REAL TIME FACE RECOGNITION

This synopsis is submitted for the partial fulfilment of Diploma in Computer Science & Technology

Under guidance of

[SHIBDAS BHATTACHARYA]

Lecturer, Department of Computer Science & Technology

TECHNIQUE POLYTECHNIC INSTITUTE

Submitted by

Rajdeep Das	D141512169
Tanima Sau	D141512181
Rajasree Maity	D141512180
Tanusree Bhar	D141512171

ACKNOWLEDGEMENT

We are highly grateful to **Mr. P.K.Mitra, In-charge of DCST, Technique Polytechnic Institute, Panchrokhi, Sugandhya, Hooghly,** for providing this opportunity to carry out the major project of **REAL TIME FACE RECOGNITION.**

We would like to express my gratitude to other faculty members of Computer Science & Technology department of Technique Polytechnic Institute, Panchrokhi, and Sugandhya for providing high academic inputs, guidance and encouragement throughout this period.

Authors would like to express a deep sense of gratitude and thank to everyone.

The help rendered by **Mr. Shibdas Bhattacharya** as a project guide for experiment is greatly acknowledged.

……...

……...

……...

……...

……...

Technique Polytechnic Institute
November, 2016

Department of Computer Science and Technology Technique Polytechnic Institute

Certificate of Approval

The foregoing project entitled **"REAL TIME FACE RECOGNITION"** is approved as a research oriented case studies**.** The project is quite interesting as now days Image Processing has a huge demand in various fields. We had gone through the project requirements and specification, project progress plan, not only that but also we discussed a lot on the scope of the project. After all the discussion we satisfied on the project and we allow this group to perform this research oriented case studies as their final year project.

…….
Signature of the H.O.D / In-Charge,
Computer Science & Technology

…….
Signature of the Project Guide,
Computer Science & Technology

Project Progress Plan

This Project Progress Plan is submitted for the approval of the Final Year Project for Academic Session 2016-17

Project Title: REAL TIME FACE RECOGNITION

Institute Name : Technique Polytechnic Institute

Address: Panchrokhi, Sugandhya, Hooghly, PIN-712102

Department: Computer Science & Technology

Project Guide : Mr. Shibdas Bhattacharya, Lecturer, CST

Group Members:

Rajdeep Das (DCST/5th Sem/04)
Tanima Sau (DCST/5th Sem/06)
Rajasree Maity(DCST/5th Sem/15)
Tanusree Bhar (DCST/5th Sem/16)

Academic Session : 2016-17

GANTT Chart of the Final Year Project for 5th Semester

Particulars of the Task	Weeks										
	1	2	3	4	5	6	7	8	9	10	11
Topic selection & Preparation of Project Progress Plan											
Congregation of prerequisite knowledge & Literature Survey											
Implementation of basic Image Processing Technique of Image Manipulation using MTLAB											
Implementation of LBPH Algorithm in Matlab and C++											
Preparation of Project Synopsis and to be submitted at the end of 5th Semester											

GANTT Chart of the Final Year Project for 6th Semester

Particulars of the Task	Weeks								
	1	2	3	4	5	6	7	8	9
Implementation LBPH and Generating Training Model in Details									
Testing & debugging of the Source Code									
Analysis of Output Result with the LBPH Algorithm									
Preparation of Final Documentation and to be submitted at the end of 6th Semester									

Submitted By:

.....

……...

……...

……...

The Project Progress Plan is prepared by the students under my look after. The distribution of the project work throughout the academic session is made considering all the other academic activities as per the Academic Calendar of WBSCTVESD. The effort made by the students for preparing this work plan is commendable.

……..
Signature of the Project Guide
Mr. Shibdas Bhattacharya
Department of Computer Science & Technology,
Technique Polytechnic Institute

TABLE OF CONTENTS

LIST OF FIGURES

1.1 ABSTRACT

In this project, we present face detection and recognition algorithm in real time camera input environment. The entire face tracking algorithm is divided into two modules. The first module is face detection and second is face tracking. To detect the face in the image, Haar based algorithm is used. We try to recognize each of the detected faces as our chosen person.
If successful, draw a green rectangle around the face and display the person name. Otherwise, display Unknown.
We are tried to control electrical hardware appliance by face recognition and detection.

Key Words: Face detection, face tracking, and face recognition.

