Mini Project on

FACE DETECTION AND RECOGNITION USING OPENCV

By

Mayank Chhabra (1SI12CS156)

Faraz Ahmad Khan (1SI12CS041)

Guide

Mrs. Nayana S. M.Tech

Assistant Professor

Dept. of CSE

Dept. of Computer Science and Engineering

Siddaganga Institute of Technology

Tumkur-572103

2014-15

ABSTRACT

An application for automatic face detection and recognition on video streams from surveillance cameras in public or commercial places is discussed in this paper. In many situations it is useful to detect and recognize people due to many reasons such as security reasons, differentiating segregation reasons in exhibits, commercial malls, prisons, high alerted places and public places in buildings.

Prototype is designed to work with web cameras for the face detection and recognition system based on OpenCV using Microsoft Visual Studio. The system is based on AdaBoost algorithm and abstracts faces Haar-Like features. This system can be used for security purpose to record the visitor face as well as to detect and recognize the face if person is in database. A program is developed using OpenCV that can detect people's face and also recognize faces.

This method utilizes the idea of the principal component analysis and decomposes face images into a small set of characteristic feature images called eigenfaces. Recognition is performed by projecting a new face onto a low dimensional linear “face space” defined by the eigenfaces, followed by computing the distance between the resultant position in the face space and those of known face classes. A number of experiments were done to evaluate the performance of the face recognition system we have developed. The results demonstrate that the eigenface approach is quite robust to head/face orientation, but sensitive to scale and illumination. At the end of the report, a couple of ways are suggested to improve the recognition rate. The report is organized as follows: the first part provides an overview of face recognition algorithms; the second part states the theory of the eigenfaces approach for face recognition.

1. INTRODUCTION

The research purpose of computer vision aims to simulate the manner of human eyes directly by using computer. Computer vision is such kind of research field which tries to percept and represent the 3D information for world objects. Its essence is to reconstruct the visual aspects of 3D object by analyzing the 2D information extracted accordingly. 3D objects surface reconstruction and representation not only provide theoretical benefits, but also are required by numerous applications.

Face detection is a process, which is to analysis the input image and to determine the number, location, size, position and the orientation of face. Face detection is the base for face tracking and face recognition, whose results directly affect the process and accuracy of face recognition. The common face detection methods are: knowledge-based approach, Statistics-based approach and integration approach with different features or methods. The knowledge-based approach can achieve face detection for complex background images to some extent and also obtain high detection speed, but it needs more integration features to further enhance the adaptability. Statistics-based approach detects face by judging all possible areas of images by classifier, which is to look the face region as a class of models, and use a large number of "Face" and "non-face" training samples to construct the classifier.

The method has strong adaptability and robustness, however, the detection speed needs to be improved, because it requires test all possible windows by exhaustive search and has high computational complexity. The AdaBoost algorithm arose in recent years; it trains the key category features to the weak classifiers, and cascades them into a strong classifier for face detection. The method has real-time detection speed and high detection accuracy, but needs long training time. The digital image of the face generated is a representation of a two-dimensional image as a finite set of digital values, called picture elements or pixels. Pixel values typically represent gray levels, colours, heights, opacities etc. It is to be noted that digitization implies that a digital image is an approximation of a real scene. Recently there has been a tremendous growth in the field of computer vision. The conversion of this huge amount of low level information into usable high level information is the subject of computer vision. It deals with the development of the theoretical and algorithmic basis by which useful information about the 3D world can be automatically extracted and analyzed from a single or multiple 2D images of the world as shown in figure below.

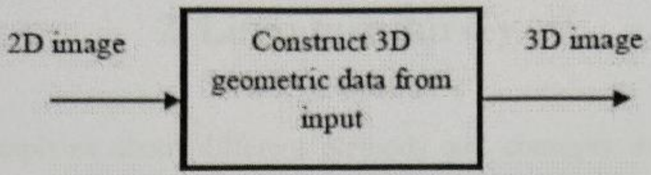


Fig.1.1: Typical Data Processing in Computer Vision

This project describes a system that can detect and track human face in real time using haar-like features where the detection algorithm is based on wavelet transform. In computer vision, low level processing involves image processing tasks in which the quality of the image is improved for the benefit of human observers and higher level routines to perform better (Viola & Jones). Intermediate level processing involves the processes of feature extraction and pattern detection tasks.

