

SP04049

**A Project Report
On
Fingerprint Recognition System**

As partial fulfillment of the Degree of Bachelor of Technology

Submitted by

GAURAV CHAKARVARTY	041268
SUJOY CHAND	041273
AMIT SINHA	041274



MAY- 2008
DEPARTMENT OF COMPUTER SCIENCE
JAYPEE UNIVERSITY OF INFORMATION
TECHNOLOGY-WAKNAGHAT

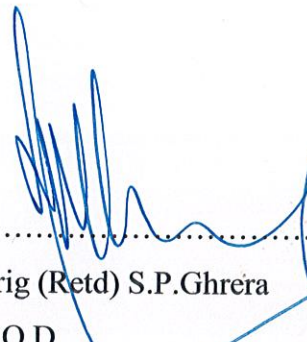
CERTIFICATE

This is to certify that the work entitled, "**Fingerprint Recognition System**" submitted by **Amit Sinha, Gaurav Chakarvarty and Sujoy Chand** in partial fulfillment for the award of degree of Bachelor of Technology in Computer Science of Jaypee University of Information Technology has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

PROJECT SUPERVISOR:


.....

Ms. Shipra Sharma
Lecturer
CSE & IT
JUIT


.....

Brig (Retd) S.P. Ghrera
H.O.D
CSE & IT
JUIT

ACKNOWLEDGEMENT

We would most of all like to thank our project supervisor Ms. Shipra Sharma for the invaluable advice and positive encouragement she have provided us throughout the course of this project and helps at all stages during the project and through the preparation of the dissertation. Thanks also go to all our fellow students for making the four years of B.Tech most memorable ones. Finally, we would like to express our gratitude to all friends and family members for their continuous love and support, and for always being there for us.

GAURAV CHAKARVARTY (041268)

SUJOY CHAND (041273)

AMIT SINHA (041274)

Gaurav Chakravarty

Sujoy Chand

Amit Sinha

LIST OF FIGURES:

Fig -1.1 Common Human Biometric Characteristics

Fig-1.2 Basic Block Diagram of Biometric systems

Fig -1.3 Market Share of Finger Print Recognition in Biometric

Fig-1.4 Annual Biometric Industrial Revenue

Fig -2.1 Level 0 DFD

Fig-2.2 Level 1 DFD

Fig-2.3 Gantt Chart.

Fig-3.1 Complete Process of Finger print Matching.

Fig -3.2 Formation of DT net.

Fig -3.3 Local Structures and Features DT Net Edges and Triangles.

Fig-3.4 Bifurcation and Ridge Ending.

Fig-3.5 Types of false minutiae

Fig -3.6 Filtered and Unfiltered Minutiae sets

Fig-4.1 Fingerprint Image Showing Salient Features for Authentication.

Fig -4.2 Two types of minutiae points.

Fig-4.3 Angle between minutiae points.

Fig -8.1 Modular Architecture.

Table of contents

Problem Specification.

Chapter 1. Introduction

1.1 Overview

1.2 Biometrics

1.2.1 Comparison of different Biometrics Techniques with Fingerprint

1.3 Why Fingerprint Recognition

1.4 A brief look at how fingerprint system works

1.5 Application of fingerprint recognition

Chapter 2. Software development life cycle

2.1 Process model

2.2 DFD

2.3 GANTT CHART

Chapter 3 Analysis and research

3.1 DT algorithm of minutia sets

3.2 Minutia extraction using CN number approach

3.3 Neural networks

Chapter 4 Algorithm implementation

4.1 Feature extraction method

4.2 Concept of CN number

Chapter 5 GUI (Graphical user interface)

Chapter 6 Database

Chapter 7 Testing

Chapter 8 Modular decomposition

8.1 Analysis and Research module

8.2 Algorithm development and implementation

8.2.1 Feature extraction module

8.2.2 Authentication module

8.3 Testing module

8.4 Backend module

8.5 GUI module

Chapter 9 Hardware and software requirements

Chapter 10 Installation guide

10.1 Java development kit

10.2 Java runtime application

10.3 Microsoft access 2003

Chapter 11 Result

Chapter 12 Conclusion and future work

Chapter 13 Source code

Problem Specification

To develop a Finger print authentication system which provides high level of security in today's world. Manual recognition systems are considered time consuming and less effective and more expensive to maintain. Our solution to the problem is based on Biometrics, which is a modern technological field that focuses on identifying an individual through his or her unique physical trait. Biometric spans various fields such as artificial intelligence and biology as well as various hardware related fields. Fingerprinting recognition project definitely will be a new decision in providing state of the art software security. Biometric technology will allow more protection over sensitive information. Finger print recognition project mainly deals with providing security for the organization by preventing unauthorized users to access.

Chapter 1. Introduction

1.1 Overview

1.2 Biometrics

1.2.1 Comparison of different Biometrics Techniques with

Fingerprint

1.3 Why Fingerprint Recognition

1.4 A brief look at how fingerprint system works

1.5 Application of fingerprint recognition

1. Introduction

1.1 Overview

Feature as word is pronounced has no exact definition often depends on the problem or the type of application. Feature is defined as an "interesting" part of an image when we define the term in terms of images and there manipulation [1]. The manipulation of the image is image processing.” **Image processing**” can be defined as any form of signal processing for which the input is an image, such as photographs or frames of video; the output of image processing can be either an image or a set of characteristics or parameters related to the image. Most image-processing techniques involve treating the image as a two-dimensional signal and applying standard signal-processing techniques to it. Image processing usually refers to digital image processing, but optical and analog image processing are also possible now we discuss this as we here have tried to manipulate fingerprint impression’s to produce the “**Finger Print recognition system** ”.The distribution of the information through out the organization has been going on to allow the people at each level get to know what is going around in the organization depending on the level of access granted to them at there levels .The control of the access to the information is provided by recognition system’s which identifies what access is to be given to whom .In era around seven-eight years earlier the entire access system was more of the textual data type i.e. password system type was there i.e. people were given access to information based on the defined rules for there set of information records set up by administrator. The major draw backs of the system were password were circulated and unauthorized access was granted Passwords can be cracked quiet easily by using various programs like Brute Force Search etc. To overcome such problems researchers have been developing recognition system that provides access to the information by recognizing an individual by its human organs rather than a set of alpha numeric characters. The use of human body for accessing information in scientific terms is called **biometric access**

1.2 Bio Metrics

Biometrics is defined as the study of methods for uniquely recognizing humans based upon one or more intrinsic physical or behavioral traits.

Biometrics can also be defined as an automated system that can identify an individual by measuring their physical and behavioral uniqueness or patterns, and comparing it to those on record.

Biometrics is used to identify the identity of an input sample when compared to a template, used in cases to identify specific people by certain characteristics.

Possession-based: using one specific "token" such as a security tag or a card.

Knowledge-based: the use of a code or password.

Standard validation systems often use multiple inputs of samples for sufficient validation, such as particular characteristics of the sample. This intends to enhance security as multiple different samples are required such as security tags and codes and sample dimensions is possible to understand if a human

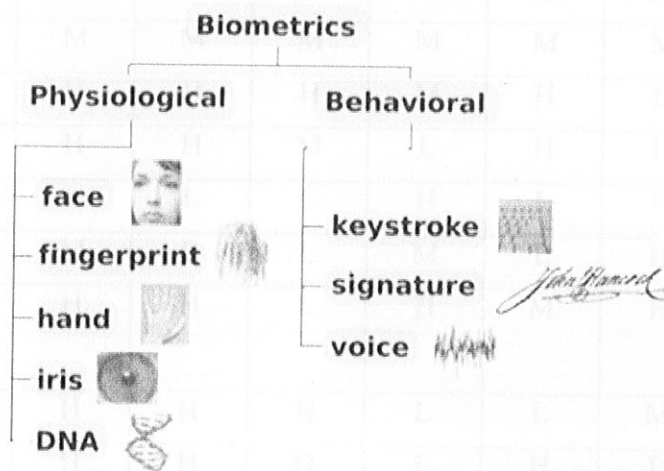


Fig -1.1 Common Human Biometric Characteristics [courtesy: wikipedia]
Characteristics can be used for biometrics in terms of the following parameters:

Universality (U): each person should have the characteristic

Uniqueness (Uq): is how well the biometric separates individually from another.

Permanence (P): measures how well a biometric resists aging.

Collect ability (C): eases of acquisition for measurement.

Performance (Pr): accuracy, speed, and robustness of technology used.

Acceptability (A): degree of approval of a technology.

Circumvention (Cr): eases of use of a substitute.

The following table shows the comparison of the various biometric technologies for high (H), medium (M), low (L) for the above mentioned following parameters

BIOMETRICS	U	Uq	P	C	Pr	A	Cr
FACE	H	L	M	H	L	H	L
FINGER PRINT	M	H	H	M	H	M	H
HAND GEOMETRY	M	M	M	H	M	M	M
KEY STROKES	L	L	L	M	L	M	M
HAND VEINS	M	M	M	M	M	M	H
IRIS	H	H	H	M	H	L	H
RETINAL SCAN	H	H	M	L	H	L	H
SIGNATURE	L	L	L	H	L	H	L
VOICE	M	L	L	M	L	H	L
FACIAL THERMOGRAPHY	H	H	L	H	M	H	H
ODOR	H	H	H	L	L	M	L
DNA	H	H	H	L	H	L	L
GAIT	M	L	L	H	L	H	M
EAR CANAL	M	M	H	M	M	H	M

Biometric systems in general follows basic block and produce the result

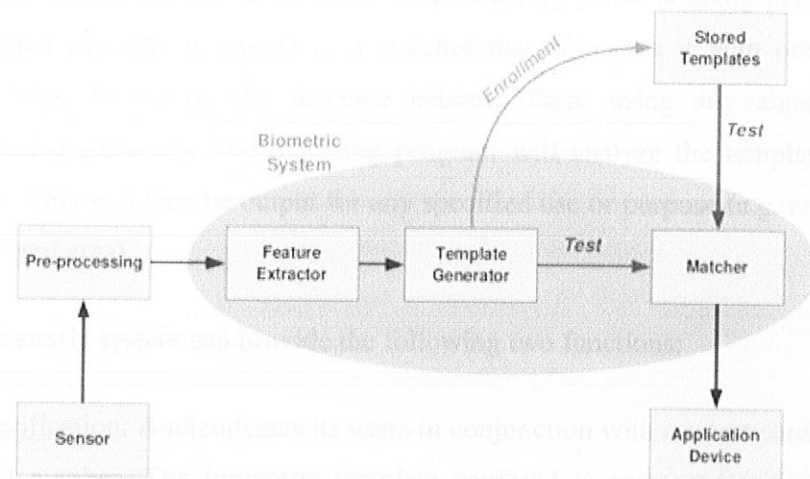


Fig 1.2 Basic Block Diagram of Bio-Metric System [courtesy: wikipedia]

The diagram shows a simple block diagram of a biometric system. When such a system is networked together with telecommunications technology, biometric systems become tele-biometric systems. The main operations a system can perform are *enrollment* and *test*. During the enrollment, biometric information from an individual is stored. During the test, biometric information is detected and compared with the stored information. Note that it is crucial that storage and retrieval of such systems themselves be secure if the biometric system is to be robust. The first block (sensor) is the interface between the real world and our system; it has to acquire all the necessary data. Most of the times it is an image acquisition system, but it can change according to the characteristics desired. The second block performs all the necessary pre-processing: it has to remove artifacts from the sensor, to enhance the input (e.g. removing background noise), to use some kind of normalization, etc. In the third block features needed are extracted. This step is an important step as the correct features need to be extracted and the optimal way. A vector of numbers or an image with particular properties is used to create a *template*. A template is a synthesis of all the characteristics extracted from the source, in the optimal size to allow for adequate identifiability.

If enrollment is being performed the template is simply stored somewhere (on a card or within a database or both). If a matching phase is being performed, the obtained template is passed to a matcher that compares it with other existing templates, estimating the distance between them using any algorithm (e.g. Hamming distance). The matching program will analyze the template with the input. This will then be output for any specified use or purpose (e.g. entrance in a restricted area).

A biometric system can provide the following two functions:

1. Verification: Authenticates its users in conjunction with a smart card, username or ID number. The biometric template captured is compared with that stored against the registered user either on a smart card or database for verification.
2. Identification: Authenticates its users from the biometric characteristic alone without the use of smart cards, usernames or ID numbers. The biometric template is compared to all records within the database and a closest match score is returned. The closest match within the allowed threshold is deemed the individual and authenticated.

Now performance of the biometric systems can be measured on the following parameters:

1. False accept rate (FAR) or false match rate (FMR): the probability that the system incorrectly declares a successful match between the input pattern and a non-matching pattern in the database. It measures the percent of invalid matches. These systems are critical since they are commonly used to forbid certain actions by disallowed people.
2. False reject rate (FRR) or false non-match rate (FNMR): the probability that the system incorrectly declares failure of match between the input pattern and the matching template in the database. It measures the percent of valid inputs being rejected.

3. Receiver (or relative) operating characteristic (ROC): In general, the matching algorithm performs a decision using some parameters (e.g. a threshold). In biometric systems the FAR and FRR can typically be traded off against each other by changing those parameters. The ROC plot is obtained by graphing the values of FAR and FRR, changing the variables implicitly. A common variation is the *Detection error trade-off (DET)*, which is obtained using normal deviate scales on both axes. This more linear graph illuminates the differences for higher performances (rarer errors).

4. Equal error rate (EER): The rates at which both accept and reject errors are equal. ROC or DET plotting is used because how FAR and FRR can be changed, is shown clearly. When quick comparison of two systems is required, the EER is commonly used. Obtained from the ROC plot by taking the point where FAR and FRR have the same value. The lower the EER, the more accurate the system is considered to be.

5. Failure to enroll rate (FTE or FER): the percentage of data input is considered invalid and fails to input into the system. Failure to enroll happens when the data obtained by the sensor are considered invalid or of poor quality.

6. Failure to capture rate (FTC): Within automatic systems, the probability that the system fails to detect a biometric characteristic when presented correctly.

7. Template capacity: the maximum number of sets of data which can be input in to the system

Now to measure the performance of the systems is based on the parameters mentioned above:

Biometrics	EER	FAR	FRR	Subjects	Comment	Reference
Face	n.a	1%	10%	37437	Varied lightning indoor/outdoor	FRVT
fingerprint	n.a	1%	0.1%	25000	US GOVT operational data	FpVTE
fingerprint	2%	2%	2%	100	Rotation and exaggerated skin distortion	FVC
Hand-geometry	1%	2%	0.1%	129	With rings and improper placement	
Iris	<1%	0.94 %	0.99%	1224	Indoor environment	ITIRT
Iris	0.01 %	0.00 01%	0.2%	132	Best conditions	NIST
keystrokes	1.8%	7%	0.1%	15	During 6 month period	
Voices	6%	2%	10%	310	Text,independent,multi lin-gual	NIST

The biometric verification systems that are currently being used include:

Fingerprint, iris, voice, retina, hand, vein and signature.

Out of all of these, the Iris can return the most accurate results. But since it requires the system or device to come very close to the individual's eyes, it is not a desirable method of verification. In addition, such devices or systems tend to be very expensive and bulky, putting limits on their practical application.

1.2.1 Comparison of different Biometrics Techniques with Fingerprint

A typical voice recognition system is usually more affordable, but it is not always reliable because the human voice is subject to changes due to illness, hoarseness, or other common throat problems. This system is also subject to extraneous noises in the surrounding area, static and varying conditions. The hand recognition system tends to take too much space and has a high return rate of False Acceptance (FAR). Therefore, this system is rarely used in high security zones. There are also cost problems in its administration, as well as concerns over access to confidential information.

Vein recognition can guarantee top security because it is almost impossible to duplicate an individual's vein. But it also has limits in practical application because its hardware structure is extremely complex, and the entire system comes at a very high cost.

The results with a signature recognition system, like the voice recognition system, are easily affected by various factors, and therefore it is not reliable.

1.3 Why Finger print Recognition?

Fingerprint recognition has long been the favored among many biometric identification technologies due to its uniqueness and permanence. Nowadays, fingerprint recognition is considered to be the best choice for most applications from network security systems to compact devices, due to its accuracy, speed, reliability, non-intrusive interfaces, and cost-effectiveness, resulting in acquiring 50% of the shares in biometric markets

Finger print recognition system contains additional features of biometrics which makes them favorable are:

1.) Except in some cases where the cut leaves a scar, a fingerprint recovers from a cut without problems, returning to its original aspect. Even if there is a cut and a scar, the fingerprint will be easily identified, because 13 minutiae are needed for the process, while a regular fingerprint has about 50 of them.

2.) Even identical twins have different fingerprints. Fingerprints have been used in identification for thousands of years and even today, with huge databases with millions of fingerprints, it hasn't been found one that is identical to another. Besides, each finger and toe has a completely different fingerprint from each other.

3.) For databases up to 1,000 fingerprints there are no big requirements. Any PC can be used, like a Celeron 1 GHz with 128MB RAM. For databases up to 10,000 fingerprints it is recommended to use a dedicated PC to keep the 1 per second speed in the identification process. A proper configuration would be a Pentium 4 with 512MB RAM. A 800Mhz Bus is very important and makes a great difference in the performance. For databases higher than 10,000 fingerprints it is important to use a multiple parallel server's architecture, the Speed Cluster. This way you will have many servers searching for the fingerprint, which makes the process faster. In case of unique use of Verification method, a heavy server structure is not required. Only one calculation of a fingerprint is computed and then, it will be compared to a stored fingerprint. Unless many requests are made at the very same time, one server is enough; even a shared one.