Chapter -2

2.1 INTRODUCTION

In today's world, the need to maintain the security of information or physical property is becoming both increasingly important and difficult. From time to time we hear about the crimes of credit card fraud, computer break-in's by hackers, or security breaches in a company or government building. In the year 1998, sophisticated cyber crooks caused well over US $100 million in losses (Reuters, 1999). In most of these crimes, the criminals were taking advantage of a fundamental flaw in the conventional access control systems: the systems do not grant access by "who we are", but "what we have', such as ID cards, keys, passwords, PIN numbers, or mother's maiden name. None of these means are really defining us. Rather, they merely are means to authenticate us. It goes without saying that if someone steals, duplicates, or acquires these identity means, he or she will be able to access our data or our personal property any time they want. Recently, technology became available to allow verification of "true" individual identity. This technology is based in a field called "biometrics". Biometrics access control are automated methods of verifying or recognizing the identity of a living person on the basis of some physiological characteristics, such as fingerprints or facial features, or some aspects of the person's behaviour, like his/her handwriting style or keystroke patterns. Since biometric systems identify a person by biological characteristics, they are difficult to forge. Among the various biometric ID methods, the physiological methods (fingerprints, face, DNA) are more stable than methods in behavioural category (keystroke, voice prints). The reason is that physiological features are often non-alterable except by severe injury. The behavioural patterns, on the other hand, may fluctuate due to stress, fatigue, or illness. However, behavioural IDs have the advantage of being no intrusiveness. People are more comfortable singing their names or speaking to a microphone than placing their eyes before a scanner or giving a drop of blood for DNA sequencing. Face recognition is one of the few biometric methods that possess the merits of both high accuracy and low intrusiveness. It has the accuracy of a physiological approach without being intrusive. For this reason, since the early 70's (Kelly, 1970), face recognition has drawn the attention of researchers in fields from security, psychology, and image processing, to computer vision. Numerous algorithms have been proposed for face recognition.

2.1.1 Face Recognition Techniques

Face recognition is the process of identifying one or more people in images or videos by analysing and comparing patterns. Algorithms for face recognition typically extract facial features and compare them to a database to find the best match. Face recognition is an important part of many biometric, security, and surveillance systems, as well as image and video indexing systems.

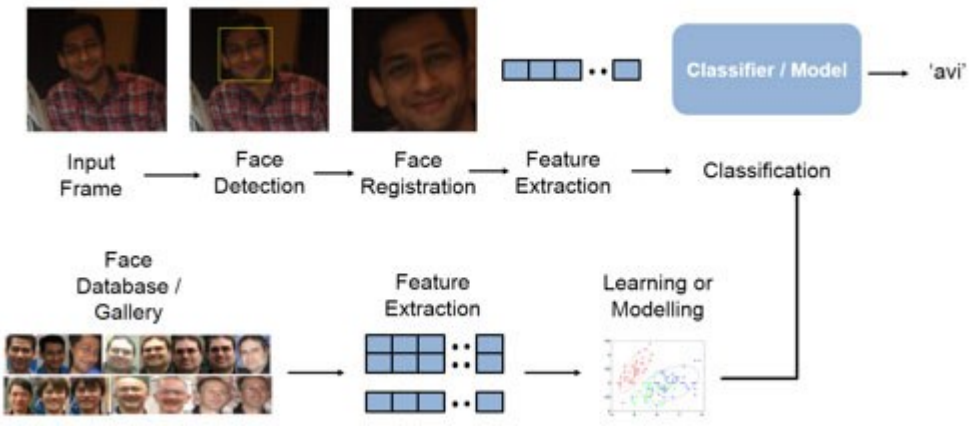


Fig 1.1 *Steps in the face recognition workflow.*

Face recognition leverages computer vision to extract discriminative information from facial images, and pattern recognition or machine learning techniques to model the appearance of faces and to classify them.

We can use computer vision techniques to perform feature extraction to encode the discriminative information required for face recognition as a compact feature vector using techniques and algorithms such as:

- Dense local feature extraction with SURF, BRISK or FREAK descriptors
- Histogram of oriented gradients
- Distance between detected facial landmarks such as eyes, noses, and lips

Machine learning techniques can applied to the extracted features to perform face recognition or classification using:

- Supervised learning techniques such as support vector machines (SVM) and decision trees
- Ensemble learning methods
- Deep neural networks

2.1.2 Face Detection

As can be assumed, detecting a face is simpler than recognizing a face of a specific person. In order to be able to determine that a certain picture contains a face (or several) we need to be able to define the general structure of a face. Luckily human faces do not greatly differ from each other; we all have noses, eyes, foreheads, chins and mouths; and all of these compose the general structure of a face.

Consider the following 5 figures:

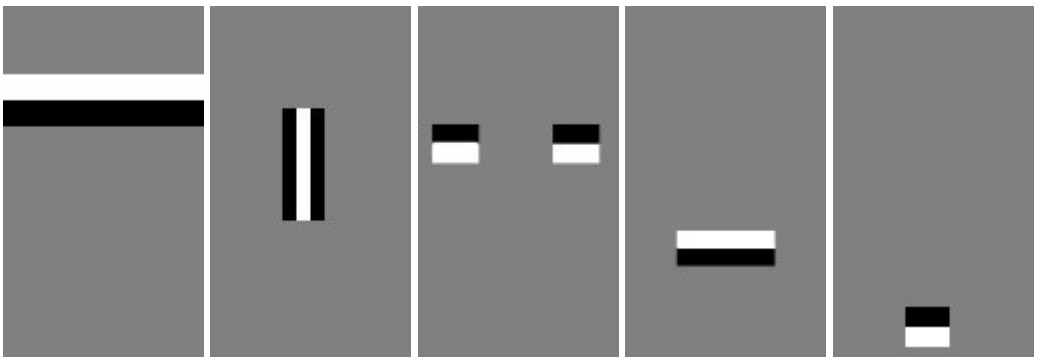


Fig-1.2

Each of these figures represents a general feature of a human face. Combining all the features together we, indeed, receive something that resembles a face.

