A Project Report on

# Face Recognition using Template Matching

Submitted to the Dept. of Information Technology, SNIST
in the partial fulfillment of the academic requirements for the award of

## B.Tech (Information Technology)

by


**B.SHANKAR LAL (15311A1296)**
**MD.NAVEED (15311A1291)**
**L.DEVENDER (15311A1297)**


under the guidance of

**Dr. K. Kranthi Kumar**
Associate Professor



**Department of Information Technology**
School of Computer Science and Informatics
Sreenidhi Institute of Science and Technology
Yamnampet, Ghatkesar Mandal, R. R. Dist., Hyderabad – 501301


Affiliated to
Jawaharlal Nehru Technological University Hyderabad
Hyderabad – 500 085

2019

# Department of Information Technology

School of Computer Science and Informatics

Sreenidhi Institute of Science and Technology

## Certificate

This is to certify that the Project report on "**Face Recognition using Template Matching**" is a bonafide work carried out by B.Shankar lal (15311A1296), Md.Naveed Ahmed (15311A1291), L.Devender (15311A1297)Science and Technology, Hyderabad, affiliated to Jawaharlal Nehru Technological University, Hyderabad under our guidance and supervision. The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

| **Internal Guide** | **Project Coordinator** | **Head of the Department** |
|---|---|---|
| (Dr.K.Kranthi Kumar) | (Dr.D.Teja santosh)<br>(Dr.Sunil Bhutada) | (Dr.V.V.S.S.S.Balaram) |

**External Examiner**

# DECLARATION

We,B.Shankarlal (15311A1296), Md. Naveed Ahmed (15311A1291),L.Devender nayak (15311A1297), Students of **Sreenidhi Institute of Science and Technology,Yamnumpet,Ghatkesar,** Studying IV Year 2nd semester, **Information Technology** solemnly declare that the group project work titled **Face Recognition using Template Matching** is submitted to **Sreenidhi Institute of Science and Technology** for partial fulfillment for the award of degree of bachelor of technology in **Information Technology**

It is declared to the best of our knowledge that the work reported does not form part of any dissertation submitted to any other University or Institute for award of any degree.

## Acknowledgements

We would like to express our immense gratitude and sincere thanks to **Dr.P. Narasimha Reddy** for giving us the opportunity to do our project work in **Sreenidhi Institute of science and technology**, Hyderabad. We would also like to express our deep felt gratitude and appreciation to **Dr.K.Kranthi Kumar, Associate Professor** for their guidance, valuable suggestions and encouragement in completing the project work within the stipulated time.

We would like to express our sincere thanks **Dr. Ch. Shiva Reddy, Principal, Prof. VVSSS Balaram,** Head of the Department of Information Technology, **Dr. Sunil Bhutada** & Project work coordinator, Sreenidhi Institute of Science and Technology, Hyderabad for permitting us to do our project work at **Sreenidhi Institute of science and technology**, Hyderabad.

Finally, we would also like to thank the people who have directly or indirectly helped us and parents and friends for their cooperation in completing the project work.

**B.Shankar lal (15311A1296)**
**Md.Naveed Ahmed (15311A1291)**
**L.Devender(15311A1297)**

# ABSTRACT

Face recognition is one of the most active research topics due to its potential applications in access control, automated crowd surveillance, law enforcement, information safety, multimedia communication, human-machine interface, etc. Compared to other biometric authentication technologies, face recognition has an obvious advantage that it does not need much cooperation from users. Just for the broad application foreground of face recognition, how to recognize fake faces is important. A weak biometric access control can be fooled with the help of a photograph of the legitimate user. This is the problem that live face detection intends to address

The problem of automatic face recognition is a composite task that involves detection and location of faces in a cluttered background, normalization, recognition and verication. Depending on the nature of the application, e.g. sizes of training and testing database, clutter and variability of the background, noise, occlusion, and nally speed requirements, some of the subtasks could be very challenging. Assuming that segmentation and normalization haven been done, we focus on the subtask of person recognition and verication and demonstrate the performance using a testing database of about many images

Present existing system of face recognition system are You're used to unlocking your door with a key, but maybe not with your face. As strange as it sounds, our physical appearances can now verify payments, grant access and improve existing security systems. Protecting physical and digital possessions is a universal concern which benefits everyone, unless you're a cybercriminal or a kleptomaniac of course. Facial biometrics are gradually being applied to more industries, disrupting design, manufacturing, construction, law enforcement and healthcare.

Proposed system of our project is developing a face recognition using LBPH(linear binary pattern of histogram),which perform recognition process . The related pakages used, numpy,os by IDLE tool using opencv, also using database called yale, which provide collection of data from which we have to recognize the image of give input, detection of face is done by HC process(Haas cascade), thinker is provide to add interface at the execution part, overall project is used for security management purpose in dailylife.

# List of Figures


| FIGURE.NO | FIGURE NAME | PAGE NO |
|---|---|---|
| 1 | Three different photos of albert | 1 |
| 2 | Successful recognition of faces | 14 |
| 3 | Architecture of system | 15 |
| 4 | Successful detection of single face | 26 |
| 5 | Successful detection of multi faces | 27 |
| 6 | Sample recognition using opencv | 32 |
| 7 | Eigen faces | 33 |
| 8 | Fisher face recognition | 33 |
| 9 | Recognised fisher faces | 34 |
| 10 | LBPH process of recognition | 35 |
| 11 | LBPH recognition in differ locations | 36 |
| 12 | Image correlation | 43 |
| 13 | Conversion of dimensions | 45 |
| 14 | Radius binary | 46 |
| 15 | Histogram representation of image | 46 |
| 16 | Transformed in dimentional image | 48 |
| 17 | Eigen faces | 51 |
| 18 | Different positions of faces | 52 |
| 19 | Average human face in gray scale | 53 |
| 20 | Area choosen for detection | 53 |
| 21 | Basis for dark intensity invariant sensitive template | 54 |
| 22 | Image detection using haar cascade | 55 |
| 23 | Detected faces for sample | 56 |
| 24 | Correlation of vertical and horizontal pixels | 58 |
| 25 | Average face in gray scale | 60 |
| 26 | Different mode of detection | 61 |
| 27 | System architecture | 62 |
| 28 | Class diagram | 63 |
| 29 | Use case diagram | 64 |
| 30 | Sequence diagram | 65 |

# List of Figures


| | | |
|---|---|---|
| 31 | Activity diagram | 66 |
| 32 | Component diagram | 67 |
| | | |

# List of Tables


| TABLE.NO | NAME | PAGE NO |
|---|---|---|
| 1 | Image of data presentation | 38 |
| 2 | Two poses of successful detection | 59 |
| 3 | Comparision of techniques | 73 |
| 4 | Observation table | 76 |
| 5 | Results | 77 |

# List of abbreviation

| | |
|---|---|
| LBPH | Linear binary pattern of histogram |
| PCA | Principal component analysis |
| HMM | Hidden markov model |
| LDA | Linear discriminant analysis |
| IDE | Integrated development environment |
| GPU | Graphical processing unit |
| DARPA | Defence advance research project agency |
| MATLAB | Matrix laboratory |

# INDEX


| S.NO | | PAGE NUMBER |
|---|---|---|
| 1 | Chapter 1 | |
| | Introduction | |
| | 1.1 Introduction | 1 |
| | 1.2 face recognition system | 1 |
| | 1.3 How face recognition works | 3 |
| | 1.4 Opencv | |
| | 1.4.1 Evolution of opencv | 4 |
| | 1.4.2 Advantages and disadvantages | 5 |
| | 1.5 Python | 6 |
| | 1.5.1 Evolution of python | 5 |
| | 1.5.2 Overview | 6 |
| | 1.5.3 Advantages and disadvantages | |
| | 1.6 Idle tool | |
| | 1.6.1 Evolution of Idle | 8 |
| | 1.6.2 Advantages and disadvantages | 9 |
| | 1.7 Objectives | 10 |
| | 1.8 Contribution of report | 11 |
| | 1.9 Report outline | 12 |
| 2 | Chapter 2 | |
| | Literature survey | |
| | 2.1 Introduction | 13 |
| | 2.2 face recognition | 13 |
| | 2.2.1 Introduction | 14 |
| | 2.2.2 Importance of face recognition | 16 |
| | 2.2.3 possible errors in face recognition | 17 |
| | 2.2.4 Applications | 18 |
| | 2.2.5 Features of face recognition | 203 |
| | 2.2.6 Existing system | 20 |
| | 2.2.7 Advantages and disadvantages | 23 |
| | 2.3 Face detection | |
| | 2.3.1 Introduction | 25 |
| | 2.3.2 Applications | 25 |
| | 2.3.3 Existing system | 26 |
| | 2.3.4 Advantages and disadvantages | 27 |
| | 2.3.5 Features | 29 |
| | 2.4 Software requirements | 30 |
| | 2.5 Conclusion | 38 |

3 Chapter 3
Implementation and analysis
3.1 Introduction 39
3.2 Face recognition algorithm 40
3.2.1 Template matching 40
3.2.2 LBPH algorithm 43
3.2.3 Understanding of eigen faces 46
3.2.4 Recognition 50
3.2.5 Pose invariant face recognition
3.3 Face detection algorithm 50
3.3.1 Introduction 51
3.3.2 Face recognition using Haar Cascades 51
3.3.3 Deformable template algorithm 53
3.3.4 Development of face search algorithm 56
3.3.5 manual face detection 56
3.4 System architecture 59
3.5 System design 61
3.6 Conclusion 62
67

4 Chapter 4
Comparison and observations 68
4.1 Introduction 68
4.2 Pseudo code 72
4.3 Comparison between all techniques 74
4.4 Database 75
4.5 Result and observation 81
4.6 Conclusion 82
5 Chapter 5
Contribution and scope
5.1 Future Extension 84
5.2 Summary of contribution 84
5.3 Conclusion 86

6 References 88
7 Appendix A:Software requirements 90
8 Case Study 94

# CHAPTER 1

# INTRODUCTION

## 1.1 Introduction

Face recognition is one of the most active research topics due to its potential applications in access control, automated crowd surveillance, law enforcement, information safety, multimedia communication, human-machine interface, etc. Compared to other biometric authentication technologies, face recognition has an obvious advantage that it does not need much cooperation from users. Just for the broad application foreground of face recognition, how to recognize fake faces is important. A weak biometric access control can be fooled with the help of a photograph of the legitimate user. This is the problem that live face detection intends to address

## 1.2 Face recognition system

Face recognition system and our own recognition ability is far more robust than any computer's can hope to be. We can recognise a familiar individual under very adverse lighting conditions, from varying angles or view points. Scaling differences (a face being near or far away), different backgrounds do not affect our ability to recognise faces and we can even recognise individuals with just a fraction of their face visible or even after several years have past. Furthermore, we are able to recognize the faces of several thousand individuals whom we have met during our lifetime as shown in figure 1.

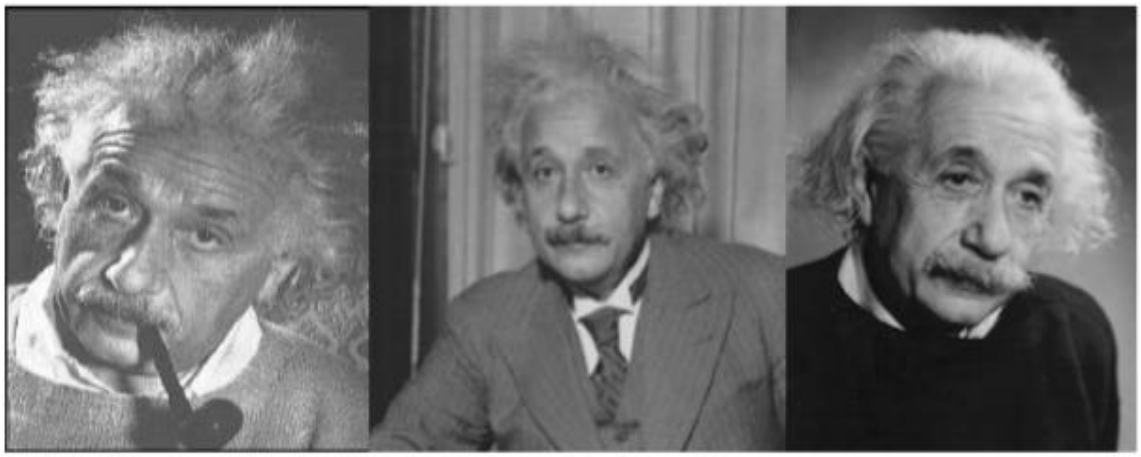


Figure 1Three different photographs of Albert Einstein, easily recognised as the eminent vary greatly from our own recollection

Just as in everyday life, most attention has been paid to the visual information conveyed by the human face. The tracking (and for registered “target” individuals, recognition) of

faces in video offers the prospect of annotating video archives with face-related metadata, and monitoring video streams for the appearance of specific or non-specific individuals. Furthermore, estimation of facial expressions, deformations associated with speech movements, and speaker and speech recognition, could potentially extend the scope of archive indexing

Face recognition is one of the most active research topics due to its potential applications in access control, automated crowd surveillance, law enforcement, information safety, multimedia communication, human-machine interface, etc. Compared to other biometric authentication technologies, face recognition has an obvious advantage that it does not need much cooperation from users. Just for the broad application foreground of face recognition, how to recognize fake faces is important. A weak biometric access control can be fooled with the help of a photograph of the legitimate user. This is the problem that live face detection intends to address

The problem of automatic face recognition is a composite task that involves detection and location of faces in a cluttered background, normalization, recognition and verication. Depending on the nature of the application, e.g. sizes of training and testing database, clutter and variability of the background, noise, occlusion, and nally speed requirements, some of the subtasks could be very challenging. Assuming that segmentation and normalization haven been done, we focus on the subtask of person recognition and verication and demonstrate the performance using a testing database of about many images

There are many face recognition algorithms in the academic field, part of the research results is relatively mature, and then a number of commercial face recognition systems and products are appeared. The Face. lt system and the Face VACS system, which are researched and developed by Identix company and Cognitech company, respectively, are very famous abroad. In the domestic, the face-recognition systems and the series products of face flux are eminent, which are researched by Beijing digital authen technology company and Hanwang technology company. It is a great significance to research the proper face recognition systems and products evaluation methods, carry out the test evaluation of the relevant systems and the products, and offer the open and fair evaluation conclusion of face recognition products. It can evaluate the merits of the algorithms, prompt them to improve and perfect gradually, shorten the R&D time,

and propel the large-scale application of face-recognition systems and products. Since 1994, more than 10 times of different scales of face recognition tests have been carried out in the international and domestic.

## 1.3 How facial recognition work

Facial recognition functions by examining the physical features of an individual's face to distinguish uniqueness from others. In order to verify someone's identity, the process can be broken down into three distinct steps: detection, unique faceprint creation, and finally, verification. In the first step, the technology captures an image of the individual's face and analyze's it to identify the user's face. Facial recognition can be a very useful and effective tool if all the key factors are satisfied.

One key factor that can strongly affect the effectiveness of facial recognition is lighting. In order for facial recognition to work, it's very important to have good lighting to clearly show all of the individual's facial features. If there are shadows or something blocking the camera's view, the system may not be able to recognize the face. It's also important that if these ke y factors are met, it will then select certain key factors such as eyes, eyebrows, nose shape and create a faceprint. The faceprint would then be compared to the key image in the system's database to verify the user. This is a quick and effective process that can identify an individual in seconds.

Though facial recognition can identify an individual in seconds, there are ways to trick and beat the system. For example, if a person is wearing disguise that covers or alters the face, the system would have a lot of difficulties identifying the individual. It's also possible that the picture in the system that is used to match with the individual is outdated, meaning that the system would likely be less accurate in identifying. Regardless of these obstacles, facial recognition is still one of the most accurate methods of identifying an individual and holds a lot of potential for the future.

## 1.4 OpenCV

OpenCV was started at Intel in 1999 by Gary Brad sky and the first release came out in 2000. Vadim Pisarev sky joined Gary Brad sky to manage Intel's russian software opencv team. In 2005, opencv was used on Stanley, the vehicle who won 2005 DARPA grand challenge. Later its active development continued under the support of Willow Garage, with Gary Brad

sky and Vadim Pisarev sky leading the project. Right now, opencv supports a lot of algorithms related to Computer Vision and Machine Learning and it is expanding day-by-day.

Currently opencv supports a wide variety of programming languages like C++, Python, Java etc and is available on different platforms including Windows, Linux, OS X, Android, iOS etc. Also, interfaces based on CUDA and Opencv are also under active development for high-speed GPU operations.

OpenCV-Python is the Python API of opencv. It combines the best qualities of opencv C++ API and Python language. Since openCV is an open source initiative, all are welcome to make contributions to this library. And it is same for this tutorial also. So, if you find any mistake in this tutorial (whether it be a small spelling mistake or a big error in code or concepts, whatever), feel free to correct it.

And that will be a good task for freshers who begin to contribute to open source projects. Just fork the opencv in git hub, make necessary corrections and send a pull request to opencv. opencv developers will check your pull request, give you important feedback and once it passes the approval of the reviewer, it will be merged to opencv. Then you become a open source contributor. Similar is the case with other tutorials, documentation etc.

As new modules are added to opencv-Python, this tutorial will have to be expanded. So those who knows about particular algorithm can write up a tutorial which includes a basic theory of the algorithm and a code showing basic usage of the algorithm and submit it to openCV.

### 1.4.1 Evolution of OpenCV

This is the collection of OpenCV Evolution proposal. OpenCV was started at Intel in 1999 by Gary Brad sky and the first release came out in 2000. Vadim Pisarev sky joined Gary Brad sky to manage Intel's russian software openCV team. In 2005, openCV was used on Stanley, the vehicle who won 2005 DARPA grand challenge. Later its active development continued under the support of Willow Garage, with Gary Brad sky and Vadim Pisarev sky leading the project. Right now, openCV supports a lot of algorithms related to Computer Vision and Machine Learning and it is expanding day-by-day.

OpenCV is quite mature library, but still there are many small and big things that the core team and the community would like to change or would like to add. For small and not so small things one can submit bug reports or feature requests, but make sure to add the evolution label so that it shows up on the evolution issue list. One can submit a pull request with the ready solution.

But what to do with big things that may take several man-weeks or even several man-months to implement and that can significantly affect the library and our users? How to document such big ideas, bring it up to some public discussion? For this purpose we've established the OpenCV Evolution mechanism, similar to Python PEP or Swift Evolution.

### 1.4.2-Advantages and Disadvantages of openCV

**Advantages:**

- First and foremost, OpenCV is available free of cost
- Since OpenCV library is written in C/C++ it is quite fast
- Low RAM usage (approx 60–70 mb)
- It is portable as OpenCV can run on any device that can run C

**Disadvantages:**

- OpenCV does not provide the same ease of use when compared to MATLAB
- OpenCV has a flann library of its own. This causes conflict issues when you try to use OpenCV library with the PCL library

## 1.5 Python

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.Van Rossum led the language community until July 2018.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python features a comprehensive standard library, and is referred to as "batteries included".

Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open-source software[30] and has a community-based development model. Python and CPython are managed by the non-profit Python Software Foundation.

### 1.5.1 Evolution of python

The incredibly talented Guido Van Rossum developed Python during the late eighties till the early nineties. Guido Van Rossum developed Python at the National Research Institute for Mathematics and Computer Science. This research institute is situated in the Netherlands. Guido Van Rossum while developing this language actually took some inspiration from many other languages. He also further derived from a number of languages. Some of the languages from which he took inspiration or derived from are ABC, C++, C, SmallTalk, Modula-3, Unix shell, and many other scripting or programming languages

You as a user of Python should also be aware of the fact that this programming language has a copyright. The source code of Python is available under the GNU General Public License (GNP) just like it is available for Perl. You should also be aware of the fact that Python is maintained by a group of core development team members present at the institute. You will acquire more information as you move forward in this Python introduction tutorial. However, this does not mean that Guido Van Rossum is out of the picture as he still plays a vital role in the whole maintaining process.

### 1.5.2 Overview

As we mentioned before python programming language that it is a kind of high-level programming language that is also interpreted, object-oriented, and interactive. Unlike many other programming languages that often punctuations, Python uses many English language keywords. Python further stands apart in terms of its syntactical construction as it has a lesser amount of syntactical construction than many other programming languages. There are a few keywords that we mentioned above regarding the Python programming language and we'll be discussing those keywords in depth now. The Python programming language is an Open Source, cross-platform, high level, dynamic, interpreted language.

The Python 'philosophy' emphasises readability, clarity and simplicity, whilst maximising the power and expressiveness available to the programmer. The ultimate compliment to a Python programmer is not that his code is clever, but that it is elegant. For these reasons Python is an excellent 'first language', while still being a powerful tool in the hands of the seasoned and cynical programmer.

- **Interpreted**

This keyword basically means that Python as a programming language is processed by the interpreter at the specific runtime. This implies that you can execute this language before even compiling your entire program. This particular feature makes this language similar to PHP and PERL. You would get to know about many other features of this language in this Python introduction tutorial.

- **Interactive**

One of the best features of Python is that it allows you to write your program by interacting with the interpreter directly while you are still sitting at a Python prompt.

- **Object Oriented**

Python as a programming language supports a more object-oriented technique or style of programming.

Python is a really good programming language for any software developer who might still be at the beginning stage. One of the best things about this language is that it allows you or provides you with a sort of a framework that would support you while you might be learning various applications like www browsers, simple text processing, or even games. There are many other uses of this language that you will get to know in this Python introduction tutorial. There are various features and information about Python language that can come quite in handy to all its users. And some of those features of Python are mentioned below.

Python is a programming language that is easier to learn in comparison to other programming languages. This feature enables a lot of students to learn this programming language relatively easily ,The codes of Python are more defined and visible which makes them easier to read,The source code of Python is also quite easy to maintain in comparison to other programming languages,The Python library is quite broad and is also portable and compatible across other platforms too like UNIX, Macintosh, and Windows

The interactive mode of Python allows users to debug snippets of codes and run interactive testing,Python further enables its users to add modules of a lower level to its interpreter python provides a better structure and support to many other large programs These are all the major features of Python. However, you will gradually become aware of many other features of Python as you become more comfortable in using the whole platform and you move forward in this Python introduction tutorial.

### 1.5.3 Advantages and Disadvantages of Python

**Advantages:**

The Python language has diversified application in the software development companies such as in gaming, web frameworks and applications, language development, prototyping, graphic design applications, etc. This provides the language a higher plethora over other programming languages used in the industry. Some of its advantages are-

- Extensive Support Libraries

It provides large standard libraries that include the areas like string operations, Internet, web service tools, operating system interfaces and protocols. Most of the highly used programming tasks are already scripted into it that limits the length of the codes to be written in Python.

- Integration Feature

Python integrates the Enterprise Application Integration that makes it easy to develop Web services by invoking COM or COBRA components. It has powerful control capabilities as it calls directly through C, C++ or Java via Jython. Python also processes XML and other markup languages as it can run on all modern operating systems through same byte code.Improved

- Programmer's Productivity

The language has extensive support libraries and clean object-oriented designs that increase two to ten fold of programmer's productivity while using the languages like Java, VB, Perl, C, C++ and C#.

**Disadvantages**

- Difficulty in Using Other Languages

The Python lovers become so accustomed to its features and its extensive libraries, so they face problem in learning or working on other programming languages. Python experts may see the declaring of cast "values" or variable "types", syntactic requirements of adding curly braces or semi colons as an onerous task.

- Weak in Mobile Computing

Python has made its presence on many desktop and server platforms, but it is seen as a weak language for mobile computing. This is the reason very few mobile applications are built in it like Carbonnelle.

- Gets Slow in Speed

Python executes with the help of an interpreter instead of the compiler, which causes it to slow down because compilation and execution help it to work normally. On the other hand, it can be seen that it is fast for many web applications too.

- Run-time Errors

The Python language is dynamically typed so it has many design restrictions that are reported by some Python developers. It is even seen that it requires more testing time, and the errors show up when the applications are finally run.

## 1.6 Idle tool

An integrated development environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development. An IDE normally consists of at least a source code editor, build automation tools, and a debugger. Some IDEs, such as NetBeans and Eclipse, contain the necessary compiler, interpreter, or both; others, such as Sharp Develop and Lazarus, do not.

The boundary between an IDE and other parts of the broader software development environment is not well-defined; sometimes a version control system or various tools to simplify the construction of a graphical user interface (GUI) are integrated. Many modern IDEs also have a class browser, an object browser, and a class hierarchy diagram for use in object-oriented software development.

### 1.6.1 Evolution of IDLE

IDEs initially became possible when developing via a console or terminal. Early systems could not support one, since programs were prepared using flowcharts, entering programs with punched cards (or paper tape, etc.) before submitting them to a compiler. Dartmouth BASIC was the first language to be created with an IDE (and was also the first to be designed for use while sitting in front of a console or terminal).[citation needed] Its IDE (part of the Dartmouth Time Sharing System) was command-based, and therefore did not look much like the menu-driven, graphical IDEs popular after the advent of the Graphical User Interface. However it

integrated editing,file management, compilation, debugging and execution in a manner consistent with a modern IDE. See also Structured Programming Facility from IBM (1974).