When it comes to commercialization fingerprint technology beats all biometric players to acquire market as result we have::

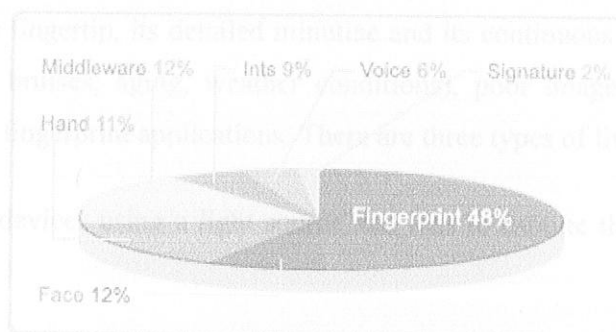


Fig 1.3 Market share of fingerprint recognition in biometrics [courtesy: wikipedia]

1.4 A brief look at how fingerprint's works on the system:

Fingerprint recognition consists of comparing a print of the characteristics of a fingertip or a template of that print with a stored template or print. A fingerprint consists of the features and details of a fingertip. There are three major fingerprint features: the arch, loop and whorl. Each finger has at least one major feature.

Loops are lines that enter and exit on the same side of the print. Arches are lines that start on one side of the print, rise into hills and then exit on the other side of the print. Whorls are circles that do not exit on either side of the print. The smaller or minor features (or minutiae) consist of the position of ridge ends (ridges are the lines that flow in various patterns across fingerprints) and of ridge bifurcations (the point where ridges split in two). There are between 50 and 200 such minor features on every finger. Fingerprint matching done on the basis of the three major features is called pattern matching while the more microscopic approach is called minutiae matching. Other features may be used for matching, but patterns and minutiae are the main ones.

1.) Acquire a sample: Enrolment and acquisition can be done by sensors reading the tip of the finger directly and in real-time. A fingerprint scan contains a lot of information but scanners normally focus only on getting an image of the information that is essential for matching. The quality of the sensed fingerprint image is of key importance for the performance of the system. Given the small area of the fingertip, its detailed minutiae and its continuous use in everyday life (e.g. cuts, bruises, aging, weather conditions), poor image quality is a major concern in fingerprint applications. There are three types of live scanners:

1. Optical devices using a light source and lens to capture the fingerprint with a camera;
2. Solid-state sensors or silicon sensors appearing on the market in the mid-1990s to address the shortcomings of the early optical sensors
3. And others, such as acoustic sensors that use acoustic signals to detect fingerprint detail.

2.) Extracting features: Getting a high quality image of the fingerprint is very important for accurate fingerprint recognition, but also feature extraction plays a crucial role. It consists of converting the fingerprint image into a usable and

comparable format that does not require lots of storage space. The format or template is a compressed version of the fingerprint characteristics. Several approaches to automatic minutiae extraction exist, but most of these methods transform fingerprint images into binary images. This means that only the coordinates of the minutiae (30 or 40) are stored, reducing it to a few hundreds of bytes

Feature extraction is also needed because even a very precise fingerprint image will have distortions and false minutiae that need to be filtered out. For example, an algorithm may search the image and eliminate one of two adjacent minutiae, as minutiae are very rarely adjacent. Anomalies can also be caused by scars, sweat, or dirt. The algorithms used for feature extraction filter the image to eliminate the distortions and would-be minutiae

3.) Comparing the templates: The identification or verification process follows the same steps as the enrolment process with the addition of matching. It compares the template of the live image with a database of enrolled templates (identification), or with a single enrolled template (authentication).

4.) Declaring a Match: The comparison between the sensed fingerprint image or template against records in a database or a chip usually yields a matching score quantifying the similarity between the two representations. If the score is higher than a certain threshold, a match is declared, i.e. belonging to the same finger(s). The decision of a match or non-match can be automated but it depends also on whether matching is done for identification or verification¹⁸ purposes. With identification applications, automated decision-making is possible when conditions are ideal. In the case of the Federal Bureau of Investigation (FBI) for instance, this means that fingerprint cards can be matched automatically when both enrolment and acquisition were done by law enforcement staff. But with latent prints (e.g. collected at a crime scene), and prints with a lower quality image, the automated process is less reliable. Automated systems imitate the way human fingerprint experts work but the problem is that these systems can not have

observed the many underlying information-rich features an expert is able to detect visually. Automatic systems are however, reliable, rapid, consistent and cost effective when matching conditions are good, but their level of sophistication cannot rival that of a well-trained fingerprint expert. Therefore, for instance a fingerprint expert can overrule an automated match. Verification applications, especially mainstream commercial fingerprint verification may be, to a certain extent, less accurate because the issues at stake are different (e.g. identifying criminals), but also because verification consists of 1-1 matching. Verification may use less information from a fingerprint compared to forensic scientists identifying a fingerprint. The former seems to be more like a possible, "close-enough correlation" of similarities. Also, because of background interference (dirt, scratches, light, etc.), and no human supervision, the quality of fingerprint images is lower. The result is a "best" matching score which would not be feasible for law enforcement.

1.5 Application's of Finger Print Recognition:

Fingerprint identification of criminals for law enforcement continues to be one of the major applications domains for this technology. Another large scale application in Europe is EURODAC for asylum requests. In New York, fingerprints are used to prevent fraudulent enrolment for benefits. Using fingerprint recognition to secure physical access is another popular application. Moreover, embedding of fingerprint readers in electronic devices opens up a whole range of digital applications that are based on online authentication. Finally, decisions have been taken for the future integration of fingerprints (with other biometrics) on travel documents and passports.

Largest biometric database in the world. It is a US national fingerprint and criminal history system maintained by the FBI. It contains the fingerprints and corresponding criminal history information for more than 47 million subjects in the Criminal Master File. The fingerprints and corresponding criminal history

information are submitted voluntarily by state, local, and federal law enforcement agencies. The IAFIS provides automated fingerprint search capabilities, electronic image storage, and electronic exchange of fingerprints and responses, 24 hours a day, 365 days a year [8]. In Europe, there is no such database. Criminal fingerprint databases are under control of national criminal authorities. The UK for instance, has a national automated fingerprint identification system (NAFI) containing more than four million records.

There is however, since January 2003, also a large central fingerprint database in the European Union, but for another purpose. It aims at preventing duplication of asylum requests in the EU Member states. EURODAC is an EU wide database (AFIS) set up to check the fingerprints of asylum seekers against the records of other EU countries. After one year of operation, an evaluation report on EURODAC highlighted satisfactory results in terms of efficiency, quality of service and cost effectiveness. The EURODAC central unit has been operating continuously. Within one year, it processed almost 250,000 fingerprints of asylum seekers. It detected 17,287 cases of multiple application (a same person having already made an asylum application in another country), which represents 7% of the total number of cases processed.

In addition to asylum requests, also illegal immigrants are identified. Almost 17,000 fingerprints of people in an illegal situation were detected and about 8,000 fingerprints related to attempts to cross borders illegally. The evaluation report also states that there were no data protection problems raised by the Member States' national data protection authorities regarding EURODAC operations

he state of New York has over 900,000 people enrolled in a system which tracks entitlement to social services and protects against fraud known as "double dipping", i.e. enrolling for a benefit under multiple names (OECD, 2004: 23). Fingerprint scanning is also being used to arrange secure access to schools and schools premises such as cafeterias and libraries. Finally, with the embedding of fingerprint scanners in electronic devices, online authentication (replacement of



passwords, PINs, etc) becomes possible for a whole range of applications including electronic payments.

Finally, at EU level, the Council of European Ministers adopted the Regulation on mandatory facial images and fingerprints in EU passports at its meeting in Brussels on 13 December 2004. This Regulation applies to passports and travel documents issued by Member States (excluding Ireland, the UK and Denmark). After the Regulation is published in the Official Journal passports issued will have to contain a facial image within 18 months, and fingerprints within three years. Also a Committee will be set up by the European Commission with representatives from 22 Member States to decide on the details such as how many fingerprints are to be taken, the equipment needed and the costs.

Thus how the fingerprint recognition is forming major part for the development process that is going around in the biometric division for development of technology whose starting point is human body itself.

Fig 1.3 Annual Biometric Industry Revenue (courtesy wikipedia)

The expected growth of this technology as predicted:

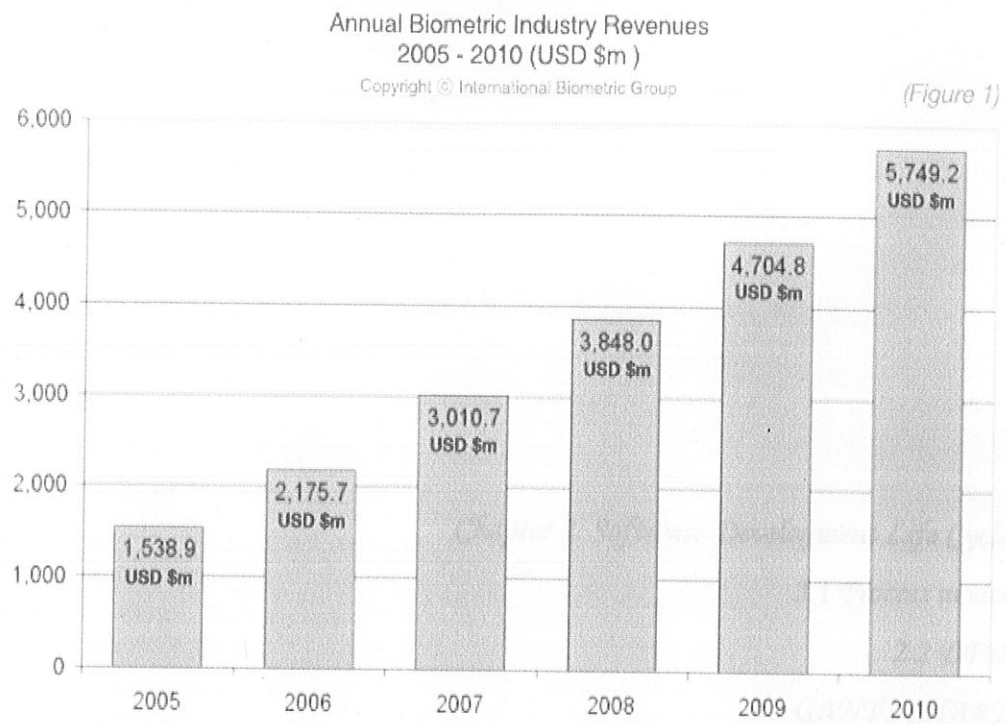


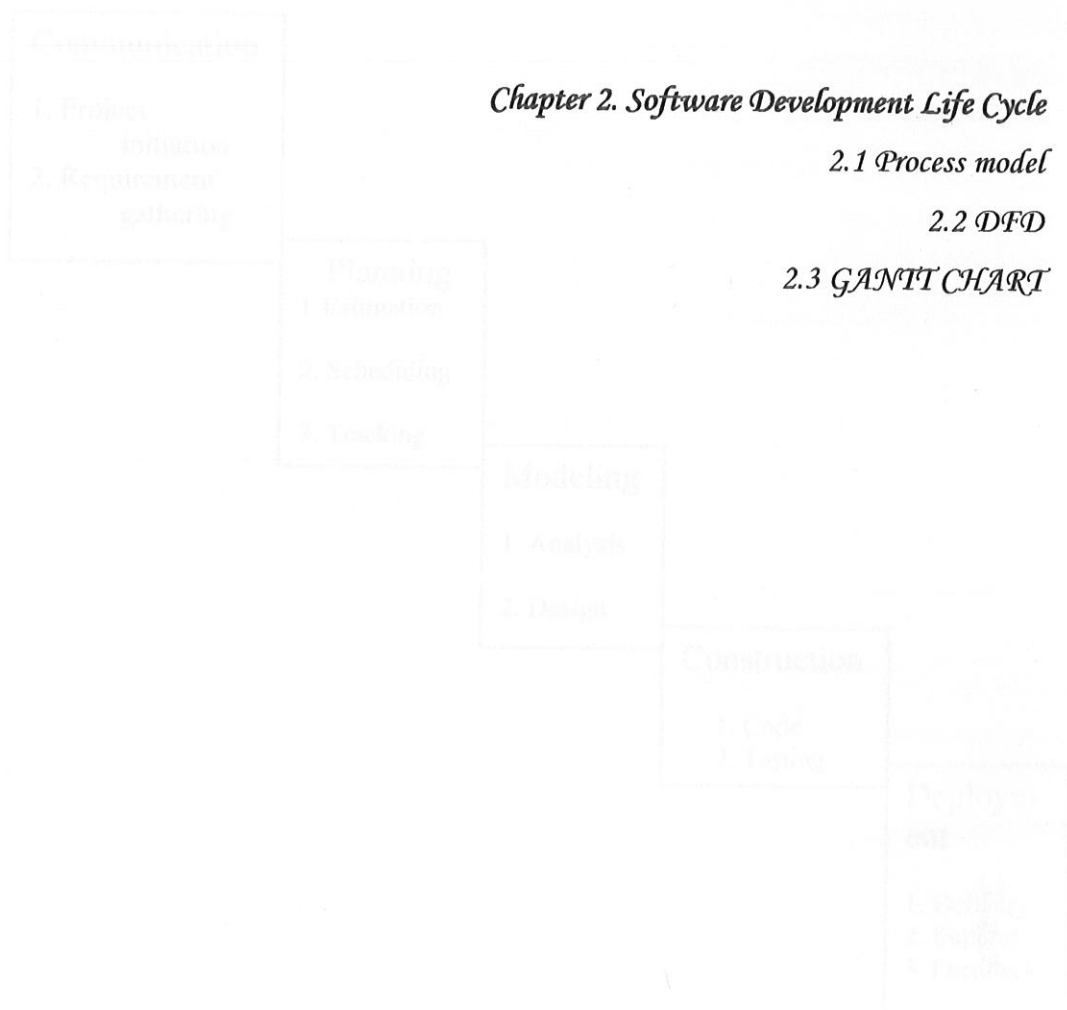
Fig-1.4 Annual Biometric Industry Revenue [courtesy: wikipedia]

Software Development Life Cycle

The development of Programmed Recognition system is a complex task and requires a systematic approach to development.

System Model

The development of our system will follow a systematic approach of software development model as a sequence of phases. Sequential approach in software development that begins at the system level and progresses through analysis, design, coding, testing, and maintenance phase.



Chapter 2. Software Development Life Cycle

2.1 Process model

2.2 DFD

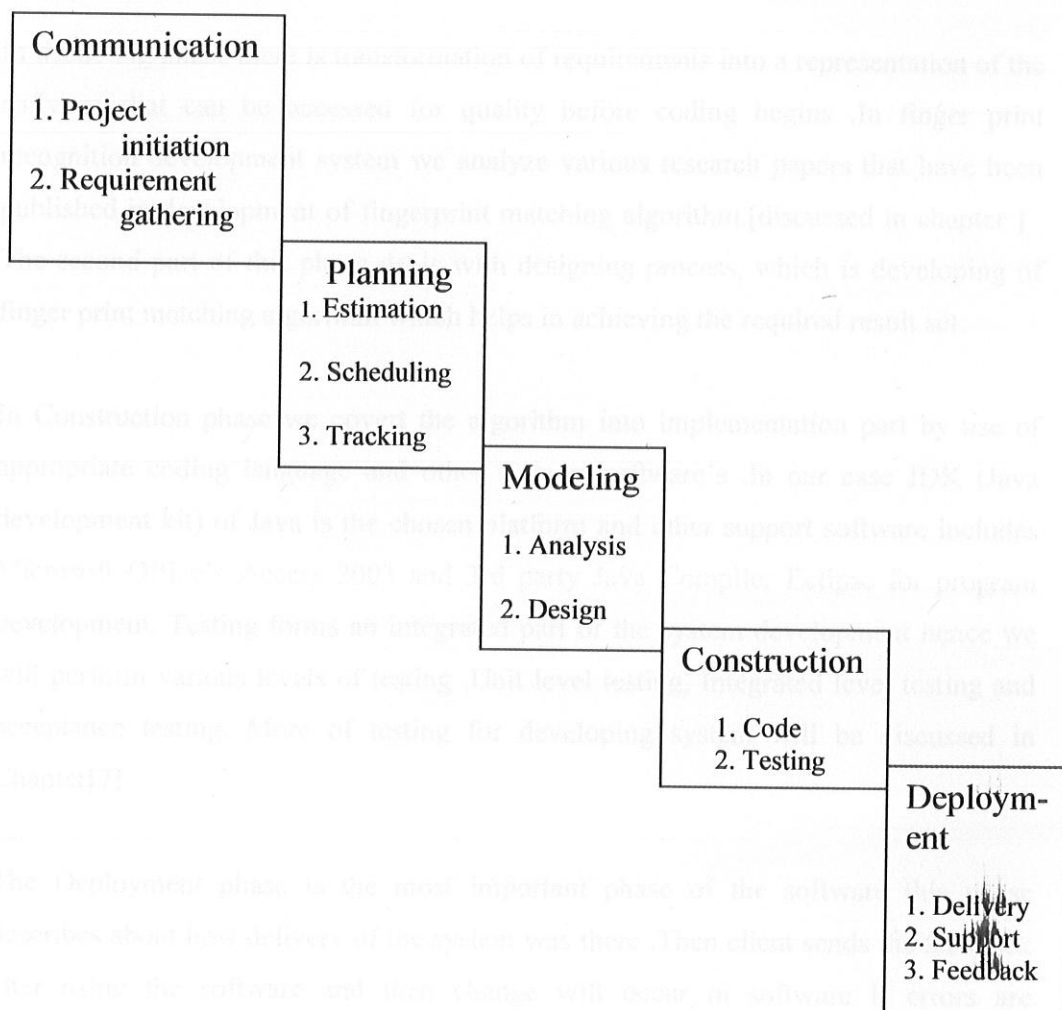
2.3 GANTT CHART

Software Development Life Cycle:

For the development of Fingerprint Recognition system we have applied software engineering at various levels of development.

2.1 Process Model

For the development of our system we have used classic life cycle or waterfall model as it suggests a systematic, sequential approach to software development that begins at the system level and progress through analysis, design, coding, testing, and support phase.



In the Communication phase, of the SDLC, we start with the analysis of the problem statement and then the detailed understanding of the objective and the goals of the problem was carried out. For understanding of the problem, the basic requirement was the detailed study of "What is finger print matching?" Why fingerprinting?"Etc. The requirements about what system view will be and how software will interact with other elements such as hardware, people and databases.

In Planning phase, we plan out the system development process by setting up the approximated time schedule and modules for system development, and then measuring the system development progress with planned schedule.