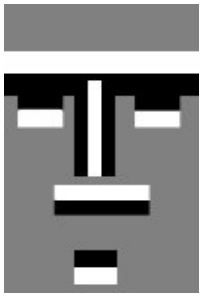


Fig -1.3

By determining if each of these features is similar to some part of our picture, we can conclude if the picture contains a face or not. Notice that this does not have to be an accurate match; we just need to know if, roughly, each of these features corresponds to some part of the image. The technique used for this purpose is Template Matching.

Fig 2.2 Face Detection Example

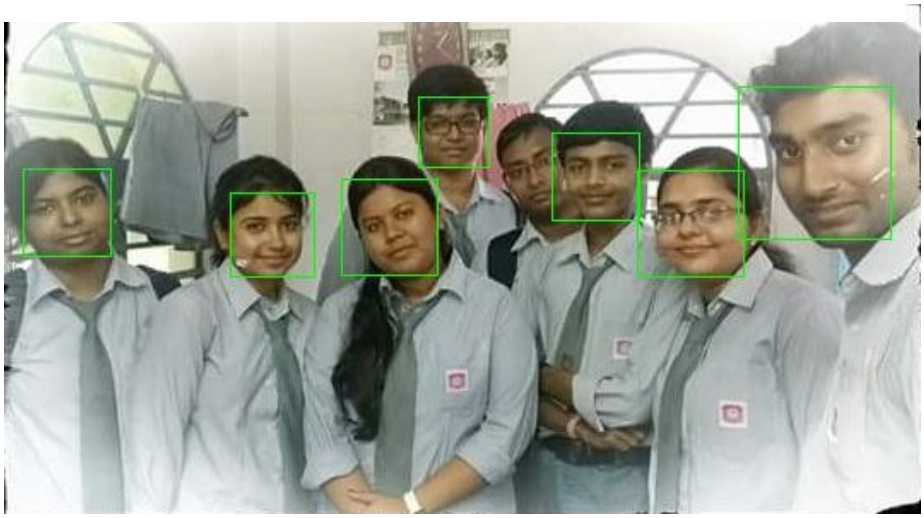


Fig 2.1 Face Detection Example

2.1.3 Face Detection Algorithms

There are many face detection algorithms to locate a human face in a scene – easier and harder ones. We are using the most popular and robust The Viola/Jones Face Detection algorithm for our problem.

THE VIOLA/JONE'S FACE DETECTOR

The characteristics of Viola–Jones algorithm which make it a good detection algorithm are:

- Robust – very high detection rate (true-positive rate) & very low false-positive rate always.
- Real time – For practical applications at least 2 frames per second must be processed.
- Face detection only (not recognition) - The goal is to distinguish faces from non-faces (detection is the first step in the recognition process).

The algorithm has four stages:

1. Haar Feature Selection
2. Creating an Integral Image
3. Adaboost Training
4. Cascading Classifiers

The features sought by the detection framework universally involve the sums of image pixels within rectangular areas. As such, they bear some resemblance to Haar basis functions, which have been used previously in the realm of image-based object detection. However, since the features used by Viola and Jones all rely on more than one rectangular area, they are generally more complex. The figure on the right illustrates the four different types of features used in the framework. The value of any given feature is the sum of the pixels within clear rectangles subtracted from the sum of the pixels within shaded rectangles. Rectangular features of this sort are primitive when compared to alternatives such as steerable filters. Although they are sensitive to vertical and horizontal features, their feedback is considerably coarser.

1. Haar Features – All human faces share some similar properties. These regularities may be matched using **Haar Features**.

A few properties common to human faces:

- The eye region is darker than the upper-cheeks.
- The nose bridge region is brighter than the eyes.

Composition of properties forming matchable facial features:

- Location and size: eyes, mouth, bridge of nose
- Value: oriented gradients of pixel intensities

The four features matched by this algorithm are then sought in the image of a face (shown at left).