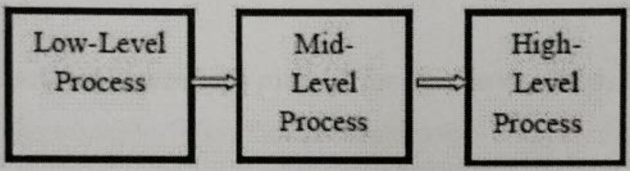


Fig. 1.2: Process levels in Computer Vision

High level vision involves autonomous interpretation of scenes for pattern classification, recognition and identification of objects in the scenes as well as any other information required for human understanding as shown in figure 2. Statistics-based approach to this project detects face by judging all possible areas of images by classifier, which is to look the face region as a class of models, and use a large number of "Face" and "Non-face" training samples to construct the classifier. The method has strong adaptability and robustness. The program can rectangle the face area with the data got from web camera.

2. Literature Survey

This chapter explains about different methods and concepts used in face detection and recognition.

Face recognition is a biometric identification by scanning a person's face and matching it against a library of known faces.It is a set of two task:

- Face Identification: Given a face image that belongs to a person in a database, tell whose image it is.
- Face Verification: Given a face image that might not belong to the database, verify whether it is from the person it is claimed to be in the database.

There are three available algorithms-Eigenfaces, Fisherfaces and Local Binary Patterns Histograms.

This prototype uses the Fisherfaces method for face recognition, because it is robust against large changes in illumination. The other necessities are-

- Path to valid Haar-Cascade for detecting a face with a cascade classifier
- Path to a valid CSV File for learning a FaceRecognizer a webcam

2.1 Fisherface Algorithm

Let X be a random vector with samples drawn from c classes:

$$X = \{X_1, X_2, \ldots, X_c\}$$
$$X_i = \{x_1, x_2, \ldots, x_n\}$$

The scatter matrices and S_{W} are calculated as:

$$S_B = \sum_{i=1}^{c} N_i(\mu_i - \mu)(\mu_i - \mu)^T$$
$$S_W = \sum_{i=1}^{c} \sum_{x_j \in X_i} (x_j - \mu_i)(x_j - \mu_i)^T$$

where μ is the total mean:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

And μ_i is the mean of class $i \in \{1,\ldots,c\}$:

$$\mu_i = \frac{1}{|X_i|}\sum_{x_j \in X_i} x_j$$

Fisher's classic algorithm now looks for a projection , that maximizes the class separability criterion:

$$W_{opt} = \arg\max_W \frac{|W^T S_B W|}{|W^T S_W W|}$$

a solution for this optimization problem is given by solving the General Eigenvalue Problem:

$$\begin{aligned} S_B v_i &= \lambda_i S_w v_i \\ S_W^{-1} S_B v_i &= \lambda_i v_i \end{aligned}$$

The rank of Sw is at most (N-c), with N samples and c classes. In pattern recognition problems the number of samples N is almost always samller than the dimension of the input data (the number of pixels), so the scatter matrix Sw becomes singular.This was solved by performing a Principal Component Analysis on the data and projecting the samples into the (N-c) dimensional space. A Linear Discriminant Analysis was then performed on the reduced data, because Sw isn't singular anymore.

The optimization problem can then be rewritten as:

$$\begin{aligned} W_{pca} &= \arg\max_W |W^T S_T W| \\ W_{fld} &= \arg\max_W \frac{|W^T W_{pca}^T S_B W_{pca} W|}{|W^T W_{pca}^T S_W W_{pca} W|} \end{aligned}$$

The transformation matrix , that projects a sample into the -dimensional space is then given by:

$$W = W_{\text{fld}}^{T} W_{\text{pca}}^{T}$$

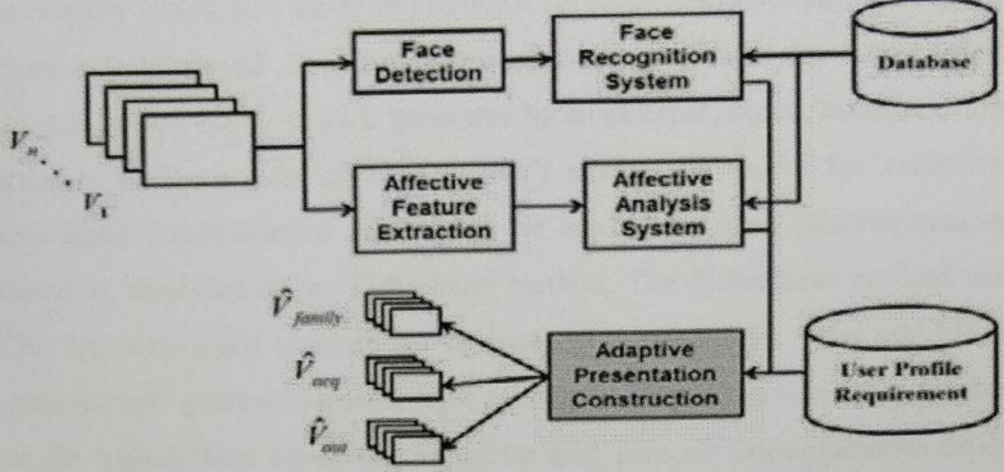


Fig. 2.1: Process model for face detection and recognition

2.2 Eigenfaces

The problem with the image representation we are given is its high dimensionality. Two-dimensional $p \times q$ grayscale images span a $m = pq$-dimensional vector space, so an image with 100×100 pixels lies in a $10,000$-dimensional image space already. The question is: Are all dimensions equally useful for us? We can only make a decision if there's any variance in data, so what we are looking for are the components that account for most of the information. The Principal Component Analysis (PCA) was independently proposed by Karl Pearson (1901) and Harold Hotelling (1933) to turn a set of possibly correlated variables into a smaller set of uncorrelated variables. The idea is, that a high-dimensional dataset is often described by correlated variables and therefore only a few meaningful dimensions account for most of the information. The PCA method finds the directions with the greatest variance in the data, called principal components.

The Fisherfaces method learns a class-specific transformation matrix, so the they do not capture illumination as obviously as the Eigenfaces method. The Discriminant Analysis instead finds the facial features to discriminate between the persons. It's important to mention, that the performance of the Fisherfaces heavily depends on the input data as well. Practically said: if you learn the Fisherfaces for well-illuminated pictures only and you try to recognize faces in bad-illuminated scenes, then method is likely to find the wrong components (just because those features may not be predominant on bad illuminated images). This is somewhat logical, since the method had no chance to learn the illumination.

2.3 Comparison

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. So to preserve some discriminative information we applied a Linear Discriminant Analysis and optimized as described in the Fisherfaces method. The Fisherfaces method works 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. Therefore a number of images of a person is required to be stored in the database to make the recognition process more accurate.

3. Requirement Specification

This chapter is about the hardware and software requirements of the project.

3.1 Specific Requirements

There are two essential and main requirements for building prototype of detecting and recognising faces:-

1 Integrated development environment (IDE)-used to develop source code

2 Image processing library-its pre-defined algorithm and functions used in code

3.1.1 Microsoft Visual Studio

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs for Microsoft Windows, as well as web sites, web applications and web services. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio supports different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++ and C++/CLI , VB.NET, C# and F# .

3.1.2 OpenCV

OpenCV (*Open* Source Computer Vision Library) is a library of programming functions mainly aimed at real time computer vision, developed by Intel and now supported by Willow Garage It is free for use under the open source BSD license. The library is cross-platform. It focuses mainly on real-time image processing. If the library finds Intel's Integrated Performance Primitives on the system [Open Source Computer Vision Library Reference Manual-intel; Gary Bradski & Adrian Kaehler O.Reilly, 2008], it will use these proprietary optimized routines to accelerate it. Interface makes OpenCV portable to some specific platforms such as digital signal processors. Wrappers for languages such as C#, Python, Ruby and Java (using JavaCV) have been developed to encourage adoption by a wider audience [Zhang, 2008]. However, since version 2.0, OpenCV includes both its traditional C interface as well as a new C++ interface. This new interface seeks to reduce the number of lines of code necessary to code up vision functionality as well as reduce common programming errors.

3.2 Hardware Requirements

In order to run the application any hardware platform capable of running Microsoft Visual Studio 2005 is suitable. The application would be developed using the following hardware:

1. Processor- 32 bit or 64 bit
2. 128MB RAM or more.

3.3 Software Requirements

In order to run the application, the following software is required:

1. Operating System : Windows Family
2. Programming Language : C++
3. Development Tools : OpenCV, Microsoft Visual Studio

4. Design and Implementation

This chapter describes the design and implementation of the fisherface algorithm. Real-time Face Detection-In this section, firstly video (live stream) is captured from web camera or surveillance camera and then cascade classifier object(implementing haar cascade library pre-defined in OpenCV) is used to detect faces. AdaBoost Algorithm is use to combine series of weak classifier into strong classifier which is the highlighting notion of detection.