Maestro I is a product from Softlab Munich and was the world's first integrated development environment for software. Maestro I was installed for 22,000 programmers worldwide. Until 1989, 6,000 installations existed in the Federal Republic of Germany. Maestro was arguably the world leader in this field during the 1970s and 1980s. Today one of the last Maestro I can be found in the Museum of Information Technology at Arlington.

One of the first IDEs with a plug-in concept was Softbench. In 1995 Computerwoche commented that the use of an IDE was not well received by developers since it would fence in their creativity.

One aim of the IDE is to reduce the configuration necessary to piece together multiple development utilities, instead it provides the same set of capabilities as one cohesive unit. Reducing setup time can increase developer productivity, especially in cases where learning to use the IDE is faster than manually integrating and learning all of the individual tools. Tighter integration of all development tasks has the potential to improve overall productivity beyond just helping with setup tasks. For example, code can be continuously parsed while it is being edited, providing instant feedback when syntax errors are introduced. Allowing developers to debug code much faster and easier with an IDE.

Some IDEs are dedicated to a specific programming language, allowing a feature set that most closely matches the programming paradigms of the language. However, there are many multiple-language IDEs.

While most modern IDEs are graphical, text-based IDEs such as Turbo Pascal were in popular use before the widespread availability of windowing systems like Microsoft Windows and the X Window System (X11). They commonly use function keys or hotkeys to execute frequently used commands or macros.

### 1.6.2 Advantages and Disadvantages of idle

**Advantages**

- You save a good amount of bandwidth by not having to poll the email server email couple of minutes to check for a new email. Server will notify you as soon as email arrives.

- - There is almost no lag in email sync. Server notifies the client as soon as a new email arrives. Better experience for the mail client users.

**Disadvantages**

- You can receive new email notification only for the IMAP folder to which you are currently connected. So if you receive a new mail in some other imap folder, the client will not be notified. The solution is to keep a connection open to all possible imap folders that you need to monitor. This means a lot of tcp connections will have to kept open from the client to server and all mail servers have some kind of upper limit to number of simultaneous connections that you can open at once.

## 1.7 Objectives

- Fully automated face detection of frontal view faces is implemented using a local binary pattern histograms algorithm relying on the image invariants of human faces
- This was chosen because a similar neural-network based face detection model would have needed far too much training data to be implemented and would have used a great deal of computing time.
- A template matching based technique was implemented for face recognition. This was because of its increased recognition accuracy when compared to geometrical features based techniques and the fact that an automated geometrical features based technique would have required complex feature detection pre-processing.
- The main thing in project is usage of Yale database which provide collection of images through which we perform LBPH operation and recognition is performed

## 1.8 Contribution of report

In computer science, face recognition is basically the task of recognizing a person based on its facial image. It has become very popular in the last two decades, mainly because of the new methods developed and the high quality of the current videos/cameras.

Although it sounds like a very simple task for us, it has proven to be a complex task for a computer, as it has many variables that can impair the accuracy of the methods, for example: illumination variation, low resolution, occlusion, amongst other.

We implemented project using Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number

First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let's see the LBPH computational steps.

Applying the LBP operation: The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters radius and neighbors

Each eigen created is used to represent each image from the training dataset. So, given an input image are choosen from yale database , we perform the steps again for this new image and creates a histogram which represents the image.

We can then use a threshold and the 'confidence' to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized if the confidence is lower than the threshold defined,at last by using tkinter we made interface for the execution of output by providing required pages for clear understanding

In this venture we have utilized OpenCV innovation, IDLE tool related pakages

## 1.9 Report outline

chapter 1,Introduction: introduction to face recognition, face detection, how facial recognition works, OpenCV ,evolution of OpenCV, python, evolution of python, IDLE tool, evolution of tool, contribution of report, objectives, advantages and disadvantages, chapter 2,Literature survey: A writing study, or writing audit, implies that you read and give an account of what the writing in the field needs to state about your theme or subject. There might be a great deal of writing on the theme or there might be a bit, chapter 3,implementation and analysis: introduction of face recognition, algorithms of recognition and detection python, importance, application, architecture, list of existing system, features, advantages and disadvantages, sysem design, architecture, conclusion,chapter 4,comparision and observation:

introduction, pseudo code, comparision of techniques, result and observation, conclusion, chapter 5,contribution and scope: future extension, summary of contribution, conclusion, chapter 6,reference,chapter 7,Appendix:software details, Case study.

# CHAPTER 2

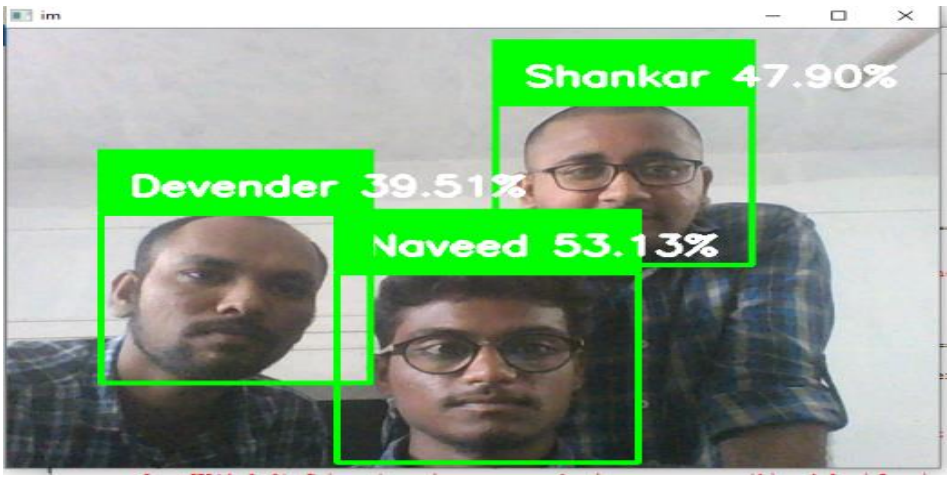
# LITERATURE SURVEY

## 2.1 Introduction

As one of the most successful applications of image analysis and understanding, face recognition has recently received significant attention, especially during the past several years. At least two reasons account for this trend: the first is the wide range of commercial and law enforcement applications, and the second is the availability of feasible technologies after 30 years of research. Even though current machine recognition systems have reached a certain level of maturity, their success is limited by the conditions imposed by many real applications. For example, recognition of face images acquired in an outdoor environment with changes in illumination and/or pose remains a largely unsolved problem. In other words, current systems are still far away from the capability of the human perception system. This paper provides an up-to-date critical survey of still- and video-based face recognition research. There are two underlying motivations for us to write this survey paper: the first is to provide an up-to-date review of the existing literature, and the second is to offer some insights into the studies of machine recognition of faces. To provide a comprehensive survey, we not only categorize existing recognition techniques but also present detailed descriptions of representative methods within each category.

## 2.2 Face recognition

Face recognition is an easy task for humans. Experiments have shown, that even one to three day old babies are able to distinguish between known faces. So how hard could it be for a computer? It turns out we know little about human recognition to date. Are inner features (eyes, nose, mouth) or outer features (head shape, hairline) used for a successful face recognition? How do we analyze an image and how does the brain encode it? It was shown by David Hubel and Torsten Wiesel, that our brain has specialized nerve cells responding to specific local features of a scene, such as lines, edges, angles or movement. Since we don't see the world as scattered pieces, our visual cortex must somehow combine the different sources of information into useful patterns. Automatic face recognition is all about extracting those meaningful features from an image, putting them into a useful representation and performing some kind of classification on them as shown in Figureure 2.




Figureure 2 successfully recognized faces

Face recognition based on the geometric features of a face is probably the most intuitive ap proach to face recognition. One of the first automated face recognition systems was described in [Kanade73]: marker points (position of eyes, ears, nose, ...) were used to build a feature vector (distance between the points, angle between them, ...). The recognition was performed by calculating the euclidean distance between feature vectors of a probe and reference image a 22-dimensional feature vector was used and experiments on large datasets have shown, that geometrical features alone my not carry enough information for face recognition.

Recently various methods for a local feature extraction emerged. To avoid the high-dimensionality of the input data only local regions of an image are described, the extracted features are (hopefully) more robust against partial occlusion, illumation and small sample size. it's still an open research question what's the best way to preserve spatial information when applying a local feature extraction, because spatial information is potentially useful information.

### 2.2.1 Introduction

Face recognition concept of feature extraction and detection, is a small capacity for human beings. Human have developed this skill to correctly and instantaneously recognize things around us after millions of years of evolution. The necessitate for machine intrusion in face recognition to create the whole process gives rise to Automated Face Recognition (AFR) that simulates the Human Vision System (HVS). The past few decades have seen AFR receive immense attention due to its myriad applications in fields of security and surveillance. Implementations in computers are much more complex though not impossible. Image processing (in this specific case, leading to face recognition) by computers usually takes place

in this order and analysis techniques. It has a wide number of applications including security, law enforcement, person verification, Internet communication, Pattern Recognition and computer entertainment on was carried out in Florida. Recent years have seen, law enforcement, person Face recognition has two main steps: feature extraction and classification. Image processing technique has been applied to evaluate feature from image database and there are some classification techniques that are applied to recognize the unknown face image. The overview of current system is demonstrated in Figureure 3.

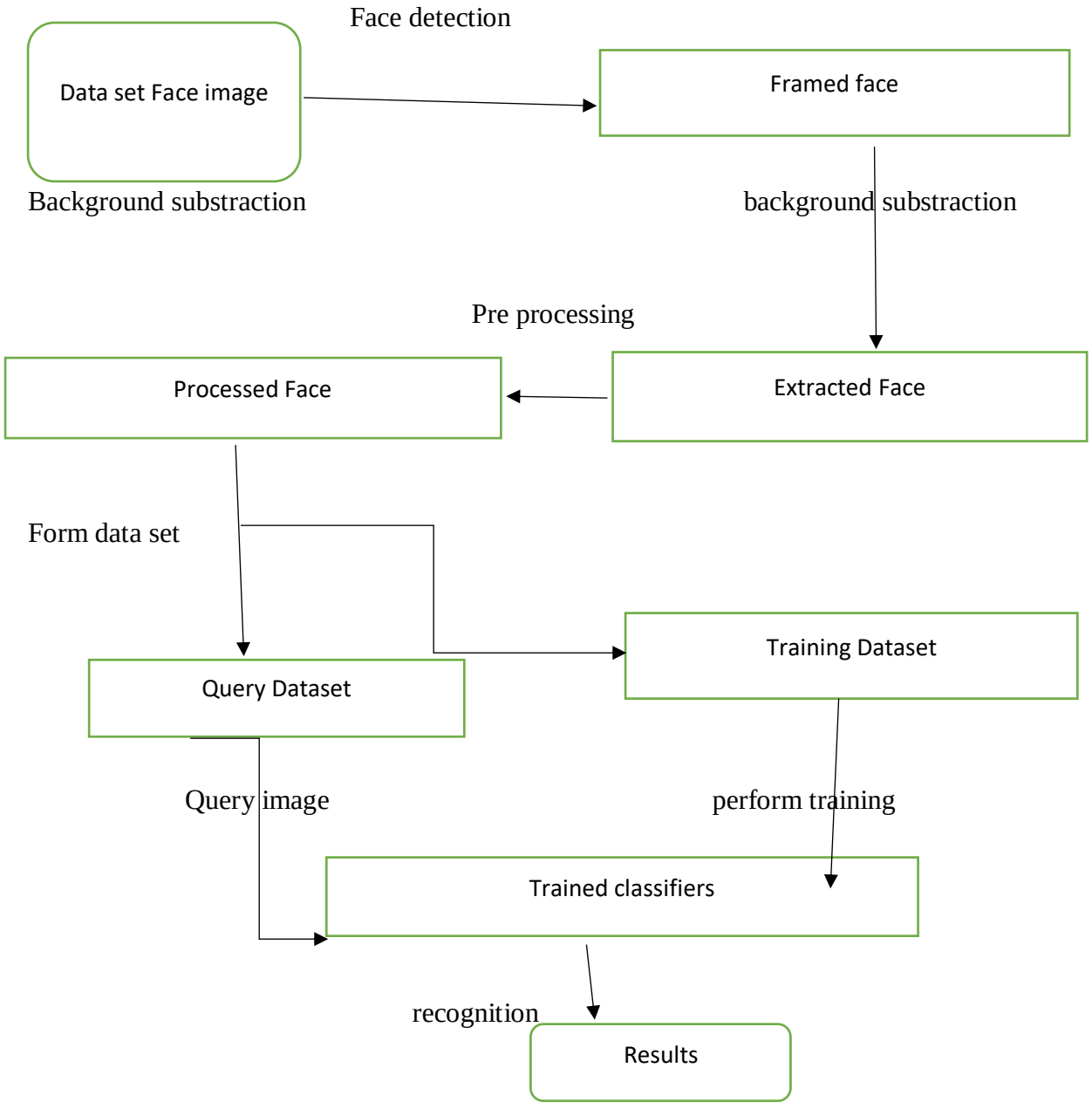



Figure 3 Architecture of system

### 2.2.2 Importance of face recognition

Facial Recognition, Identification, and Classification**:**To recognize an object certain steps must be taken. Information is received through the retina in the form of light. Visual processing occurs to organize the data by determining size, shape, contoured edges, and surface so that the information can be compared to other representations of objects in memory until recognition occurs [1].

While first-order relational information is used in object recognition, second-order relational information is needed for facial recognition. If an individual only applied first-order relational information to facial recognition it would give him or her a basic idea of what features were there and where they were located in relationship to each other. This would not be enough to distinguish one person from another as everyone has the same basic features. Second-order relational information takes the information from first-order relational information and compares it to an average face based on information each individual has accumulated on faces

When it comes to facial recognition the most significant information is the second-order relational information. Unlike objects, that can be taken apart and still recognized, faces are stored in memory as a whole image. If only a partial image is available, or if the image is turned upside down, facial recognition becomes more difficult (Diamond & Carey, 1986). According to Vecera, n.d., the task of facial recognition is made more complicated by the emotion displayed by the individual. The brain must not only recognize the face itself but also take into consideration the emotional context. This added element brings interpersonal interaction between the person doing the viewing as well as the person being viewed in to play, which adds a social element to the process.

The foundation of the arguments for multisystem memory could be deemed debatable. It assumes that some processes only apply to facial recognition when there could be other objects that have similar features. If the cognitive processes involved are not explicitly for facial recognition a single system may be all that is needed for recognition of both faces and objects. When other aspects are taken into consideration, like judgment, knowledge, and experience, the neural responses and behavior patterns for both facial and object recognition are similar .

### 2.2.3 Possible Errors in Facial Recognition

- **Misidentification**

Misidentification could occur because of a number of different reasons. One of these is unconscious transference. Basically, unconscious transference refers to being unable to distinguish between a person who is familiar in general and a person who is familiar for a specific reason. For example, someone who witnessed a crime may identify someone who looks familiar to him or her because he or she was seen at some point during the day as opposed to the person who committed the crime [2][3].

- **Self-Recognition**

Recognizing faces occurs in the fusiform face area. People who have damage in this area are unable to recognize themselves. This condition is known as prosopagnosia. For these, without this condition, one would think that self-knowledge would include not only the things we like, the things we do not like, and things we have accomplished over our lifetime, but also, knowledge of our facial features. However, studies have shown that knowledge of our own face is different from other types of knowledge. Evidence derived from brain imaging and case studies has shown that an area of the temporal lobe, known as the fusiform face area, is specified for facial recognition. This area shows more activity during brain imaging when an individual is attempting to recognize faces. The right prefrontal cortex has been shown to be more active when tasks involving the self, including self-recognition, are being performed.

when a warning should be applied. The biggest drawback associated with EEG is the difficulty in obtaining recordings under natural driving conditions, making it a somewhat unrealistic option for the detection of fatigue. Due to illumination variation, all this traditional method has some issues. So, the proposed system uses new developed pre-processing and eyes detection methods which are explained

The key drawbacks of the existing systems are

• Sun light can interfere with IR illumination

• Some devices use the external memory to store the frame

• The existing device gives more percentage false alarm

• The existing methods such as PERCLOSURE,AVGCLOSURE are not reliable

• Many existing systems use intrusive method

• The traditional devices are not reliable at day or night time traveling.

All the above drawbacks are removed in the proposed system. The new device is reliable and work all the time day and night.

### 2.2.4 Applications

**Facial recognition is being used in many businesses**

You're used to unlocking your door with a key, but maybe not with your face. As strange as it sounds, our physical appearances can now verify payments, grant access and improve existing security systems. Protecting physical and digital possessions is a universal concern which benefits everyone, unless you're a cybercriminal or a kleptomaniac of course. Facial biometrics are gradually being applied to more industries, disrupting design, manufacturing, construction, law enforcement and healthcare. How is facial recognition software affecting these different sectors, and who are the companies and organisations behind its development?

- **Payments**

It doesn't take a genius to work out why businesses want payments to be easy. Online shopping and contactless cards are just two examples that demonstrate the seamlessness of postmodern purchases. With FaceTech, however, customers wouldn't even need their cards. In 2016, MasterCard launched a new selfie pay app called MasterCard Identity Check. Customers open the app to confirm a payment using their camera, and that's that. Facial recognition is already used in store and at ATMs, but the next step is to do the same for online payments. Chinese ecommerce firm Alibaba and affiliate payment software Alipay are planning to apply the software to purchases made over the Internet.

- **Access and security**

As well as verifying a payment, facial biometrics can be integrated with physical devices and objects. Instead of using passcodes, mobile phones and other consumer electronics will be accessed via owners' facial features. Apple, Samsung and Xiaomi Corp. have all installed FaceTech in their phones. This is only a small scale example, though. In future, it looks like consumers will be able to get into their cars, houses, and other secure physical locations simply by looking at them. Jaguar is already working on walking gait ID – a potential parallel to facial

recognition technology. Other corporations are likely to take advantage of this, too. Innovative facial security could be especially useful for a company or organisation that handles sensitive data and needs to keep tight controls on who enters their facilities.

- **Criminal identification**

If FaceTech can be used to keep unauthorised people out of facilities, surely it can be used to help put them firmly inside them. This is exactly what the US Federal Bureau of Investigation is attempting to do by using a machine learning algorithm to identify suspects from their driver's licences. The FBI currently have a database which includes half of the national population's faces. This is as useful as it is creepy, giving law enforcers another way of tracking criminals across the country. AI equipped cameras have also been trialled in the UK to identify those smuggling contraband into prisons[4].

- **Healthcare**

Instead of recognising an individual via FaceTech, medical professionals could identify illnesses by looking at a patient's features. This would alleviate the ongoing strain on medical centres by slashing waiting lists and streamlining the appointment process. The question is, would you really want to find out you had a serious illness from a screen? If it's a choice between a virtual consultation or a month long wait for an appointment, then maybe so. Another application of facial biometrics within healthcare is to secure patient data by using a unique patient photo instead of passwords and usernames.

With a predicted worth of $15 billion by 2025, biometrics is an industry worth watching. It's clear that facial biometrics are a helpful tool for finance, law enforcement, advertising and healthcare, as well as a solution to hacking and identity theft. Of course, FaceTech is by no means foolproof. Gaining access to possessions using physical traits could even be counterintuitive for security. A face, as social robots like Nadine have shown us, is easily replicated. And when it comes to public adoption, some people are reluctant to switch to contactless cards, let alone abandon them completely. For the most part, though, facial recognition technology seems to be encouraging a more seamless relationship between people, payments and possessions.

### 2.2.5 Features of face recognition

For example, face recognition technology can do things like:

- Find photos you're in but haven't been tagged: This is a great feature for those who careto find their photos on the web.
- Help protect you from strangers using your photo: Thank you Facebook, this feature is the most interesting among them all.
- If you have been a victim of facial theft, you will call Zuckerberg right now to say thank you.
- There too many fake facebook accounts are existing on the platform, most of these profiles are created by fraudsters and imposters. The make use of well and highly personalty to create their profile. And they use the profile to find people who they will fraud their money. Women and old men are the victims of this scam activities.
- Tell people with visual impairments who are in your photo or video: The facial recognition can detect your face even on video, isn't that some good enough to make you shout wow
- and let you know when you might appear in photos or videos but haven't been tagged.
- You'll only be notified about photos that you're in the audience for.

### 2.2.6 Existing Systems

- **Facial Recognition**

Some facial recognition algorithms identify faces by extracting landmarks or features from an image. For example, an algorithm may analyze the relative position, size, and/or shape of the eyes, nose, cheekbones and jaw. These features are then used to search for other images with matching features. Other algorithms normalize a gallery of face images and then compress the face data, only saving the data in the image that is useful for face detection. A probe image is then compared with the face data. One of the earliest successful systems is based on template matching techniques applied to a set of salient facial features, providing a sort of compressed face representation[5][6]

- **Facial Recognition methods**

Recognition algorithms can be divided into two main approaches: i) geometric which looks at distinguishing features (feature based) and ii) photometric which is a statistical

approach that distill an image into values and compares the values with templates to eliminate variances (viewbased). Popular recognition algorithms Principal Component Analysis and Linear Discriminate Analysis are based on geometric approach. Elastic Bunch Graph Matching and the Hidden Markov model are based on statistical approach.

- **Principal Component Analysis (PCA)**

PCA involves a mathematical procedure that transforms a number of possibly correlated variables into a smaller number of uncorrelated variables Please purchase PDF Split-Merge on www.verypdf.com to remove this watermark. 5 called principal components. The first principal component accounts for as much of the variability in the data as possible and each succeeding component accounts for as much of the remaining variability as possible. PCA is mostly used as a tool in exploratory data analysis and for making predictive models. PCA involves the calculation of the eigen value decomposition of a data covariance matrix or singular value decomposition of a data matrix. PCA is the simplest of the true eigenvector-based multivariate analysis. Often, its operation can be thought of as revealing the internal structure of the data in a way which best explains the variance in the data. If a multivariate dataset is visualized as a set of coordinates in a high-dimensional data space (one axis per variable), PCA supplies the user with a lower-dimensional picture, a "shadow" of this object when viewed from its (in some sense) most informative viewpoint.

- **Elastic bunch graph mapping**

Elastic Matching (EM) is one of the pattern recognition techniques incomputer science. It is also known as deformable template, flexible matching, or nonlinear template matching. Please purchase PDF Split-Merge on www.verypdf.com to remove this watermark. 6 Elastic matching can be defined as an optimization problem of two dimensional warping specifying corresponding pixels between subjected images.

- **Hidden Markov Model (HMM)**

A hidden Markov model (HMM) is a statistical model in which the system being modeled is assumed to be a Markov process with unobserved state. An HMM can be considered as the simplest dynamic Bayesian network. Hidden Markov models are especially known for their applications in temporal pattern recognition such as speech, handwriting, gesture recognition, part-of-speech tagging, partial discharges and bioinformatics[7].

- **Image Enhancement**

The aim of image enhancement is to improve the interpretability orpe rception of information in images for human viewers or to provide `better' input for other automated image processing techniques. Image enhancement technique scan be divided into two broad

- **Histogram equalization**

Histogram equalization is a method in image processing of contrast adjustment using the image's histogram.

This method usually increases the global contrast of many images, especially when the usable data of the image is represented by close contrast values. Through this adjustment, the intensities can be better distributed on the histogram. This allows for areas of lower local contrast to gain a higher contrast without affecting the global contrast. Histogram equalization accomplishes this by effectively spreading out the most frequent intensity values[8][9].

The method is useful in images with backgrounds and foregrounds that are both bright or both dark. In particular, the method can lead to better views of bone structure in x-ray images and to better detail in photographs that are over or underexposed. A key advantage of the method is that it is a fairly straight forward technique and an invertible operator. So in theory, if the histogram equalization function is known, then the original histogram can be recovered. The calculation is not computationally intensive. A disadvantage of the method is that it is in discriminate. It may increase the contrast of background noise, while decreasing the usable signal.

- **Histogram equalization of color images**

The above histogram equalization describes on a gray scale image. How ever it can also be used on color images by applying the same method separately to the Red, Green and Blue components of the RGB color values of the image. Still, it should be noted that applying the same method on the Red, Green, and Blue components of an RGB image may yield dramatic changes in image's color balance since the relative distributions of the color channels change as a result of applying the algorithm. However if the image is first converted into another color space, Lab color space, or HSL/HSV color space in particular, then the algorithm can be applied to the luminance or value channel without resulting in changes to the hue and saturation of the image.

### 2.2.7 Advantages and Dis advantages of face recognition

**Advantages**

**The Improvement of Security Level**

As we said in the first paragraph, a face biometric system greatly improves your security measures. All corporation's premises would be protected since you'll be able to track both the employees and any visitors that come into the area. Anyone who doesn't have access or permission to be there will be captured by the recognition system that alerts you instantly about the trespassing.

As an example, let's take a 24/7 drugstore. Any owner prefer to keep their money and clients safe, avoiding unpleasant troubles with difficult visitors. When you have a FRT in place, you'd be instantly alerted as soon as the wanted or suspicious character arrives. Which leads to a significant reduces of expenses one usually spends on security staff.

- **Easy Integration Process**

Most of the time, integratable facial recognition tools work pretty flawlessly with the existing security software that companies have installed. And they're also easy to program for interaction with a company's computer system.