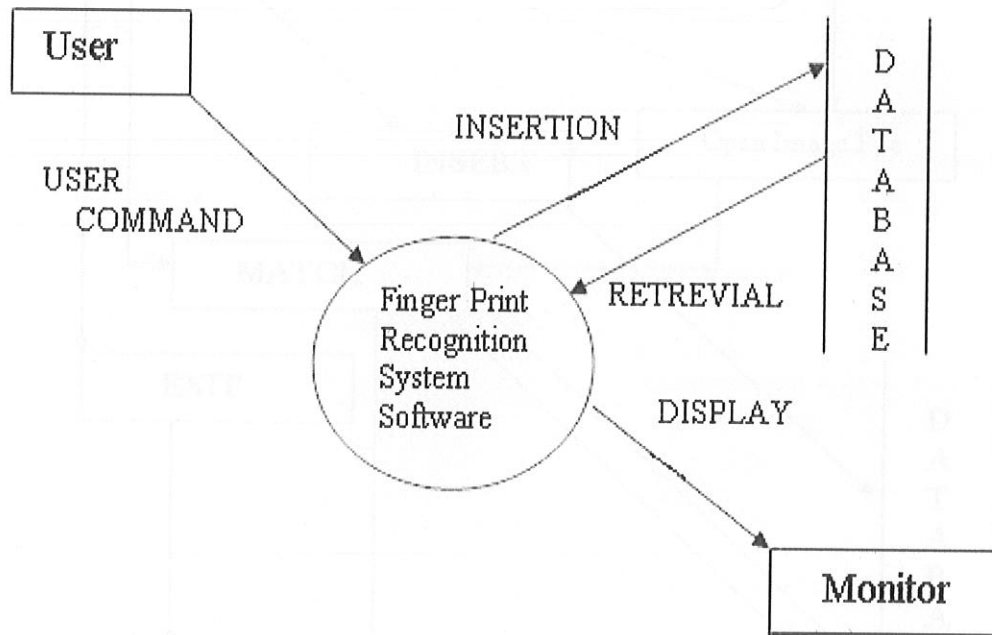
In Modeling phase there is transformation of requirements into a representation of the software that can be accessed for quality before coding begins .In finger print recognition development system we analyze various research papers that have been published in development of fingerprint matching algorithm.[discussed in chapter] . The second part of this phase deals with designing process, which is developing of finger print matching algorithm which helps in achieving the required result set.

In Construction phase we covert the algorithm into implementation part by use of appropriate coding language and other support software's .In our case JDK (Java development kit) of Java is the chosen platform and other support software includes Microsoft Office's Access 2003 and 3rd party Java Compiler Eclipse for program development. Testing forms an integrated part of the system development hence we will perform various levels of testing .Unit level testing, Integrated level testing and acceptance testing. More of testing for developing system will be discussed in Chapter[7].

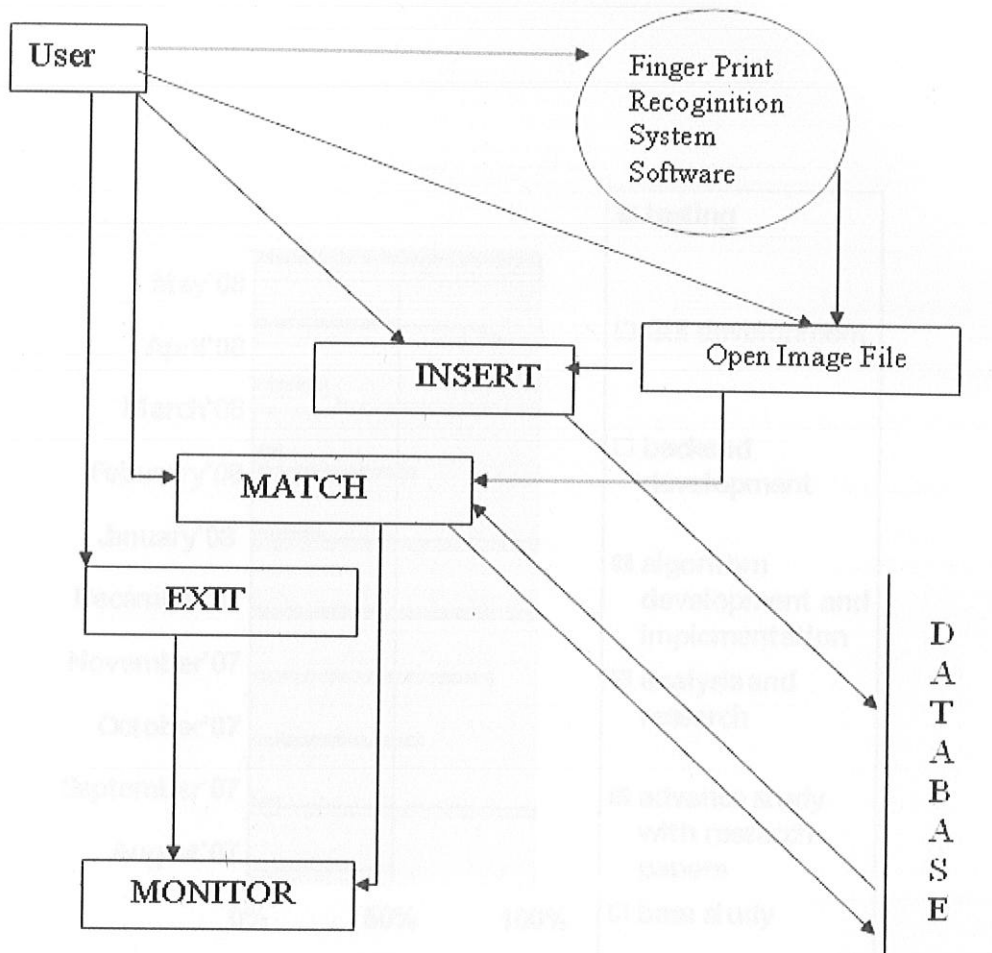
The Deployment phase is the most important phase of the software this phase describes about how delivery of the system was there .Then client sends the feedback after using the software and then change will occur in software if errors are encountered. Maintenance for software after delivery to client is most important.

2.2 DFD (Data Flow Diagram):

LEVEL 0:

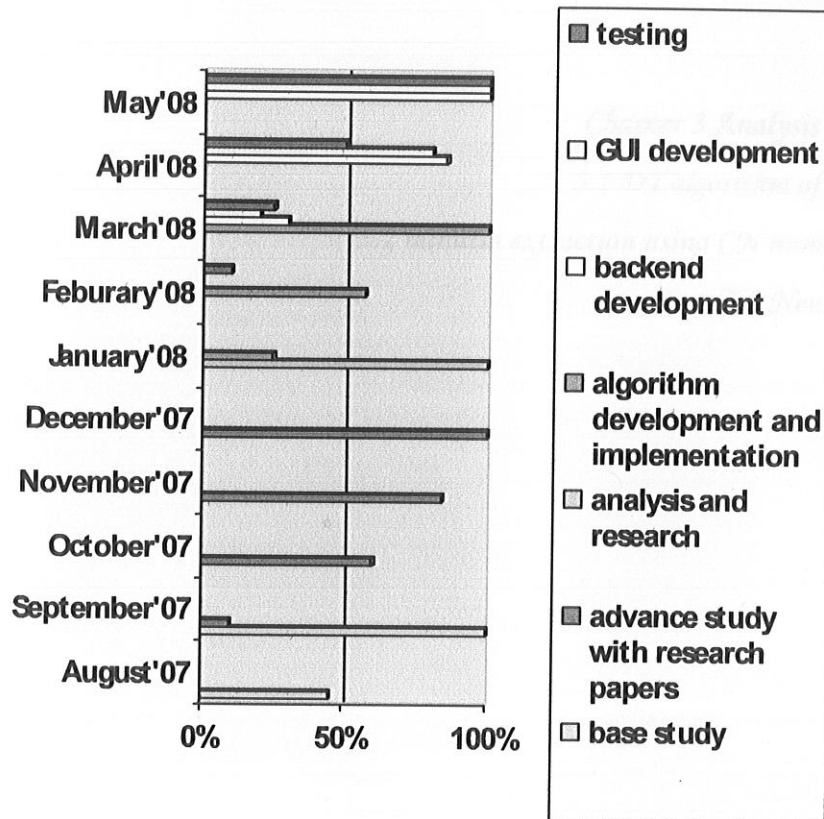


LEVEL 1:



2.3 GANTT CHART

Gantt chart is a popular bar chart which is used to describe the entire project schedule.



Chapter 3 Analysis and research

3.1 DT algorithm of minutia sets

3.2 minutia extraction using CN number approach

3.3 Neural networks

ANALYSIS AND RESEARCH

In this chapter we carried out extensive literature survey studied the existing techniques of feature extraction and matching in the field of fingerprinting. We have gone through various research papers and books on fingerprint matching and data available on internet and finally made a layout of our algorithm. Research helped us in developing a new effective algorithm which is also very efficient in terms of accuracy and time and space complexities.

The whole processes can be described as:

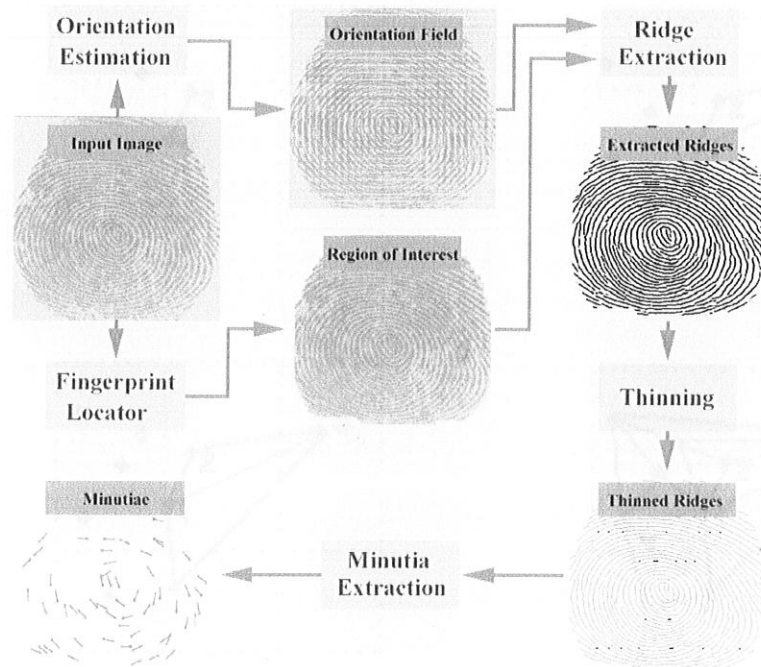


Fig 3.1 complete process.[Courtesy:imageg.google.com]

A Fingerprint Matching Algorithm Based On Delaunay Triangulation Net In this paper we studied following features:

On the basis of Delaunay triangulation (DT) in computational geometry, we proposed a fingerprint matching algorithm based on DT net. Triangulation is an effective tool in dealing with scattered data set. DT net is rotation and translation invariant. Then uses RMPs(ref. minutae pairs) to align input fingerprint.

3.1 DT Algorithm of Minutiae Set

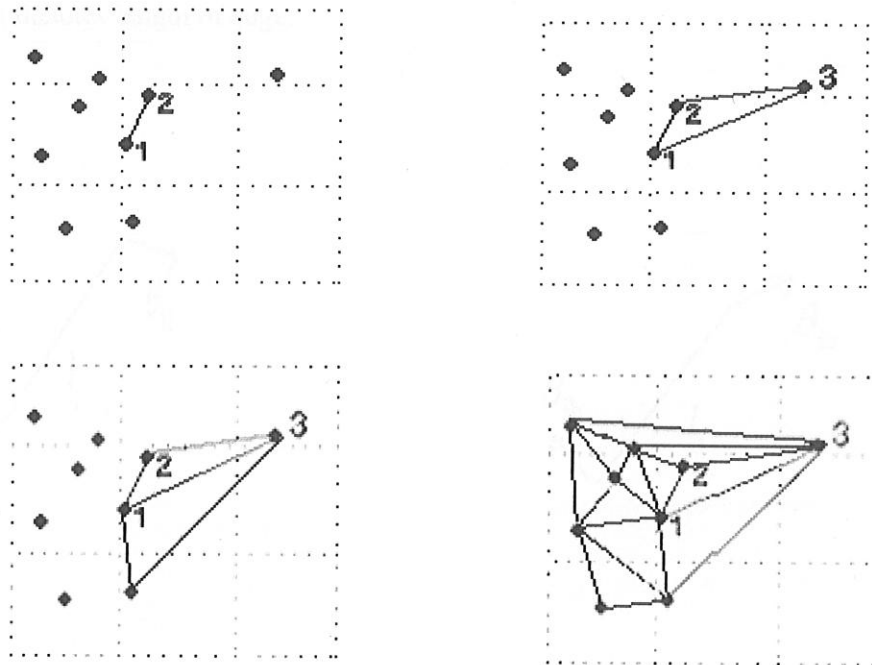


Fig 3.2 Formation of DT net.

Extracting RMPs

Matching the DT nets was used by us for finding several RMP. In the beginning, some pairs of similar edge pairs (SEPs) can be found through matching. Having the SEPs, the corresponding similar triangle pairs (STPs) can be found quickly. The three pairs of vertexes of each STP are three RMPs needed in alignment.

Given the template image and input image, two corresponding DT nets have produced by the algorithm. Minutia can be described as a vector $(x_i, y_i, \Theta_i)^T$ in template image or as $(x_i, y_i, \Theta_i)^T$ in input image. Meanwhile, edge can be described as a vector $(\theta_{ih}, \theta_{ib}, \gamma_i, l_i)^T$ in input DT net. In vectors X,Y denotes coordinates of minutia, θ denotes minutia angle and γ denotes orientation of edge. And l denotes length of edge.

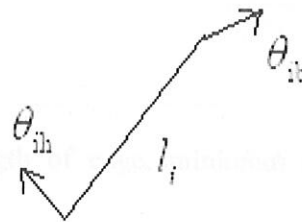




Fig 3.3 Local structure and features DT net edges and triangles.

Edge matching is used to find SEP. Choose two edges T and I from the template and input DT net, respectively. Only when the equations (1),(2) and (3) are satisfied, T and I are considered a SEPs. The equations (1),(2) and (3) are defined as follows:

$$|\lambda_t - \lambda_i| < T_1$$

$$|\theta_{th} - \theta_{ih}| - |\theta_{tb} - \theta_{ib}| < T_2$$

$$|\gamma_T - \gamma_i| < T_3$$

Where T1, T2 and T3 are thresholds of length of edge, minimum angle and orientation

3.2 MINUTAE EXTRACTION USING CROSSING NUMBER APPROACH

Minutiae extraction was carried out using the crossing number approach. Crossing number of pixel 'p' is defined as half the sum of the differences between pairs of adjacent pixels defining the 8-neighborhood of 'p'. Mathematically

$$CN = 0.5 \sum_{i=1}^8 |P_i - P_{i+1}| \quad \text{Where } p_0 \text{ to } p_7 \text{ are the pixels belonging to an ordered}$$

sequence of pixels defining the 8-neighborhood of p and Val (p) is the pixel value.

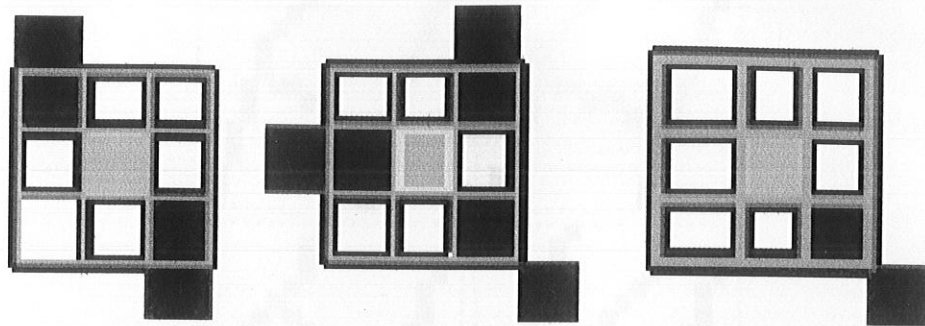


Fig. 3.4:: $cn(p)=2$, $cn(p)=3$ and $cn(p)=1$ representing a non minutiae region, a bifurcation and a ridge ending.

Crossing numbers 1 and 3 correspond to ridge endings and ridge bifurcations respectively. An intermediate ridge point has a crossing number of 2. The minutiae obtained from this algorithm must be filtered to preserve only the true minutiae. The different types of false minutiae introduced during minutiae extraction include spike, bridge, hole, break, Spur, Ladder, and Misclassified Border areas.

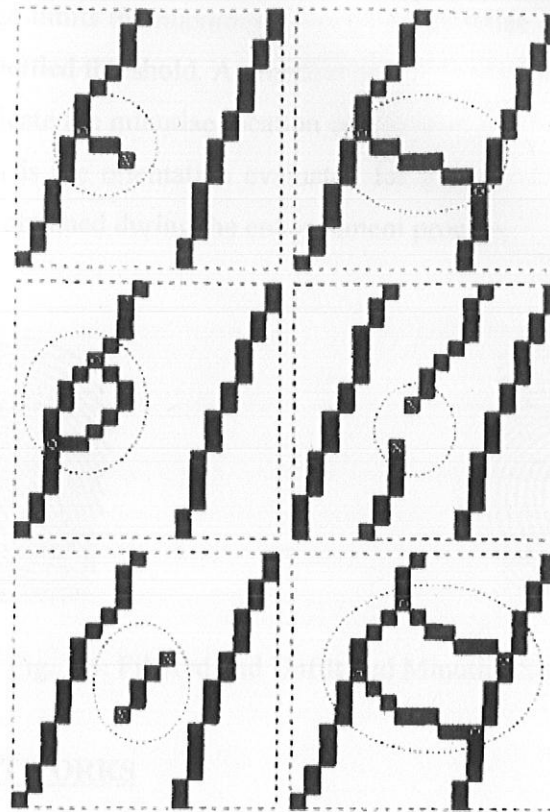


Fig. 3.5: Types of false minutiae

A B

C D

E F

A. Spike, B. Bridge, C. Hole, D.

Break, E. Spur F. Ladder

The number of minutiae in a given area is also limited therefore the minutiae density must also be kept in check. In order to filter out these false minutiae a 3 level-filtering process is applied.