4.1 Haar-like feature

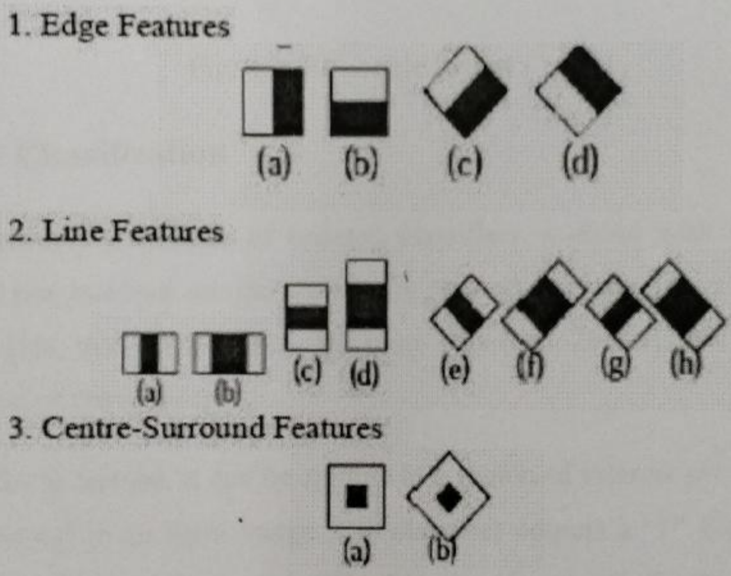


Fig. 4.1: Haar features

In the implementation of face detection, Xi contains a huge number of face features, and some of the features with low i to train our strong classifier are selected. By AdaBoost algorithm this can be achieved automatically [Lu et al., 1999]. For each iteration i with each feature in Xi can be calculated and then the lowest one is what we need. For doing this, the face detection rapid could be very fast. In next part, you will find there are many haar-like features, so it is hard to make use of all them. Face features are abstracted from the input image and are used to train the classifier, modify weights [Introduction to OpenCV].

Face features are abstracted from the input images and are used to train the classifiers, modify weights as mentioned in [Viola & Jones, 2001]. In 2001, Viola et al. first introduced the haar-like features. The haar-like features are rectangle features and value is that the sum of pixels in black district subtracts the sum of pixels in white district [Guo & Wang, 2009] as shown in figure 6 and figure 8. Rainer Lienhart had done an extended set of haar-like features which significantly enrich the basic set of simple haar-like features, and can get a better hit rate

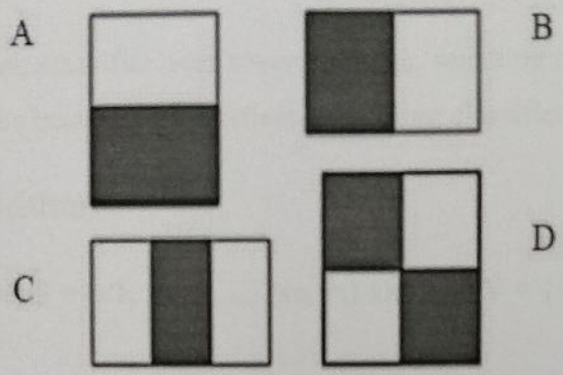


Fig. 4.2: Rectangle features

4.2 Cascade Classification

A classifier (namely a cascade of boosted classifiers working with haar-like features) is trained with a few hundred sample views of a particular object (i.e., a face or a car), called positive examples, that are scaled to the same size (say, 20x20), and negative examples - arbitrary images of the same size.

After a classifier is trained, it can be applied to a region of interest (of the same size as used during the training) in an input image. The classifier outputs a "1" if the region is likely to show the object (i.e., face/car), and "0" otherwise. To search for the object in the whole image one can move the search window across the image and check every location using the classifier. The classifier is designed so that it can be easily "resized" in order to be able to find the objects of interest at different sizes, which is more efficient than resizing the image itself. So, to find an object of an unknown size in the image the scan procedure should be done several times at different scales.

The word "cascade" in the classifier name means that the resultant classifier consists of several simpler classifiers (stages) that are applied subsequently to a region of interest until at some stage the candidate is rejected or all the stages are passed. The word "boosted" means that the classifiers at every stage of the cascade are complex themselves and they are built out of basic classifiers using one of four different boosting techniques (weighted voting).

