so that the eyes and lips are always in the sample place in the image. This will make it a lot easier for us to compare faces in the next steps. To do this, we are going to use an algorithm called face landmark estimation. There are lots of ways to do this, but we are going to use the approach invented in 2014 by Vahid Kazemi and Josephine Sullivan. 14 | P a g e The basic idea is we will come up with 68 specific points (called landmarks) that exist on every face — the top of the chin, the outside edge of each eye, the inner edge of each eyebrow, etc. Then we will train a machine learning algorithm to be able to find these 68 specific points on any face: Now that we know were the eyes and mouth are, we'll simply rotate, scale and shear the image so that the eyes and mouth are centered as best as possible. We won't do any fancy 3d warps because that would introduce distortions into the image. We are only going to use basic image transformations like rotation and scale that preserve parallel lines (called affine transformation): Now no matter how the face is turned, we are able to center the eyes and mouth are in roughly the same position in the image. This will make our next step a lot more accurate.

Step 3: Encoding Faces

Researchers have discovered that the most accurate approach is to let the computer figure out the measurements to collect itself. Deep learning does a better job than humans at figuring out which parts of a face are important to measure. The solution is to train a Deep Convolutional Neural Network. But instead of training the network to recognize pictures objects, we are going to train it to generate 128 measurements for each face. The training process works by looking at 3 face images at a time:

1. Load a training face image of a known person

2. Load another picture of the same known person

3. Load a picture of a totally different person

Then the algorithm looks at the measurements it is currently generating for each of those three images. It then tweaks the neural network slightly so that it makes sure the measurements it generates for #1 and #2 are slightly closer while making sure the measurements for #2 and #3 are slightly further apart.

After repeating this step millions of times for millions of images of thousands of different people, the neural network learns to reliably generate 128 measurements for each person. Any ten different pictures of the same person should give roughly the same measurements.

Machine learning people call the 128 measurements of each face an embedding. The idea of reducing complicated raw data like a picture into a list of computergenerated numbers comes up a lot in machine learning (especially in language translation).

Encoding our face image

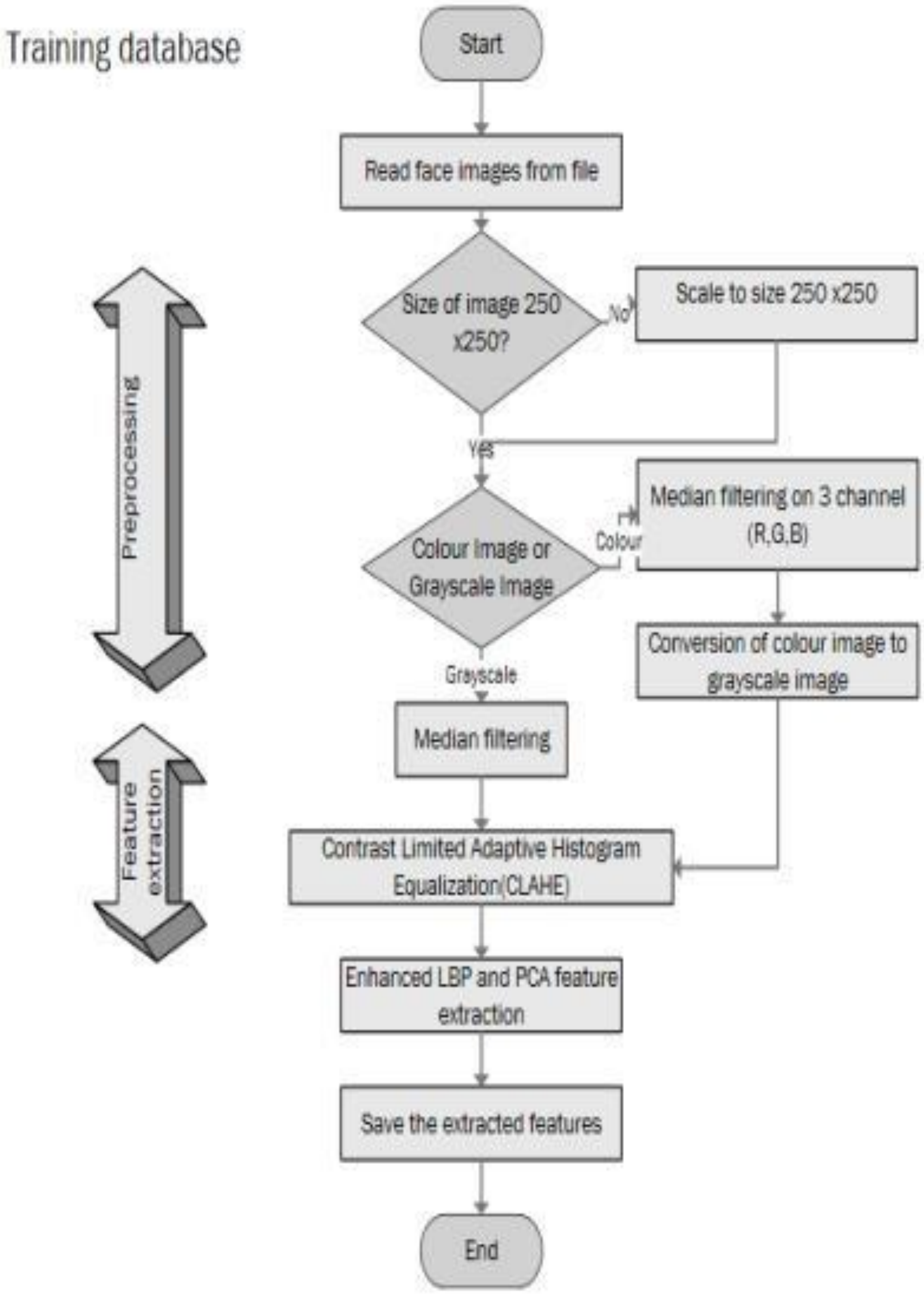
This process of training a convolutional neural network to output face embeddings requires a lot of data and computer power. But once the network has been trained, it can generate measurements for any face, even ones it has never seen before! So this step only needs to be done once. So all we need to do ourselves is run our face images through their pre-trained network to get the 128 measurements for each face. Here’s the measurements for our test image:

So what parts of the face are these 128 numbers measuring exactly? It turns out that we have no idea. It doesn’t really matter to us. All that we care is that the network generates nearly the same numbers when looking at two different pictures of the same person.

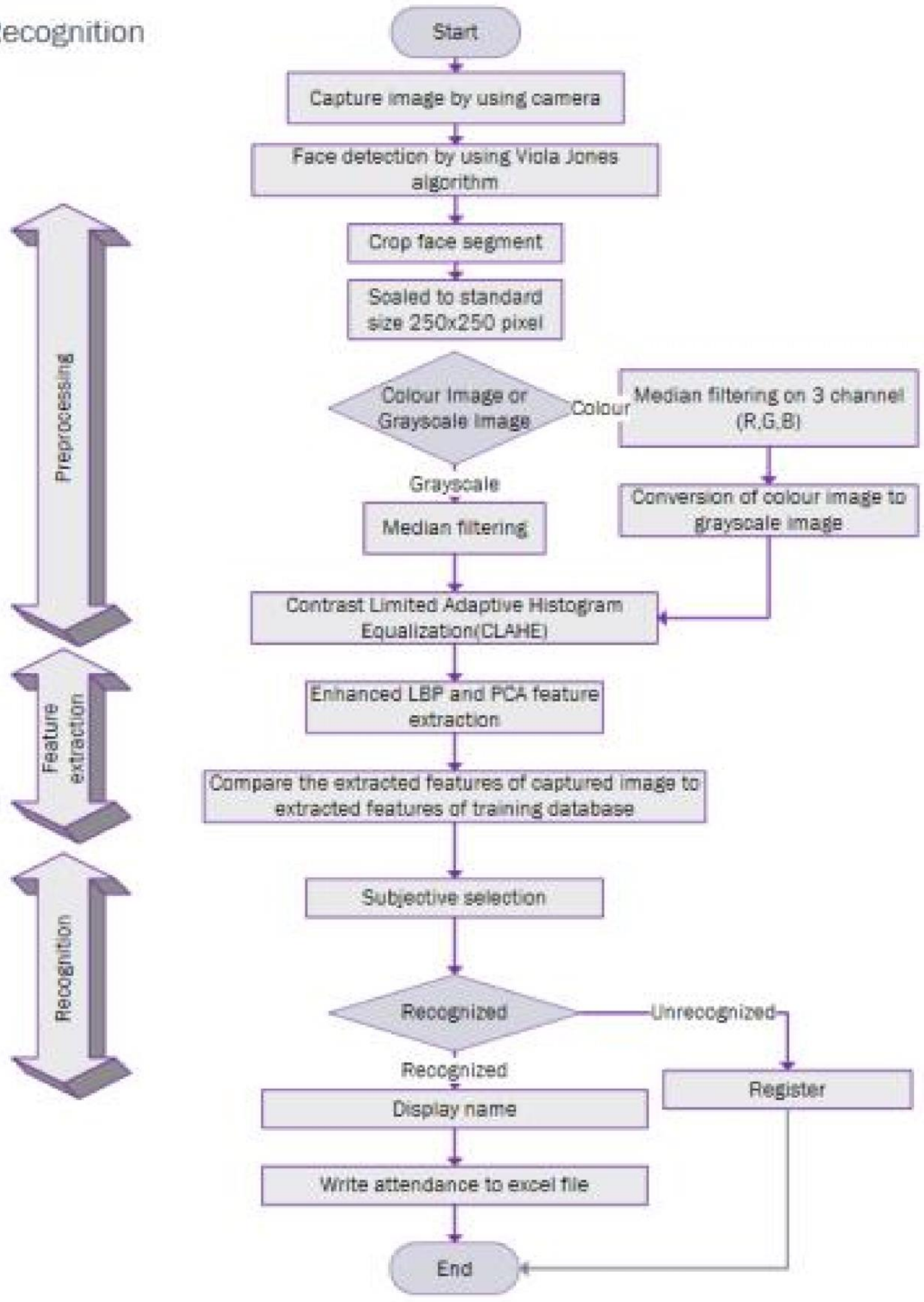
Step 4: Finding the person’s name from the encoding