Why is it great for business? Well, you won't need to spend additional money and time on redeveloping your own software to make it suitable for FRT integration. Everything will be already adaptable.

- **Full Automation**

Instead of manual recognition, which is done by security guards or the official representatives outside of company's premises, the facial recognition tech automates the identification process and ensures its flawlessness every time without any haltings. You won't even need an employee to monitor the cameras 24/7.

Automation means convenience and reduces the expenses too. Therefore, any entrepreneur would be fond of the fact that image identification systems are fully automated.

- **Forget the Time Fraud**

One of the big benefits that facial recognition technology companies offer is the time attendance tracking that allows excluding the time fraud among the workers. No more buddy favours from securities for staff members, since everyone now has to pass a face scanning devices to check-in for work. And the paid hours begin from this moment till the same check-

out procedure. And the process will be fast due to the fact that employees don't have to prove their identities or clock in with their plastic cards.

It's crucial for businessmen to trust their workers but keep an eye on them just in case. Unfortunately, time fraud is one of the most common violations of the work ethics, but the facial identification tech will spare you a headache regarding this matter

**Disadvantages**

- **Processing & Storing**

Storages are like gold in a digital world since you have to save huge amounts of data for future usage. Even though you get HD-video in a pretty low resolution, it still requires a significant space. Just as the high-quality image visuals. There is no need to process every video's frame – it's an enormous waste of resources. That's why most of the time only a fraction (around 10 – 25%) is actually being put through an FRT.

Professional agencies use whole clusters of computers in order to minimize total processing time. But every added computer means considerable data transfer via network, which can be influenced by input-output limitations that lower a processing speed.

- **Image Size & Quality**

It's obvious that a facial recognition is a super advanced software that requires HQ digital cameras for algorithms to operate accurately. A face-detection system captures a face in the photo or screen-shot from a video, then the relative size of that face image will be compared with the size of enrolled one. So, the photo's quality here affects the whole face recognition process, how well it would be done. Imagine, the already small size picture is coupled with a distance that was between a target and a CCTV… What proportions will the detected face have? No more than 100×200 pixels.

Pretty hard to get a clear identity in such case. What's more, scanning a photo for varying face sizes is a processor-intensive task. Most systems allow identification of a face-size range to eliminate false recognition and speed up image processing. But the initial investment in such face tracking software is not a cheap one, however, it will pay off in no time.

- **Surveillance Angle**

The identification process is also under a great pressure of the surveillance angle that was responsible for the target's face capturing. To enroll a face through the recognition software, the multiple angles are being used – profile, frontal, 45-degree, etc. But to generate a clear

template for the face, you'll need nothing less than a frontal view. The higher resolution photo has and the more direct its angle is (goes for both enrolled and compared images) the more accurate resulting matches would be.

Then, there are also troubles with such things as facial hair or sunglasses. One can still fool the FRT with a suddenly appeared or removed beard, same goes for obscuring face's parts with glasses or masks. To avoid such failures, the databases must be regularly updated with the most up-to-date images.

## 2.3 Face detection

Face detection is a computer technology being used in a variety of applications that identifies human faces in digital images. Face detection also refers to the psychological process by which humans locate and attend to faces in a visual scene

Face-detection algorithms focus on the detection of frontal human faces. It is analogous to image detection in which the image of a person is matched bit by bit. Image matches with the image stores in database. Any facial feature changes in the database will invalidate the matching process[10].

### 2.3.1 Introduction

Face detection is the first step of face recognition as it automatically detects a face from a complex background to which the face recognition algorithm can be applied. But detection itself involves many complexities such as background, poses, illumination etc. There are many approaches for face detection such as, colour based, feature based (mouth, eyes, nose), neural network. The approach studied and applied in this thesis is the skin colour based approach. Thealgorithm is pretty robust as the faces of many people can be detected at once from an image consisting of a group of people.

Face Detection has been one of the hottest topics of computer vision for the past few years. This technology has been available for some years now and is being used all over the place. From cameras that make sure faces are focused before you take a picture, to Facebook when it tags people automatically once you upload a picture (before you did that manually remember?).Or some shows like CSI used them to identify "bad guys" from security footage (ENHANCE! – then insert crime pun) or even unlocking your phone by looking at it!

You look at your phone, and it extracts your face from an image (the nerdy name for this process is face detection). Then, it compares the current face with the one it saved before during training and checks if they both match (its nerdy name is face recognition)and, if they do, it ulocks itself as clearly shown in Figure 4, 5.

Figure 4 face detected successfully

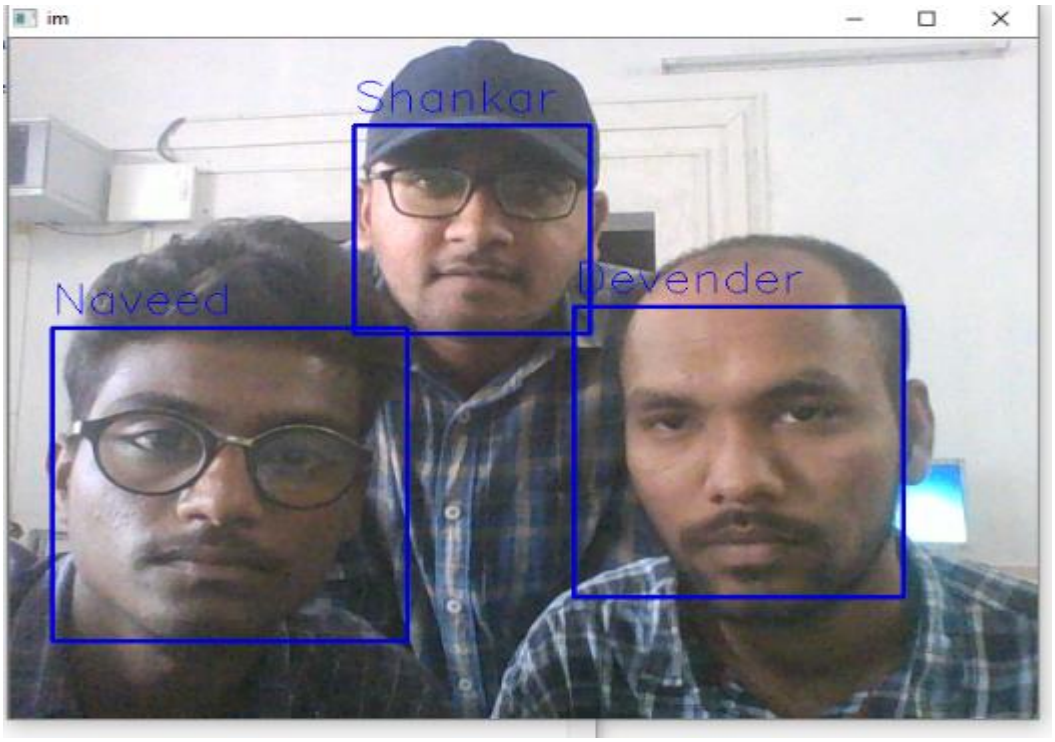



Figure 5 successful detection of multi faces

### 2.3.2 Applications

- **Facial motion capture**

Is the process of electronically converting the movements of a person's face into a digital database using cameras . This database may then be used to produce CG (computer graphics) computer animation for movies, games, or real-time avatars. Because the motion of

CG characters is derived from the movements of real people, it results in more realistic and nuanced computer character animation than if the animation were created manually.

A facial motion capture database describes the coordinates or relative positions of reference points on the actor's face. The capture may be in two dimensions, in which case the capture process is sometimes called "expression tracking", or in three dimensions. Two dimensional capture can be achieved using a single camera and low cost capture software such as Zign Creations' Zign Track. This produces less sophisticated tracking, and is unable to fully capture three-dimensional motions such as head rotation. Three-dimensional capture is accomplished using multi-camera rigs or laser marker system. Such systems are typically far more expensive, complicated, and time-consuming to use. Two predominate technologies exist; marker and markerless tracking systems.

Facial Motion Capture is related to body motion capture, but is more challenging due to the higher resolution requirements to detect and track subtle expressions possible from small movements of the eyes and lips. These movements are often less than a few millimeters, requiring even greater resolution and fidelity and different filtering techniques than usually used in full body capture. The additional constraints of the face also allow more opportunities for using models and rules

- **Facial recognition system**

Is a technology capable of identifying or verifying a person from a digital image or a video frame from a video source. There are multiple methods in which facial recognition systems work, but in general, they work by comparing selected facial features from given image with faces within a database. It is also described as a Biometric Artificial Intelligence based application that can uniquely identify a person by analysing patterns based on the person's facial textures and shape[11].

- **Human sensing**

It also called human detection or human presence detection encompasses a range of technologies for detecting the presence of a human body in an area of space, typically without the intentional participation of the detected person. Common applications include search and rescue, surveillance, and customer analytics (for example, people counters).

### 2.3.3 Existing system

- **Holistic detection**

Detectors are trained to search for pedestrians in the video frame by scanning the whole frame. The detector would “fire” if the image features inside the local search window meet certain criteria. Some methods employ global features such as edge template ,others uses local features like histogram of oriented gradients descriptors. The drawback of this approach is that the performance can be easily affected by background clutter and occlusions.

- **Part-based detection**

Pedestrians are modeled as collections of parts. Part hypotheses are firstly generated by learning local features, which include edgelet and orientation features. These part hypotheses are then joined to form the best assembly of existing pedestrian hypotheses. Though this approach is attractive, part detection itself is a difficult task. Implementation of this approach follows a standard procedure for processing the image data that consists of first creating a densely sampled image pyramid, computing features at each scale, performing classification at all possible locations, and finally performing non-maximal suppression to generate the final set of bounding boxes.

- **Patch-based detection**

Recently Leibe et al. proposed an approach combining both the detection and segmentation with the name Implicit Shape Model (ISM). A codebook of local appearance is learned during the training process. In the detecting process, extracted local features are used to match against the codebook entries, and each match casts one vote for the pedestrian hypotheses. Final detection results can be obtained by further refining those hypotheses. The advantage of this approach is only a small number of training images are required.

- **Detection using multiple cameras**

Fleuret et al.suggested a method for integrating multiple calibrated cameras for detecting multiple pedestrians. In this approach, The ground plane is partitioned into uniform, non-overlapping grid cells, typically with size of 25 by 25 (cm). The detector produces a Probability Occupancy Map (POM), it provides an estimation of the probability of each grid cell to be occupied by a person. Given two to four synchronized video streams taken at eye level and from different angles, this method can effectively combine a generative model with dynamic

programming to accurately follow up to six individuals across thousands of frames in spite of significant occlusions and lighting changes. It can also derive metrically accurate trajectories for each one of them.

### 2.3.4 Advantages and disadvantages face detection

**Advantages**

The key benefits of "Face Detection" feature are:

• Accurate focusing, ensures that the important parts of a picture, typically faces, are in focus

• Improved Exposure, optimizes automatic exposure control

• Avoid ruining the picture, reduces over and under exposure for image capture

• No need to use "Focus Lock" feature and recomposing the frame

- **Government/ Identity Management**

The governments in Australia have built large biometric databases through registration of people as drivers with a photograph of the driver through registration for passports, aviation/maritime security and other purposes. The U.S. Department of State with 117 mn American adults in its database is one of the largest face recognition systems in the world.

- **Healthcare/ Healthcare provisioning**

Facial recognition has great advantages over other biometrics solutions. Face recognition technology is contactless with the target, making it convenient for wide-ranging applications. The potential application of facial recognition technology used for genetic screening is an easy identity management facilitating easy check into hospitals, clinics and other medical facilities. MR records could be fetched in a flash, and Insurance companies could check and authorize transactions in minutes instead of hours or days. Besides, facial recognition could also power emotional and sentiment analysis in mental healthcare environments, as well add another piece to Telemedicine to make it a real possibility. In addition, face and head tracking can be utilized for physiotherapy treatment through measuring and recording of head position, gaze direction and eye closure.

- **Financial services/ Authentication systems**

Since facial recognition could ease Point of Sale payments, with no swiping of cards, and potential of reducing lines at retails counters, financial services industry has been remarkably disrupted by the new business model, with customers are demanding more customer-centric, digital solutions and expecting banks to create solutions that not only solve the problem of security but also increase the advantage, it could also reduce spend and inconvenience for pension and other financial disbursements that require Proof of Life for older citizens who often have to manage this with difficulty.

### 2.3.5 Features

It's customary in any other novel approach to business – new or established – to field questions on the effectiveness of face recognition software too, particularly in cases of railway and airport security. And, therefore, a technology this promising has to have some hiccups. There are areas where this software may have its restrictions since accuracy in facial recognition technology remains less than desired. Fearing malfunction occurring, it may often

Other important issue is that of privacy & consent, which we have touched upon earlier also. Many consumers are concerned about the privacy and consent or acquiring permission for facial recognition technology usage that may be difficult to accomplish. The silver lining is that with Snapchat's over 200 million, Facebook base exceeding 2 billion people, LinkedIn acquiring 2 new users every second with a projected goal of 3 billion (currently 467 million) this concern may be genuine but not something impossible to accomplish[12].

## 2.4 Software Requirements

- **Theory of OpenCV face recognizers**

Thanks to OpenCV, coding facial recognition is now easier than ever. There are three easy steps to computer coding facial recognition, which are similar to the steps that our brains use for recognizing faces. These steps are:

- **Data Gathering**: Gather face data (face images in this case) of the persons you want to identify.
- **Train the Recognizer**: Feed that face data and respective names of each face to the recognizer so that it can learn.
- **Recognition**: matches the faces of input image to that of the database images


Figure 6 sample image of recognition using opencv

By performing LBPG algorithm it calculates all the eigen faces values by rate it recognises the selected face from given input dataset as showh in Figureure 6, recognition output is displayed in this format,OpenCV has three built-in face recognizers and thanks to its clean coding, you can use any of them just by changing a single line of code. Here are the names of those face recognizers and their OpenCV calls:

- **Eigenfaces face recognizer**

This algorithm considers the fact that not all parts of a face are equally important or useful for face recognition. Indeed, when you look at someone, you recognize that person by his distinct features, like the eyes, nose, cheeks or forehead; and how they vary respect to each other.

In that sense, you are focusing on the areas of maximum change. For example, from the eyes to the nose there is a significant change, and same applies from the nose to the mouth. When you look at multiple faces, you compare them by looking at these areas, because by catching the maximum variation among faces, they help you differentiate one face from the other.

In this way, is how EigenFaces recognizer works. It looks at all the training images of all the people as a whole and tries to extract the components which are relevant and useful and discards the rest. These important features are called principal components.

**Note**: We will use the terms: principal components, variance, areas of high change and useful features indistinctly as they all mean the same.

- **Fisher faces face recognizer**

This algorithm is an improved version of the last one. As we just saw, EigenFaces[13] looks at all the training faces of all the people at once and finds principal components from all of them combined. By doing that, it doesn't focus on the features that discriminate one individual from another. Instead, it concentrates on the ones that represent all the faces of all the people in the training data, as a whole.But here's the kicker:Consider the lighting changes in following images Figureure 7.

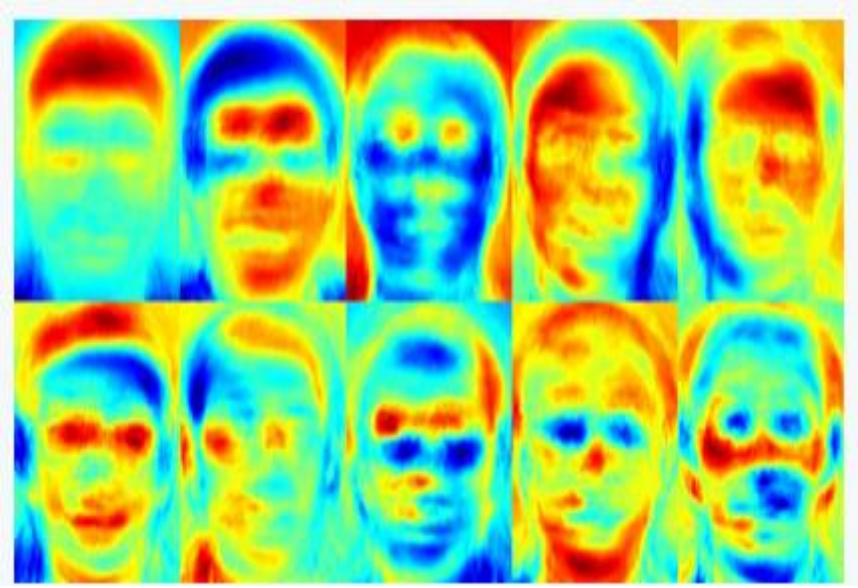


Figure 7 eigen faces detected using fisher face method

You can see that the useful features represent faces which receive the name of Eigen Faces. I mean, how else do you think the algorithm got its name?

In simple words, it's a game of matching .However, one thing to note in above image is that Eigen Faces algorithm also considers illumination as an important feature. In consequence, lights and shadows are picked up by Eigen Faces(shown in Figureure8), which classifies them as representing a 'face.'

Figure 8 eigen faces illumination

Since EigenFaces also finds illumination as a useful component, it will find this variation very relevant for face recognition and may discard the features of the other people's faces, considering them less useful. In the end, the variance that EigenFaces has extracted represents just one individual's facial features.

Precisely, FisherFaces[14] face recognizer algorithm extracts principal components that differentiate one person from the others. In that sense, an individual's components do not dominate (become more useful) over the others.

Below is an image (Figureure 9) of principal components using FisherFaces algorithm.

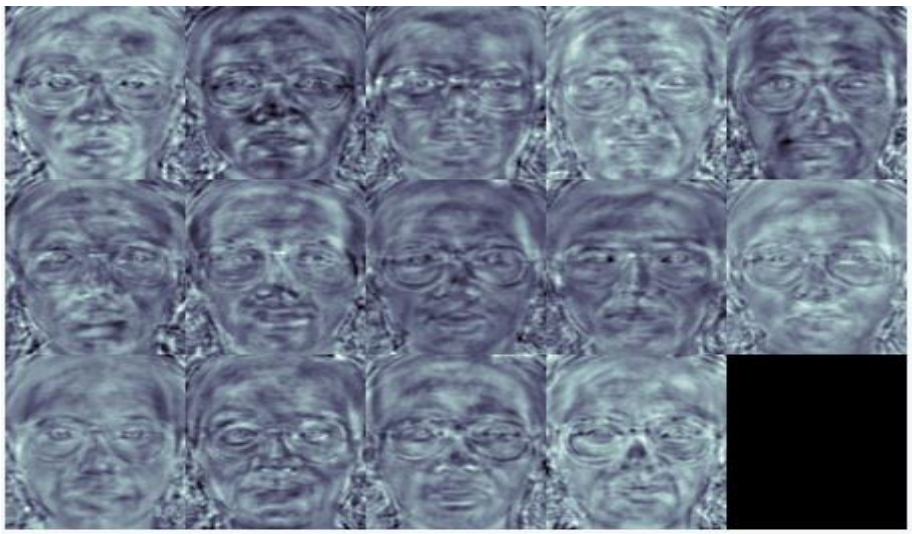


Figure 9 fisher face recognition

FisherFaces Face Recognizer Principal Components. Source: docs.opencv.org You can see that the features represent faces which receive the name of Fisher Faces. Are you noticing a theme with the names of the algorithms?

One thing to note here is that FisherFaces only prevents features of one person from becoming dominant, but it still considers illumination changes as a useful feature. We know that light variation is not a useful feature to extract as it is not part of the actual face.Then, how to get rid of this problem? Here is where our next face recognizer comes in.

- **Local binary patterns histograms (LBPH) Face Recognizer**

I wrote a detailed explanation of Local Binary Patterns Histograms[15] in my previous article on face detection, which I'm sure you've read by now. So, here I will just give a brief overview of how it works.

We know that Eigenfaces and Fisherfaces are both affected by light and, in real life, we can't guarantee perfect light conditions. LBPH face recognizer is an improvement to overcome this drawback.The idea with LBPH is not to look at the image as a whole, but instead, try to find its local structure by comparing each pixel to the neighboring pixels.

- **The LBPH Face Recognizer Process**

Take a 3×3 window and move it across one image. At each move (each local part of the picture),compare the pixel at the center, with its surrounding pixels. Denote the neighbors with intensity value less than or equal to the center pixel by 1 and the rest by0.After you read these 0/1 values under the 3×3 window in a clockwise order, you will have a binary pattern like 11100011 that is local to a particular area of the picture. When you finish doing this on the whole image, you will have a list of local binary patterns as shown in Figure 10.

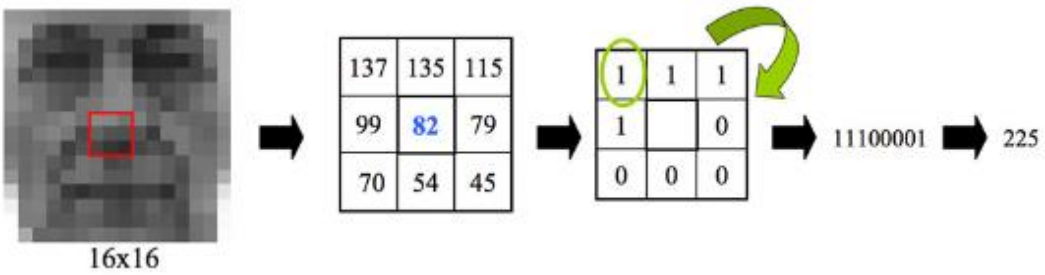



Figure 10 Lbph face recognition process

Now, after you get a list of local binary patterns, you convert each one into a decimal number using binary to decimal conversion (as shown in above image) and then you make a histogram of all of those decimal values. A sample histogram looks like this:

In the end, you will have one histogram for each face in the training data set. That means that if there were 100 images in the training data set then LBPH will extract 100 histograms after training and store them for later recognition. Remember, the algorithm also keeps track of which histogram belongs to which person.

Later during recognition, the process is as follows:

- Feed a new image to the recognizer for face recognition.
- The recognizer generates a histogram for that new picture.
- It then compares that histogram with the histograms it already has.
- Finally, it finds the best match and returns the person label associated with that best match.

Below is a group of faces and their respective local binary patterns images. You can see that the LBP faces are not affected by changes in light conditions as shown in Figureure 11.

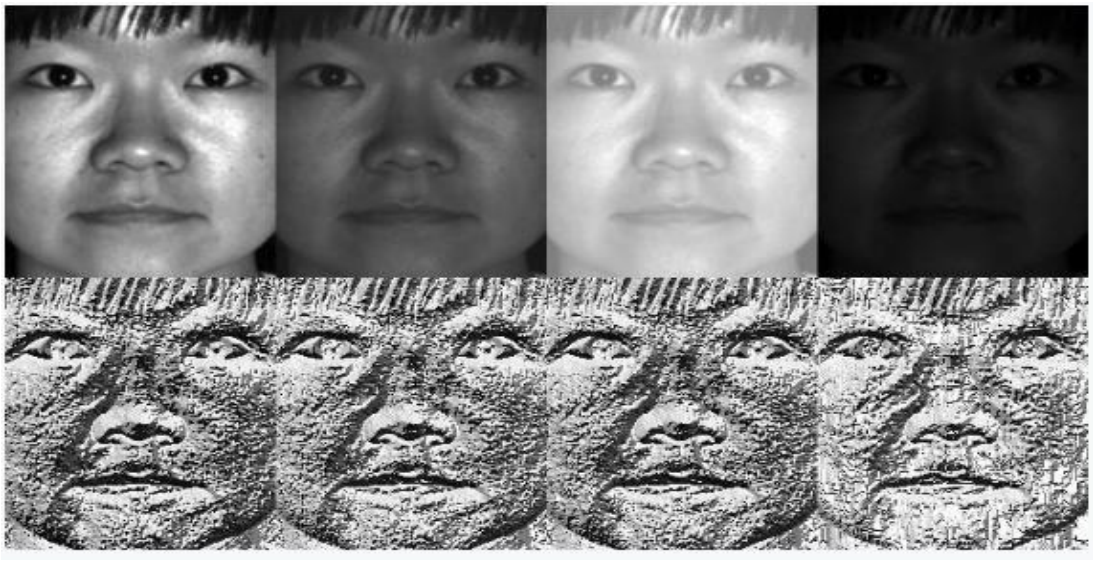


Figure 11 Lbph face recognition in different light locations

LBPH Face Recognizer Principal Components. Source: docs.opencv.org

The theory part is over, and now it is time to unmask this moron. Brace yourself. Here comes the coding!

Coding Face Recognition using Python and OpenCV

We are going to divide the Face Recognition process in this tutorial into three steps:

- Prepare Training Data: Read training images for each person/subject along with their labels, detect faces from each image and assign each detected face an integer label of the person it belongs.
- Train Face Recognizer: Train OpenCV's LBPH recognizer by feeding it the data we prepared in step 1.
- Prediction: Introduce some test images to face recognizer and see if it predicts them correctly.

To detect faces, I will use the code from my previous article on face detection. Before we start the actual coding, we need to install the Code Dependencies and import the Required M

- **Code dependencies**

Install the following dependencies:OpenCV 3.2.0 ,Python v3.5

NumPy that makes computing in Python easy. It contains a powerful implementation of N-dimensional arrays which we will use for feeding data as input to OpenCV functions.