Currently Discrete Adaboost, Real Adaboost, Gentle Adaboost and Logitboost are supported. The basic classifiers are decision-tree classifiers with at least 2 leaves. Haar-like features are the input to the basic classifiers

4.3 Adaboost

It tries out multiple weak classifier over several rounds, selecting best weak classifier in each round and combining the best weak classifiers into strong classifier.

The AdaBoost Algorithm

1. Input: Give sample set S = (x1, y1),.........(xn,yn) xiϵX, yiϵY = {-1,+1}, number of iterations T

2. Initialize: wi,j=1/N i = 1,......,N

3. For t = 1,2,...,T,

i) Train weak classifier using distribution Wt.

ii) Calculate the weight (wi) training error for each hypoproject

hn Ɛt=Sigma $Wt,i|hi-yi|$Ni=1

iii) Set:

at=1/2log1−Ɛt/Ɛt

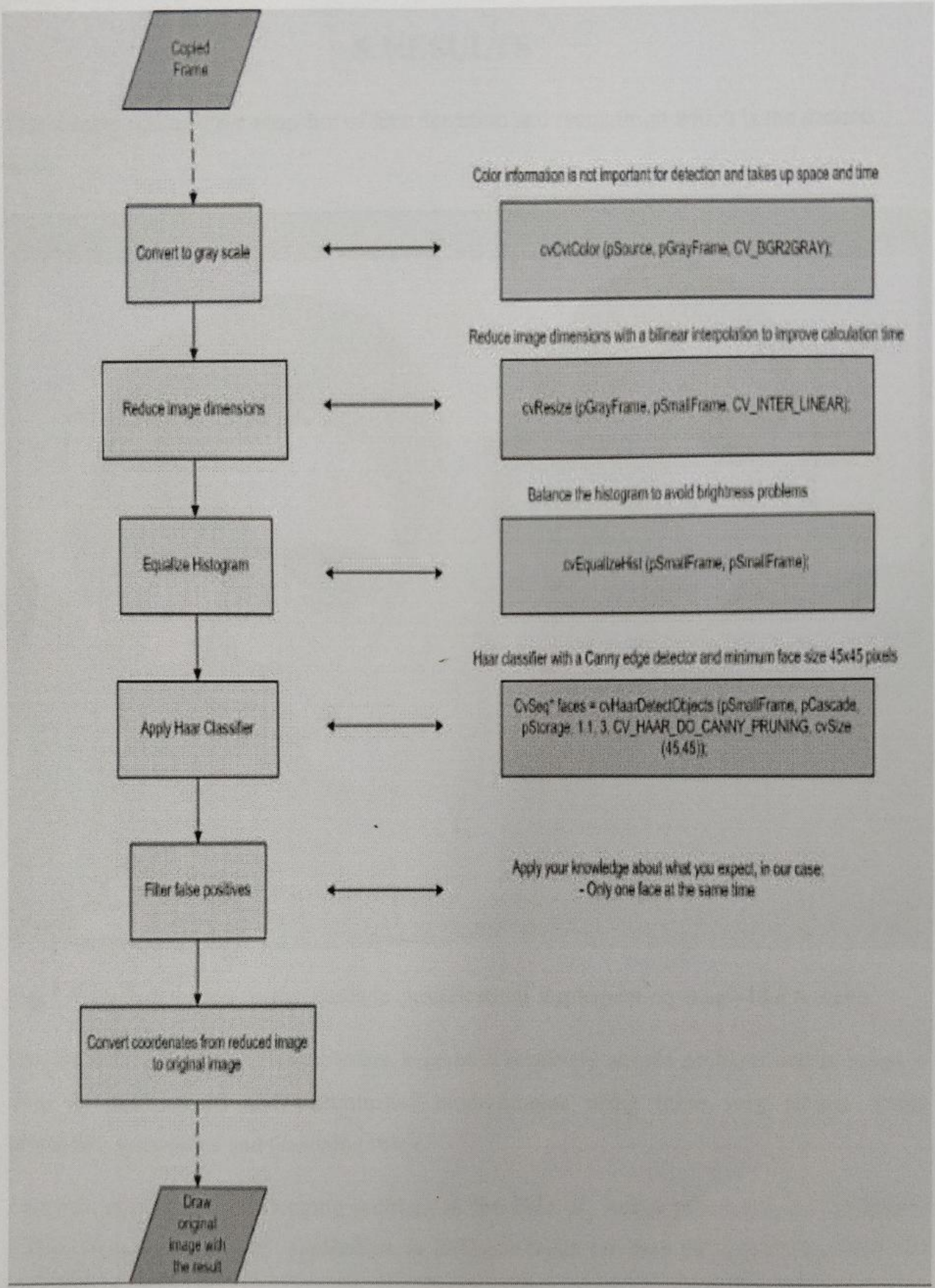
iv) Update the weights:

Wt+1,i=1+$Wt,i/Zt$×{e−at eat} =wt,i exp −$atyiht$ xi Zt

Zt is normalization constant

4. Output: the final hypoproject, also the stronger classifier.

(x) =(Sigma at ht$T$$t$=1 ($x$))


Copied Frame
Convert to gray scale
Color information is not important for detection and takes up space and time
Reduce image dimensions
Reduce image dimensions with a bilinear interpolation to improve calculation time
Equalize Histogram
Balance the histogram to avoid brightness problems
Apply Haar Classifier
Haar classifier with a Canny edge detector and minimum face size 45x45 pixels
Filter false positives
Apply your knowledge about what you expect, in our case:
- Only one face at the same time
Convert coordenates from reduced image to original image
Draw original image with the result

```
cvCvtColor (pSource, pGrayFrame, CV_BGR2GRAY);
cvResize (pGrayFrame, pSmallFrame, CV_INTER_LINEAR);
cvEqualizeHist (pSmallFrame, pSmallFrame);
CvSeq* faces = cvHaarDetectObjects (pSmallFrame, pCascade, pStorage, 1.1, 3, CV_HAAR_DO_CANNY_PRUNING, cvSize (45,45));
```

Fig 4.3: Overview of Face Detection and Recognition

5. RESULTS

This chapter contains the snapshot of face detection and recognition which is the desired result.