Rectangle features:

- Value = Σ (pixels in black area) - Σ (pixels in white area)
- Three types: two-, three-, four-rectangles, Viola & Jones used two-rectangle features
- For example: the difference in brightness between the white &black rectangles over a specific area
- Each feature is related to a special location in the sub-window

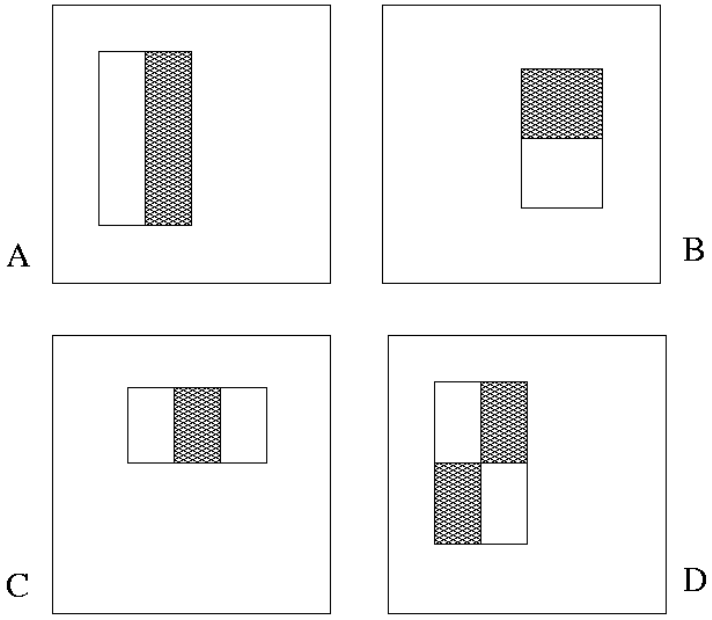
2. An image representation called the integral image evaluates rectangular features in *constant* time, which gives them a considerable speed advantage over more sophisticated alternative features. Because each feature's rectangular area is always adjacent to at least one other rectangle, it follows that any two-rectangle feature can be computed in six array references, any three-rectangle feature in eight, and any four-rectangle feature in nine.

The integral image at location (x, y), is the sum of the pixels above and to the left of (x ,y), inclusive.

Image Features:

"Rectangle filters"

Value = ∑ (pixels in white area) – ∑ (pixels in black area)



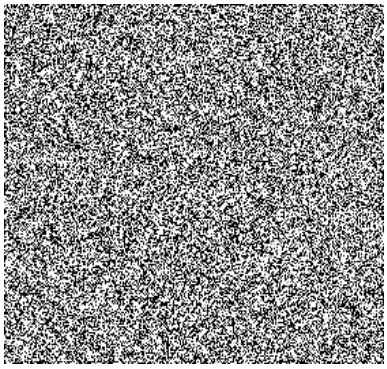
Haar Feature that looks similar to the bridge of the nose is applied onto the face

Haar Feature that looks similar to the eye region which is darker than the upper cheeks is applied onto a face

Fig 2.3 Working of Face Detector

EXAMPLE

Source



Result

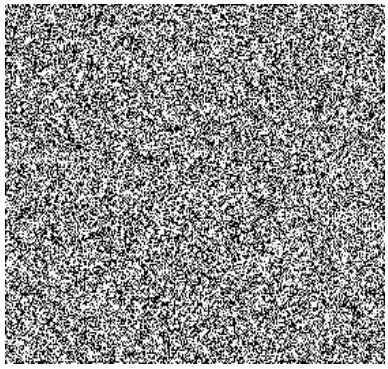


Fig 2.4 Example of Face Detection

2.1.4 Face Recognition

A facial recognition system is a computer application capable of identifying or verifying a person from a digital image or a video frame from a video source. One of the ways to do this is by comparing selected facial features from the image and a facial database

The difference between face detection and recognition is that in detection we just need to determine if there is some face in the image, but in recognition we want to determine whose face it is. In the above example we detected a face, which we recognize as our Project Guide Mr. Shibdas Bhattacharya.

In order to understand the methods for recognizing faces, more advanced mathematical knowledge is required; namely linear algebra and statistics.

Open CV and Mat lab Computer Vision toolbox provides three methods of face recognition: Eigen faces, Fisher faces and Local Binary Patterns Histograms (LBPH).

All three methods perform the recognition by comparing the face to be recognized with some training set of known faces. In the training set, we supply the algorithm faces and tell it to which person they belong. When the algorithm is asked to recognize some unknown face, it uses the training set to make the recognition. Each of the three aforementioned methods uses the training set a bit differently.

Eigen faces and Fisher faces find a mathematical description of the most dominant features of the training set as a whole. LBPH analyses each face in the training set separately and independently.

An example training set:

2.1.5 Eigen faces & Fisher faces

Those familiar with linear algebra will remember that every vector space has an orthogonal basis. By combining elements of this basis we can compose every vector in this vector space. And vice versa, every vector in the vector space can be decomposed to the elements of the basis.

Images (gray scale) are nothing more than a series of numbers, each number corresponding to some intensity level. So why not treat images as vectors? Say, for example, we have a collection of face images of size 150 by 150 pixels; each of these images can be thought of as a vector of size 22,500 (150*150). We can now talk about the vector space in which these vectors reside. By treating the images as samples of data, we can perform a Principal Components Analysis and obtain the eigenvectors which make up the basis of the vector space.