- **Required Modules**

Import the following modules:

cv2: This is the OpenCV module for Python used for face detection and face recognition.

os: We will use this Python module to read our training directories and file names.

numpy: This module converts Python lists to numpy arrays as OpenCV face recognizer needs them for the face recognition process.

- **Prepare training data**

The premise here is simple: The more images used in training, the better.Being thorough with this principle is important because it is the only way for training a face recognizer so it can learn the different ‘faces’ of the same person; for example: with glasses, without glasses, laughing, sad, happy, crying, with a beard, without a beard, etc.

So, our training data consists of total two people with 12 images of each one. All training data is inside the folder: training-data.

This folder contains one subfolder for every individual, named with the format: sLabel (e.g. s1, s2) where the label is the integer assigned to that person. For example, the subfolder called s1 means that it contains images for person 1.With that in mind, the directory structure tree for training data is as follows

On the other hand, The foldertest-data contains images that we will use to test our face recognition program after we have trained it successfully.

On the other hand, The foldertest-data contains images that we will use to test our face recognition program after we have trained it successfully.

Considering that the OpenCV face recognizer only accepts labels as integers, we need to define a mapping between integer tags and the person's actual name. So, below I am defining the mapping of a person's integer labels and their respective names.

I have a sneaking suspicion that my follower thief is none other than Elvis Presley. Why else do I keep listening to 1950s rock and roll?

- **Data Preparation for Face Recognition**

Perhaps you are thinking: Why are are we talking about preparing data?

Well, to know which face belongs to which person, OpenCV face recognizer accepts information in a particular format. In fact, it receives two vectors:

One is the faces of all the people.

The second is the integer labels for each face.

For example, if we had two individuals and two images for each one:

Table 1 images for data preparation

| Person 1 | Person 2 |
|---|---|
| Img 1 | Img 1 |
| Img 2 | Img2 |

Then, the data preparation step will produce following face and label vectors:

In detail, we can further divide this step into the following sub-steps:

- Read all the sub folders names provided in the folder training-data. In this tutorial; we have folder names:s1, s2.

- Extract label number. Remember that all the sub folders containing images of a person following the format:sLabel where Label is an integer representing each person. So for example, folder name: s1 means that the person has label 1, s2 means the person's label is 2, and so on. We will assign the integer extracted in this step to every face detected in the next one.
- Read all the images of the person, and apply face detection to each one.
- Add each face to face vectors with the corresponding person label (extracted in above step)

## 2.5 Conclusion

The task of face recognition has been actively researched in recent years. This paper provides an up-to-date review of major human face recognition research. We first present an overview of face recognition and its applications. Then, a literature review of the most recent face recognition techniques is presented. Description and limitations of face databases which are used to test the performance of these face recognition algorithms are given. A brief summary of the face recognition vendor test (FRVT) 2002, a large scale evaluation of automatic face recognition

Progress has advanced to the point that face recognition systems are being demonstrated in real-world settings. The rapid development of face recognition is due to a combination of factors: active development of algorithms, the availability of a large databases of facial images, and a method for evaluating the performance of face recognition algorithms. In the literatures, face recognition problem can be formulated as: given static (still) or video images of a scene, identify or verify one or more persons in the scene by comparing with faces stored in a database. When comparing person verification to face recognition, there are several aspects which differ. First, a client – an authorized user of a personal identification system . Face recognition has the benefit of being a passive, non intrusive system to verify personal identity in a “natural” and friendly way.

# CHAPTER 3

# IMPLEMENTATION AND ANALYSIS

## 3.1 Introduction

Face recognition is an important research problem spanning numerous fields and disciplines. This because face recognition, in additional to having numerous practical applications such as bankcard identification, access control, Mug shots searching, security monitoring, and surveillance system, is a fundamental human behaviour that is essential for effective communications and interactions among people. A formal method of classifying faces was first proposed. The author proposed collecting facial profiles as curves, finding their norm, and then classifying other profiles by their deviations from the norm. This classification is multi-modal, i.e. resulting in a vector of independent measures that could be compared with other vectors in a database. Progress has advanced to the point that face recognition systems are being demonstrated in real-world settings. The rapid development of face recognition is due to a combination of factors: active development of algorithms, the availability of a large databases of facial images, and a method for evaluating the performance of face recognition algorithms.

In the literatures, face recognition problem can be formulated as: given static (still) or video images of a scene, identify or verify one or more persons in the scene by comparing with faces stored in a database. When comparing person verification to face recognition, there are several aspects which differ. First, a client – an authorized user of a personal identification system – is assumed to be co-operative and makes an identity claim. Computationally this means that it is not necessary to consult the complete set of database images (denoted model images below) in order to verify a claim. An incoming image (referred to as a probe image) is thus compared to a small number of model images of the person whose identity is claimed and not, as in the recognition scenario, with every image (or some descriptor of an image) in a potentially large database. Second, an automatic authentication system must operate in near-real time to be acceptable to users.

Finally, in recognition experiments, only images of people from the training database are presented to the system, whereas the case of an imposter (most likely a previously unseen person) is of utmost importance for authentication. Face recognition is a biometric approach that employs automated methods to verify or recognition of face

of a living person based on his/her physiological characteristics. In general, a biometric identification system makes use of either physiological characteristics (such as a fingerprint, iris pattern, or face) or behaviour patterns (such as hand-writing, voice, or key-stroke pattern) to identify a person. Because of human inherent protectiveness of his/her eyes, some people are reluctant to use eye identification systems. Face recognition has the benefit of being a passive, non intrusive system to verify personal identity in a "natural" and friendly way.

In general, biometric devices can be explained with a three step procedure a sensor takes an observation. The type of sensor and its observation depend on the type of biometric devices used. This observation gives us a Face recognition starts with the detection of face patterns in sometimes cluttered scenes, proceeds by normalizing the face images to account for geometrical and illumination changes, possibly using information about the location and appearance of facial landmarks, identifies the faces using appropriate classification algorithms, and post processes the results using model-based schemes and logistic feedback. The application of face recognition technique can be categorized into two main parts: law enforcement application and commercial application. Face recognition technology primarily used in law enforcement applications, especially Mug shot albums (static matching) and video surveillance (real-time matching by video image sequences).

The commercial applications range from static matching of photographs on credit cards, ATM cards, passports, driver's licenses, and photo ID to real-time matching with still images or video image sequences for access control. Each application presents different constraints in terms of processing. All face recognition algorithms consistent of two major parts: face detection and normalization and face identification. Algorithms that consist of both parts are referred to as fully automatic algorithms and those that consist of only the second part are called partially automatic algorithms. Partially automatic algorithms are given a facial image and the coordinates of the center of the eyes. Fully automatic algorithms are only given facial images

## 3.2 Face recognition algorithm

### 3.2.1 Template matching

A simple version of template matching is that a test image represented as a two-dimensional array of intensity values is compared using a suitable metric, such as the Euclidean distance, with a single template representing the whole face. There are several other more sophisticated versions of template matching on face recognition. One can use more than one

face template from different viewpoints to represent an individual's face. A face from a single viewpoint can also be represented by a set of multiple distinctive smaller templates . The face image of gray levels may also be properly processed before matching . In, Bruneli and Poggio automatically selected a set of four features templates, i.e., the eyes, nose, mouth, and the whole face, for all of the available faces. They compared the performance of their geometrical matching algorithm and template matching algorithm on the same database of faces which contains 188 images of 47 individuals. The template matching was superior in recognition (100 percent recognition rate) to geometrical matching (90 percent recognition rate) and was also simpler. Since the principal components (also known as eigenfaces or eigenfeatures) are linear combinations of the templates in the data basis, the technique cannot achieve better results than correlation , but it may be long computationally expensive.

One drawback of template matching is its computational complexity. Another problem lies in the description of these templates. Since the recognition system has to be tolerant to certain discrepancies between the template and the test image, this tolerance might average out the differences that make individual faces unique. In general, template-based approaches compared to feature matching are a more logical approach In summary, no existing technique is free from limitations. Further efforts are required to improve the performances of face recognition techniques, especially in the wide range of environments encountered in real world.

- **Concept**

Template Matching techniques are expected to address the following need: provided a reference image of an object (the template image) and an image to be inspected (the input image) we want to identify all input image locations at which the object from the template image is present. Depending on the specific problem at hand, we may (or may not) want to identify the rotated or scaled occurrences.

We will start with a demonstration of a naive Template Matching method, which is insufficient for real-life applications, but illustrates the core concept from which the actual Template Matching algorithms stem from. After that we will explain how this method is enhanced and extended in advanced Grayscale-based Matching and Edge-based Matching routines.

- **Image Correlation**

One of the subproblems that occur in the specification above is calculating the similarity measure of the aligned template image and the overlapped segment of the input image, which is equivalent to calculating a similarity measure of two images of equal dimensions. This is a classical task, and a numeric measure of image similarity is usually called image correlation as shown in Figure 15.

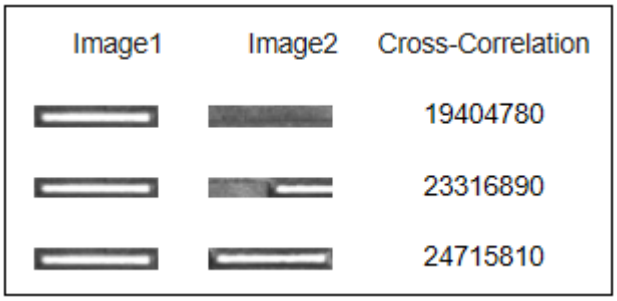



Figure 15 image correlation

- **Cross-Correlation**

The fundamental method of calculating the image correlation is so called cross-correlation, which essentially is a simple sum of pairwise multiplications of corresponding pixel values of the images.

Though we may notice that the correlation value indeed seems to reflect the similarity of the images being compared, cross-correlation method is far from being robust. Its main drawback is that it is biased by changes in global brightness of the images - brightening of an image may sky-rocket its cross-correlation with another image, even if the second image is not at all similar as in equation (1)

$$\boldsymbol{cross - correlation}(\boldsymbol{img_1}, \boldsymbol{img_2}) = \boldsymbol{\Sigma_{x,y}}(\boldsymbol{img_1}(\boldsymbol{x},\boldsymbol{y}) * \boldsymbol{img_2}(\boldsymbol{x},\boldsymbol{y}) - - - \ \boldsymbol{equation}(\mathbf{1})$$

- **Normalized Cross-Correlation**

Normalized cross-correlation is an enhanced version of the classic cross-correlation method that introduces two improvements over the original one:

- The results are invariant to the global brightness changes, i.e. consistent brightening or darkening of either image has no effect on the result (this is accomplished by subtracting the mean image brightness from each pixel value).

### 3.2.2 Linear binary pattern histogram

- **Introduction**

Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number**.**

It was first described in 1994 (LBP) and has since been found to be a powerful feature for texture classification. It has further been determined that when LBP is combined with histograms of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets.Using the LBP combined with histograms we can represent the face images with a simple data vector.As LBP is a visual descriptor it can also be used for face recognition tasks, as can be seen in the following step-by-step explanation.

- **Step-by-Step**

Now that we know a little more about face recognition and the LBPH, let's go further and see the steps of the algorithm:

**Parameters**: the LBPH uses 4 parameters:

**Radius**: the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.

**Neighbors**: the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.

**Grid X**: the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

Don't worry about the parameters right now, you will understand them after reading the next steps.

- **Training the Algorithm**:

First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name

of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let's see the LBPH computational steps.

- **Applying the LBP operation**

The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters radius and neighbors.

The image(Figure 16) below shows this procedure:

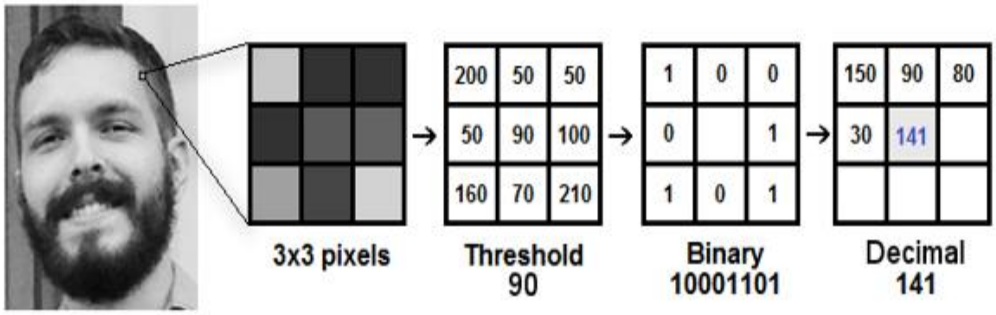



Figure 16 conversion of image to binary and decimal

Based on the image above, let's break it into several small steps so we can understand it easily:

- Suppose we have a facial image in grayscale.
- We can get part of this image as a window of 3x3 pixels.
- It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255).
- Then, we need to take the central value of the matrix to be used as the threshold.
- This value will be used to define the new values from the 8 neighbors
- For each neighbor of the central value (threshold), we set a new binary value. We set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.
- Now, the matrix will contain only binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10001101). Note: some authors use other approaches to concatenate the binary values (e.g. clockwise direction), but the final result will be the same.

- Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original image.

At the end of this procedure (LBP procedure), we have a new image which represents better the characteristics of the original image as shown in Figure 17.

Note: The LBP procedure was expanded to use a different number of radius and neighbors, it is called Circular LBP.

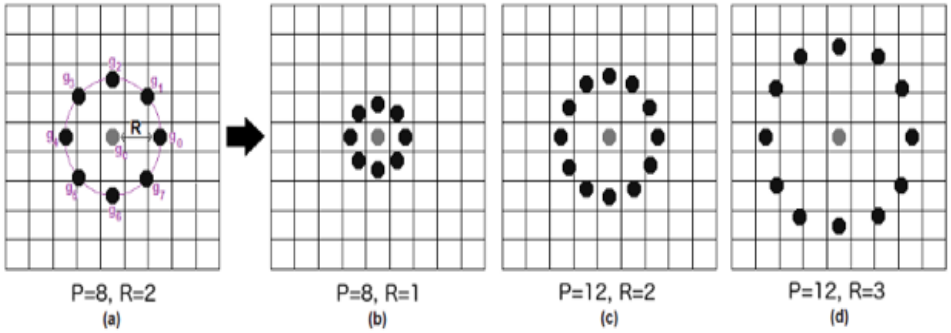



Figure 17 radius binary interpolation of image

It can be done by using bilinear interpolation. If some data point is between the pixels, it uses the values from the 4 nearest pixels (2x2) to estimate the value of the new data point.

➢ **Extracting the Histograms**

Now, using the image generated in the last step, we can use the Grid X and Grid Y parameters to divide the image into multiple grids, as can be seen in the following Figure 18:

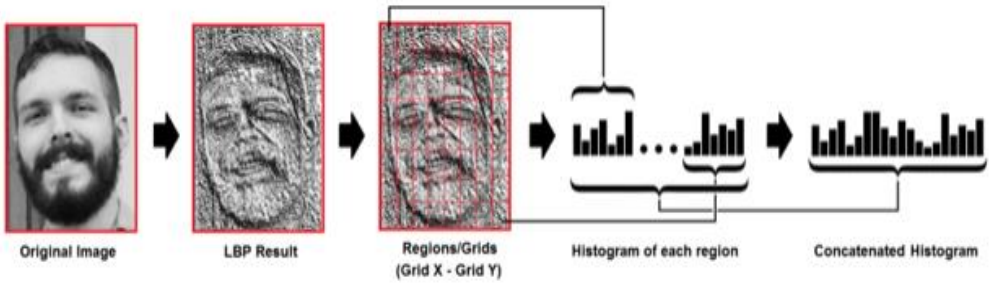



Figure 18 histogram representation of image

Based on the image above, we can extract the histogram of each region as follows:

- As we have an image in grayscale, each histogram (from each grid) will contain only 256 positions (0~255) representing the occurrences of each pixel intensity.

- Then, we need to concatenate each histogram to create a new and bigger histogram. Supposing we have 8x8 grids, we will have 8x8x256=16.384 positions in the final histogram. The final histogram represents the characteristics of the image original image.

The LBPH algorithm is pretty much it.

➢ **Performing the face recognition**

In this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, we perform the steps again for this new image and creates a histogram which represents the image.

So to find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram.

We can use various approaches to compare the histograms (calculate the distance between two histograms), for example: euclidean distance, chi-square, absolute value, etc. In this example, we can use the Euclidean distance (which is quite known) based on the following Equation 2:

$$\eta=\sqrt{\sum_{i=1}^{n}(hist1i - hist2i)^2}$$ ---------Equation 2

- So the algorithm output is the ID from the image with the closest histogram. The algorithm should also return the calculated distance, which can be used as a 'confidence' measurement. Note: don't be fooled about the 'confidence' name, as lower confidences are better because it means the distance between the two histograms is closer.
- We can then use a threshold and the 'confidence' to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized if the confidence is lower than the threshold defined.

### 3.2.4 Understanding Eigen faces

Any grey scale face image I(x,y) consisting of a NxN array of intensity values may also be consider as a vector of N2. For example, a typical 100x100 image used in this thesis will have to be transformed into a 10000 dimension vector,as shown in Figure 18.

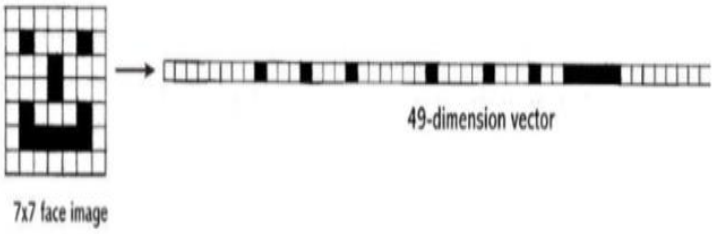



Figure 19 7x7 face image transformed into a 49 dimension vector

This vector can also be regarded as a point in 10000 dimension space. Therefore, all the images of subjects' whose faces are to be recognized can be regarded as points in 10000dimension space. Face recognition using these images is doomed to failure because all human face images are quite similar to one another so all associated vectors are very closeto each other in the 10000-dimension space.

Therefore classification of a new vector (image) would be a very sensitive process sinceeven a slight change in the image would cause it to be nearer another face image than thesubject's face in the face database.The original variables (vectors) which described the face (pixel intensity at [1,1], pixelintensity at [1,2], ……) are highly correlated. With PCA, the researcher tried to find a better representation of faces by finding the specific vectors that account for thedistribution of face images. These vectors will define the subspace of face images (sometimes called 'face space'). Face space will be a better representation for face images than image space which is the space which containing all possible images since there will be increased variation between the faces in face space (O'Toole et al. 1993).

The vectors that describe faces in face space are eigenfaces. These are in fact the eigenvectors of the covariance matrix of a set of mean subtracted face images (subtract the average face from each of the face images). Since a typical face image used in this thesis is100x100 (therefore associated vectors 10000x1), and if there are 30 face images in the training set for PCA, the covariance matrix (C) as in Equation(3):

**C = X XT**---------Equation (3)

Here X, is a 10000x30 matrix containing the mean subtracted face images. Therefore, the covariance matrices dimensions would be 10000x10000. Calculating this matrix would be an impossible task for most modern computers. This is one of the problems of using PCAin pattern

recognition since high dimension vectors (i.e. images) are used. A computationally feasible method must be found to calculate eigenfaces as in Equation (4).

$$\mathbf{c\ v = \lambda\ v}\text{---------- Equation (4)}$$

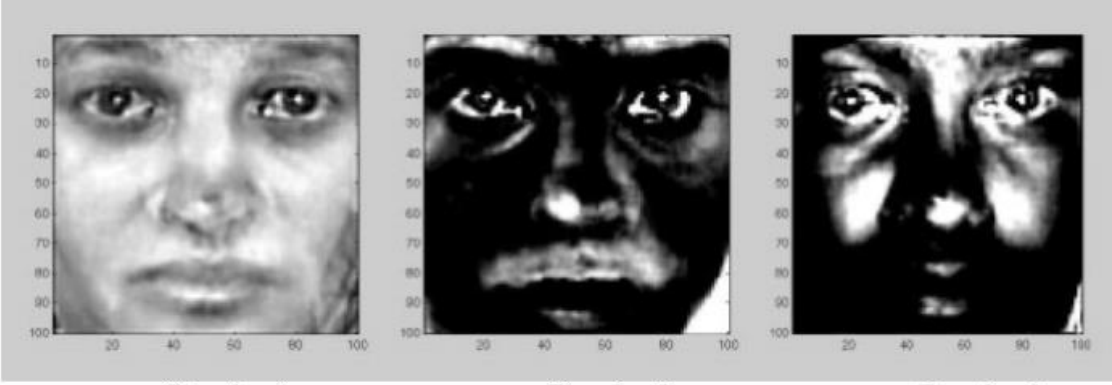
All the eigenvectors, which were calculated, need not be used. Further, dimensionality reduction can be done by sorting the eigenvectors according to their associated eigenvalues and just taking (say 40) eigenvectors with the largest eigenvalues. These describe the greatest variation in human faces. Now that the eigenvectors of c have been found, the eigenvectors of C (eigenfaces) are in the matrix U where shown in Equation (5),

$$\mathbf{U=X\ v}\text{---------- Equation (5)}$$

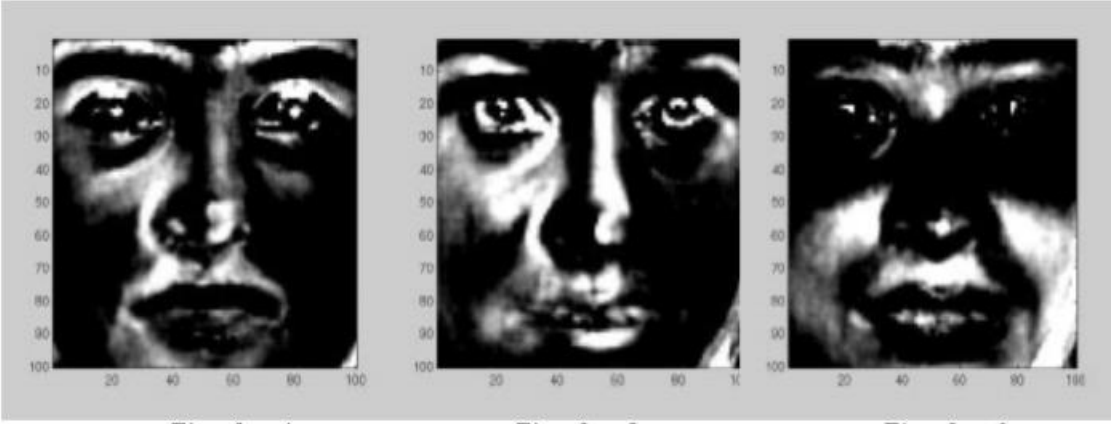
Any face can be described using these eigenfaces. For a detailed description of Principal Component Analysis for face images the reader is encouraged to refer Turk and Pentland (1991a). Further details and proofs of the PCA results used in this thesis can be found in highly enjoyable, specialised mathematics textbooks such as Dunteman(1989), and Narayanaswamy and Raghavarao(1991) .