Level 1: Removes the false ridge endings created as a result of the application of minutiae extraction algorithm at the ends of the thinned image.

Level 2: Removes the first five types of minutiae mentioned above using the rule based morphological minutiae filtering approach.

Level 3: This stage limits the maximum number of minutiae present in the thinned image to a pre-specified threshold. A minutiae m is described by the triplet $m=\{x, y, \theta\}$, where x, y indicate the minutiae location coordinates and θ denotes the minutiae orientation, which is the orientation evaluated for the minutiae location from the orientation image obtained during the enhancement process.



Fig. 3.6: Filtered and Unfiltered Minutiae Sets

3.3 NEURAL NETWORKS

We also studied about neural networks and their implementation so that if we could apply in our algorithm for dealing with speed complexities.

Back-propagation algorithm for multiple O/P classes

Step 0: Initialize the weights , biases & learning rate suitably.

Step 1: Check for stopping condition ; if it is false , perform Steps 2-6.

Step 2: Perform Steps 3-5 for each bipolar or binary training vector pair

Step 3: Set activation of each I/P unit $i = 1$ to n , that is calculate new weights and bias $\Delta w_i = \eta(t) x_i$, $b_0 = b_0 + \eta(t)$

Step 4: Calculate O/P response of each O/P unit $j = 1$ to m :

First , the net I/P is calculated Then , activations are applied over the net I/P to calculate the O/P response

Step 5: Make adjustments in weights & bias for $j=1$ to m & $i = 1$ to n .

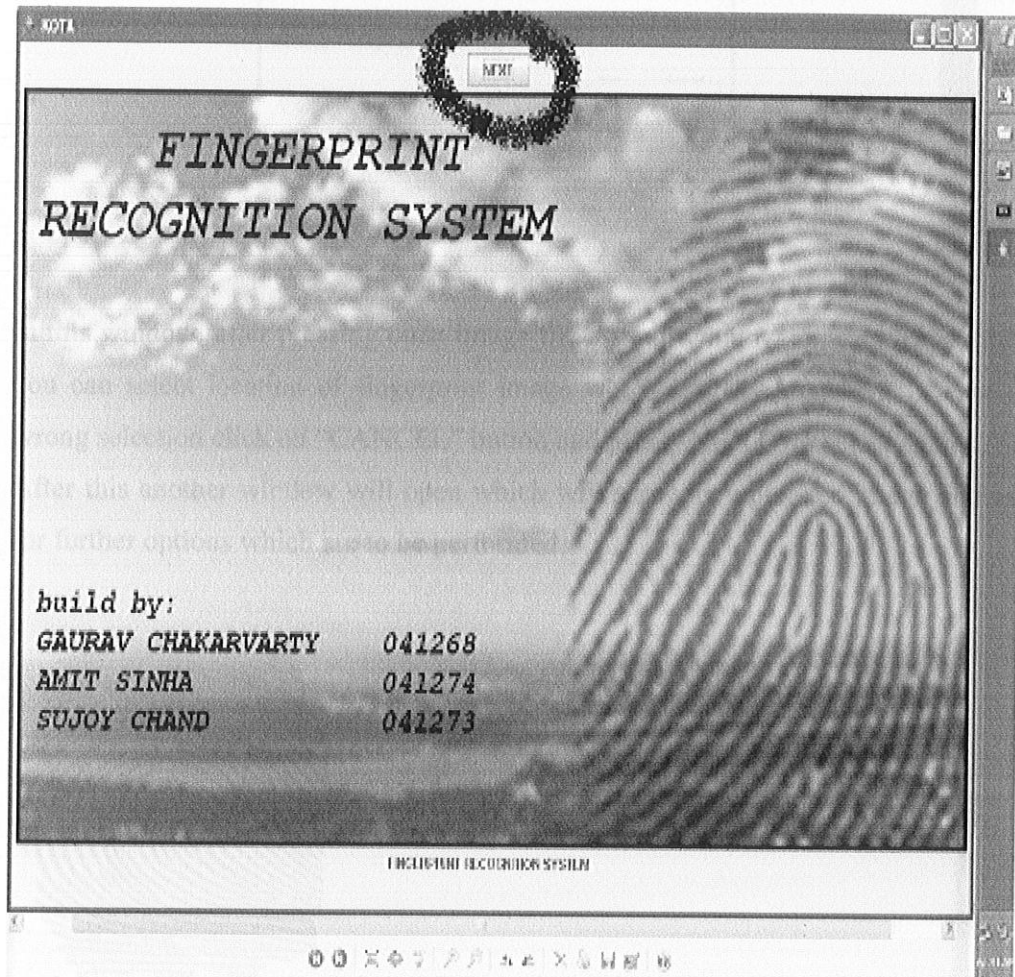
Step 6: Test for the stopping condition , i.e., if there is no change in weights then stop the training process , else start again from step 2.

GUI (GRAPHICAL USER INTERFACE)

A graphical user interface (GUI) presents a pictorial interface to a program.

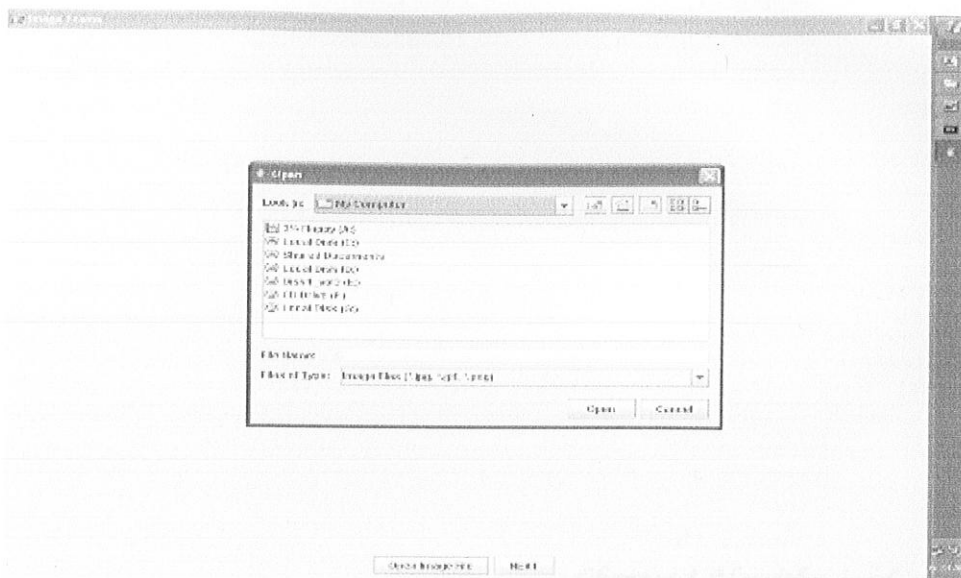
Our GUI is built in java swing and is very user friendly.

Following are screen shorts of our GUI along with detailed explanation:

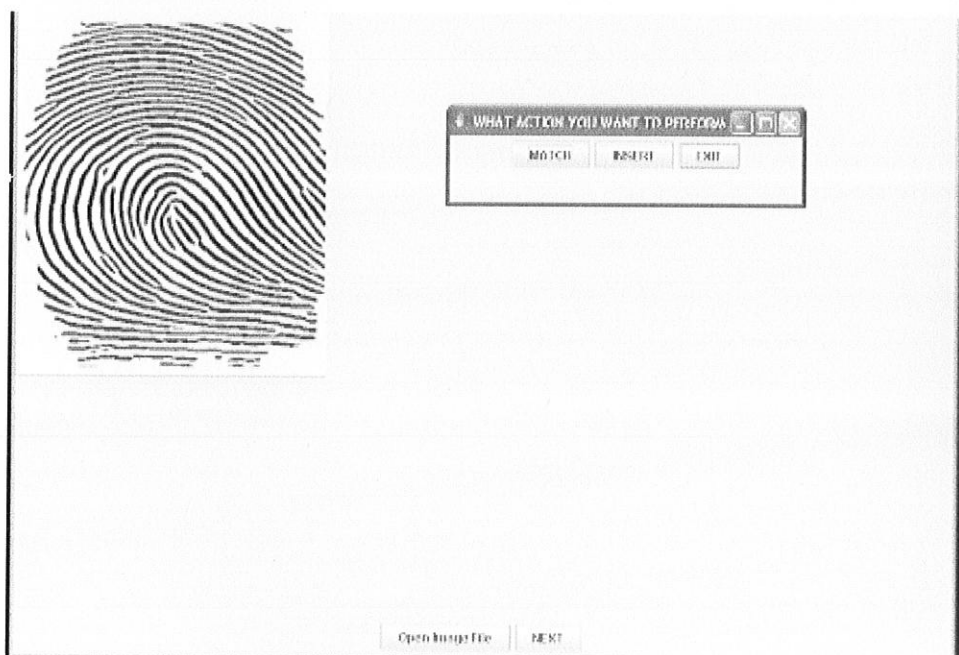


This is the welcome screen of our software which contains names of group members and a button labeled as "NEXT".

When user presses the next button then another window gets open which is given below:



In this window, after pressing open image file a small window appears from where you can select location of fingerprint image which is to be matched. In case of wrong selection click on “CANCEL” button and press “NEXT” to select again. After this another window will open which will show the uploaded image and ask for further options which are to be performed.



ALGORITHM IMPLEMENTATION

Chapter 4 Algorithm implementation

4.1 Feature extraction method

4.2 Concept of CN number

Fig 1: Two sample of fingerprint images showing the ridge features used for identification.

This stage is a crucial stage in the process of all the subsequent steps. It involves the extraction of features from the input image and the representation of these features in a form that can be used for identification.

4.2 CONCEPT OF CN NUMBER

Counting number of pixels 'p' is against which the value of 'CN' is determined. It is defined as pixels defining ridge valley.

ALGORITHM IMPLEMENTATION

In this chapter we have discussed about the process involved in building of our algorithm. This whole process can be divided into following steps:

4.1 FEATURE EXTRACTION MODULE

This module deals with the extraction of unique features from the fingerprint image like minutae points, bifurcation points, ridge ending points etc.

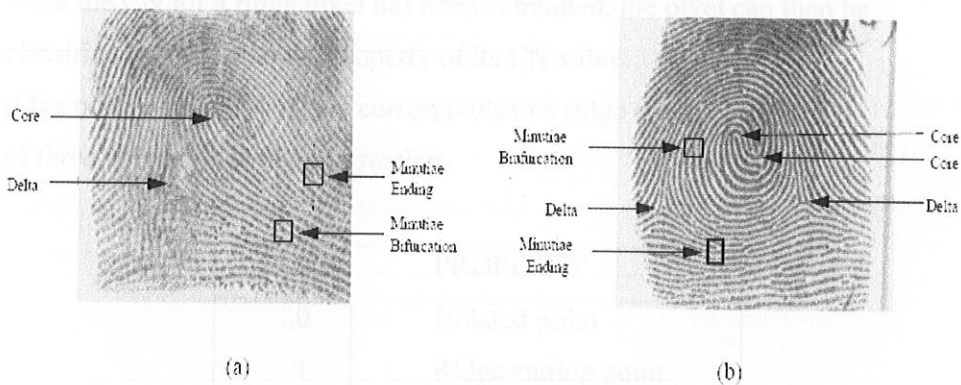


Fig4.1: Two example of fingerprint images showing the salient features used for authentication.

This stage is crucial since the success of all the subsequent steps depends upon the correct detection and extraction of these features. Minutae extractions was carried out using CROSS NUMBER approach.

4.2 CONCEPT OF CN NUMBER

Crossing number of pixel 'p' is defined as half the sum of differences between pairs of adjacent pixels defining eight neighbors of 'p'.

$$CN = 0.5 \sum_{i=1}^8 |P_i - P_{i+1}|$$

For a pixel P , its eight neighbouring pixels are scanned in an anti-clockwise direction as follows:

P4	P3	P2
P5	P	P1
P6	P7	P8

After the CN for a ridge pixel has been computed, the pixel can then be classified according to the property of its CN value.

ridge pixel with a CN of one corresponds to a ridge ending , and a CN of three corresponds to a bifurcation.

CN	PROPERTY
0	Isolated point
1	Ridge ending point
2	Continuing ridge point
3	Bifurcation point
4	Crossing point

For each extracted minutae point the following information is recorded:

- 1.X and Y Coordinates
2. Type of minutae (ridge end or bifurcations)

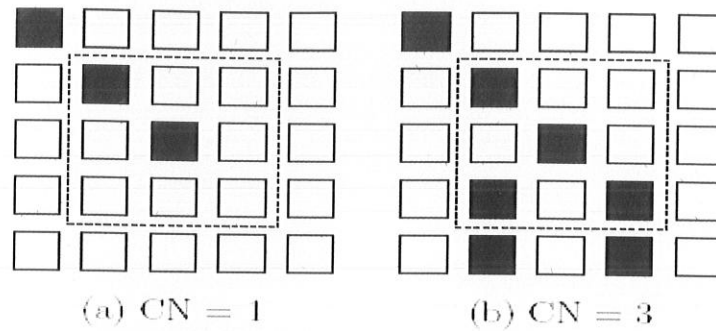


Fig4.2: Two types of minutae point

Most fingerprint based authentication systems use information extracted from minutiae bifurcation and endings, as well as the ridge flow and singularities to assess the degree of similarity between two fingerprints.

After extraction this is another very important step in our process, here we have taken the distance and angle between two minutae points as the basis of matching. What we are doing here is that we have predefined distance between all minutae points and angles between them in our database and now we are matching the given image on its basis.

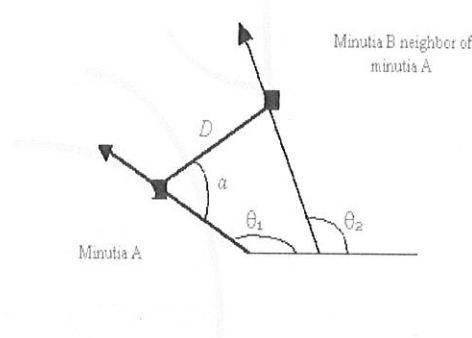


Fig.4.3: Angle between minutae point

It compares all the images and their values in database and look for the best match where all angles and distances are same and then displays whether match found or not.

Chapter 5 GUI (Graphical user interface)

After this window, if we'll press "MATCH" button then our code for matching will start running in backend.

If we'll press "INSERT" then all data related to selected image will be added in the database.

If we'll select "EXIT" then program will get terminated.

(Figure 1- Dashboard)

DATABASE

A database is a collection of related data which helps in creation of table and insertion, deletion, retrieval and storage of data. We require database to store, and subsequently add the matching parameters of image which are image number, pixel values, angle and distance between minutae points.

We have used Microsoft office Access to build our two databases, one of which stores originally stored parameters and other one acts like a buffer and is used to store values for comparing.

Further we have connected it to our program using JDBC (java database connectivity) which also helps in retrieval of values from data base and allow insertion and deletion using SQL queries.

Once matching parameters are fed into database it will look like

x	y	w	z
1	1.570696	1	1
3	1.570762	1	2
58	1.570794	1	3
1	1.570696	1	4
2	1.570746	1	5
1	1.570696	1	6
7	1.570782	1	7
1	1.570696	1	8
1	1.570696	1	9
1	1.570696	1	10
2	1.570746	1	11
1	1.570696	1	12
17	1.57079	1	13
1	1.570696	1	14
1	1.570696	2	15
3	1.570762	2	16
58	1.570794	2	17
1	1.570696	2	18
2	1.570746	2	19
1	1.570696	2	20
7	1.570782	2	21
1	1.570696	2	22
1	1.570696	2	23
1	1.570696	2	24
2	1.570746	2	25
1	1.570696	2	26
17	1.57079	2	27
1	1.570696	2	28
11	1.570787	2	29

Above screenshot is of main database which contains different matching parameters of two images represented by match_id 1 and 2.

Table2 : Table			
	x	y	z
▶	1	1.57069635391	1
	3	1.57076299191	2
	58	1.57079458237	3
	1	1.57069635391	4
	2	1.57074630260	5
	1	1.57069635391	6
	7	1.57078206539	7
	1	1.57069635391	8
	1	1.57069635391	9
	1	1.57069635391	10
	2	1.57074630260	11
	1	1.57069635391	12
	17	1.57079041004	13
	1	1.57069635391	14
	11	1.57078719139	15
*			

This screenshot is of buffer database which stores parameters only till matching is performed and when new image is loaded old values get deleted and new get entered.

Testing

Chapter 7 Testing

Testing:

The design process of finger print recognition system involves testing that should "begin in small" and progress towards "testing in large". White box testing which involves testing internal program logic is main form of testing which was performed on our system. Since our code development process can be easily sub divided into modules we have performed unit level testing and white box testing together. The first level of the of this combined testing was tested over binarisation code i.e. code which converts image into binary form. Second unit would comprise of checking that whether CN number calculated over a set of matrix values generates CN number value for each element of matrix or not. Third unit would implement CN number calculated over the set of values obtained by loading of the image into the program.