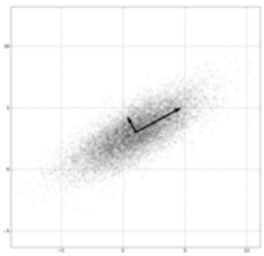


Fig-2.5 Principal components of a dataset (Source: Wikipedia).

These eigenvectors represent the most prominent features of the dataset, and since we talk about face images, the eigenvectors actually represent the strongest characteristics of the faces in the dataset.

2.1.6 Local Binary Patterns Histogram

The **LBPH** method takes a different approach than the Eigen faces method. In **LBPH** each images is analysed independently, while the Eigen faces method looks at the dataset as a whole. The LBPH method is somewhat simpler, in the sense that we characterize each image in the dataset locally; and when a new unknown image is provided, we perform the same analysis on it and compare the result to each of the images in the dataset. The way which we analyse the images is by characterizing the local patterns in each location in the image.

Local binary pattern (LBP) is a type of visual descriptor used for classification in computer vision. LBP is the particular case of the Texture Spectrum model proposed in 1990. LBP was first described in 1994.It has since been found to be a powerful feature for texture classification; it has further been determined that when LBP is combined with the **Histogram of oriented gradients (HOG)** descriptor, it improves the detection performance considerably on some datasets.

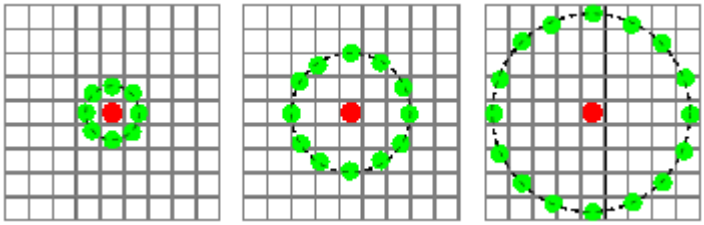


Fig 2.6

Three neighbourhood examples used to define a texture and calculate a local binary pattern (LBP)

2.1.7 Why we use Local Binary Patterns Histogram

Eigenfaces and Fisherfaces take a somewhat holistic approach to face recognition. You treat your data as a vector somewhere in a high-dimensional image space. We all know high-dimensionality is bad, so a lower-dimensional subspace is identified, where (probably) useful information is preserved. The Eigenfaces approach maximizes the total scatter, which can lead to problems if the variance is generated by an external source, because components with a maximum variance over all classes aren't necessarily useful for classification (see http://www.bytefish.de/wiki/pca_lda_with_gnu_octave). So to preserve some discriminative information we applied a Linear Discriminant Analysis and optimized as described in the Fisherfaces method. The Fisherfaces method worked great. At least for the constrained scenario we've assumed in our model.

Now real life isn't perfect. You simply can't guarantee perfect light settings in your images or 10 different images of a person. So what if there's only one image for each person? Our covariance estimates for the subspace *may*be horribly wrong, so will the recognition. Remember the Eigenfaces method had a 96% recognition rate on the AT&T Facedatabase? How many images do we actually need to get such useful estimates? Here are the Rank-1 recognition rates of the Eigenfaces and Fisherfaces method on the AT&T Facedatabase, which is a fairly easy image database:

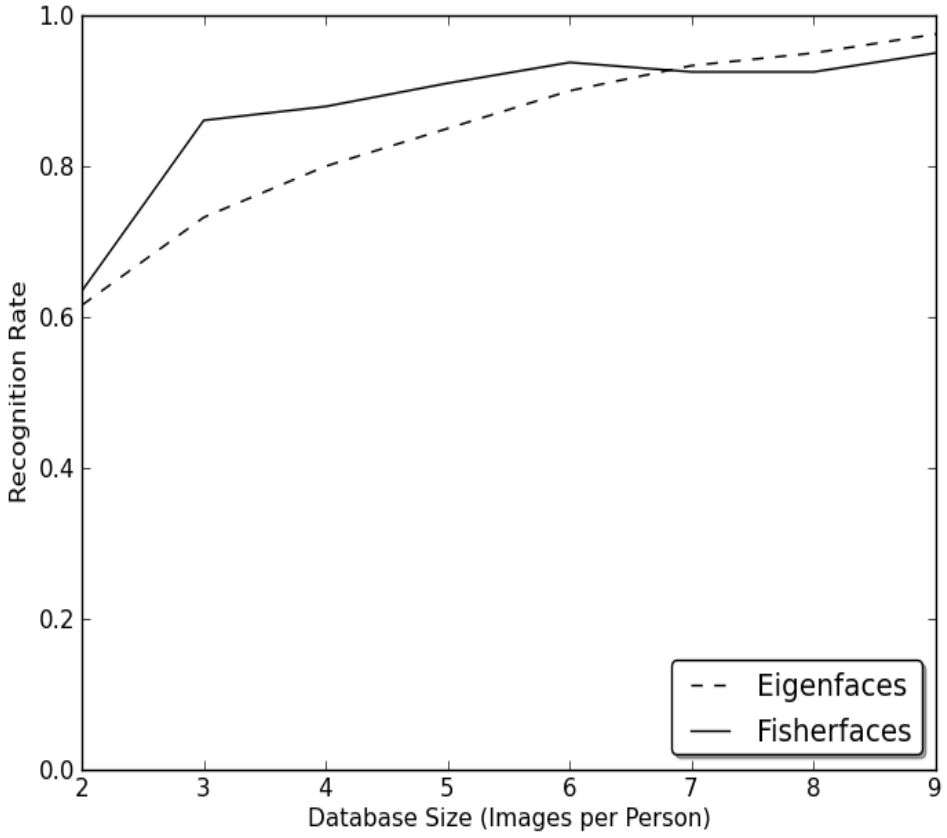


Fig-2.7 Experiment Result Graph

So in order to get good recognition rates you'll need at least 8(+-1) images for each person and the Fisher faces method doesn't really help here. The above experiment is a 10-fold cross validated result carried out with the facer framework at: https://github.com/bytefish/facerec.

So some research concentrated on extracting local features from images. The idea is to not look at the whole image as a high-dimensional vector, but describe only local features of an object. The features you extract this way will have a low-dimensionality implicitly. A fine idea! But you'll soon observe the image representation we are given doesn't only suffer from illumination variations. Think of things like scale, translation or rotation in images - your local description has to be at least a bit robust against those things. Just like SIFT, the Local Binary Patterns methodology has its roots in 2D texture analysis. The basic idea of Local Binary Patterns is to summarize the local structure in an image by comparing each pixel with its neighbourhood. Take a pixel as centre and threshold its neighbours against. If the intensity of the centre pixel is greater-equal its neighbour, then denote it with 1 and 0 if not. You'll end up with a binary number for each pixel, just like 11001111. So with 8 surrounding pixels you'll end up with 2^8 possible combinations, called Local Binary Patterns or sometimes referred to as LBP codes. The first LBP operator described in literature actually used a fixed 3 x 3 neighbourhood just like this:

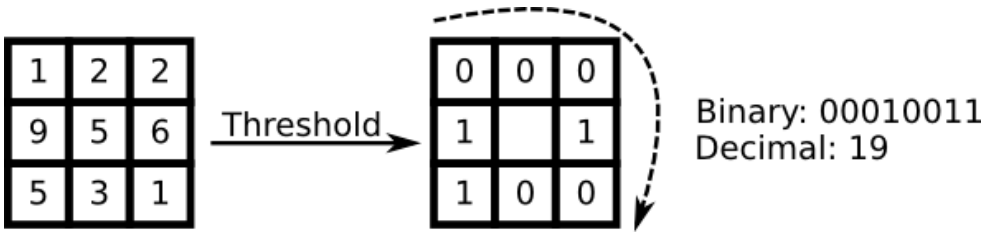


Fig 2.8

2.2 OBJECTIVE

The Project Objective is to implements face recognition in an optimum way in terms of run time into the system. Various algorithms and methodologies are studied and hardware resources planning will be done to achieve the goal. This kind of face recognition system can be widely used in our daily life in different sectors. We hope that human life can be greatly helped with this technology.

Chapter- 3

3.1 ALGORITHM

3.1.1 Local Binary Patterns Histogram Algorithmic Description

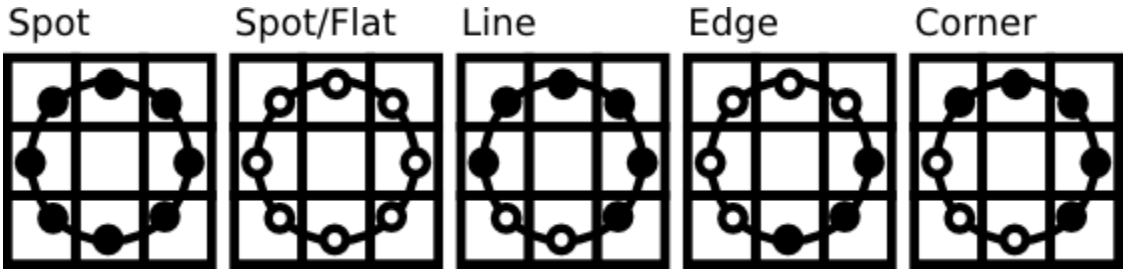
A more formal description of the LBP operator can be given as:

$$LBP(x_c, y_c) = \sum_{p=0}^{P-1} 2^p s(i_p - i_c)$$

, with (x_c, y_c) as central pixel with intensity i_c; and i_n being the intensity of the neighbour pixel. s is the sign function defined as:

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{else} \end{cases} \qquad (1)$$

This description enables you to capture very fine grained details in images. In fact the authors were able to compete with state of the art results for texture classification. Soon after the operator was published it was noted, that a fixed neighbourhood fails to encode details differing in scale. So the operator was extended to use a variable neighbourhood. The idea is to align an arbitrary number of neighbours' on a circle with a variable radius, which enables to capture the following neighbourhoods:



For a given Point (x_c, y_c) the position of the neighbour $(x_p, y_p), p \in P$ can be calculated by:

$$x_p = x_c + R\cos\left(\frac{2\pi p}{P}\right)$$

$$y_p = y_c - R\sin\left(\frac{2\pi p}{P}\right)$$

Where R is the radius of the circle and P is the number of sample points.