The following Figures 20 contain the first 18 eigenfaces calculated from 30 frontal view face

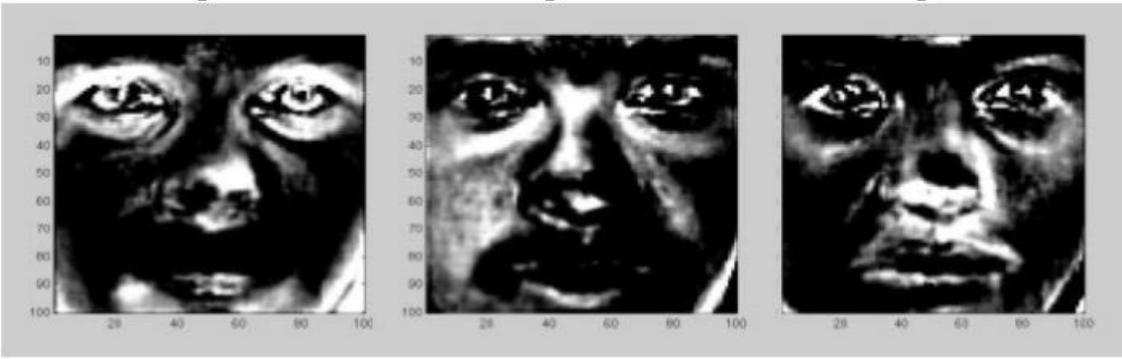
Images



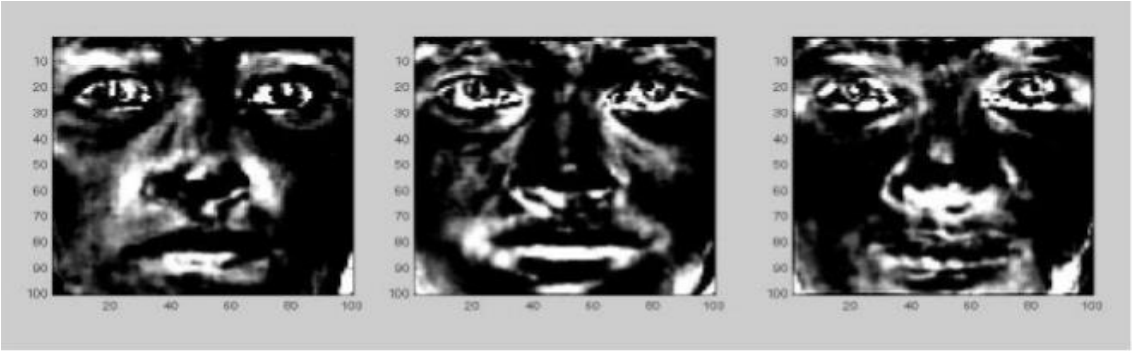
Eigenface 1 Eigenface 2 Eigenface 3



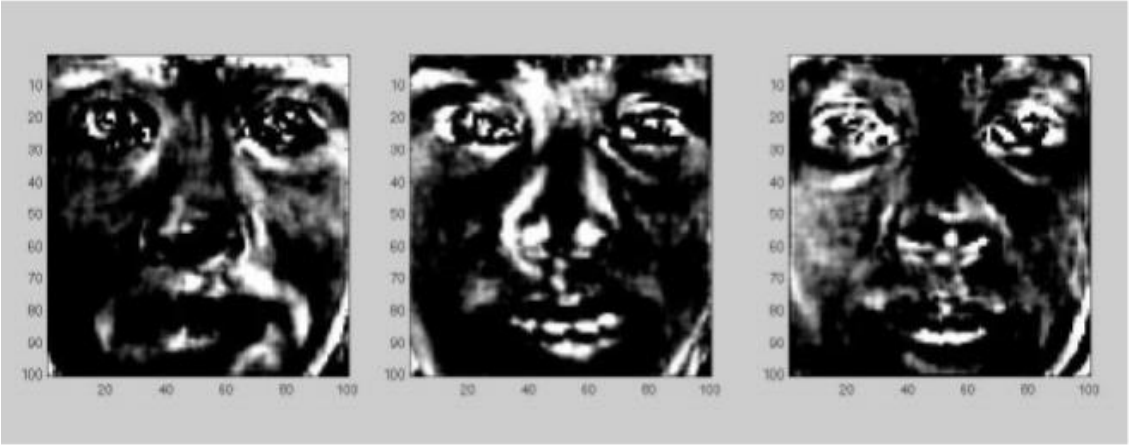
Eigenface 4 Eigenface 5 Eigenface 6



Eigenface 7 Eigenface 8 Eigenface 9



Eigenface 10 Eigenface 11 Eigenface 12



Eigenface 13 Eigenface 14 Eigenface 15

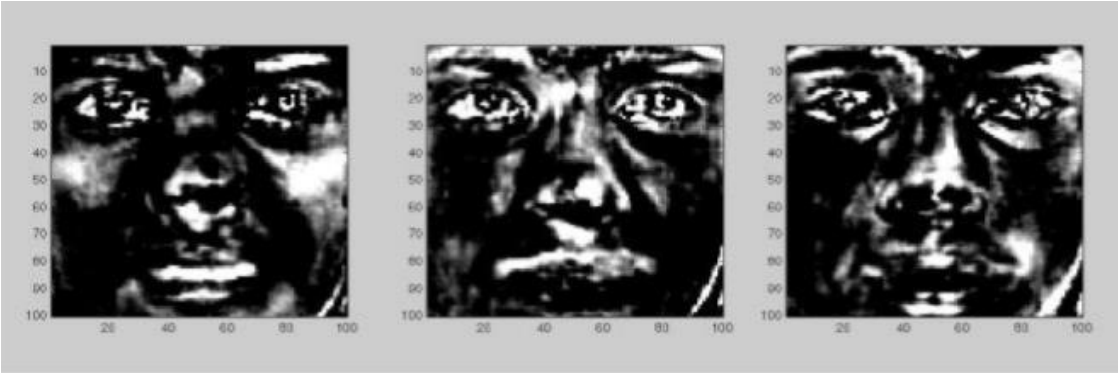



Figure 20 Eigenfaces with gray scale

### 3.2.4 Recognition

The transformation of a face from image space (I) to face space (f) involves just a simple matrix multiplication. If the average face image is A and U contains the (previously calculated) eigenfaces,

This is done to all the face images in the face database (database with known faces) and to the image (face of the subject) which must be recognized. The possible results when

projecting a face into face space are given in the following Figure.

There are four possibilities:

1. Projected image is a face and is transformed near a face in the face database

2. Projected image is a face and is not transformed near a face in the face database

3. Projected image is not a face and is transformed near a face in the face database

4. Projected image is not a face and is not transformed near a face in the face

### 3.2.5 Pose invariant face recognition

Extending the frontal view face recognition system to a pose-invariant recognition system is quite simple if one of the proposed specifications of the face recognition system is relaxed. Successful pose-invariant recognition will be possible if many images of a known individual are in the face database. Nine images from each known individual can be taken as shown below. Then if an image of the same individual is submitted within a 30o angle from the frontal view he or she can be identified.

Pose invariant face recognition highlights the generalisation ability of PCA. For exampleshown in Figure 21, when an individual's frontal view and 30o left view known, even the individual's 15o leftview can be recognised.

Figure 21 different facial expressions of a person

Although good results can be obtained for pose invariant face recognition under controlled conditions, pose invariant face detection is extremely hard and successful template matching or traditional neural network strategy to deal with the problem has still not been proposed. Furthermore, gathering multiple views of an individual for a face database is not realistic since in most situations only a single known face for each subject is available.Since the face was not isolated in the image, variations in the subject's clothing, hairstyle or the background would adversely effect recognition performance. The pose invariant face database and testing images gathered in this thesis were obtained under highly controlled conditions since the pose invariant face recognition system is not as robust as the frontal view system.

## 3.3 Face detection algorithm

### 3.3.1 Introduction

While some may regard face detection as simple pre-processing for the face recognition system, it is by far the most important process in a face detection and recognition system. However face recognition is not the only possible application of a fully automated face

detection system. There are applications in automated colour film development where information about the exact face location is useful for determining exposure and colour levels during film development. The are even uses in face tracking for automated camera control in the film and television news industries.

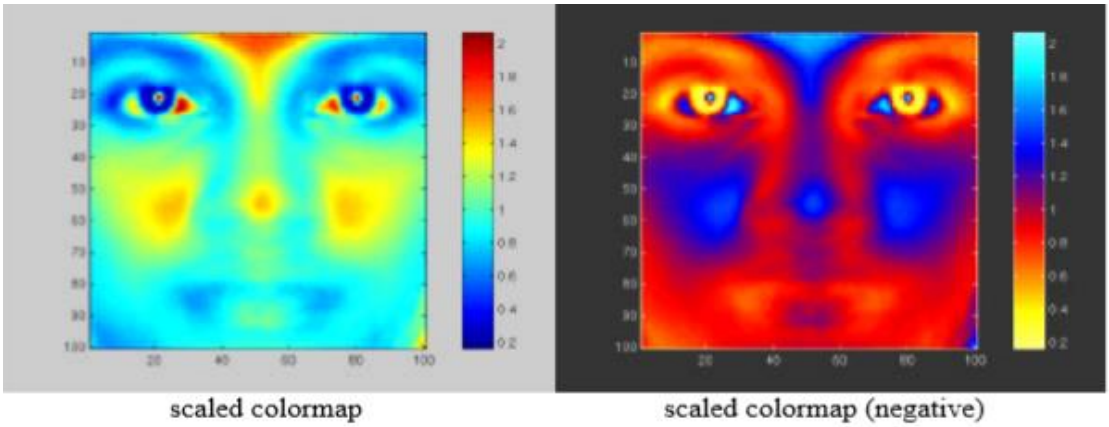



Figure 22 average human face in gray scale

In this project the author will attempt to detect faces in still images by using image invariants. 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. A suitably scaled colormap has been used to highlight grey-scale intensity differences as shown in Figure 23.

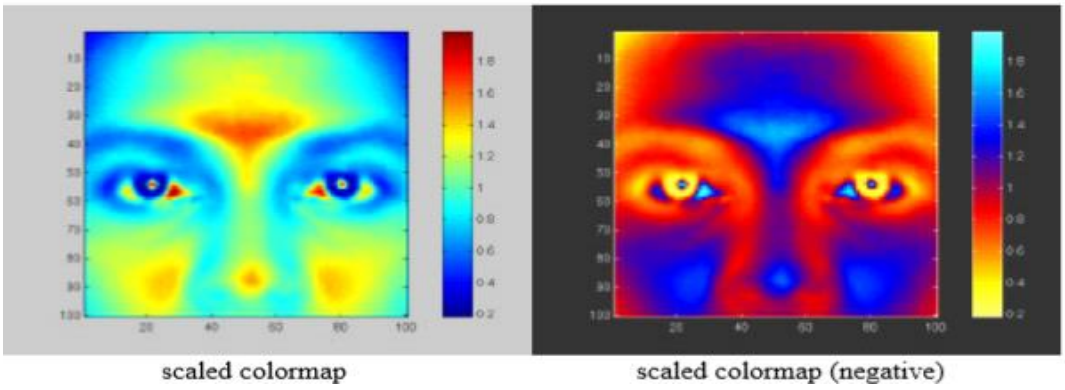



Figure 23 area choosen for detection

The grey-scale differences, which are invariant across all the sample faces are strikingly apparent. The eye-eyebrow area seem to always contain dark intensity (low) gray-levels while nose forehead and cheeks contain bright intensity (high) grey-levels. After a great deal of experimentation, the researcher found that the following areas of the human face were suitable for a face detection system based on image invariants and a deformable template

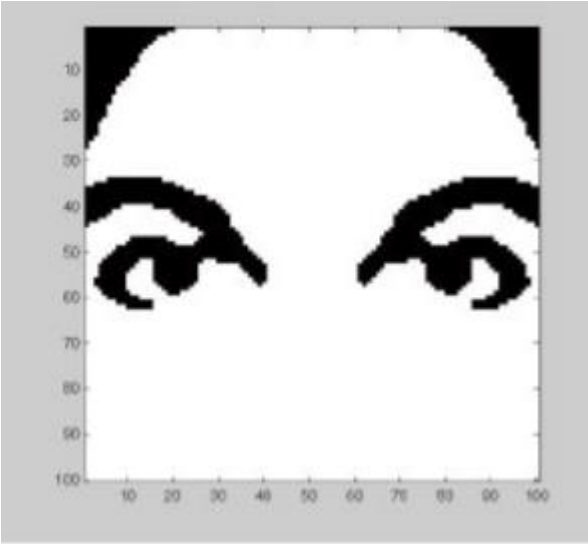



Figure 24 basis for dark intensity invariant sensitive template

The above facial (Figure 24) 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.Once this pixel area is located by the face detection system, any particular area required can be segmented based on the proportions of the average human face After studying the above images it was subjectively decided by the author to use the following as a basis for dark intensity sensitive and bright intensity sensitive templates.Once these are located in a subject's face, a pixel area 33.3% (of the width of the square window) below this area will be segmented.

Note the slight differences which were made to the bright intensity invariant sensitive template (compare Figures 3.4 and 3.2) which were needed because of the pre-processing done by the system to overcome irregular lighting (chapter six).Now that a suitable dark and bright intensity invariant templates have been decided on, it is necessary to find a way of using these to make 2 A-units for a perceptron, i.e. a computational model is needed to assign neurons to the distributions displayed

### 3.3.2 Face Detection using Haar Cascades

Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. just like process show in Figure 25.

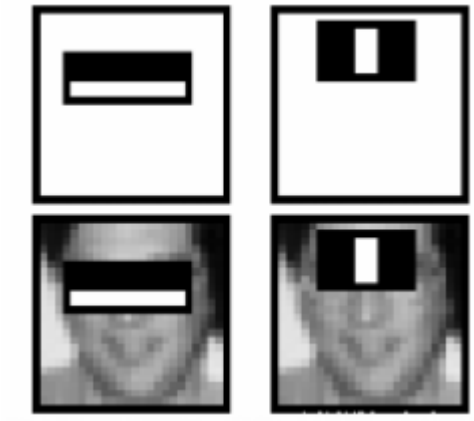


Figure 25 images of face detection using HC method

Here we will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, Haar features shown in the below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle.

Now, all possible sizes and locations of each kernel are used to calculate lots of features. (Just imagine how much computation it needs? Even a 24x24 window results over 160000 features). For each feature calculation, we need to find the sum of the pixels under white and black rectangles. To solve this, they introduced the integral image. However large your image, it reduces the calculations for a given pixel to an operation involving just four pixels. Nice, isn't it? It makes things super-fast.

But among all these features we calculated, most of them are irrelevant. For example, consider the image below. The top row shows two good features. The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose. But the same windows applied to cheeks or any other place is irrelevant. So how do we select the best features out of 160000+ features? It is achieved by Adaboost.

The final classifier is a weighted sum of these weak classifiers. It is called weak because it alone can't classify the image, but together with others forms a strong classifier. The paper says even 200 features provide detection with 95% accuracy. Their final setup had around 6000 features. (Imagine a reduction from 160000+ features to 6000 features. That is a big gain).

In an image, most of the image is non-face region. So it is a better idea to have a simple method to check if a window is not a face region. If it is not, discard it in a single shot, and don't process it again. Instead, focus on regions where there can be a face. This way, we spend more time checking possible face regions as shown in Figure 26.

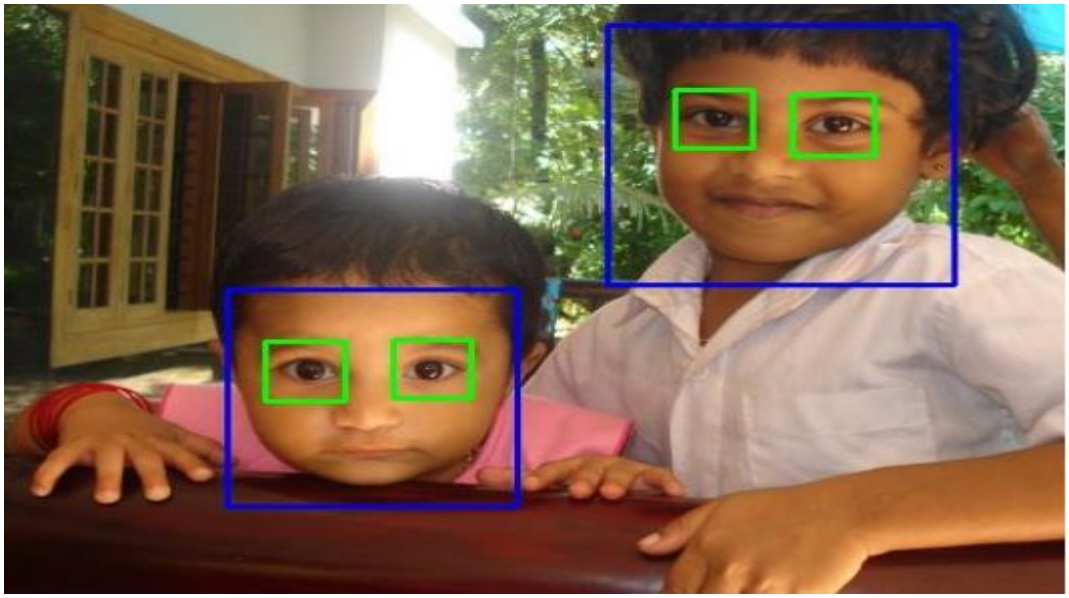


Figure 26 detected faces of sample

For this they introduced the concept of Cascade of Classifiers. Instead of applying all 6000 features on a window, the features are grouped into different stages of classifiers and applied one-by-one. (Normally the first few stages will contain very many fewer features). If a window fails the first stage, discard it. We don't consider the remaining features on it. If it passes, apply the second stage of features and continue the process. The window which passes all stages is a face region. How is that plan!

The authors' detector had 6000+ features with 38 stages with 1, 10, 25, 25 and 50 features in the first five stages. (The two features in the above image are actually obtained as the best two features from Adaboost). According to the authors, on average 10 features out of 6000+ are evaluated per sub-window.

So this is a simple intuitive explanation of how Viola-Jones face detection works. Read the paper for more details or check out the references in the Additional Resources section.

### 3.3.3 Deformable template algorithm

The dark and bright intensity invariant templates which have been created can be implemented in a deformable template algorithm. For example, if a 100x100 template is needed to check whether a certain 100x100 pixel area contains a face, the weight vectors in the dark and bright templates have to be simply multiplied by 100. Now the pixels in the areas that are indicated by the templates can be sampled to see if they match the dark and bright patterns

However since individually sampling pixels for the templates is expensive, an efficient, simple implementation of a deformable template is needed. Remember that computer vision problems are by nature computationally expensive. Therefore, very simple algorithms to implement the system must be found. Since computers are efficient at array indexing, it occurred to the author that this would be an ideal way to implement the deformable template. Once the deformable template has been expanded to match a particular window (for example, 100x100), the weight vectors of each template are converted to array indexes. Then the elements in the image (i.e. 100x100 pixel area), which can be regarded as a 2-dimensional array are extracted. image(indexes) = (pixel intensities at indexed positions)

Therefore, to keep the face detection algorithm simple, extracted pixel intensities could be merely added. As a result, if the total added pixel intensities from the bright intensity invariant template is high and the total added intensities from the dark intensity invariant template is low, a high heuristic value will be output by the perceptron.

### 3.3.4 Development of the face search algorithm

An exhaustive search of an image for a face is clearly impossible since there are almost an infinite number of possible places where a face may be. A face may be anywhere from the upper left to the lower right corner of the image, scaled to almost fit the whole image or far away in the distance. An efficient search algorithm is essential for face detection in near real-time. One possible way of reducing the search space in static images is to use the fact that frontal view faces are symmetrical (Saber and Tekalp, 1996). If this (vertical) line of symmetry can be found, the deformable template can be used only along this line of symmetry. After experimentation, it was discovered that trying to find a high correlation coefficient of pixel areas on either side of the line of symmetry was a suitable model to reduce the search space reference shown in Figure 27.

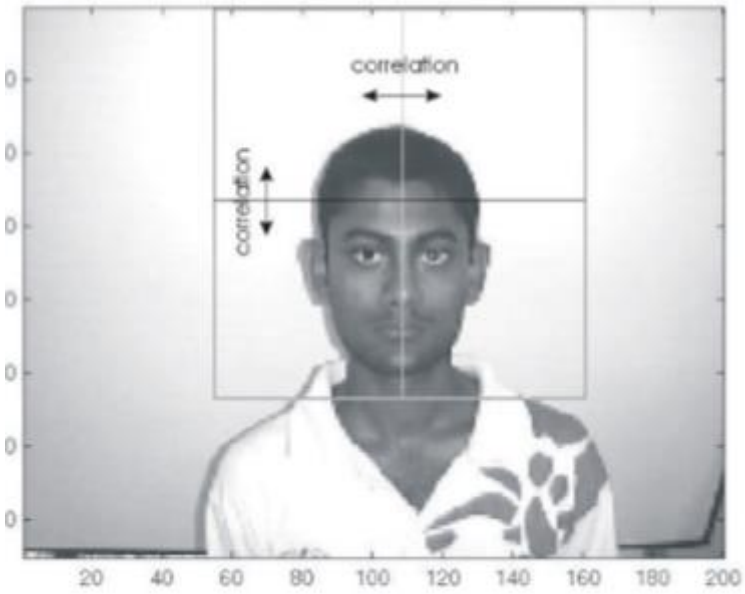
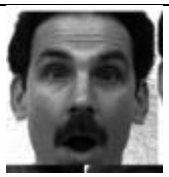
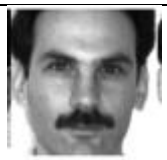



Figure 27 checking coorelation of vertical and horizontal pixel areas

Using correlation as a statistic of similarity is preferable to comparing the two pixel areas on the left and right of the (tested) line of symmetry. This is because the correlation coefficient measures the strength of the relation between the two pixel areas (Frank and Althoen, 1994) and is therefore less sensitive to lighting variations between the left and right halves. The element (i,j)of the correlation coefficient matrix is related to the corresponding element of the covariance matrix .

X and Y are the two pixel regions on either side of the (tested) line of symmetry.To find the best line of symmetry (highest correlation coefficient) a window is run from left to right on the upper edge of the image and the window position, which resulted in the highest correlation coefficient, was regarded as containing the best line of symmetry frontal view face images. After experimenting with this model it was found that there may be several areas in an image which display a high degree of symmetry around a vertical axis. Often a plain (even) background would result in a undesirable line of symmetry. Therefore it was decided to look for areas (window positions in Figure 28) with high correlation around a vertical axis and low correlation around a horizontal axis .

Table 2 process of successful detection

| | |
|---|---|
|  |  |
| Subject 01 | Subject 01 |
|  |  |
| Subject 02 | Subject 02 |
|  |  |
| Subject 03 | Subject 03 |
|  |  |
| Subject 04 | Subject 04 |

When searching for a face's line of symmetry the particular pixel window that is considered has to be moved from the upper left to upper right of the image. The optimal distance that the window is moved has to be determined. There is a speed versus accuracy trade off here, with small 'jumps' of the window finding a the exact line of symmetry while large 'jumps'being much faster in finding an approximate solution since less pixel windows are being evaluated. A perfect face detection algorithm which takes an hour to finish processing would have no real-world application. These decisions should be made depending on the platform the face detection system is implemented an exhaustive search (limited only by the maximum and minimum pixel window sizes) was done down the line of symmetry.

Some face detection examples are given belowIt is obvious that the face detection output, on the lower right of Figure 3.24 is inaccurate. Here the best 'faceness' heuristic value does not correspond to the best face location, therefore an incorrect pixel area was segmented. The other three testing results are accurate to a high degree and may be called successful frontal view

face detections. In chapter four the face detection system will be further optimised to reject false face detections and improve face detection accuracy.

### 3.3.5 Manual Face Detection

Besides a fully automated frontal view face detection system, a manual frontal view face detection system was also implemented. This was done because of the obvious improved accuracy a human operator could provide to the face detection process. In the system, the operator is asked to manually locate positions just under the subject's eyes, and the system thereby determines the exact face location.In the ideal frontal like Figure 29 as reference.

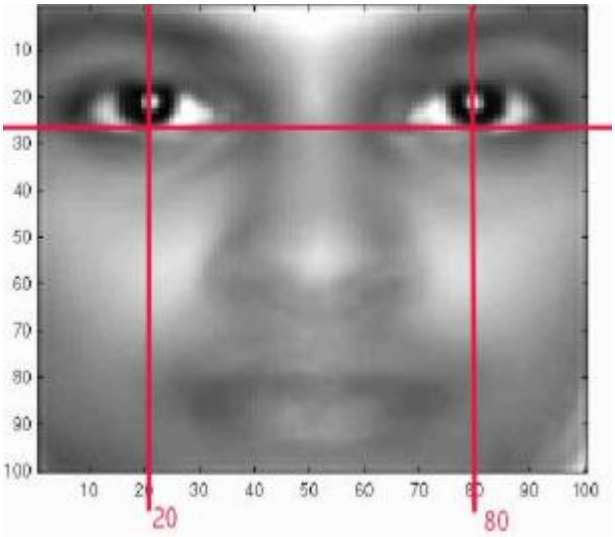
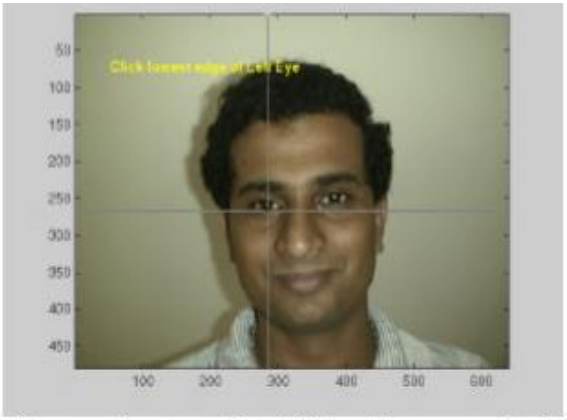


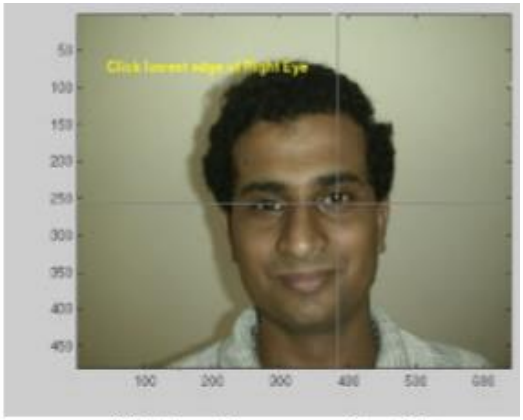

Figure 28 average face in gray scale using manual face detection control points

view segmented facial image for face recognition, the lower edge of each eye is 27% from the top of the image and the left and right eyes (operator's view point) are 20% and 80% from the left border of the image respectively Average face in gray scale with manual face detection control points.