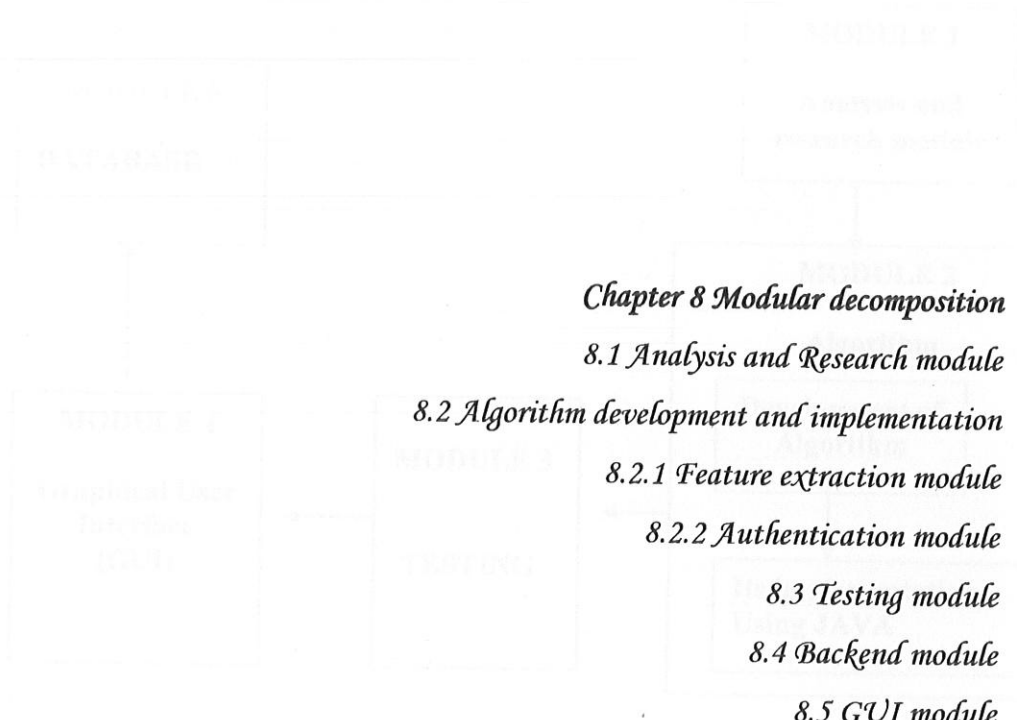
The functional level testing along with white box testing is implemented in the phase when Java connectivity with Access was done. In this the integration testing also sets in because we input the values into access database through the program which contains calculation performed over CN number method program for an image. White box test case tests internal logic for the programs at each step.

Query development for matching of the values stored in two different tables of the database by doing programming at front end is itself a module for which white box testing along with structural level testing has been specified. Before query was tested over the program it had to go under unit test over almost similar set of values that were obtained from image on SQL.

Integration of the entire system i.e. Insert button on GUI for insertion of data into main data base table, Match button on GUI for matching of the values of the image loaded with the one already there in the database and finally enabling of Exit button on button panel as soon as the image is loaded on the system, was done. Entire above scenario was tested one by one on the integrated system.

MODULAR DECOMPOSITION

Module is divided into following modules:



Chapter 8 Modular decomposition

8.1 Analysis and Research module

8.2 Algorithm development and implementation

8.2.1 Feature extraction module

8.2.2 Authentication module

8.3 Testing module

8.4 Backend module

8.5 GUI module

Fig 8.1 Modular decomposition

MODULAR DECOMPOSITION

Our project can be divided into following modules

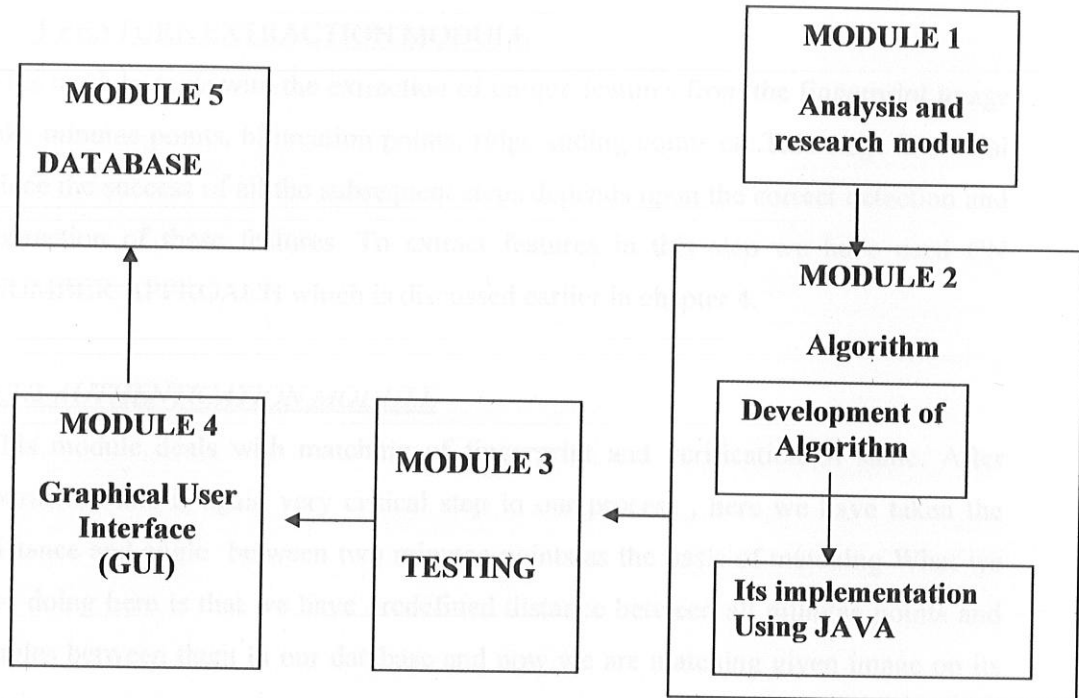


Fig 8.1 Modular architecture

8.1. ANALYSIS AND RESEARCH MODULE

In this module we carried out extensive literature survey studied the existing techniques of feature extraction and matching in the field of fingerprinting. We have gone through various research papers and books on fingerprint matching and data available on internet and finally made a layout of our algorithm. Research helped us in developing a new effective algorithm which is also very efficient in terms of accuracy and time and space complexities.

8.2. ALGORITHM DEVELOPMENT AND IMPLEMENTATION

MODULE

The whole process can be divided into following steps:

8.2.1 FEATURE EXTRACTION MODULE

This module deals with the extraction of unique features from the fingerprint image like minutae points, bifurcation points, ridge ending points etc. This stage is crucial since the success of all the subsequent steps depends upon the correct detection and extraction of these features. To extract features in this step we have used CN NUMBER APPROACH which is discussed earlier in chapter 4.

8.2.2 AUTHENTICATION MODULE

This module deals with matching of fingerprint and verification of same. After extraction this is again very critical step in our process, here we have taken the distance and angle between two minutae points as the basis of matching. What we are doing here is that we have predefined distance between all minutae points and angles between them in our database and now we are matching given image on its basis.

8.3. TESTING MODULE

Also known as unit or component testing phase, module testing is concerned with the testing of the smallest piece of software for which a separate specification exists. In this module what we have done is we took 5 fingerprint images and their gray values, pixel location of minutae points, angle and distance between minutae points into the database and matched them using our program. In some cases when same image was taken program gave perfect match while in other cases when image was taken from outside it did not match with database.

8.4. BACK END MODULE (DATABASE)

This module deals with the creation, insertion, deletion and retrieval of matching parameters which are angle and distance from the database.

Here we have used JDBC which is APPLICATION PROGRAMMING INTERFACE. Its function is to document a standard framework for dealing with data which may be tabular and relational data. This database helps in creating table and storing image and their corresponding pixel values, angle and distance between minutae points which are further matched by our program.

This module also helps in retrieval of values from database and allows insertion and deletion in databases. Through JDBC we can connect our code with databases and run SQL query in it.

8.5. GRAPHICAL USER INTERFACE MODULE

A graphical user interface (GUI) presents a pictorial interface to a program . A GUI (pronounced “GOO-EE”) gives a program a distinctive “look” and “feel” providing different programs with a consistent set of intuitive user interface components provides user with a basic level of familiarity with each program before they ever use it. In turn, this reduces the time which users require to learn a program and increases their ability to use the program in a productive manner.

GUIs are built from GUI components. A GUI component is an object with which the user interacts via the mouse, the keyboard or another form of inputs, such as voice recognition. Several common GUI components are JLabel, JTextField, JButton, JList, JCheckBox, JPanel etc.

The classes that create the GUI components are part of the swing GUI components from package ‘javax.swing’.

We have used swing components in our GUI to make our program more user friendly and included facilities like windows, buttons, panels etc which are built in java swing.

Chapter 9- Hardware and Software Requirements.

H/W and S/W Requirement

For any software which is developed over a time period there are certain hardware and software components that are required to form a system. Hardware and software requirements of the system can be of two type's minimum and recommended where recommended is suggested by the software vendor and minimum are those which must be satisfied for the software to be usable at all. For our Fingerprint Recognition system hardware and software requirements are:

HARDWARE REQUIREMENTS:

1. Memory (RAM): Since, the fingerprint recognition involves consideration of the minutiae point details for each fingerprint .The data sizes of these fingerprints are very large sizes hence processor with low ram will take a large amount of time in processing hence recommended and minimum RAM for the system are 512 MB and 256 MB respectively.
2. Hard Drive (HDD): Hard-disk requirements vary, depending on the sizes of software installation, temporary files created and maintained .The hard disk requirement for our system is very low recommended is 80GB and minimum is 40GB.
3. Processing Power (CPU): A system with Intel Pentium IV processor and x86 architecture with 800 MHZ FSB (Front Side Bus) is recommended.
4. Display Adapter: Recommended: 32bits
Minimum : 16bits.
5. Peripherals: Peripherals required for the system are Standard PS/2mouse and Standard keyboard and Cd-Rom for the installation guide.

SOFTWARE REQUIREMENTS:

1. Operating System: Windows Xp Professional.
2. Java Development Kit: Version: jdk-1_5_0_15-windows-i586-p.
3. Java Runtime Environment: Version: jre-6-windows-i586.
4. Microsoft Office 2003: Microsoft Office Access 2003.

INSTALLATION GUIDE

The following steps describe the installation of the software. The software is installed on the hard disk of the computer. The software is installed on the hard disk of the computer. The software is installed on the hard disk of the computer.

JAVA DEVELOPMENT KIT (JDK) 1.4.2_01

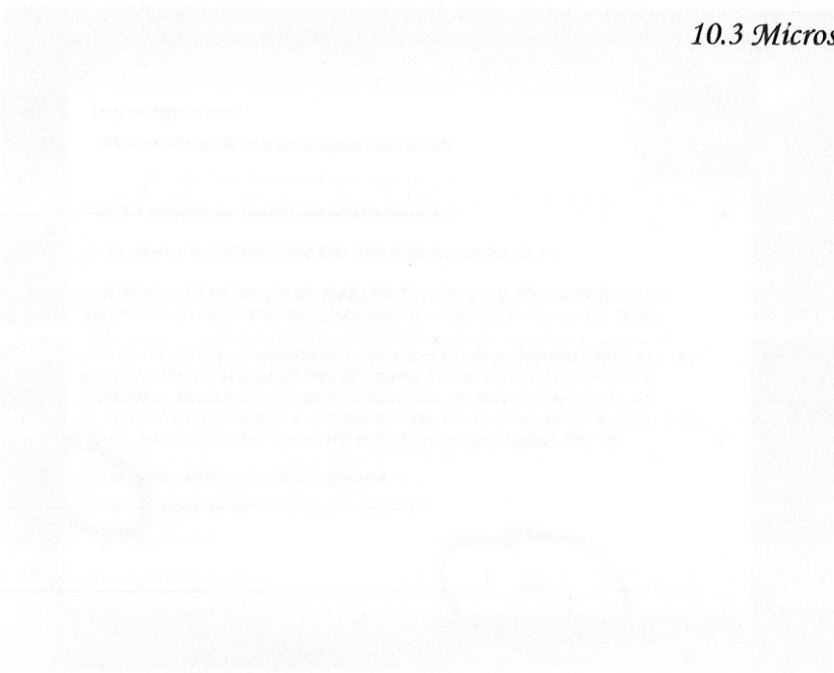
Java development kit or popularly known as JDK is the most important part of the Java environment. The JDK is a software development kit which is used to develop Java applications. The JDK is a software development kit which is used to develop Java applications. The JDK is a software development kit which is used to develop Java applications.

10 Installation guide

10.1 Java development kit

10.2 Java runtime application

10.3 Microsoft access 2003



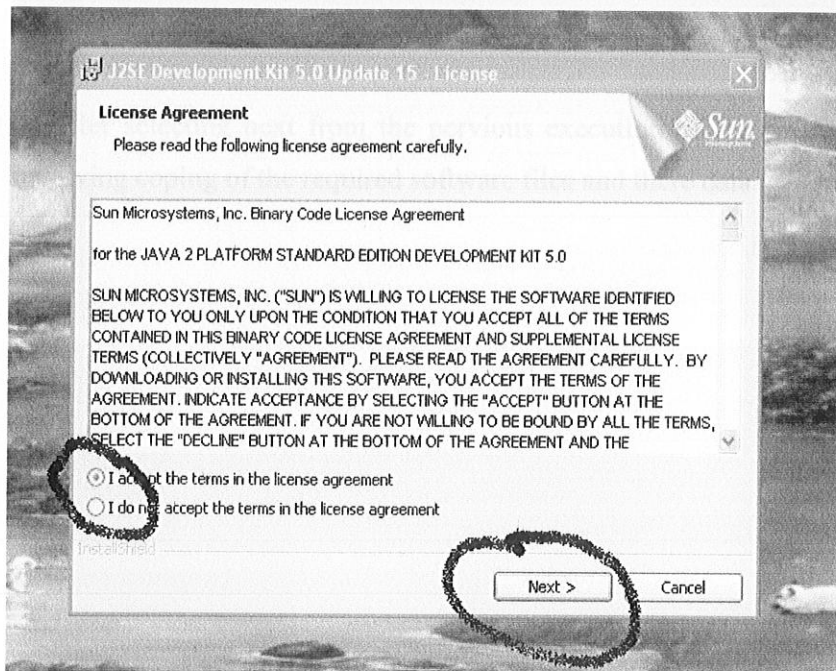
INSTALLATION GUIDE

This section of the project report describes about how the various software's which are used up in the project are installed .as ours is a project where we would be using the software's which are already developed by third party software developers .The project requires the machine to be installed with the following list of software's

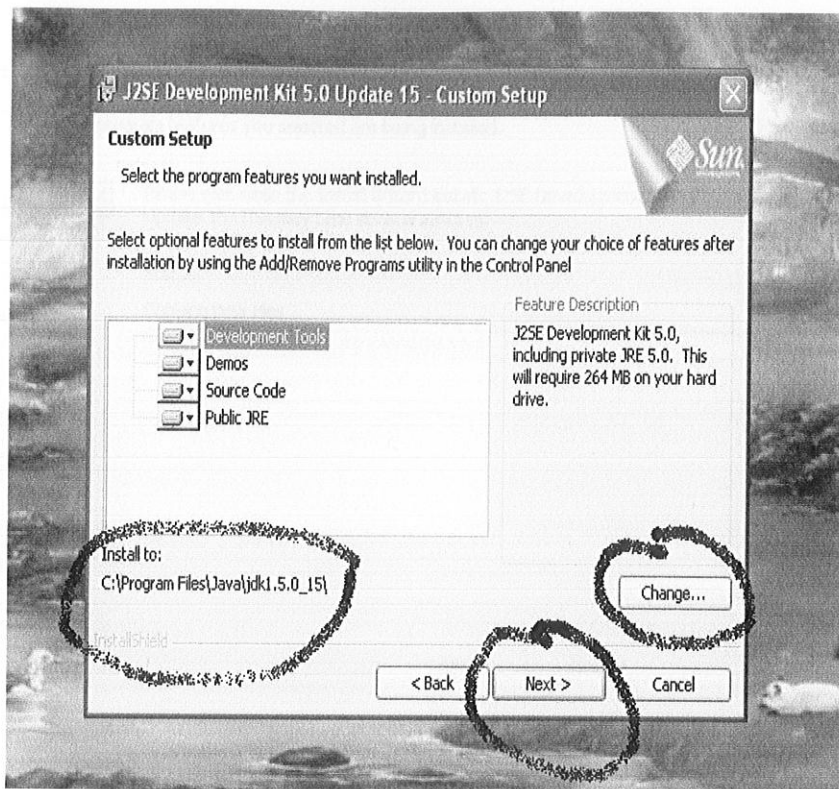
10.1. JAVA DEVELOPMENT KIT (jdk-1_5_0_15-windows-i586-p)

Java development kit or popularly known as jdk form the most important part of project development as the entire project is built over java which is provided by it jdk-1_5_0_15-windows-i586-p is a direct executable file. The installation instruction for this along with graphical interface has been provided here

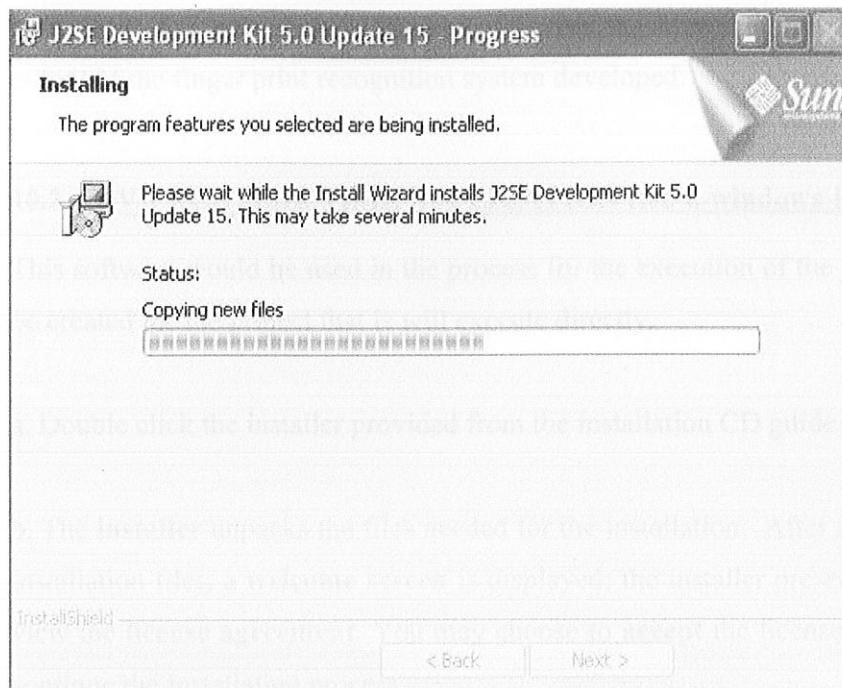
- a. Double click the installer provided from the installation CD guide provided
- b. Following screen appears that identifies the option to be selected



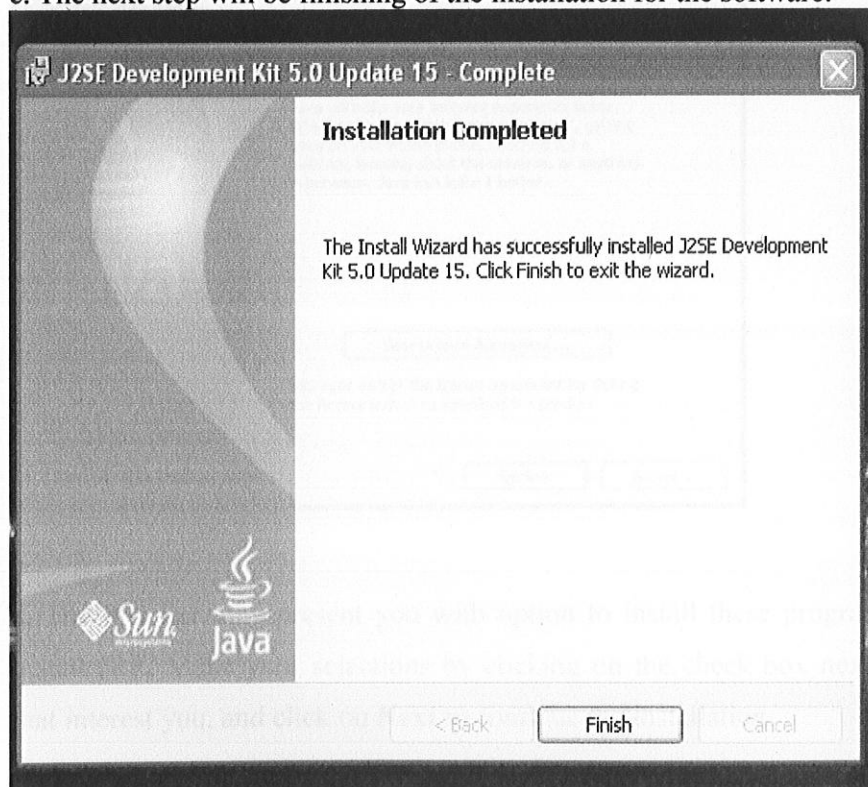
- c. We move to next screen where user should select where he will install the software the location for the software can also be changed



d. After selecting next from the pervious execution of the entire set up will run involving coping of the required software files and there data



e. The next step will be finishing of the installation for the software.