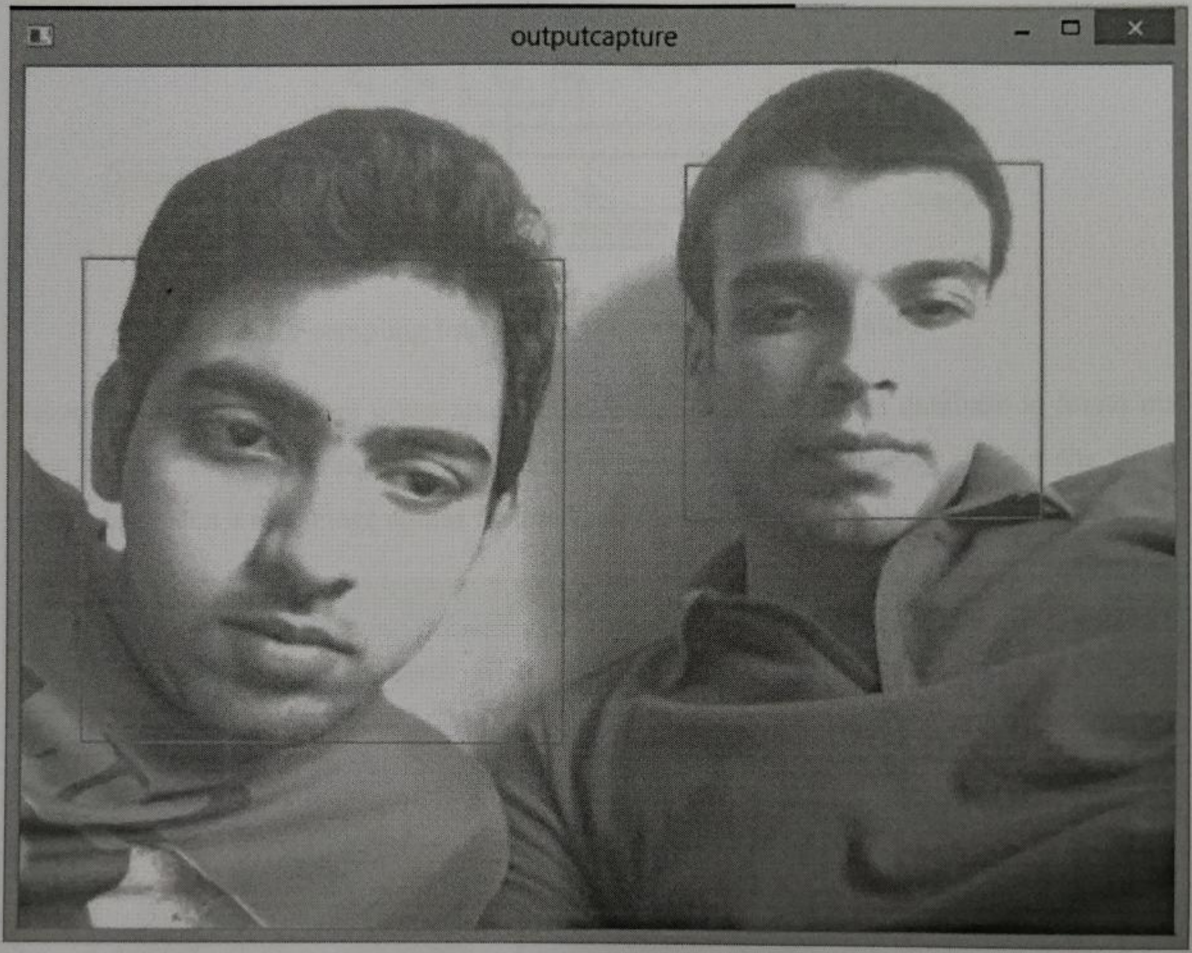


Fig. 5.1: Face detection using cascade classification(implementing haar –like feature)

Real-time face detection has therefore become a relatively simple problem and is possible even in unstructured and uncontrolled environments using these very simple image processing techniques and reasoning rules.

Face recognition is a challenging problem in the field of image processing and computer vision. Because of lots of application in different fields the face recognition has received great attention.

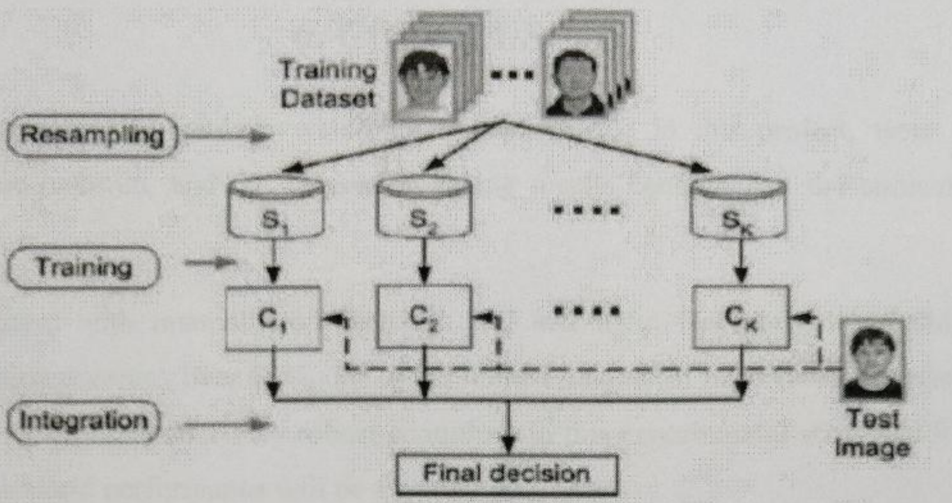


Fig. 5.2: Re-sampling Integration Scheme for face recognition

According to the procedure given above "fisher faces" of the stored database is found out. Then feature vector of the each individual is calculated by projecting it onto the set of fisherface. When a test image comes feature vector is calculated exactly in the same way.