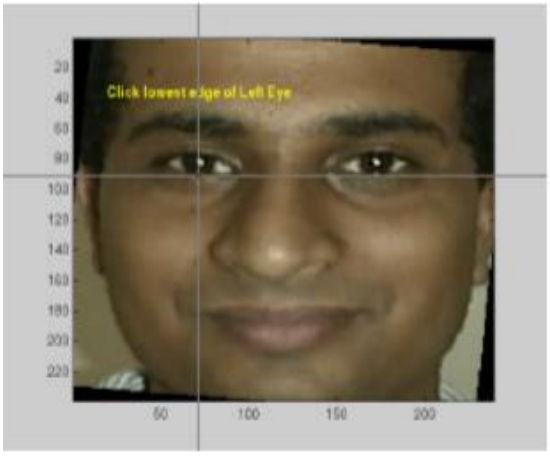
Here are some (Figure30) ideal facial area to segment using manual face detection was determined by examining the average face of 30 test subjects (the same average face was used to develop the deformable template). This segmented pixel area, which should be better than that obtained using fully automated face detection, can be used for automated face recognition



Operator instructed to click under eye on the left



Click under eye on the right



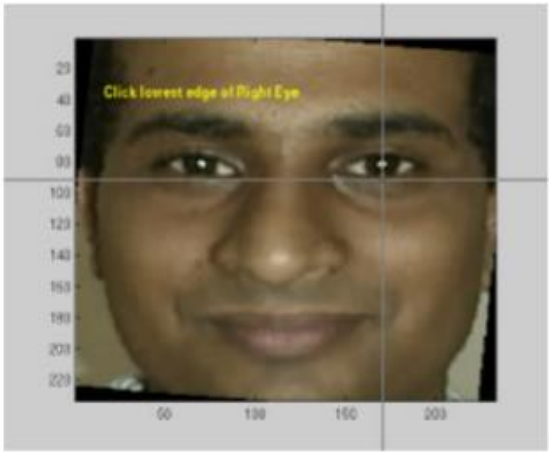




Figure 29 different mode of dtection

It would also have been possible to use more control points to determine the position of a subject's face in an image. For example, an operator can be instructed to click just under a subject's left and right eye, and then on the left and right edges of the subjects nose etc. However, the researcher decided to just use a single However, the researcher decided to just use a single statistic (vector between lower edge of eyes) so as not to lose the natural variation between human faces. For example, if individual A has a long nose and individual B has a short nose, and if a manual face detection system centres and normalises face images based on eye width and nose length, a large part of the variation between the individual A's and B's faces is lost. Using a single statistic this does not occur. However, when one uses several control points to manually detect a face, the segmented face area would probably be more exact and the reader is encouraged to experiment with different combinations and test the face recognition accuracy of the segmented pixel area.

## 3.4 System architecture

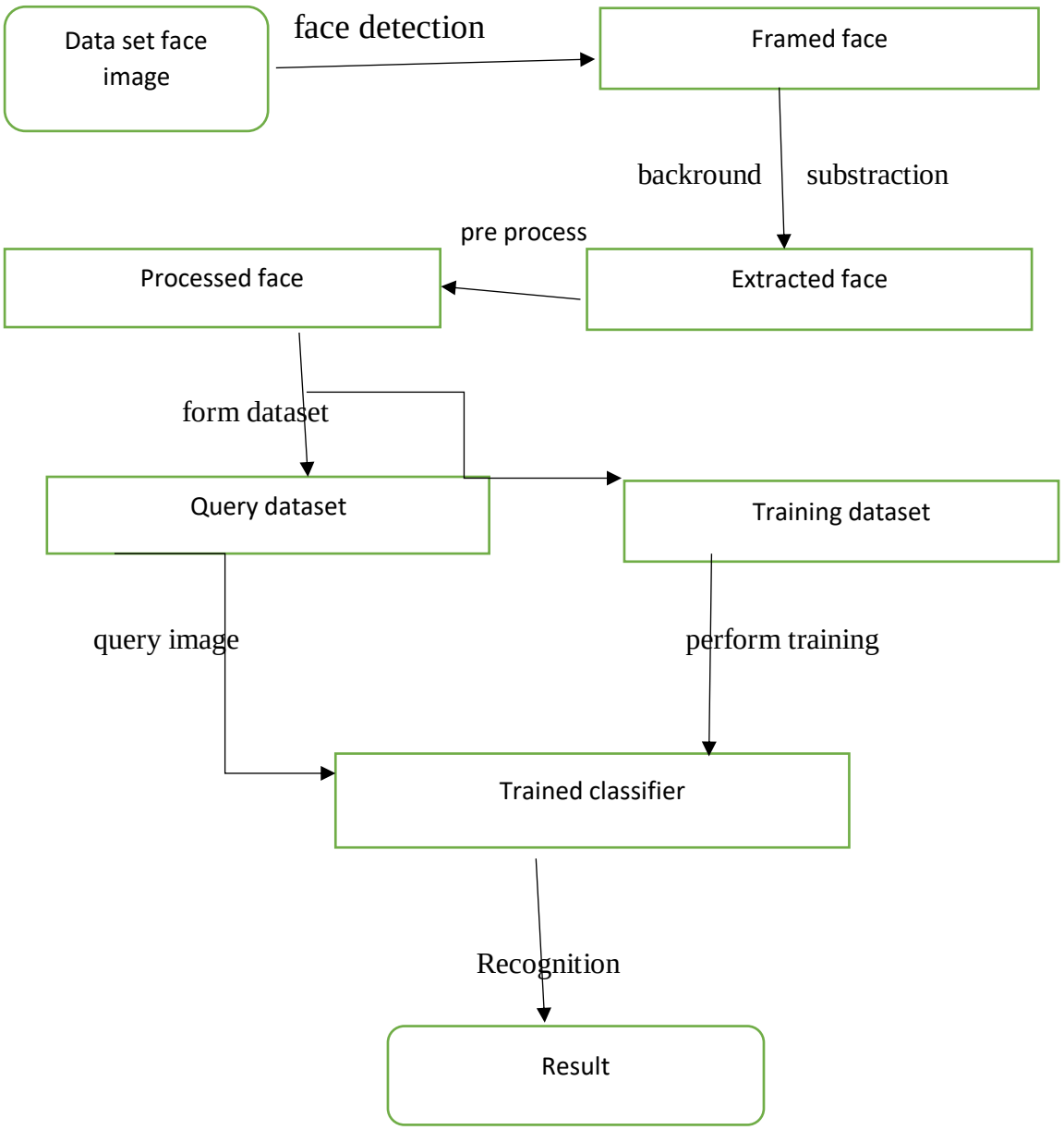



Figure 30 System architecture

## 3.5 System design:

In system design phase, we discuss about UML diagrams. Which represents the design of the proposed system,

Above diagram represents the system design in class model

Class diagram :-

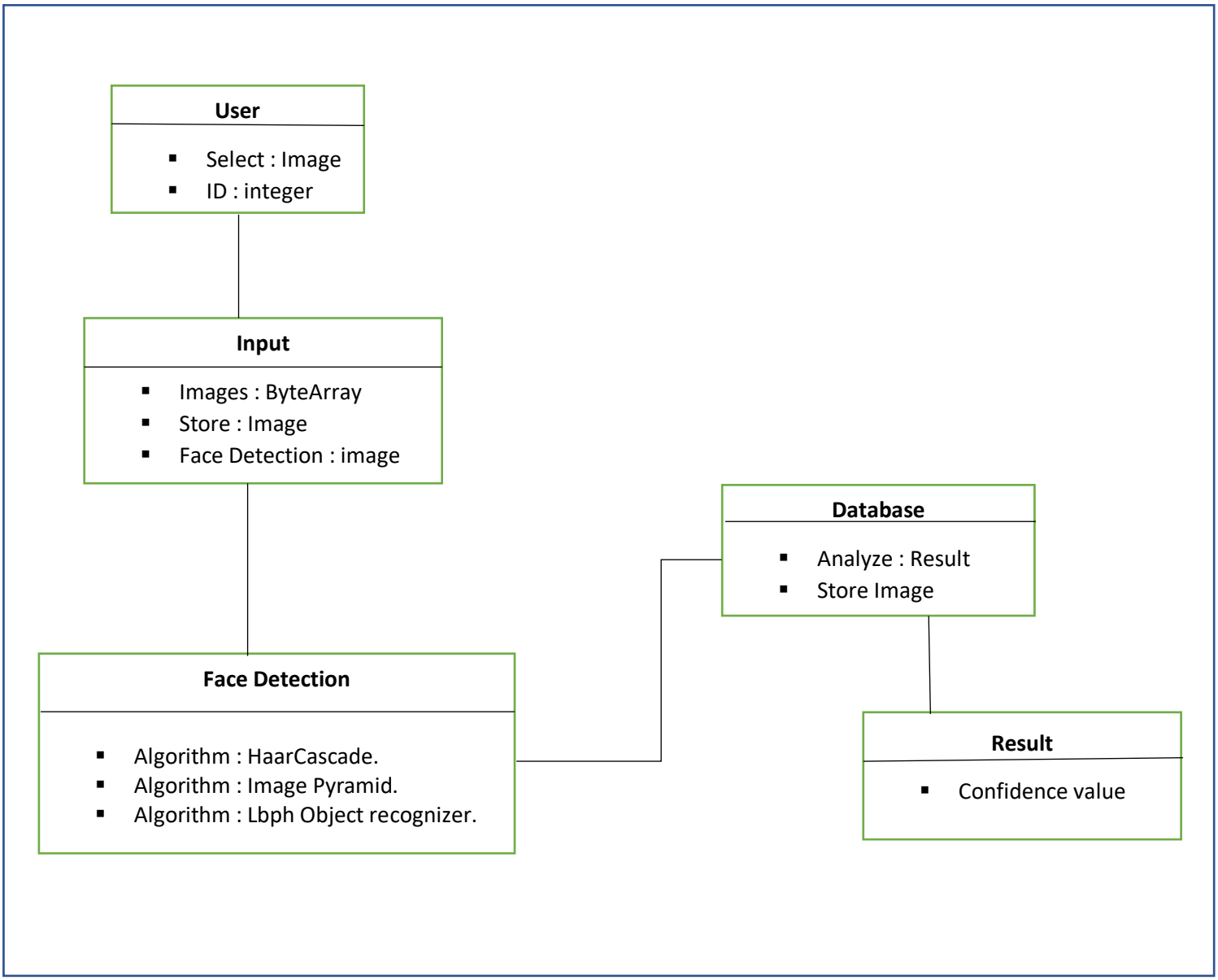



Figure 31 class diagram

**Use-case diagram**

In this diagram we represent the system design in use case model

It define the system in simple understanding format, so it is easy to read the process of the proposed system

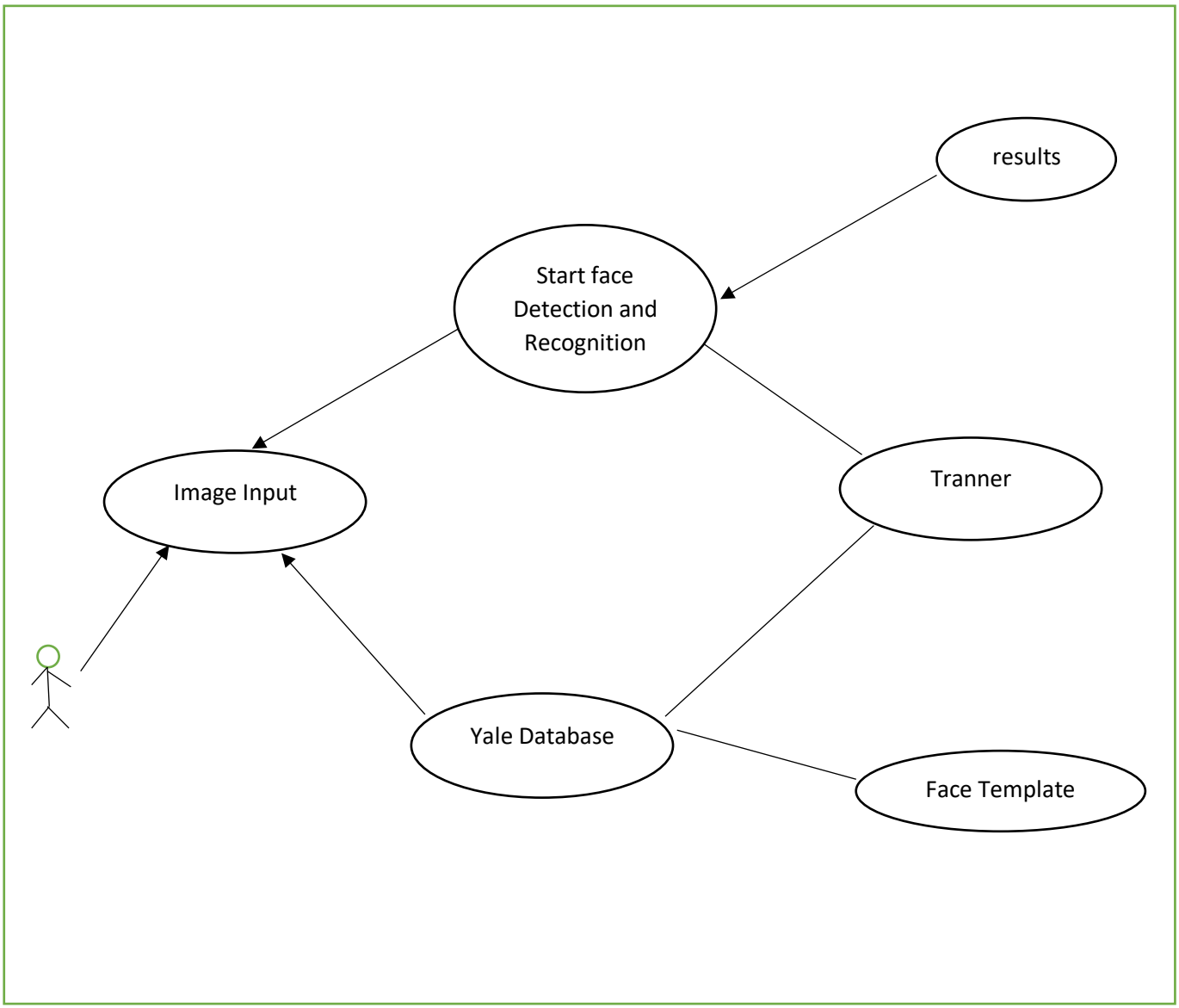



Figure 32 use case diagram

**Sequence diagram**

Representation of system design in sequence diagram

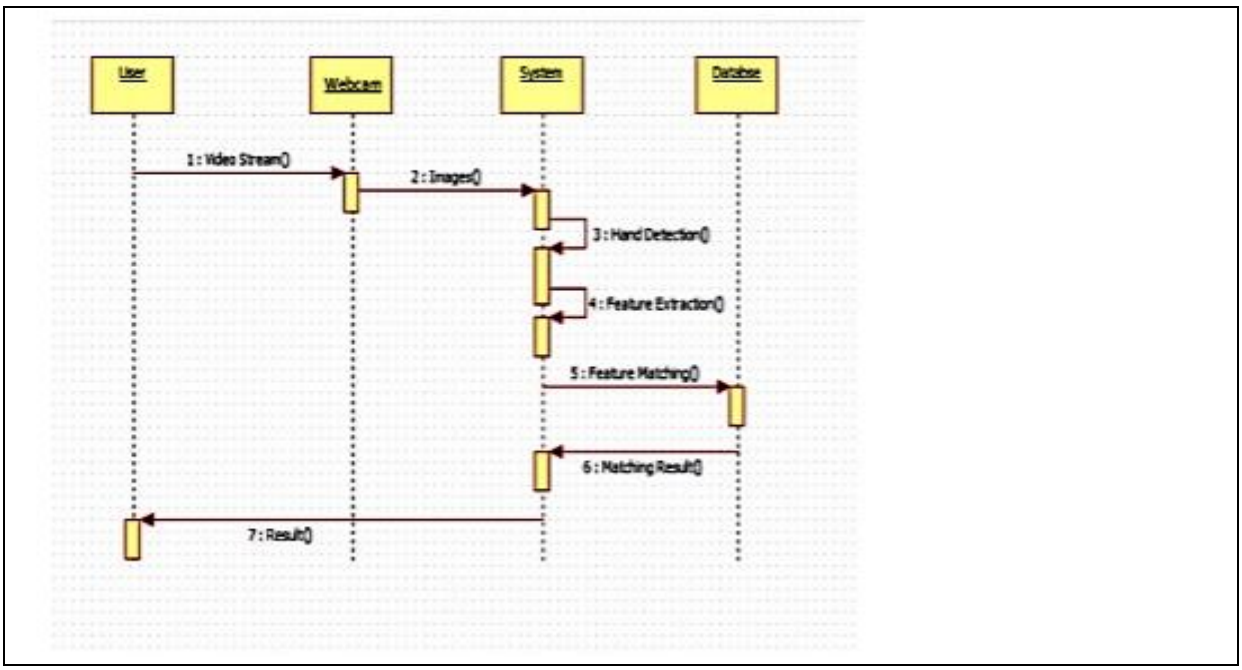



Figure 33 sequence diagram

**Activity diagram**

System design in shown in activity model as shown below

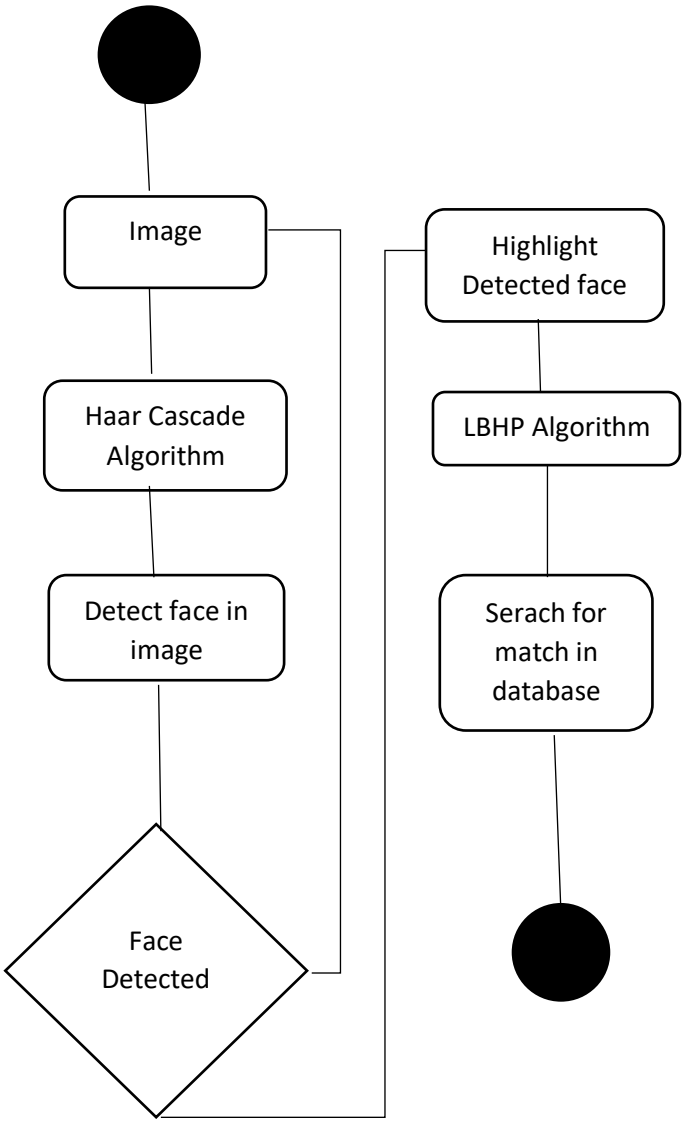



Figure 34 activity diagram

**Component Diagram**

Defining the system design in component diagram model as shown below

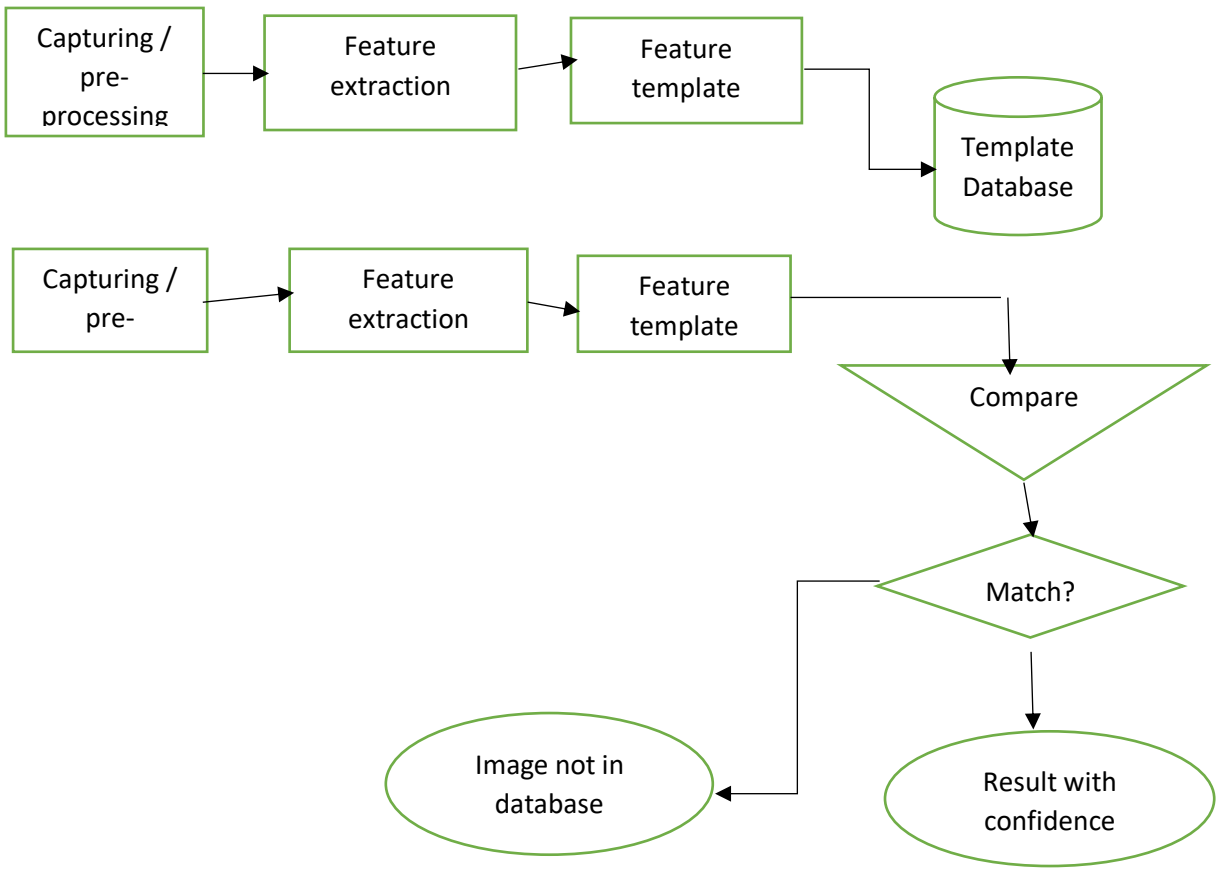



Figure 35 component diagram

### 3.6 Conclusion

In this chapter we have discussed about the algorithms used for face recognition and face detection, also the design of the system like UML diagrams system architecture etc., generally this module says about the implementation face of face recognition using template matching methods.

# CHAPTER 4

# COMPARISON AND OBSERVATION

## 4.1 Introduction

Face Recognition System (FRS) process can be subdivided into two main parts. The first part is image processing and the second part is recognition techniques. The image processing part consists of face image acquisition through scanning, image enhancement, image clipping, filtering, edge detection and feature extraction . The second part consists of the artificial intelligence which is composed by genetic algorithms and there are many approaches for face recognition. Feature-based approaches first process the input image to identify and extract (and measure) distinctive facial features such as the eyes, mouth, nose, etc., as well as other fiducial marks, and then compute the geometric relationships among those facial points, thus reducing the input facial image to a vector of geometric features. Standard statistical pattern recognition techniques are then employed to match faces using these measurements.

However, if the facial features are manually extracted, it is reasonable to assume that the recognition performance would have been much lower if an automated, and hence less precise, feature extraction method had been adopted. In general, current algorithms for automatic feature extraction do not provide a high degree of accuracy and require considerable computational capacity.