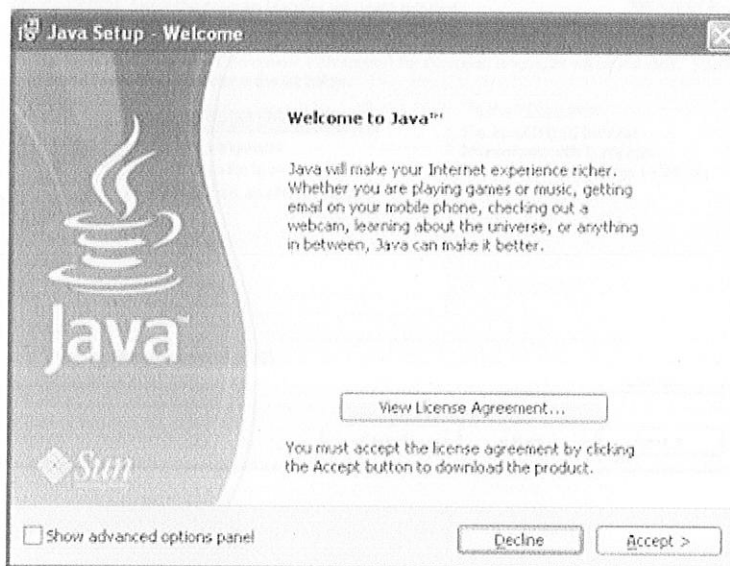


Now, user will have java installed in his /her machine required for viewing the output of the finger print recognition system developed.

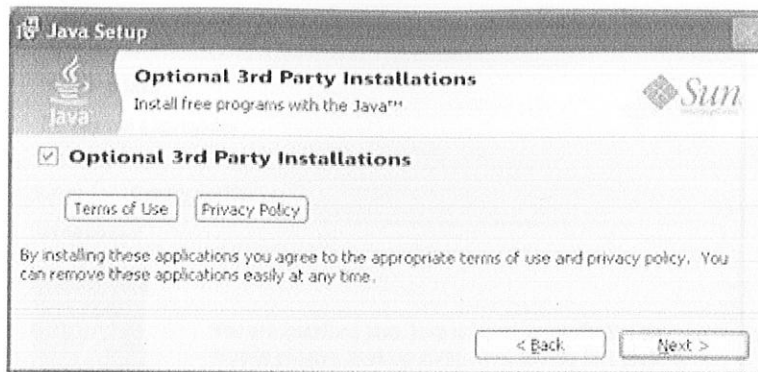
10.2. JAVA RUNTIME TIME APPLICATION (jre-6-windows-i586)

This software would be used in the process for the execution of the jar file that will be created for the project that is will execute directly.

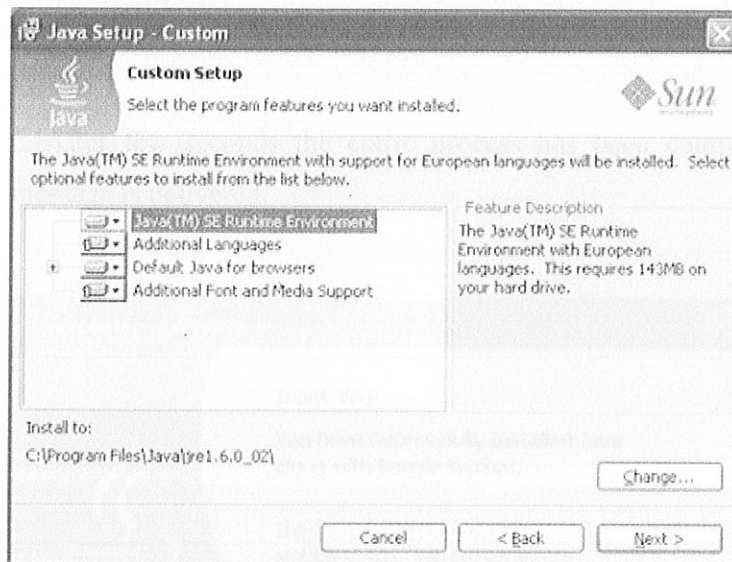
- a. Double click the installer provided from the installation CD guide provided .
- b. The **installer** unpacks the files needed for the installation. After unpacking the installation files, a **welcome screen** is displayed; the installer presents an option to view the **license agreement**. You may choose to **accept** the license agreement and continue the installation process



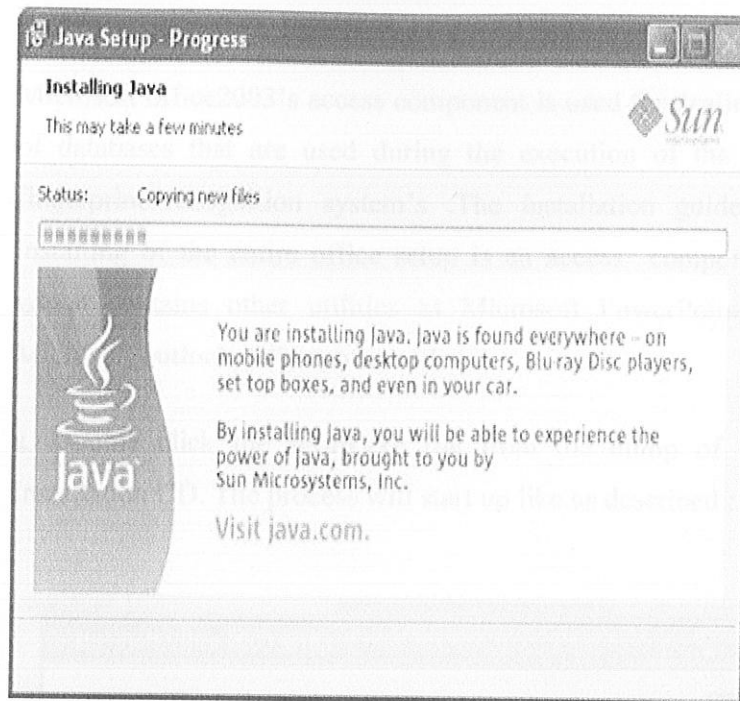
- c. The installer may present you with option to install these programs when you install JRE. Make your selections by clicking on the check box next to programs that interest you, and click on **Next** to continue the installation.



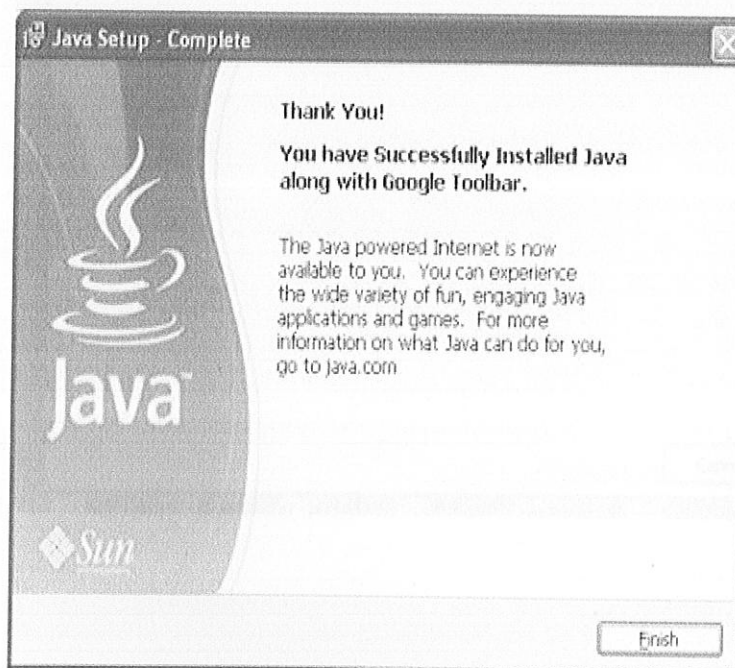
d. The installer displays a **Custom Setup** screen that allows you to choose program features to set up. Desired program features are selected, click the **Next** button to continue with the installation



e. After **next option** is clicked software starts copying the files required for the operation over long term application



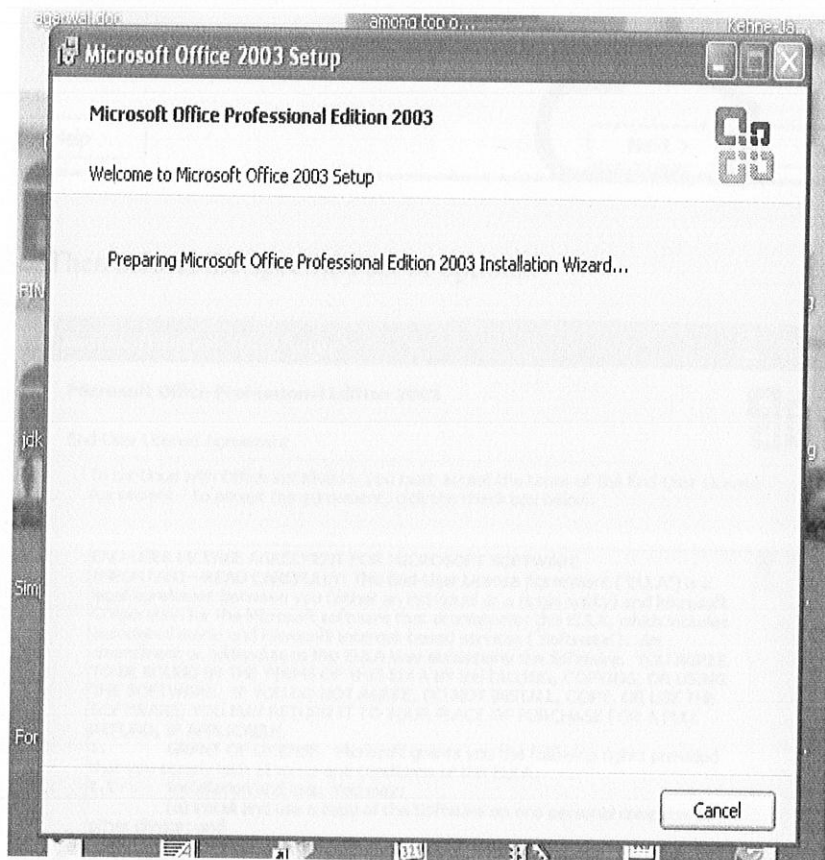
f. After few seconds the entire process has been completed and thanking you message is delivered.



10.3. MICROSOFT OFFICE ACCESS 2003

Microsoft office2003's access component is used for dealing with the manipulation of databases that are used during the execution of the jar file created for the fingerprint recognition system's .The installation guide for this involves the installing of the entire office setup is an access component of Microsoft office which contains other utilities as Microsoft PowerPoint, Microsoft font page, Microsoft outlook, Microsoft word

- a. Double click the **setup.exe** file from the dump of office2003 provided in Installation CD. The process will start up like as described :



b. The product key will be asked for and enter it as shown and then press next

In the boxes below, type your 25-character Product Key. You will find this number on the sticker on the back of the CD case or on your Certificate of Authenticity.

Product Key: - - - -

c..Then selects the specified set of options:

Microsoft Office 2003 Setup

Microsoft Office Professional Edition 2003

End-User License Agreement

To continue with Office installation, you must accept the terms of the End-User License Agreement. To accept the agreement, click the check box below.

END-USER LICENSE AGREEMENT FOR MICROSOFT SOFTWARE
IMPORTANT—READ CAREFULLY: This End-User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and Microsoft Corporation for the Microsoft software that accompanies this EULA, which includes associated media and Microsoft Internet-based services ("Software"). An amendment or addendum to this EULA may accompany the Software. YOU AGREE TO BE BOUND BY THE TERMS OF THIS EULA BY INSTALLING, COPYING, OR USING THE SOFTWARE. IF YOU DO NOT AGREE, DO NOT INSTALL, COPY, OR USE THE SOFTWARE; YOU MAY RETURN IT TO YOUR PLACE OF PURCHASE FOR A FULL REFUND, IF APPLICABLE.

1. GRANT OF LICENSE. Microsoft grants you the following rights provided that you comply with all terms and conditions of this EULA:

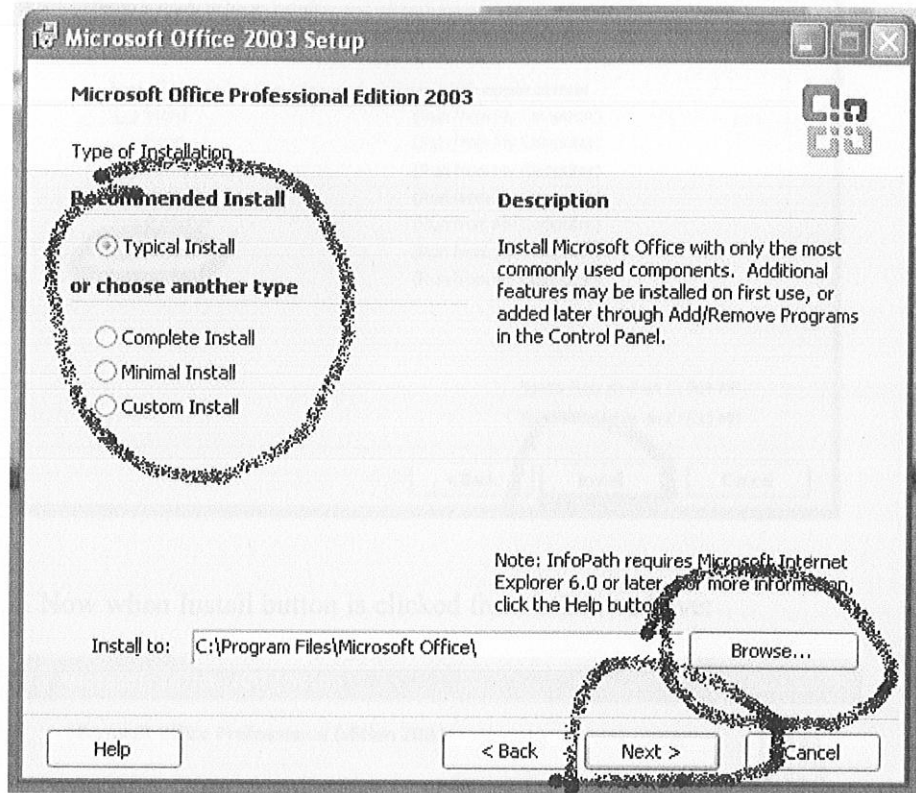
1.1 Installation and use. You may:

(a) install and use a copy of the Software on one personal computer or other device; and

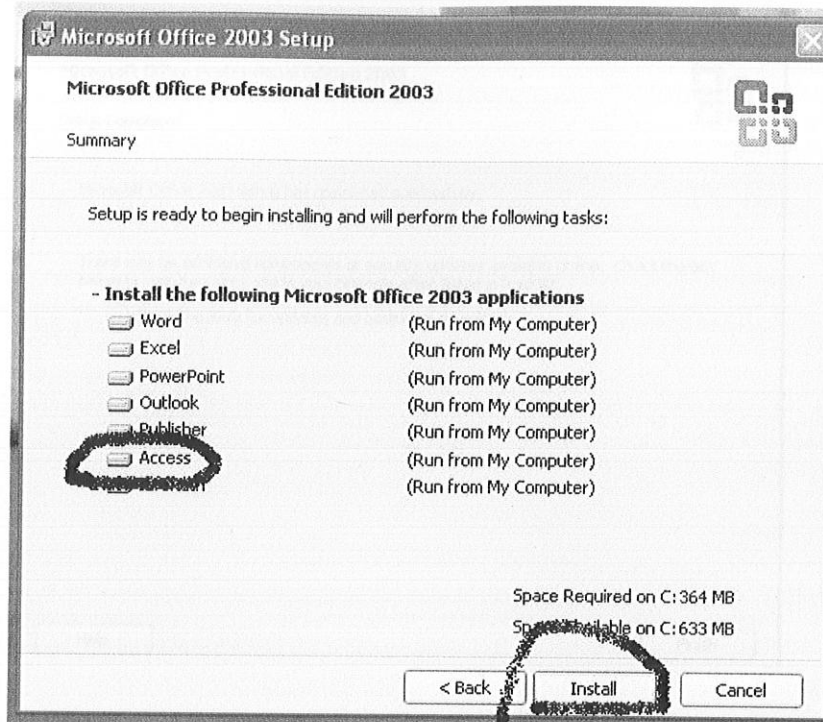
(b) install an additional copy of the Software on a second, portable device for the personal use of the primary user of the first copy of the Software.