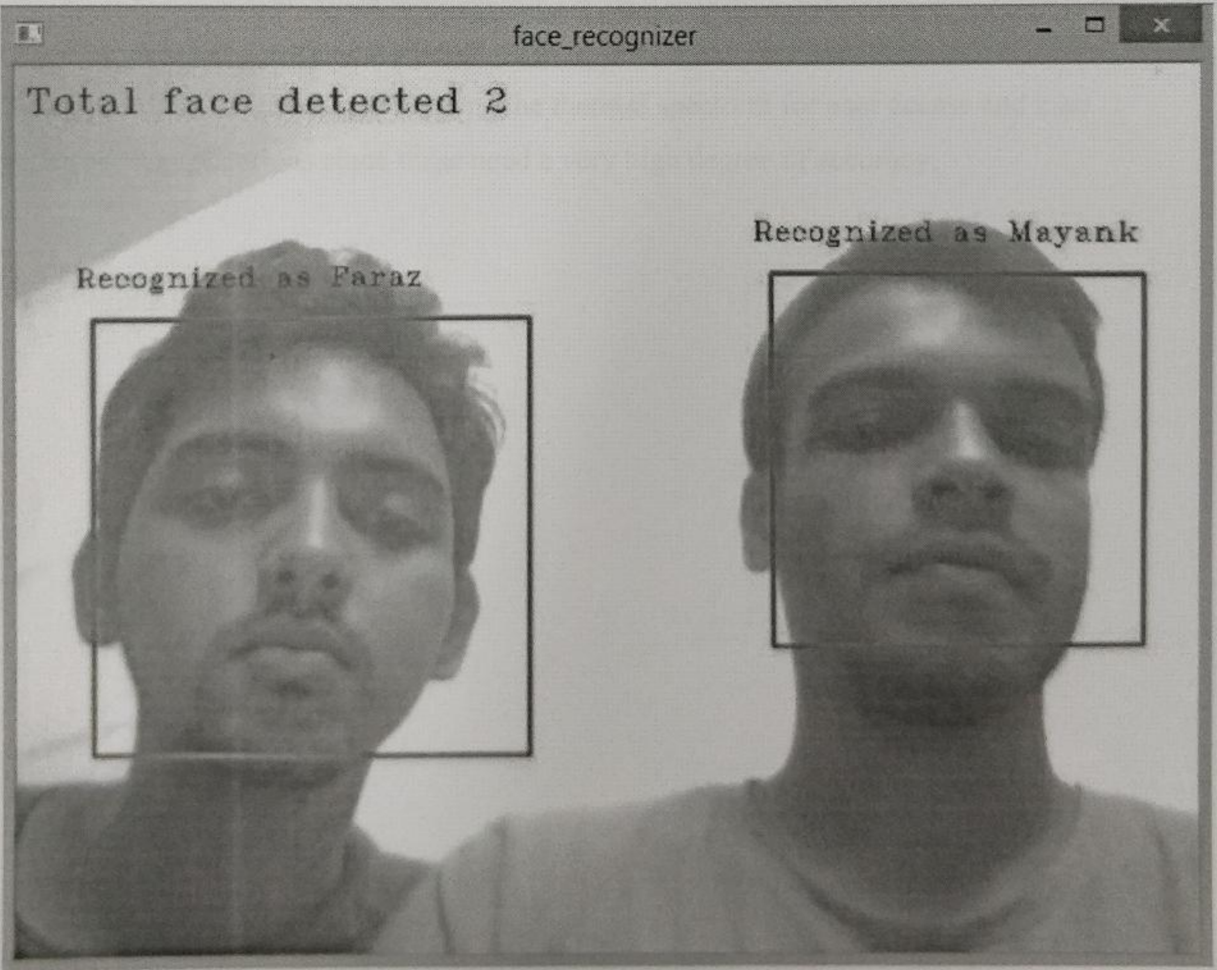


Fig. 5.3: Face Recognition using fisherface approach

6. CONCLUSION

The computational models, which were implemented in this project, were chosen after extensive research, and the successful testing results confirm that the choices made were reliable.

The system with manual face detection and automatic face recognition did not have a recognition accuracy over 90%, due to the limited number of fisherfaces that were used . This system was tested under very robust conditions in this experimental study and it is envisaged that real-world performance will be far more accurate.

The fully automated frontal view face detection system displayed virtually perfect accuracy and in the researcher's opinion further work need not be conducted in this area. The face recognition and detection algorithms were thoroughly studied taking a number of test images and varying the conditions and variables. All the work mentioned above involved real time data.

The most suitable real-world applications for face detection and recognition systems are for mugshot matching and surveillance. There are better techniques such as iris or retina recognition and face recognition using the thermal spectrum for user access and user verification applications since these need a very high degree of accuracy.

REFERENCES

[1] M. Turk, A. Pentland, Eigenfaces for Recognition, Journal of Cognitive Neurosicence, Vol. 3, No. 1, Win. 1991, pp. 71-86

[2] Discriminant analysis for recognition of human face images
Kamran Etemad and Rama Chellappa

[3] Baron, R. J. (1981). *Mechanisms of human facial recognition.* International Journal of Man Machine Studies, 15:137-178

[4] 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

[5] Goldstein, A.J., Harmon, L.D., and Lesk, A.B. (1971). *Identification of human faces*. In Proc. IEEE, Vol. 59, page 748