Beside the mentioned approach to face recognition, some researchers also used other methods to perform the studies on face recognition, i.e., the rules of the shape and albedo of a face under all possible illumination conditions, Bayesian decision, etc. In order to develop a universal face recognition system which can handle all face recognition factors, the integrated approach could be a choice. A method that integrates the above different methods and applies different techniques would be the answer to all the drawbacks

## 4.2 pseudo code

Here we discuss about the pseudo code of face recognition system,which s implemented using LBPH algorithm for face recognition and for detection haas cascade xml file for detection of face image

```
def get_images_and_labels(path):
    # Append all the absolute image paths in a list image_paths
    # We will not read the image with the .sad extension in the training set
    # Rather, we will use them to test our accuracy of the training
    image_paths = [os.path.join(path, f) for f in os.listdir(path) if not f.endswith('.sad')]
    # images will contains face images
    images = []
    # labels will contains the label that is assigned to the image
    labels = []
    for image_path in image_paths:
        # Read the image and convert to grayscale
        image_pil = Image.open(image_path).convert('L')
        # Convert the image format into numpy array
        image = np.array(image_pil, 'uint8')
        # Get the label of the image
        nbr = int(os.path.split(image_path)[1].split(".")[0].replace("subject", ""))
        # Detect the face in the image
         faces = faceCascade.detectMultiScale(image)
        # If face is detected, append the face to images and the label to labels
        for (x, y, w, h) in faces:
            images.append(image[y: y + h, x: x + w])
            labels.append(nbr)
            cv2.imshow("Adding faces to traning set...", image[y: y + h, x: x + w])
            cv2.waitKey(50)
```

```
# return the images list and labels list

return images, labels
```

- **Preparing the training set**

We pass the get_images_and_labels function with the path of the database directory. This path has to be the absolute path. This functions returns the features (images) and labels (labels) which will be used to train the face recognizer in the next step.

```
# Path to the Yale Dataset

path = 'yalefaces'

# The folder yalefaces is in the same folder as this python script

# Call the get_images_and_labels function and get the face images and the

# corresponding labels

images, labels = get_images_and_labels(path)

cv2.destroyAllWindows()
```

- **Perform the training**

We perform the training using the FaceRecognizer.train function. It requires 2 arguments, the features which in this case are the images of faces and the corresponding labels assigned to these faces which in this case are the induvidual number that we extracted from the image names.

```
# Perform the training

recognizer.train(images, np.array(labels)
```

- **Testing the face recognizer**

We will test the results of the recognizer by using the images with .sad extension which we had not used earlier. As done in the get_images_and_labels function, we append all the image names with the .sadextension in a image_paths list. Then for each image in the list, we read it in grayscale format and detect faces in it. Once, we have the ROI containing the faces, we pass the ROI to the FaceRecognizer.predict function which will assign it a label and it will also tell us how confident it is about the recognition. The label is an integer that is one of the individual

numbers we had assigned to the faces earler. This label is stored in nbr_predicted. The more the value of confidence variable is, the less the recognizer has confidence in the recognition. A confidence value of 0.0 is a perfect recognition.

```
# Append the images with the extension .sad into image_paths
image_paths = [os.path.join(path, f) for f in os.listdir(path) if f.endswith('.sad')]
for image_path in image_paths:
predict_image_pil = Image.open(image_path).convert('L')
predict_image = np.array(predict_image_pil, 'uint8')
faces = faceCascade.detectMultiScale(predict_image)
for (x, y, w, h) in faces:
    nbr_predicted, conf = recognizer.predict(predict_image[y: y + h, x: x + w])
    nbr_actual = int(os.path.split(image_path)[1].split(".")[0].replace("subject", ""))
    if nbr_actual == nbr_predicted:
        print "{} is Correctly Recognized with confidence {}".format(nbr_actual, conf)
    else:
        print "{} is Incorrectly Recognized as {}".format(nbr_actual, nbr_predicted)
    cv2.imshow("Recognizing Face", predict_image[y: y + h, x: x + w])
    cv2.waitKey(1000)
```

We check if the recognition was correct by comparing the predicted label “nbr_predicted” with the actual label “nbr_actual”. The label “nbr_actual” is extracted using the os module and the string operations from the name of the image. We also display the confidence score for each recognition.

## 4.3 Comparision between all techniques

Table 3 comparision table

| Criteria | description | Implemented using | application | accuracy | advantages |
|---|---|---|---|---|---|
| Eigen face | In the usage of computer vision ,eigen vector are represented with names | Programming, python Package, opencv OpenIMAJ | Use of eigen faces for recognition | Speed and efficiency are high,gray(72.12%) | Computation of efficiency when the eigenfaces stored is not much dimension |
| Fisher face | Authentication of the object For which the technology is used | C++,java, python Matlablib, GNU octave, | Social media,ID verification solution,policing, National security | Fisher face gray(83.9%), high accuracy compare to eigen face | insensitive to the maximum variation in lighting direction and facial expression. |
| PCA | Orthogonal linear transformation that transforms the data to a new coordinate source such that is greatest variance by some projection | Matplotlib, Linear algebra, | Quantitative finance, neuroscience | Performance of the pca is more accurate compare to all the methods | Linearity, Large variance implies more structure, Orthogonality |

| | | | | | |
|---|---|---|---|---|---|
| LDA | Analysis of discriminant function is a generalization to Fisher's linear discriminants | Matplotlib, numpy, Pandas, seaborn | Biomedical studies, earth science, marketing | Accuracy and rate has greater level,,and it has lower computational time,,t has weaker tolerance than EBGM | Discriminant analysis works by creating one or more linear combinations of predictors, |
| HMM | It is simplest of dynamic Bayesian network process | Viterbi, FBBW | Computational finance, partial discharge, gene prediction, protein folding | Accuracy greater the pca | The object is directly visualized to observer |

## 4.4 Data base used

The database contains 5760 single light source images of 10 subjects each seen under 576 viewing conditions (9 poses x 64 illumination conditions). For every subject in a particular pose, an image with ambient (background) illumination was also captured. Hence, the total number of images is in fact 5760+90=5850. The total size of the compressed database is about 1GB.

The 65 (64 illuminations + 1 ambient) images of a subject in a particular pose have been "tarred" and "gzipped" into a single file. There are 90 (10 subjects x 9 poses) '*.tar.gz' files. Each '*.tar.gz' file is about 11MB big. All filenames begin with the base name 'yaleB' followed by a two digit number signifying the subject number (01 - 10). The 2 digit number after '_P' signifies the pose number (00 - 08). (See below for the relative pose positions.) The images in each '*.tar.gz' file can be unpacked using the following two commands (under Unix)

The coordinates of faces in each set (e.g., 'yaleB01_P00.tar') can be found here. For the set 'yaleB01_P00.tar', for example, the coordinates are in the file 'yaleB01_P00.crop'. Each 'yaleB**_P**.crop' file contains two columns corresponding to the x- and the y-coordinates.

For all the sets in the frontal pose (i.e., for the files 'yaleB**_P00.tar') the coordinates of the left eye, right eye, and mouth in each image have been appended on top of each other into two columns of length 195. The top 65 rows are for the left eye, the next 65 are for the right eye, and the rest are for the mouth centers. Files other than for the frontal pose (e.g., 'yaleB01_P07.crop') contain only the coordinates of the face centers (i.e., columns have a length of 65). As a final note, each of the 65 rows in the 'yaleB**_P0*.crop' files correspond (in the same order) to the images whose filenames appear in the file 'yaleB**_P**.info'. This '*.info' is unpacked together with the images in 'yaleB**_P0*.tar'.now, a word about the naming of each image: The first part of the filename of an image follows the same convention as the filename of one of the "tarred" (and "gzipped") files. It begins with the base name 'yaleB' and is followed by the two digit number signifying the subject number and then by the two digit number signifying the pose. The rest of the filename deals with the azimuth and elevation of the single light source direction. For example, the image with the filename as shown in Figure36.

Figure 36 images of database

## 4.5 Result and observation

### a) Manual face detection and automated face recognition

These tests will investigate the robustness and accuracy of the lbph

Analysis frontal-view face recognition system. The 27 images of condition A (as described earlier) were manually face detected and lbph transformed, creating a face database of known images. The author divided the condition of images submitted for recognition into three types,

1) Perfect conditions - the known images are submitted again

2) Good conditions - manual face detection(again) of condition A images

3) Normal/Adverse conditions - manual face detection of condition B images.

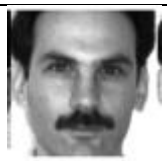
The faces in the images were of different sizes, positions and many subjects had even slightly rotated their heads. Further more, lighting conditions were intentionally varied by

the researcher to strain the recognition system. While the FRS performance is strong on perfect and good inputs of images its perform on 'normal/adverse' condition image could be good.here we used 22 images to test which are perfect enough in which the successful recognition are of 22 percent, lower eigen value of eigenfaces are the most useful for recognition of image, if the eigen values are high in rate then there is a huge chance of perfect recognition

Table 4 observation table

| Conditionof image | Total tested | Tested successful | Recognitionfailures |
|---|---|---|---|
| perfect | 22 | 27(100%) | 0 |
| good | 22 | 25(93%) | 2(7%) |
| nominal | 80 | 58(53%) | 22(22%) |

Table 5 results

| | |
|---|---|
|  |  |
| Subject 01 | Subject 01 |
|  |  |
| Subject 02 | Subject 02 |
|  |  |
| Subject 03 | Subject 03 |
|  |  |
| Subject 04 | Subject 04 |

- **Observation**

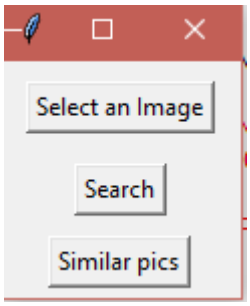
Simply attempting the recognition of the output of the fully automated face detection system resulted in a recognition rate close to zero (0%). Linear binary pattern is extremely sensitive to even slight variations in scale, position and rotation so it is extremely unlikely that the extracted segment would be suitable for recognition.

Slightly moving the extracted segment around in the hope of getting better recognition results was viewed as a crude solution to the problem and also was too computationally expensive to implement on the testing platform. The author investigated using the Hough transform after edge detection on the extracted segment as a means of eye detection. Detecting the eyes would enable normalising the face (in fact, extracting the exact face position from the segment) thus vastly improving recognition accuracy. Eye detection would let us rotate the face to make the eye horizontal and scale the face exactly to match the calculated average face. Unfortunately sufficient literature was not available to understand and implement such a system.

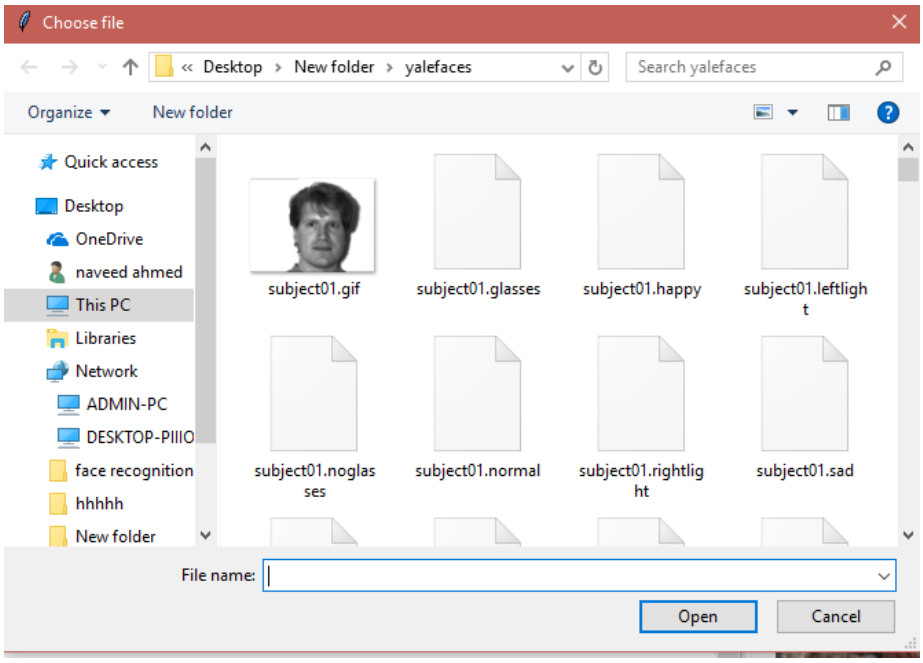
## ➢ Result

- Training data set is the initial process of execution

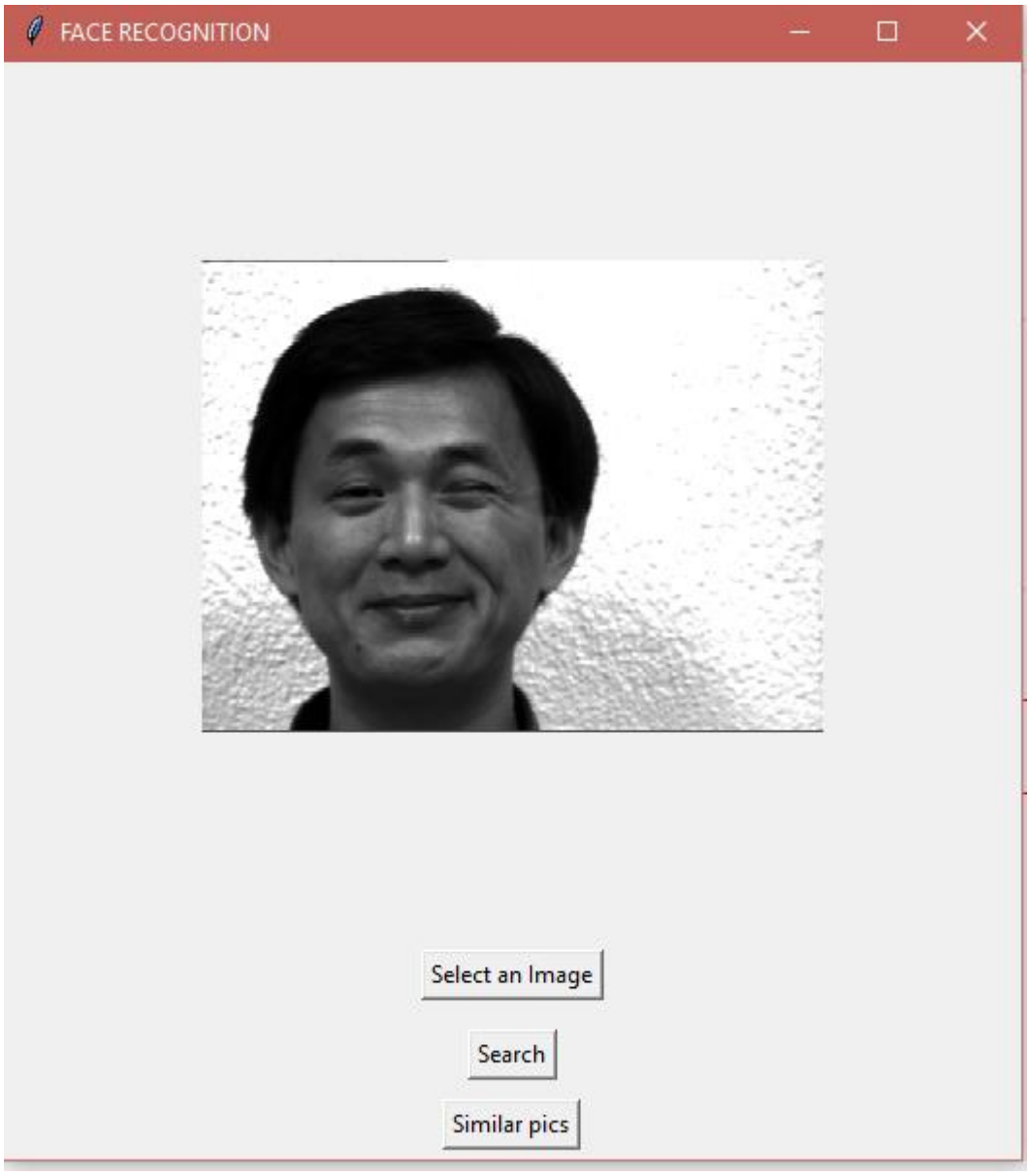
GUI for selecting image




Select image from the file

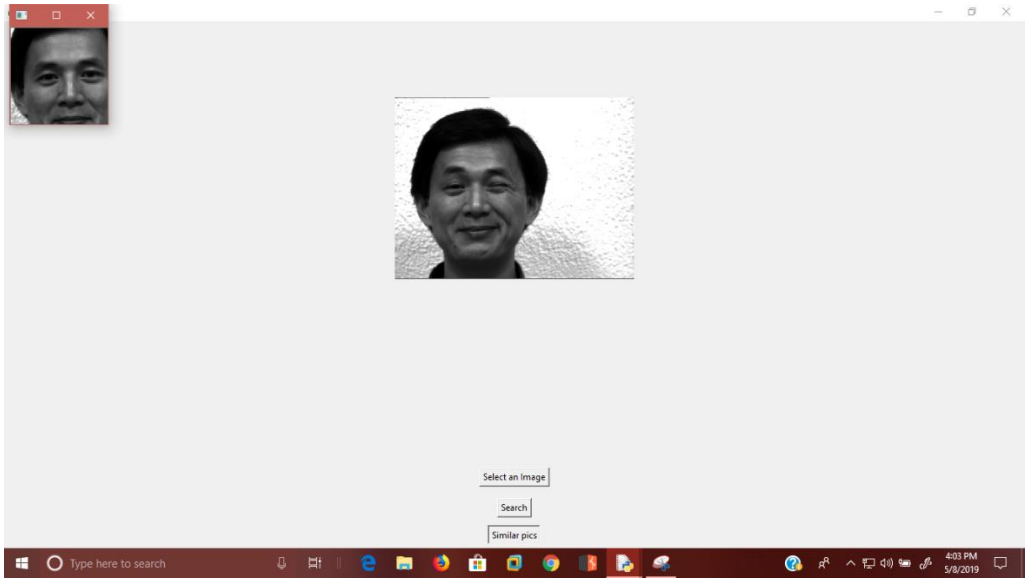


Displaying the selected image

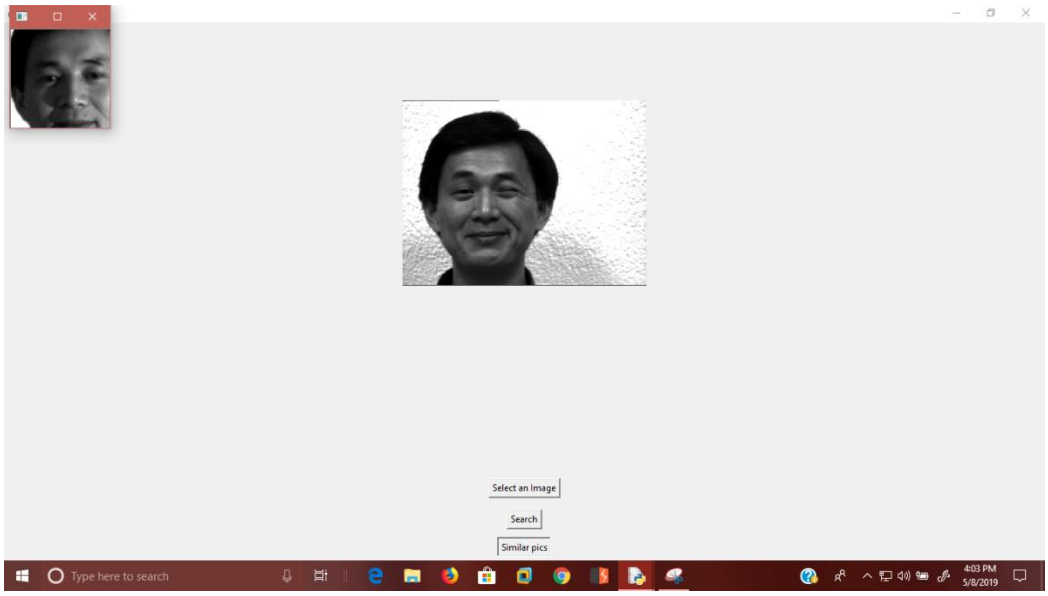


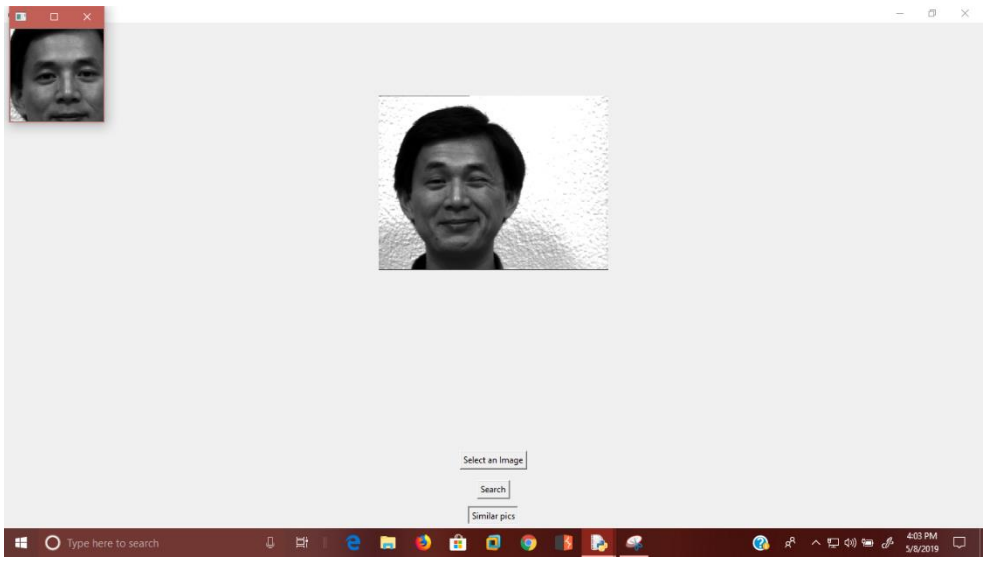
Confidence value of a input image

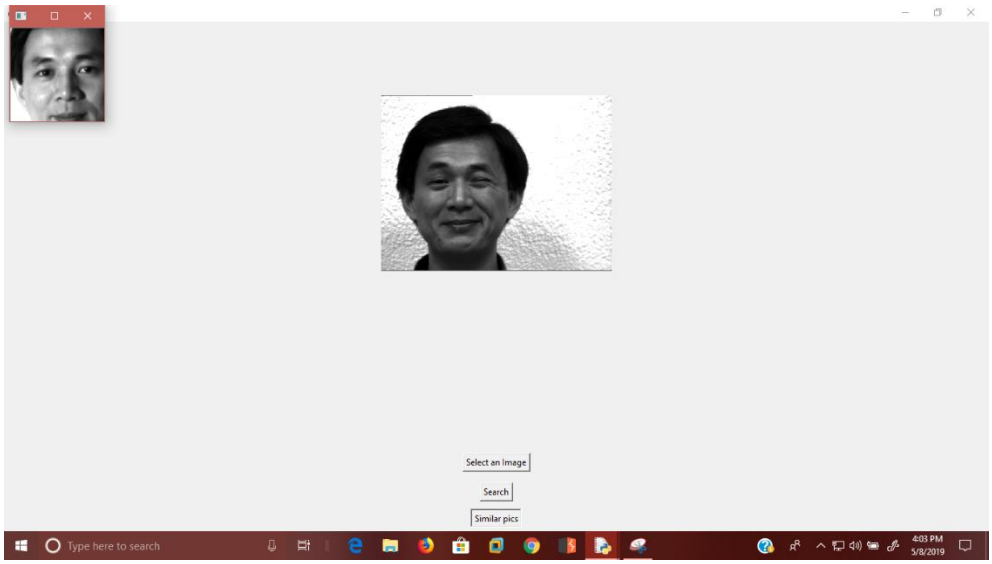
```
============== RESTART: C:\Users\hp\Desktop\New folder\browse.py ==============
>>> 6 is Correctly Recognized with confidence 28.60643286755775
```