This last step is actually the easiest step in the whole process. All we have to do is find the person in our database of known people who has the closest measurements to our test image. You can do that by using any basic machine learning classification algorithm. No fancy deep learning tricks are needed. We’ll use a simple linear SVM classifier, but lots of classification algorithms could work. All we need to do is train a classifier that can take in the measurements from a new test image and tells which known person is the closest match. Running this classifier takes milliseconds. The result of the classifier is the name of the person!

FLOWCHART


Training database
Start
Read face images from file
Size of image 250 x250?
No
Scale to size 250 x250
Yes
Colour Image or Grayscale Image
Colour
Median filtering on 3 channel (R,G,B)
Conversion of colour image to grayscale image
Grayscale
Median filtering
Contrast Limited Adaptive Histogram Equalization(CLAHE)
Enhanced LBP and PCA feature extraction
Save the extracted features
End
Preprocessing
Feature extraction

Recognition



Start
Capture image by using camera
Face detection by using Viola Jones algorithm
Crop face segment
Scaled to standard size 250x250 pixel
Colour image or Grayscale image
Colour
Median filtering on 3 channel (R,G,B)
Conversion of colour image to grayscale image
Grayscale
Median filtering
Contrast Limited Adaptive Histogram Equalization(CLAHE)
Enhanced LBP and PCA feature extraction
Compare the extracted features of captured image to extracted features of training database
Subjective selection
Recognized
Unrecognized
Register
Recognized
Display name
Write attendance to excel file
End
Preprocessing
Feature extraction
Recognition

CONCLUSION :

Face recognition systems are currently associated with many top technological companies and industries making the work of face recognition easier. The use of python programming and OpenCV makes it an easier and handy tool or system which can be made by anyone according to their requirement. The proposed system discussed in this project will be helpful for many as it is user friendly and cost-efficient system. Hence by the use of python and Open CV the face recognition system can be designed for various purposes. Face recognition systems are part of facial image processing applications and their significance as a research area are increasing recently. Implementations of system are crime prevention, video surveillance, person verification, and similar security activities. The face recognition system implementation will be part of humanoid robot project at Atılım University. The goal is reached by face detection and recognition methods. Knowledge-Based face detection methods are used to find, locate and extract faces in acquired images. Implemented methods are skin color and facial features. Neural network is used for face recognition. RGB color space is used to specify skin colorvalues, and segmentation decreases searching time of face images. Facial components on face candidates are appeared with implementation of LoG filter. LoG filter shows good performance on extracting facial compoments under different illumination conditions.

Reference:

:Adelson, E. H., and Bergen, J. R. (1986) The Extraction of Spatio-Temporal Energy in Human and Machine Vision, Proceedings of Workshop on Motion: Representation and Analysis (pp. 151-155) Charleston, SC; May 7-9 AAFPRS(1997). A newsletter from the American Academy of Facial Plastic and Reconstructive Surgery. Third Quarter 1997, Vol. 11, No. 3. Page 3. Baron, R. J. (1981). Mechanisms of human facial recognition. International Journal of Man Machine Studies, 15:137-178 Beymer, D. and Poggio, T. (1995) Face Recognition from One Example View, A.I. Memo No. 1536, C.B.C.L. Paper No. 121. MIT Bichsel, M. (1991). Strategies of Robust Objects Recognition for Automatic Identification of Human Faces. PhD thesis, Eidgenossischen Technischen Hochschule, Zurich. Brennan, S. E. (1982) The caricature generator. M.S. Thesis. MIT. Brunelli, R. and Poggio, T. (1993), Face Recognition: Features versus Templates. IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(10):1042- 1052 Craw, I., Ellis, H., and Lishman, J.R. (1987). Automatic extraction of face features. Pattern Recognition Letters, 5:183-187, February. Deffenbacher K.A., Johanson J., and O'Toole A.J. (1998) Facial ageing, attractiveness, and distinctiveness. Perception. 27(10):1233-1243 Dunteman, G.H. (1989) Principal Component Analysis. Sage Publications. Frank, H. and Althoen,

S. (1994). Statistics: Concepts and applications. Cambridge University Press. p.110 Gauthier, I., Behrmann, M. and Tarr, M. (1999). Can face recognition really be dissociated from object recognition? Journal of Cognitive Neuroscience, in press.