The operator is an extension to the original LBP codes, so it's sometimes called *Extended LBP* (also referred to as *Circular LBP*). If a point's coordinate on the circle doesn't correspond to image coordinates, the point gets interpolated. Computer science has a bunch of clever interpolation schemes; the Open CV implementation does a bilinear interpolation:

$$f(x,y) \approx \begin{bmatrix} 1-x & x \end{bmatrix} \begin{bmatrix} f(0,0) & f(0,1) \\ f(1,0) & f(1,1) \end{bmatrix} \begin{bmatrix} 1-y \\ y \end{bmatrix}.$$

By definition the LBP operator is robust against monotonic gray scale transformations. We can easily verify this by looking at the LBP image of an artificially modified image:

Fig 3.1 LBPH Face

So what's left to do is how to incorporate the spatial information in the face recognition model. The idea is to divide the LBP image into m local regions and extract a histogram from each. The spatially enhanced feature vector is then obtained by concatenating the local histograms (**not merging them**). These histograms are called *Local Binary Patterns Histograms*.

Chapter- 4

4.1 The Plan

Before we start coding, we better understand the different components in our application**.** As was already mentioned, our goal is to determine in which frames of the video our chosen person appears.

1. Create a Face Database Form Live Video Frame of Each Person and Link a label to it (label = face name).

2. Train the LBPH (Local Binary Patterns Histograms) Face Recognition Classifier

3. Read the video frame of the live input video.

4. Detect all the faces in the frame.

5. Try to recognize each of the detected faces as our chosen person.

6. If successful, draw a green rectangle around the face and display the person name. Otherwise, display Unknown.

7. If the Known person is Higher Authority then turn on the necessary electrical appliances in the particular person cabin.

8. Repeat steps 3-7 until the Application is closed by the user.

Evidently, our application should have the following major components:

1. Live Video Frame Reader

2. Faces Detector

3. Person Recognizer

4. External Hardware (Electrical appliances) Controller

A graphical representation of the implemented features and software architecture can be seen in fig 4.1

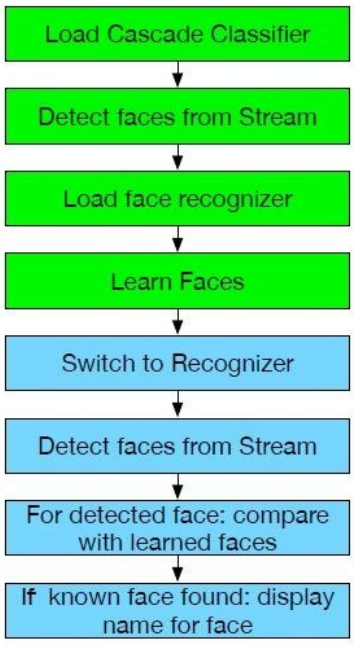


Fig 4.1 Software Architecture

4.2. Architecture

The overall system architecture is shown as a use case diagram in fig. 4.2. It shows the detection and recognition process. Subject A and B are positioned in front of the webcam. The detector wills then loads the cascade classifier for detecting the haar-like features of the image sequence. After detecting the faces of **Subject A and B** inside of an image the detector will forward the coordinates of the detected faces to the recognizer. Then the recognizer will loop through the learned samples of each subject and tries to recognize the subject. If the recognition process was successful the recognizer will highlight the face of a subject and display the name over the previously drawn rectangle. For this whole process the face samples need to be learned first.