☒ I accept the terms in the License Agreement

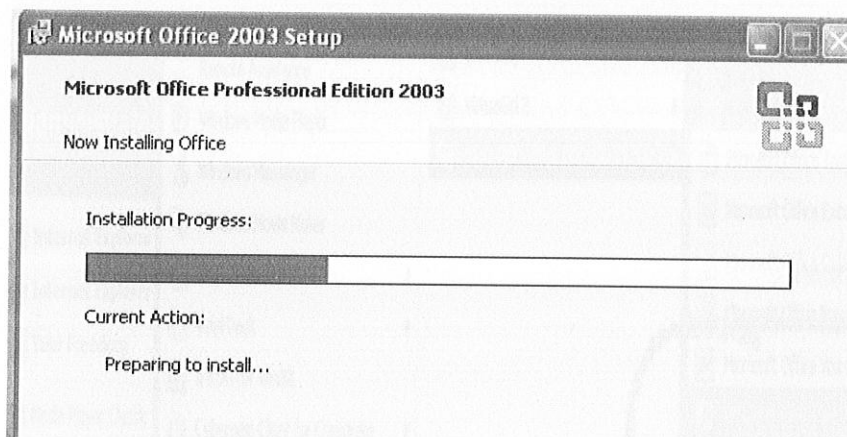
d. Then select the type of the installation then where to install the software and then next button



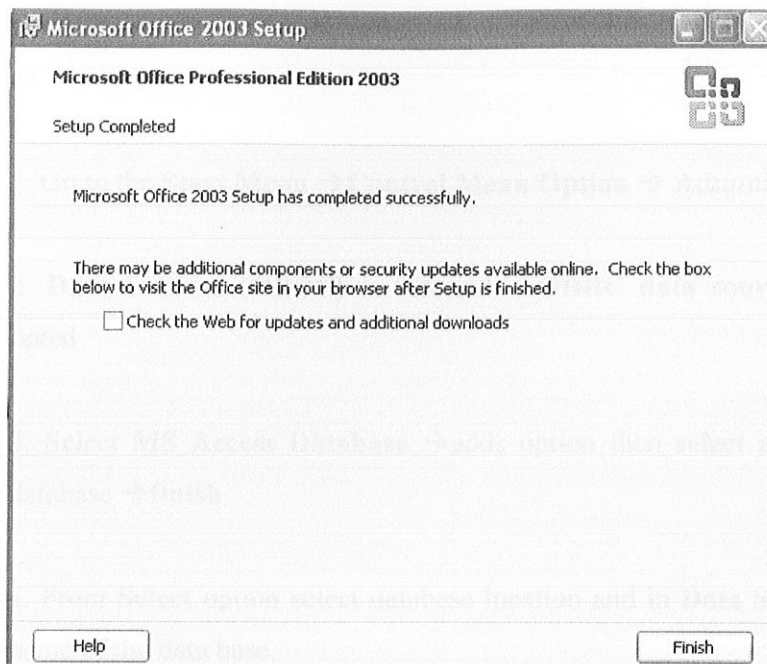
e. This will show what applications are going to be installed in for the along with required access:



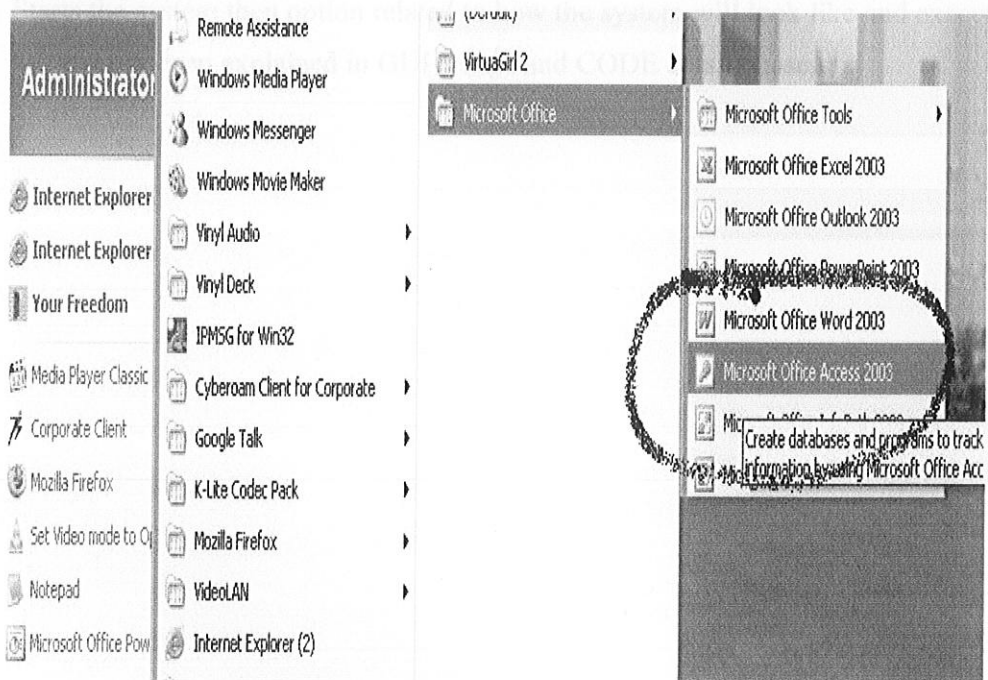
f. Now when Install button is clicked from other we have:



g. Finishing of the installation process is defined :



h..The Microsoft's access can be accessed from start menu option:



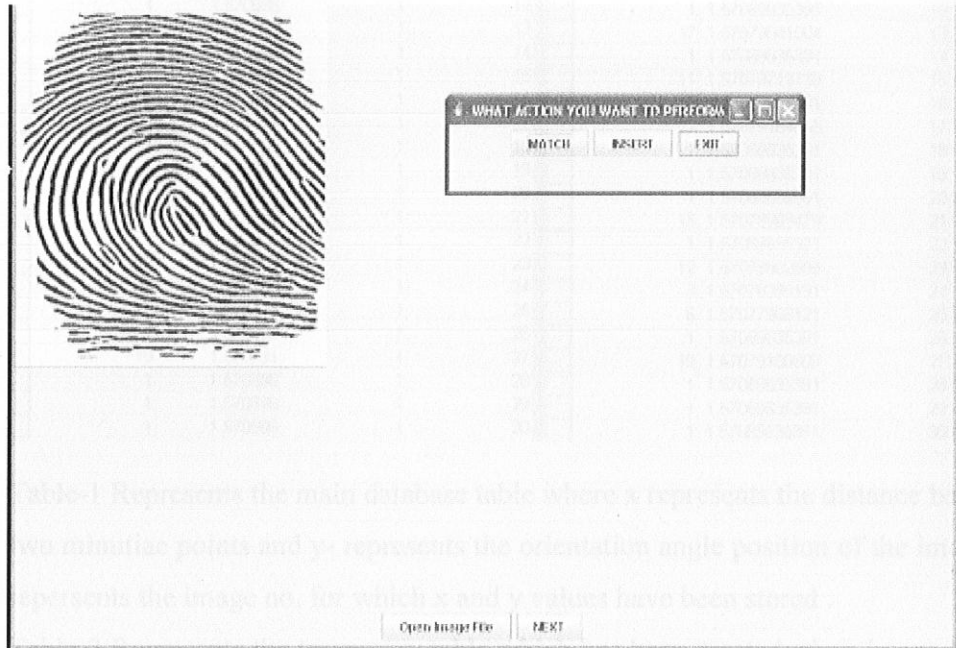
Now, connection has to be set up for the ACCESS: ODBC network which is done as

1. Go to the **Start Menu** → **Control Menu Option** → **Administrative tools** from it
2. **Data Sources (ODBC)** is selected → **ODBC data source Administrator** is opted
3. Select **MS Access Database** → adds option then **select access driver** for the database → **finish**
4. From **Select** option select database location and in **Data Source Name** add the name of the data base.

Thus, how the entire setup has been generated for the execution the Finger Print Recognition System that has been built up for production of demo that algorithm develops work. The software developed by us will make use of all these when user Starts the system then option related to how the system will look like and execution has already been explained in GUI design and CODE detail phase.

Result

Result is the output of the program. Objective of the system was to produce an efficient fingerprint matching algorithm. The main stress was on matching of the finger print .To produce result -we took two images and store there value in the main database table .To check precession of our algorithm we took one of the two images and load it in the system .



Then press the match button, at data base level this was observed:

x	y	w	z
1	1.570696	1	1
3	1.570762	1	2
58	1.570794	1	3
1	1.570696	1	4
2	1.570746	1	5
1	1.570696	1	6
7	1.570782	1	7
1	1.570696	1	8
1	1.570696	1	9
1	1.570696	1	10
2	1.570746	1	11
1	1.570696	1	12
17	1.57079	1	13
1	1.570696	1	14
11	1.570787	1	15
1	1.570696	1	16
22	1.570791	1	17
1	1.570696	1	18
1	1.570696	1	19
1	1.570696	1	20
15	1.570789	1	21
1	1.570696	1	22
12	1.570788	1	23
3	1.570762	1	24
6	1.570779	1	25
1	1.570696	1	26
19	1.570791	1	27
1	1.570696	1	28
1	1.570696	1	29
1	1.570696	1	30

x	y	z
1	1.57069635391	1
3	1.57076299191	2
58	1.57079458237	3
1	1.57069635391	4
2	1.57074630260	5
1	1.57069635391	6
7	1.57078206539	7
1	1.57069635391	8
1	1.57069635391	9
1	1.57069635391	10
2	1.57074630260	11
1	1.57069635391	12
17	1.57079041004	13
1	1.57069635391	14
11	1.57078719139	15
1	1.57069635391	16
22	1.57079184055	17
1	1.57069635391	18
1	1.57069635391	19
1	1.57069635391	20
15	1.57078969479	21
1	1.57069635391	22
12	1.57078802586	23
3	1.57076299191	24
6	1.57077968121	25
1	1.57069635391	26
19	1.57079100609	27
1	1.57069635391	28
1	1.57069635391	29
1	1.57069635391	30

Table-1 Represents the main database table where x represents the distance between two minutiae points and y- represents the orientation angle position of the image.w-repersents the image no. for which x and y values have been stored .

Table-2 Represents the temporary table which has been created when image loaded is asked for matching.

When Match button executes it matches for the x and y values in Table-1 and Table-2 and when match is found between these values based on the query analysis There is display of image no. for which values match .Match function produces the threshold based match where (97-100) % precision has been defined for set of values .which are compared.

The particular kind of output has been produced for the dataset that has been entered in the database.

Match from image no1

Thus, we are able to achieve our objective of achieving a precision in the match.

Chapter 17 Conclusion and Future work

Conclusion Work:

In this work we have successfully developed a program working on the use of Coordination as method. The main emphasis was on the handling of the fragments with template which was already stored in the database. Time taken by the system for every file template is a secondary to production of match for an image. The template match for an image is based on what percentage of image loaded for match matches the with each stored fragment data and then by use of threshold value we produce the result. Threshold value will be in range from (57-100) %, thus accuracy of the result will be (0.77-1.0) on scale of 1.0.

Thus our system enhances the human computer communication facilitating security mechanism. **Chapter 11 Conclusion and Future work**

Conclusion Work:

In this project we have successfully developed a fingerprint matching algorithm by the use of Coordination no. method. The main emphasis was given on matching of the fingerprint with template which was already stored in the database. Time taken by the system for successful template match is a secondary to production of match for an image .The template match for an image is based on what percentage of image loaded for match matches the with each stored fingerprint data and then by use of threshold value we produce the result. Threshold value will be in range from (97-100) %, thus accuracy of the result will be (0.97 -1.0) on scale of 1.0.

Thus our system enhances the human computer communication facilitating security mechanism.

Future Work:

This project can be extended in various directions like improving the developed algorithm to reduce the time taken by the system to produce the match. Application of the various other image enhancement techniques such as region mark and frequency estimation. Project can be implemented on Web Server as it is created in Java Applet function. This project can also be implemented on LAN and can actually be used to provide real life utilities like marking attendance, access grant.

Chapter 12 Implemented Code

CODING

INSERTION OF NEW IMAGEG INTO THE DATABASE

```
import javax.swing.*;
```

```
/*The Swing related classes are contained  
and its sub packages.*/
```

```
import java.awt.*;
```

```
/*It include all Abstract Windows Toolkit  
since all applets run in a window, it is  
Necessary to include support for that window*/
```

```
import javax.swing.JDialog;
```

```
/* All the GUI component packages. This includes
```

```
import java.applet.*;
```

```
/*This imports our native java applet class  
for us to use and contains several methods  
that give you detailed control over the execution  
of your panel*/
```

```
/* GUI class in the class */
```

```
import java.awt.image.*;
```

```
/*They provide support for imaging( the display  
and manipulation of graphical images).and several  
others that offer enhanced control over the imaging  
process and that support advanced.*/
```

```
import java.lang.*;
```

```
/*This package defines a class called System,  
Which encapsulates several aspects of the run-time  
Environment */
```

```
import java.sql.*;
```

```
/*this package invokes connection method and all  
various methods like submit SQL query to the database.*/
```

```
import java.math.*;
```

```
/*this package include all the math operation like  
Square root*/
```

```
import javax.swing.JFrame;
```

```
/*All the GUI component package. this include  
Frame in the GUI.*/
```

```
import javax.swing.JTextField;
```

```
/*All the GUI component package. this include  
Text field in the GUI.*/
```

```

/*
<APPLET
CODE=insert.class
WIDTH=248
HEIGHT=248 >
</APPLET>
*/

/*This define the applet width and height*/

public class insert extends JApplet /*Create an insert class*/
{

Image image; /*Create an object image of Image class*/
int xx1,zz1; /*Globally declare the variable*/

public void init()
/*Called once by the applet container when an applet is
Loaded for execution. This method initializes an applet.
Typical actions performed here are initializing fields,
creating GUI components, loading image to display.*/
{
image = getImage(getDocumentBase(),"aa.jpg");

/*Class Image is an abstract class---the applet cannot create
an object of class Image directly.getImage takes two argument
the location of image file and the file name of the image.
In the first argument, applet method getDocumentBase returns a
URL representing the location of the image on the internet(or on
your computer if the applet was loaded from your computer).Method
getDocumentBase returns the location of the HTML file as an object
of class URL. The second argument specifies an image file name.*/

```



```

int pixels[] = new int[248*248];
/*Declare one dimensional array i.e.pexels which contains
Value of the RGB image.*/

float z[]=new float[248*248];
float xx[][]=new float[248][248];
float yy[][]=new float[248][248];
int u[] = new int [248*248];
int a[][] = new int[248][248];
int t[][] = new int[248][248];
int w[][][] = new int[248][248][3];
int i,j,k,b,c,d,m,f,g,h,l;
int n=0;
int s=0;
Connection c1;
PreparedStatement s1;

/*above variable are initialize and declare in the init() function.*/

PixelGrabber pg = new PixelGrabber(image, 0, 0, 248, 248,pixels, 0, 248);

/*create a PixelGraber object to grab the (x,y,w,h) section
of pixels from the given image and places them in an array.
PixelGrabber(Image image, int x, int y, int w, int h, int[]pixels,
int off ,int scansize)*/
try {
pg.grabPixels();
}
catch (InterruptedException e) {}

int count = 0;

```

/* in the below image converted into the binary form and stored in one dimensional array because PixelGrabber works on one dimensional array.*/

```
for (int loop_index = 0; loop_index < 248 * 248; loop_index++)
{
//System.out.println("pixels values " +pixels[loop_index]);
If (pixels[loop_index] >= -16777216 && pixels[loop_index]<=-8388608)
pixels[loop_index]=1;
else
pixels[loop_index]=0;

//System.out.println("pixels values are " +pixels[loop_index]);
```

```
count++;
```

```
}
```

/* in the below two for loop image converted into two dimensional array.
and array store binary form of image.*/

```
for(i=0; i<248; i++)
```

```
{
```

```
    for( j=0; j<248; j++)
```

```
    {
```

```
        t[i][j]= pixels[248*i+j];
```

```
        //System.out.println("pixels values---- " +t[i][j]);
```

```
    }
```

```
}
```

```
    //System.out.println("at particular values " +t[24][24]);
```

```
    //System.out.println("loops r " +count);
```

/*In next operation we apply the CN number algorithm to abstract
the minutiac point.*/

```

for (i=0;i<248;i++)
{
for ( k=0;k<248;k++)
{
i=i+1;k=k+1;
if(i>=0 && k>=0 && i<248 && k<248)
b=t[i][k];
else
b=0;
i=i-1;
k=k-1;
i=i+1;
if(i>=0 && k>=0 && i<248 && k<248)
c=t[i][k];
else
c=0;
i=i-1;
k=k+1;
if(i>=0 && k>=0 && i<248 && k<248)
d=t[i][k];
else
d=0;
k=k-1;
i=i-1;
k=k-1;
if(i>=0 && k>=0 && i<248 && k<248)
m=t[i][k];
else
m=0;
i=i+1;
k=k+1;
}
}