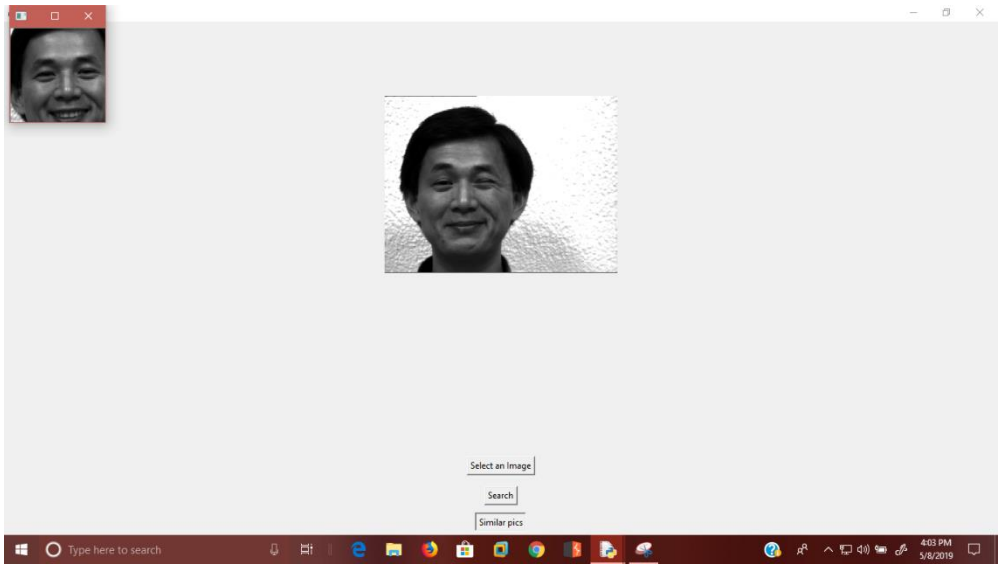


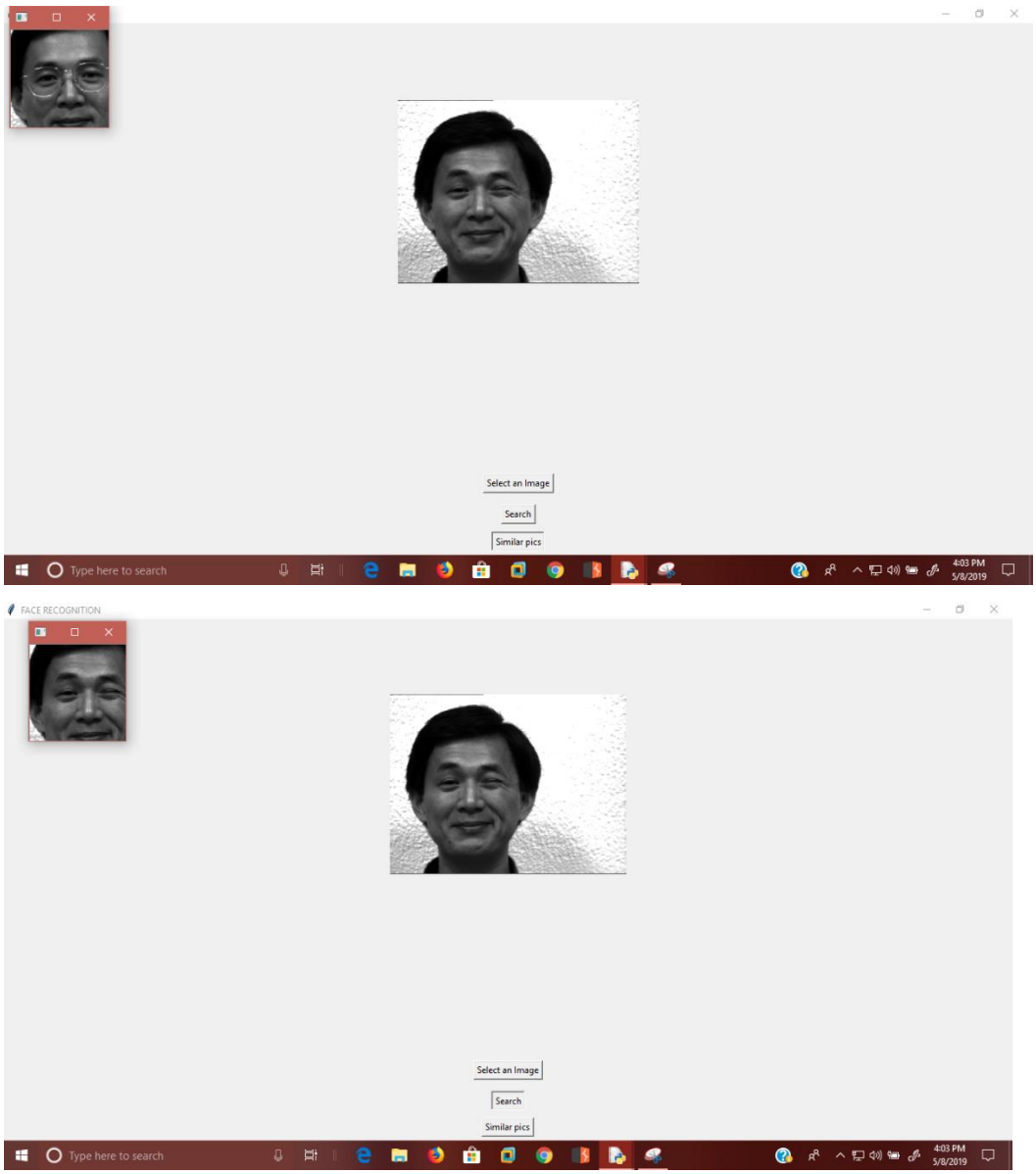

Similar images of selected image




Select an Image
Search
Similar pics
Type here to search
4:03 PM
5/8/2019


Select an Image
Search
Similar pics
Type here to search
4:03 PM
5/8/2019


Select an Image
Search
Similar pics
Type here to search
4:03 PM
5/8/2019




## 4.6 conclusion

In the LBP approach for texture classification, the occurrences of the LBP codes in an image are collected into a histogram. The classification is then performed by computing simple histogram similarities. However, considering a similar approach for facial image representation results in a loss of spatial information and therefore one should codify the texture information while retaining also their locations. One way to achieve this goal is to use the LBP texture descriptors to build several local descriptions of the face and combine them into a global description. Such local descriptions have been gaining interest lately which is understandable

given the limitations of the holistic representations. These local feature based methods are more robust against variations in pose or illumination than holistic method

we have discussed the pseudo code for face recognition and also comparision between existing techniques We have checked theoretical as well as statistical aspect of the three different statistical approach face orientations recognition algorithm (Eigenface, Lbph and Elastic Bunch Graph Matching). Finally, we have made a comparison of these algorithms and have discussed the advantages and Disadvantages of each of them. eigenface and elastic bunch graph matching. For the elastic bunch graph matching, it makes use of Gabor feature, which are insensitive to lighting variation, rigid, and deformable matching. This allows for the position and facial expression variation, because features only taken from a key points in the face image rather than the whole images. about 70 graphs. With this algorithms the face is represented as a graph. The two-dimensional face description method has been extended into spatiotemporal domain depicts facial expression description using LBP. Excellent facial expression recognition performance has been obtained with this approach.

# CHAPTER 5

# CONTRIBUTION AND SCOPE

## 5.1 Future extention

The face database of the implemented system should be expanded to as many individuals as possible. Face recognition using Linear binary pattern promises to be highly scalable so recognition of even over 10000 individuals should be possible. This is in contrast to a traditional neural network based technique. However, the real-world problems that arise when expanding the face database can only be found out by actually experimenting with a large face database.

Additional test subjects are also needed because there were insufficient eigenfaces calculated to perform recognition accurately. As we increase the size of the face database the recognition accuracy of the system should increase. In the author's opinion a minimum of 40 eigenfaces should be used for face recognition. An eye detection system should be implemented to further normalise the area segmented by the face detection system. If the subject's eyes were accurately identified, the image could be transformed to make the eyes horizontal and the face scaled to a constant proportion of the segmented pixel area. This would let us implement a fully automated face detection and recognition system since Lbph is far too sensitive to scale, rotation and shift, to perform recognition with the raw output from a face detection system. Fortunately there are well known techniques for circle detection which can also be used for eye detection.

Only the whole-face template was used for frontal view face recognition. The template matching strategy that was used can be expanded to use the whole face, left eye, right eye, nose and mouth templates for recognition. The problems associated in extracting these template areas and assigning weights to the Euclidean distances found by each template(closest known left eyes, closest known right eyes, etc) can be investigated.

If a real-world implementation of this system is needed, the face detection system should be optimised to suit the specific platform. There are several speed versus accuracy trade-off decisions which will have to be reconsidered depending on each computing environment that is used. Besides the pose invariant recognition scheme that was implemented, recognition even with facial expressions is possible by taking multiple images of each known individual with a range of facial expressions.

### 5.1.1 Transforming frontal view face images for pose invariant face recognition

While the results for pose invariant face recognition were quite strong, the face database contained nine known faces for each individual to be identified. With more known data points in face space, recognition performance increased sharply. Unfortunately having nine known images per individual is not realistic (in most real-world situations only a single frontal view face image is presented as the known image), and therefore a great deal of research has been done on transforming frontal view face images to create 'virtual views' of the other poses. These 'virtual views' can then be added to the face database as additional known faces of an individual from different poses. There are two approaches to transforming or rotating human faces.

**a) 3D modelling techniques**

These involve transforming the facial image to a 3D model, rotating the model and transforming the face back into a (2D) image. It is a 2D to 3D to 2D transformation. While this technique produces accurate results many implementations of this strategy require a frontal and a profile image per individual to create the 3D model (Lee and Thalmann, 1998) and is therefore once again not practical..

**b) 2D transformations**

The other approach to the problem is to use a 2D to 2D parallel deformation to rotate the (2D) frontal view face image. Beymer and Poggio (1995) used an example based strategy to create prototype transformations from a frontal view face to a angled pose. These were then used in a pose invariant face recognition system

## 5.2 summary of contribution

In this project we discuss about face recognition using LBPH algorithm using pakages OpenCV ,python, numpy, using tool IDLE(Integreted development learning environment)

Firstly we discuss about LBPH algorithm-is a two-dimensional face description method has been extended into spatiotemporal domain (Zhao and Pietikäinen 2007). depicts facial expression description using LBP-TOP. Excellent facial expression recognition performance has been obtained with this approach. to the traditionally divergent statistical and structural models of texture analysis. Perhaps the most important property of the LBP operator in real-world applications is its robustness to monotonic gray-scale changes caused, for example, by

illumination variations. Another important property is its computational simplicity, which makes it possible to analyze images in challenging real-time settings.

Secondy we discuss about the training algorithm which is choosen for project Training the Algorithm: First, we need to train the algorithm. To do so, we need to use a dataset (yale databse) with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let's see the LBPH computational steps.

Next we discuss about applying LBPH algorithm The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters radius and neighbor.

Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original imagea t the end of this procedure (LBP procedure), we have a new image which represents better the characteristics of the original image.

Next here in this stage it perform face detection process which detects the face of person by given input to system detected faces leads to face recognition, it basically compares the input facial image with the facial image related to the user which is requiring the authentication, input facial image with all facial images from a dataset with the aim to find the user that matches that face. It is basically a 1xN comparison.

Next Performing the face recognition in this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, we perform the steps again for this new image and creates a histogram which represents the image. so to find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram.

We can use various approaches to compare the histograms (calculate the distance between two histograms), for example: euclidean distance, chi-square, absolute value, etc. In this example, we can use the Euclidean distance (which is quite known) based on the following formula.

So the algorithm output is the ID from the image with the closest histogram. The algorithm should also return the calculated distance, which can be used as a 'confidence' measurement. Note: don't be fooled about the 'confidence' name, as lower confidences are better because it means the distance between the two histograms is closer.we can then use a threshold and the 'confidence' to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized if the confidence is lower than the threshold defined

## 5.3 Conclusion

We have proposed and implemented a face recognition using template matching approach using openCV. In our approach, we don't assume limiting assumptions of rigid (or approximately rigid) surfaces but have demonstrated with faces having widely different facial expressions for each human subject. Four different facial expressions are used. The extracted range data sets are registered together, the registration being defined by the rigid parts of the face. In this way, a library of faces can be built up. Recognition of a test face is relative fast with the efficient indexing into the model library. The most appropriate models are ranked according to their similarity with the test scene. Verification of each model proceeds according to their ranking. In this way, the correct model face can be quickly and efficiently identified. Experimental results involving six human subjects, each with four different facial expressions, have demonstrated the validity and effectiveness of our algorithm. Since a deformable template match requires several thousand times more computation than computing a match score between the extracted features and each template, significant computational savings result. The degradation of recognition performance is relatively modest and may be outweighed by the reduction in computation. Computational burden is a significant issue for automatic video recognition and indexing systems which are required to process large volumes of image data, ideally in real time or faster. It is possible that the kind of reduction in that burden evidenced in the present work might make for increased practicality of this or related deformable template based recognition systems. not treating the unmodified system as a theoretical system with 100% nominal accuracy, but gauging the performance under real conditions, preferably with tracking operational.

The holistic approaches have the main advantage of distinctly capturing the most prominent features within the given facial images, so as to uniquely identify individuals amongst the given set also automatically finding features. However, disadvantages are that face recognition

performance could be drastically be affected by lighting, orientation and scale; or, features found from faces may not form part of the face but may be some other feature has been captured. For (example), features from the background of a facial image. So in order to develop a universal face recognition system which can handle all face recognition factors, the integrated approach could be a choice. The automated vision systems implemented in this thesis did not even approach the performance, nor were they as robust as a human's innate face recognition system,however, they give an insight into what the future may hold in computer vision

Therefore, when evaluating face-recognition technology in future, whether conventional single face-recognition product test, or largescale multiple face-recognition systems test, we should not only carry out the static face-recognition algorithm test, but also ought to carry out the dynamic applied face recognition test. Meanwhile, the impact of hardware configuration of face-recognition products or systems should be considered and their parameters should be paid close attention.

## References


[1]Sirovich, L. and M. Kirby, “Low-Dimensional procedure for the characterization of human faces”, J. Optical Soc. Am., 4: 519-524, 1987.

[2]Turk, M. and A. Pentland, “Eigenfaces for recognition”, J. Cognitive Neurosci., 3: 71-86, 1991.

[3] Pentland, B. Moghaddam and T. Starner, “View- Based and modular eigenspaces for face recognition”, Proceeding of IEEE CS Conference on Computer Vision and Pattern Recognition, pp: 84-91, 1994.

[4] Shermina, J., “Face recognition system using multi linear principal component analysis and locality preserving projection”, IEEE GCC Conference and Exhibition, 19-22 Feb., Stirling, UK, pp: 283-286, 2011.

[5] Dimitri, “Eigenface-based facial recognition”, available online at http://openbio.sourceforge.net/resources/eigenfaces/eigenfaceshtml/facesOptions.html , February 13, 2003.

[6] Sung K. and Poggio T., “Example-based learning for view-based human face detection”, A.I. Memo 1521, CBCL Paper 112, MIT, December 1994.

[7] Rowley H. A., Kanade T, “Neural Network-Based Face Detection”, IEEE Transactions on Pattern analysis and Machine Intelligence, Vol. 20,No. 1, pp. 23–30, 1998.

[8] Fraud R.. et al, “A Fast and Accurate Face Detector Based on Neural Networks”, IEEE Transactions on Pattern analysis and Machine Intelligence, Vol. 23, No. 1, pp. 42–53, 2001.

[9] Jafri R., Arabnia H.R., “A Survey of Face Recognition Techniques”, Journal of Information Processing Systems, Vol. 5, No. 2, June 2009.

[10] Chinese social and economic research center'”report on the research progress and trend of development biological recognition technology in China between 2011 and 2015” (In Chinese)

[11] P.Jonathon Philips, Partrick Rauss 掌 and S.Der.FERET(face recognition technology),ŷrecognition algorithm development and test report.' Technical Report TR 995,U.S. Army Reseach Laboratory,1996.

[12] P. J. Phillips, M. Hyeonjoon, S. A. Rizvi, and P. J. Rauss,ˇThe FERET evaluation methodology for face recognition algorithms,' IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 10, pp. 1090–1104,Oct. 2000.

[13] P. J. Phillips, P. Grother, R. J. Micheals, D. M. Blackburn, E. Tabassi, and J. M. Bone,ˇFace recognition vendor test 2002: Evaluation report,' Nat.Inst. Standards Technol., Gaithersburg, MD, Tech. Rep. NISTIR 6965,2003.

[14] Wen Gao, Senior Member, IEEE, Bo Cao, Shiguang Shan, Member, IEEE,Xilin Chen, Member, IEEE, Delong Zhou, Xiaohua Zhang, and Debin Zhao掌The CAS-

PEAL Large-Scale Chinese Face Database and Baseline Evaluations' IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, VOL. 38, NO. 1, JANUARY 2008

[15] P. J. Phillips, P. J. Flynn, T. Scruggs, K.W. Bowyer, J. Chang, K. Hoffman,J. Marques, J. Min, and W. Worek, "Overview of the face recognition grand challenge,'

# Appendix-A

## Software requirements

### IDLE

An integrated development environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development. An IDE normally consists of at least a source code editor, build automation tools, and a debugger. Some IDEs, such as NetBeans and Eclipse, contain the necessary compiler, interpreter, or both; others, such as Sharp Develop and Lazarus, do not.

The boundary between an IDE and other parts of the broader software development environment is not well-defined; sometimes a version control system or various tools to simplify the construction of a graphical user interface (GUI) are integrated. Many modern IDEs also have a class browser.

### Python lang

You as a user of Python should also be aware of the fact that this programming language has a copyright. The source code of Python is available under the GNU General Public License (GNP) just like it is available for Perl. You should also be aware of the fact that Python is maintained by a group of core development team members present at the institute. You will acquire more information as you move forward in this Python introduction tutorial. However, this does not mean that Guido Van Rossum is out of the picture as he still plays a vital role in the whole maintaining process.

### Packages

### Numpy

Numpy can be abbreviated as Numeric Python, is a Data analysis library for Python that consists of multi-dimensional array-objects as well as a collection of routines to process these arrays. In this tutorial, you will be learning about the various uses of this library concerning data science.

NumPy is a linear algebra library for Python, and it is so famous and commonly used because most of the libraries in PyData's environment rely on Numpy as one of their main building blocks. Moreover, it is fast and reliable

**OS**

The functions that the OS module provides allows you to interface with the underlying operating system that Python is running on – be that Windows, Mac.

**OpenCV-Python**

The Python API of opencv. It combines the best qualities of opencv C++ API and Python language. Since openCV is an open source initiative, all are welcome to make contributions to this library. And it is same for this tutorial also. So, if you find any mistake in this tutorial (whether it be a small spelling mistake or a big error in code or concepts, whatever), feel free to correct it.

And that will be a good task for freshers who begin to contribute to open source projects. Just fork the opencv in git hub, make necessary corrections and send a pull request to opencv. opencv developers will check your pull request, give you important feedback and once it passes the approval of the reviewer, it will be merged to opencv. Then you become a open source contributor. Similar is the case with other tutorials, documentation etc.

**Tkinter**

Python provides the standard library for creating the graphical user interface for desktop based applications developing desktop based applications with python Tkinter is not a complex task. An empty Tkinter top-level window can be created by using the following steps.

import the Tkinter module.

- Create the main application window.
- Add the widgets like labels, buttons, frames, etc. to the window.
- Call the main event loop so that the actions can take place on the user's

## Code

- **Face detection using haar cascade xml file**

```
import cv2, os
import numpy as np
import glob
import tkinter
from tkinter import *
from tkinter import filedialog
from PIL import Image, ImageTk
root = Tk()
root.title("FACE RECOGNITION")
# Code to add widgets will go here...

# For face detection we will use the Haar Cascade provided by OpenCV.
cascadePath = "haarcascade_frontalface_default.xml"
faceCascade = cv2.CascadeClassifier(cascadePath)

recognizer = cv2.face.LBPHFaceRecognizer_create()
#sizetuple=(320,243)
def get_images_and_labels(path):

    image_paths = [os.path.join(path, f) for f in os.listdir(path) if not
f.endswith('.sad')]
    # images will contains face images
    images = []
    # labels will contains the label that is assigned to the image
    labels = []
    for image_path in image_paths:
        # Read the image and convert to grayscale
        image_pil = Image.open(image_path).convert('L')
```

- Face recognition using LBPH algorithm

```
def recog():
    path = './input'
    # Append the images with the extension .sad into image_paths
    image_paths = [os.path.join(path, f) for f in os.listdir(path) ]#if
f.endswith('.sad')]
    for image_path in image_paths:
        predict_image_pil = Image.open(image_path).convert('L')
        predict_image = np.array(predict_image_pil, 'uint8')
        faces = faceCascade.detectMultiScale(predict_image)
        image_pil = Image.open(image_path).convert('L')
        image = np.array(image_pil, 'uint8')

        for (x, y, w, h) in faces:
        #cv2.imshow("Recace", predict_image[y: y + 130, x: x + 130])
            nbr_predicted, conf = recognizer.predict(predict_image[y: y +
130, x: x + 130])
            nbr_actual =
int(os.path.split(image_path)[1].split(".")[0].replace("subject", ""))
            if nbr_actual == nbr_predicted:
                print ("{} is Correctly Recognized with confidence
{}".format(nbr_actual, conf))
            else:
                print ("{} is Incorrect Recognized as
{}".format(nbr_actual, nbr_predicted))
            cv2.imshow("Recognizing Face", predict_image[y: y + 130, x: x
+ 130])
            cv2.waitKey(1300)
    cv2.destroyAllWindows()
C = tkinter.Button(root, text ="Search" , command = recog )
C.pack(padx=5, pady=5, side=BOTTOM)
```

# CASE STUDY

The whole process can be divided in three major steps -

1. The first step is to find a good database of faces with multiple images for each individual.
2. The next step is to detect faces in the database images and use them to train the face recognizer.
3. The last step is to test the face recognizer to recognize faces it was trained for.

## 1. Database

we use the Yale Face Database that contains 165 grayscale images of 15 individuals in gif format, There are 11 images for each individual. In each image, the individual has a different facial expression like happy, sad, normal, surprised, sleepy etc. so replace the jpeg format with different facial expressions.

## 2. Implementation

- Import the modules, here we use CV2, OS, NUMPY, PIL.

Cv2 - we will use this module for the functions of face detection and face recognition.

Os - we will use this module to extract the image names in the database directory and then from these names we will extract the individual number, which will be used as a label for the face in that image.

Numpy - Our images will be stored in numpy arrays.

PIL - we will use Image module from PIL to read the image in grayscale format.

- **Load the face detection Cascade**

For the purpose of face detection, we will use the Haar Cascade provided by OpenCV. The haar cascades that come with OpenCV are located in the directory of your OpenCV installation. We will use "**haarcascade_frontalface_default.xml**" for detecting the face. So, we load the cascade using the **cv2.CascadeClassifier** function which takes the path to the cascade xml file.

- **Create the Face Recognizer Object**

The next step is creating the face recognizer object. The face recognizer object has functions like FaceRecognizer.train to train the recognizer and FaceRecognizer.predict to recognize a face.

- **Create the function to prepare the training set**

Now, we will define a function "**get_images_and_labels**" that takes the absolute path to the image database as input argument and returns tuple of 2 list, one containing the detected faces and the other containing the corresponding label for that face.

For example, if the ith index in the list of faces represents the 5th individual in the database, then the corresponding ith location in the list of labels has value equal to 5.

We load the current image in a 2D numpy array image. We cannot read the images directly using cv2.imread because as of now, OpenCV doesn't support gif format images and unfortunately, our database images are in this format. So, we use the Image module from PIL to read the images in grayscale format and convert them into numpy arrays which are compatible with OpenCV. **From the image name, we extract the individual number. This** number will be the label for that face. We use "**CascadeClassifier.detectMultiScale**" to detect faces in the image.

- **Code of function to return images and labels**

```
def get_images_and_labels(path):
    # Append all the absolute image paths in a list image_paths
    # We will not read the image with the .sad extension in the training set
    # Rather, we will use them to test our accuracy of the training
    image_paths = [os.path.join(path, f) for f in os.listdir(path) if not f.endswith('.sad')]
    # images will contains face images
    images = []
    # labels will contains the label that is assigned to the image
    labels = []
    for image_path in image_paths:
        # Read the image and convert to grayscale
        image_pil = Image.open(image_path).convert('L')
        # Convert the image format into numpy array
```

```
        image = np.array(image_pil, 'uint8')
        # Get the label of the image
        nbr = int(os.path.split(image_path)[1].split(".")[0].replace("subject", ""))
        # Detect the face in the image
        faces = faceCascade.detectMultiScale(image)
        # If face is detected, append the face to images and the label to labels
        for (x, y, w, h) in faces:
            images.append(image[y: y + h, x: x + w])
            labels.append(nbr)
            cv2.imshow("Adding faces to traning set...", image[y: y + h, x: x + w])
            cv2.waitKey(50)
    # return the images list and labels list
    return images, labeL
```

- **Preparing the training set**

We pass the get_images_and_labels function with the path of the database directory. This path has to be the absolute path. This functions returns the features (images) and labels (labels) which will be used to train the face recognizer in the next step.

Code: -

```
------------------------------------------------
# Path to the Yale Dataset
path = 'yalefaces'
# The folder yalefaces is in the same folder as this python script
# Call the get_images_and_labels function and get the face images and the
# corresponding labels
images, labels = get_images_and_labels(path)
cv2.destroyAllWindows()
------------------------------------------------
```

- **Perform the training**

We perform the training using the FaceRecognizer.train function. It requires 2 arguments, the features which in this case are the images of faces and the corresponding labels assigned to these faces which in this case are the induvidual number that we extracted from the image names.

## 3. Testing the face recognizer

We will test the results of the recognizer by using the images with **.sad** extension which we had not used earlier. As done in the get_images_and_labels function, we append all the image names with the **.sad**extension in a image_paths list. Then for each image in the list, we read it in grayscale format and detect faces in it. Once, we have the ROI containing the faces, we pass the ROI to the FaceRecognizer.predict function which will assign it a label and it will also tell us how confident it is about the recognition. The label is an integer that is one of the individual numbers we had assigned to the faces earler. This label is stored in nbr_predicted. The more the value of confidence variable is, the less the recognizer has confidence in the recognition. A confidence value of 0.0 is a perfect recognition.

We check if the recognition was correct by comparing the predicted label "nbr_predicted" with the actual label "nbr_actual". The label "nbr_actual" is extracted using the os module and the string operations from the name of the image. We also display the confidence score for each recognition.

Doc5.docx

ORIGINALITY REPORT

| 14% | 10% | 1% | 6% |
| --- | --- | --- | --- |
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
| --- | --- | --- |
| 1 | www.dcs.gla.ac.uk<br>Internet Source | 3% |
| 2 | drrajivdesaimd.com<br>Internet Source | 1% |
| 3 | towardsdatascience.com<br>Internet Source | 1% |
| 4 | Submitted to Higher Education Commission Pakistan<br>Student Paper | 1% |
| 5 | cvc.yale.edu<br>Internet Source | 1% |
| 6 | Submitted to Minnesota State University, Mankato<br>Student Paper | 1% |
| 7 | iopencv.com<br>Internet Source | 1% |
| 8 | www.rroij.com<br>Internet Source | 1% |