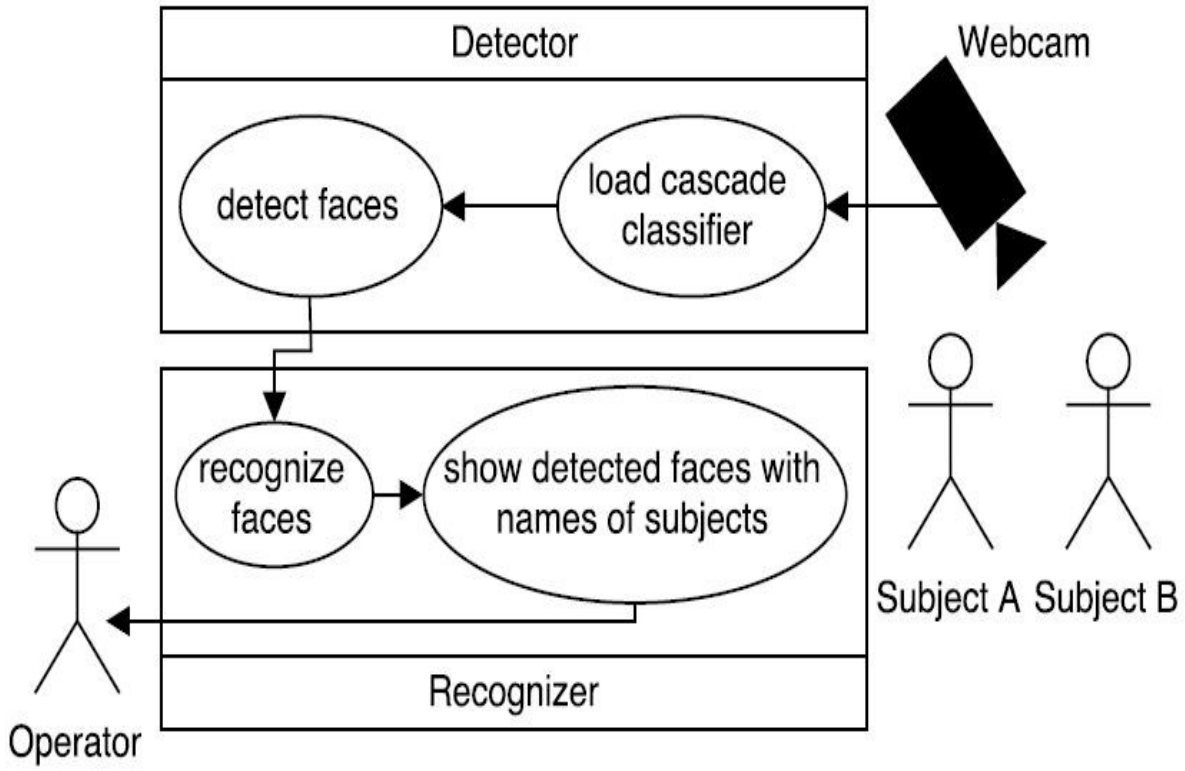


Fig 4.2 Use Case Diagram

Chapter -5

5.1 MINIMUM HARDWARE REQUIREMENTS

- Intel® Core™ i3-530 CPU @ 2.93 GHz
- 250 GB Hard Drive
- 2 GB Ram
- Standard Mouse
- Standard Keyboard
- Arduino Uno R3 (Atmega328 Microcontroller)
- 4 Channel Relay Module
- Jumper Wires (male to male)
- Jumper Wires (female to male)
- REES52 1.64FT USB 2.0 A-B Male Printer Cable 0.5m

5.2 MINIMUM SOFTWARE REQUIREMENTS

- Matlab R2015a
- Microsoft Windows 7 Home Basic
- Mexopencv Library (Matlab mex functions that interface a hundred of OpenCV APIs)
- OpenCV 3.0 Computer Vision (C++) Library
- Microsoft Visual C++ 2010 Compiler
- CMake 3.6.0
- Microsoft Visual Studio 2010
- Arduino IDE 1.5.0
- Microsoft Word 2003

Chapter -6

6.1 ACHIEVEMENTS TILL DATE

The following have been completed till date

- Basic idea about digital image processing
- Preliminary idea about Face Recognition
- Compiling OpenCV Library using Cmake and Visual C++
- Compiling MexOpenCV In Matlab
- Basic Implementation of LBPH(Local Binary Patterns Histogram)
- Basic idea about Arduino
- Basic idea about Relay Module
- Basic idea about Microcontroller Programming in C++

6.2 REMAINING WORK

The following work remaining

- Detail Implementation of LBPH Algorithm
- Coding of LBPH Algorithm
- Coding of Microcontroller (Atmega 2560)
- Modification of LBPH Algorithm for improvement in time and better accuracy.
- Export as a Standalone Software for Distribution

Chapter-7

7.1 BIBLIOGRAPHY

- Ahonen, T., Hadid, A., and Pietikainen, M. Face Recognition with Local Binary Patterns. Computer Vision - ECCV 2004 (2004), 469–481.
- Belhumeur, P. N., Hespanha, J., and Kriegman, D. *Eigen faces vs. Fisher faces: Recognition Using Class Specific Linear Projection.* IEEE Transactions on Pattern Analysis and Machine Intelligence 19, 7 (1997), 711–720.
- Opncv C++ Documentation
- https://kyamagu.github.io/mexopencv/
- http://docs.opencv.org/2.4/doc/tutorials/introduction/windows_visual_studio_Opencv/windows_visual_studio_Opencv.html
- https://in.mathworks.com/products/computer-vision/
- https://in.mathworks.com/help/vision/deep-learning-object-detection-and-recognition.html
- https://in.mathworks.com/help/vision/ug/local-feature-detection-and-extraction.html
- http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html