```

```

i=i-1;
if(i>=0 && k>=0 && i<248 && k<248)
f=t[i][k];
else
f=0;
i=i+1;
k=k-1;
if(i>=0 && k>=0 && i<248 && k<248)
g=t[i][k];
else
g=0;
k=k+1;
i=i+1;
k=k-1;
if(i>=0 && k>=0 && i<248 && k<248)
h=t[i][k];
else
h=0;
i=i-1;
k=k+1;
i=i-1;
k=k+1;
if(i>=0 && k>=0 && i<248 && k<248)
l=t[i][k];
else
l=0;
i=i+1;
k=k-1;

```

```

z[n] = (float) (0.5*((Math.abs(d-l)+Math.abs(l-f)+Math.abs(f-m)+Math.abs(m-g)
+ Math.abs(g-h)+Math.abs(h-c)+Math.abs(c-b)+Math.abs(b-d))));

```

```
/* z[n] array stored the integer value of each pixel after operation of CN number like  
0,1,2,3 and 4. */
```

```
    n++;
```

```
    }
```

```
    }
```

```
int v=0;
```

```
int p=0;
```

```
for(i=0;i<248;i++)
```

```
    {
```

```
        //int s=0;
```

```
        for( j=0;j<248;j++)
```

```
            {
```

```
                if(z[v]==1 || z[v]==3)
```

```
                {
```

```
                    //System.out.println("X coordinate r: " +i);
```

```
                    //System.out.println("Y coordinate r: " +j);
```

```
                    //System.out.println("a[i][j]:"+i + "," +j);
```

```
                    w[i][s][1]=i;
```

```
                    w[i][s][2]=j;
```

```
                    //System.out.println("values in the array:" +w[i][s][1] + "," +w[i][s][2]);
```

```
                    //System.out.println("value of i and s:" +i + "," +s);
```

```
                    s++;
```

```
                    p++;
```

```
                }
```

```
                v++;
```



```

}
}
int count1=0;

/* here we apply the algorithm to calculate the distance and
angle between the minutia point and stroed in two diffent arrays.*/

for(i=0;i<248;i++)
{
for(j=0;j<s;j++)
{

xx[i][j] = (float)(Math.sqrt(((w[i][j][1]-w[i][j+1][1]) * (w[i][j][1]-w[i][j+1][1]))
+ ((w[i][j][2]-w[i][j+1][2]) * (w[i][j][2]-w[i][j+1][2]))));

double aa=w[i][j+1][1]-w[i][j][1];
if(aa==0)
{
aa=0.0001;
}

yy[i][j] = (float)(Math.atan(((w[i][j+1][2]-w[i][j][2])/aa)));
System.out.println("Angle between minutia points are:" +yy[i][j]);

if(xx[i][j]>0)
{
System.out.println("Distance between minutia points are:" +xx[i][j]);
}

/* here we connect to the database and insert the distance and
angle values to the database.*/

```

```

try
{
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    c1=DriverManager.getConnection("jdbc:odbc:db1");
    if(ref == 0)
    {
        xx1=0;
        Statement stmt=c1.createStatement();
        ResultSet rs=stmt.executeQuery("Select distinct(w) from Table1 ");
        while(rs.next())
        {
            int yyy;
            yyy=rs.getInt("w");
            if(xx1 < yyy)
            {
                xx1=yyy;
            }
            //System.out.println(rs.getString ("z"));
        }

        xx1=xx1+1;
        zz1=0;
        rs=stmt.executeQuery("Select z from Table1 ");
        while(rs.next())
        {
            int yyy;
            yyy=rs.getInt("z");
            if(zz1 < yyy)
            {

```

```

zz1=yyy;
}
//System.out.println(rs.getString ("z"));

}
zz1=zz1+1;
count1=zz1;
ref=1;
}

s1=c1.prepareStatement("insert into Table1 values(?,?,?,?)");
s1.setFloat(1,xx[i][j]);
s1.setFloat(2,yy[i][j]);
s1.setInt(3,xx1);
System.out.println("count " +count1);
s1.setInt(4,count1);
s1.execute();

count1++;
}
catch(Exception e)
{
System.out.println("Exception due to " +e);
}

}
}

// JTextField t1=new JTextField("data has been inserted", 10);
// t1.setEditable(false);
//this.add(t1);

}

```

```
//System.out.println("angle and distance:" +xx[146][94] + "," +yy[146][94]);  
//System.out.println("number of (X,Y) coordinates are:--" +p);  
//System.out.println("v" +v);
```

```
public void paint(Graphics g)  
{  
    g.drawImage(image, 100, 100, this);  
}  
}
```

Matching from the database

```
import java.io.*;
import java.sql.*;
import javax.swing.*;
import javax.swing.JFrame;
import javax.swing.JTextField;
import java.awt.*;
import java.applet.*;
import java.awt.image.*;
import java.lang.*;
import java.math.*;
import java.awt.Graphics;
```

```
public class match extends JFrame
```

```
{
```

```
int h;
```

```
public Container contentPane;
```

```
public JPanel panel;
```

```
public void init()
```

```
{
```

```
int i,il;
```

```
i=0;
```

```
il=0;
```

```
try
```

```
{
```



```

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection A;
A=DriverManager.getConnection("jdbc:odbc:db1");
Statement stmt=A.createStatement();
ResultSet rs=stmt.executeQuery("Select distinct(w) from Table1 ");
while(rs.next())
{
//System.out.println(rs.getString("w"));
i=i+1;
//System.out.println(rs.getString ("z"));

}
int a[]=new int[i];

rs=stmt.executeQuery("Select distinct(w) from Table1 ");
while(rs.next())
{
//System.out.println(rs.getString("w"));
a[i1]=rs.getInt("w");
System.out.println(a[i1]);
i1=i1+1;
//System.out.println(rs.getString ("z"));

}
double d[]=new double[i1];

double count1,count2;
count1=0;
count2=0;

ResultSet rs1;

```

```

int i2;
for(i2=0;i2<i1;i2++)
{
    rs1=stmt.executeQuery("Select w from Table1 where x in(select x from Table2 ) and
    y in(select y from Table2) and w="+a[i2]);

    //System.out.println("Select      Table1.w      from      Table1,Table2      where
    ((Table1.x=Table2.x) and (Table1.y=Table2.y) and w="+a[i2]+")");

    count1=0;
    count2=0;
    while(rs1.next())
    {
        count1=count1+1;
    }

    rs1=stmt.executeQuery("select * from Table1 where w="+a[i2]);

    //System.out.println("select * from Table1 where w="+a[i2]);
    while(rs1.next())
    {
        count2=count2+1;
    }

    d[i2]=count1/count2;
    System.out.println("xxxxxxxxxxxxxx");
    System.out.println(a[i2]+" "+d[i2]);

}

```

```

h=0;
for(i2=0;i2<i1;i2++)
{
if(d[i2] > 0.98 || d[i2] == 1.0 )
{
h=a[i2];
//System.out.println("*****\n");
//System.out.println(a[i2]);
//System.out.println("*****\n");
}
}
JTextField t=new JTextField("Match from image no"+h,10);
t.setEditable(false);
panel = new JPanel();
panel.add(t);
contentPane = getContentPane();
contentPane.add(panel);
setVisible (true);
//paint(Graphics g);
//Graphics g;
//g.drawString("abc"+h,10,10);
}
catch(Exception e)
{
System.out.println(e);
}
}

//public void paint(Graphics g)
//{

```

```
//g.drawString("abc"+h,10,10);  
//}  
public static void main(String args[])  
{  
    match m=new match();  
    m.init();  
  
}  
  
}
```

Creating temporary table

```
import javax.swing.*;
import java.awt.*;
import java.applet.*;
import java.awt.image.*;
import java.lang.*;
import java.sql.*;
import java.math.*;
import java.net.*;
```

```
/*
```

```
<APPLET
```

```
CODE=insert3.class
```

```
WIDTH=248
```

```
HEIGHT=248 >
```

```
</APPLET>
```

```
*/
```

```
public class insert3 extends Applet
```

```
{
```

```
Image image;
```

```
int xx1,zz1;
```



```

public void init()
{
    image = getImage(getDocumentBase(), "aa1.jpg");
    int pixels[] = new int[248*248];
    float z[]=new float[248*248];
    float xx[][]=new float[248][114];
    float yy[][]=new float[248][114];
    int u[] = new int [248*113];
    int a[][] = new int[248][248];
    int t[][] = new int[248][248];
    int w[][][] = new int[248][248][3];
    int i,j,k,b,c,d,m,f,g,h,l;
    int n=0;
    Connection c1;
    PreparedStatement s1,s4;

    PixelGrabber pg = new PixelGrabber(image, 0, 0, 248, 248,pixels, 0, 248);

    Try
    {
        pg.grabPixels();
    }

    catch (InterruptedException e) {}

    int count = 0;
    for (int loop_index = 0; loop_index < 248 * 248; loop_index++)

    {

```

```

if(pixels[loop_index] >= -16777216 && pixels[loop_index]<=-8388608)
pixels[loop_index]=1;
else
pixels[loop_index]=0;
//System.out.println("pixels values are " +pixels[loop_index]);
count++;
}
for(i=0; i<248; i++)
{
for( j=0; j<248; j++)
{
t[i][j]= pixels[248*i+j];
}
}
for (i=0;i<248;i++)
{
for ( k=0;k<248;k++)
{
i=i+1;k=k+1;
if(i>=0 && k>=0 && i<248 && k<248)
b=t[i][k];
else
b=0;
i=i-1;
k=k-1;
i=i+1;
if(i>=0 && k>=0 && i<248 && k<248)
c=t[i][k];
else
c=0;

```

```

        i=i-1;
        k=k+1;
    if(i>=0 && k>=0 && i<248 && k<248)
        d=t[i][k];
    else
        d=0;
        k=k-1;
        i=i-1;
        k=k-1;
    if(i>=0 && k>=0 && i<248 && k<248)
        m=t[i][k];
    else
        m=0;
        i=i+1;
        k=k+1;
        i=i-1;
    if(i>=0 && k>=0 && i<248 && k<248)
        f=t[i][k];
    else
        f=0;
        i=i+1;
        k=k-1;
    if(i>=0 && k>=0 && i<248 && k<248)
        g=t[i][k];
    else
        g=0;
        k=k+1;
        i=i+1;
        k=k-1;
    if(i>=0 && k>=0 && i<248 && k<248)
        h=t[i][k];

```

```

        else
            h=0;
            i=i-1;
            k=k+1;
            i=i-1;
            k=k+1;
        if(i>=0 && k>=0 && i<248 && k<248)
            l=t[i][k];
        else
            l=0;
            i=i+1;
            k=k-1;

z[n] = (float) (0.5*((Math.abs(d-l)+Math.abs(l-f)+Math.abs(f-m)+Math.abs(m-g)
+ Math.abs(g-h)+Math.abs(h-c)+Math.abs(c-b)+Math.abs(b-d))));

        n++;
    }
}

int v=0;
int p=0;
for(i=0;i<248;i++)
{
    int s=0;
    for(j=0;j<248;j++)
    {
        if(z[v]==1 || z[v]==3)
        {
            w[i][s][1]=i;

```

```

w[i][s][2]=j;
s++;
p++;
}
v++;
}
}

int count1=1;
int ref=0;
for(i=0;i<248;i++)
{
for(j=0;j<114;j++)
{
xx[i][j] = (float)(Math.sqrt(((w[i][j][1]-w[i][j+1][1]) * (w[i][j][1]-w[i][j+1][1]))+
((w[i][j][2]-w[i][j+1][2]) * (w[i][j][2]-w[i][j+1][2]))));

double aa=w[i][j+1][1]-w[i][j][1];
if(aa==0)
{
aa=0.0001;
}
yy[i][j] = (float)(Math.atan(((w[i][j+1][2]-w[i][j][2])/aa)));
//System.out.println("Angle between minutia points are:" +yy[i][j]);
if(xx[i][j]>0)
{
//System.out.println("Distance between minutia points are:" +xx[i][j]);
}
try
{
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
c1=DriverManager.getConnection("jdbc:odbc:db1");

```



```

if(ref == 0)
{
xx1=0;
Statement stmt=c1.createStatement();
ResultSet rs=stmt.executeQuery("Select distinct(w) from Table1 ");
while(rs.next())
{
int yyy;
yyy=rs.getInt("w");
if(xx1 < yyy)
{
xx1=yyy;
}
}
xx1=xx1+1;
zz1=0;
rs=stmt.executeQuery("Select z from Table1 ");
while(rs.next())
{
int yyy;
yyy=rs.getInt("z");
if(zz1 < yyy)
{
zz1=yyy;
}
}
zz1=zz1+1;
count1=zz1;
xx1=1;
zz1=1;
count1=1;

```

```

s1=c1.prepareStatement("drop table Table2 ");
s1.execute();
s1=c1.prepareStatement("create table Table2(x number,y number,z number)");
s1.execute();
ref=1;
}
s1=c1.prepareStatement("insert into Table2 values(?,?,?)");
s1.setFloat(1,xx[i][j]);
s1.setFloat(2,yy[i][j]);
s1.setInt(3,count1);
s1.execute();
System.out.println("record added");
count1++;
}
catch(Exception e)
{
System.out.println("Exception due to " +e);
}
}
}
match mm1=new match();
mm1.init();
mm1.setSize(100,100);
mm1.setVisible(true);

}

public void paint(Graphics g)
{
g.drawImage(image, 100, 100, this);
}
}

```

GUI INTERFACE

MAIN GUI PAGE

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class labelimage extends JFrame implements ActionListener
{
    JButton button1;
    JPanel p1;
    public labelimage()
    {
        super("FINGERPRINT RECOGNITION SYSTEM");
        button1 = new JButton("NEXT");
        p1=new JPanel();
        add(p1);
        p1.add(button1);
        button1.addActionListener(this);
        Container contentPane = getContentPane();
        JLabel jlabel = new JLabel("FINGERPRINT RECOGNITION SYSTEM ", new);
        ImageIcon("label.jpg"),
        JLabel.CENTER);
    }
}
```



```
/* linked page with the main GUI page.*/
```

IMAGE SELECTION AND LOADING THE IMAGE

```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;
import java.io.*;

public class PictureFrame extends JFrame implements ActionListener
{
    Image img;
    JButton getPictureButton;
    JButton button2;
    public PictureFrame()
    {
        this.setSize(300, 300);
        this.setTitle("Image Frame");
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel picPanel = new JPanel();
        this.add(picPanel, BorderLayout.CENTER);
        JPanel buttonPanel = new JPanel();
        getPictureButton = new JButton("Open Image File");
        getPictureButton.addActionListener(this);
        buttonPanel.add(getPictureButton);
        this.add(buttonPanel, BorderLayout.SOUTH);

        this.setVisible(true);
    }
}
```



```

button2 = new JButton("NEXT");
button2.addActionListener(this);
buttonPanel.add(button2);
this.add(buttonPanel, BorderLayout.SOUTH);
this.setVisible(true);
}

public void actionPerformed(ActionEvent e)
{
    int m=0;
    while (m<=0)
    {
        String file = getImageFile();
        if (file != null)
        {
            Toolkit kit = Toolkit.getDefaultToolkit();
            img = kit.getImage(file);
            img = img.getScaledInstance(300, -1, Image.SCALE_SMOOTH);
            this.repaint();
        }
        m++;
    }
    EventObjectWindow n=new EventObjectWindow();
    n.setSize(300,100);
    n.setVisible(true);
    setVisible(true);
    // call the EventObjectWindow
}

private String getImageFile()
{
    JFileChooser fc = new JFileChooser();
    fc.setFileFilter(new ImageFilter());

```

```

int result = fc.showOpenDialog(null);
File file = null;
if (result == JFileChooser.APPROVE_OPTION)
{
    file = fc.getSelectedFile();
    return file.getPath();
}
else
return null;
}

private class PicturePanel extends JPanel
{
    public void paint(Graphics g)
    {
        g.drawImage(img, 0, 0, this);
    }
}

```

```

private class ImageFilter
extends javax.swing.filechooser.FileFilter
{
    public boolean accept(File f)
    {
        if (f.isDirectory())
            return true;
        String name = f.getName();
        if (name.matches(".*((.jpg)|(.gif)|(.png))"))
            return true;
        else
            return false;
    }
}

```

```
}
```

```
public String getDescription()
```

```
{
```

```
return "Image files (*.jpg, *.gif, *.png)";
```

```
}
```

```
}
```

```
}
```

SELECT BUTTONS FOR PERFORMING INDIVIDUAL TASK

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class EventObjectWindow extends JFrame implements ActionListener
{
    public JButton button1,
    button2,
    button3;
    public JPanel panel;
    public Container contentPane;
    public final int WINDOW_WIDTH=400 , WINDOW_HEIGHT=80;
    public EventObjectWindow()
    {
        //set the title bar text .
        super("WHAT ACTION YOU WANT TO PERFORM");
        //set the size of window.
        setSize(WINDOW_WIDTH, WINDOW_HEIGHT);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // create three buttons.
        button1 = new JButton("MATCH");
        button2 = new JButton("INSERT");
        button3 = new JButton("EXIT");
        //register eventlistener with all buttons.
```

```

button1.addActionListener(this);
button2.addActionListener(this);
button3.addActionListener(this);

// create panel and adds buttons to it
panel = new JPanel();
panel.add(button1);
panel.add(button2);
panel.add(button3);

// add panel to content pane
contentPane = getContentPane();
contentPane.add(panel);
setVisible (true);
}

public static void main(String[] args){
EventObjectWindow n=new EventObjectWindow();
}

public void actionPerformed(ActionEvent event)
{
if(event.getSource()==button1)
{
match ss1=new match();
ss1.init();
ss1.setSize(400,500);
ss1.setVisible(true);
//Graphics g=new Graphics();
//ss1.paint(g);
}
if(event.getSource()==button2)

```

```

{
insert ss2=new insert();
system("appletviewer insert.java");
//ss2.init();
//ss2.setSize(400,500);
//ss2.setVisible(true);

```

```

}
}
}
}

```

```

public class Insert extends JApplet
{

```

```

JTextField tf;

```

```

Image img;

```

```

JButton btn1, btn2, btn3;

```

```

JButton btn4;

```

```

public Container getContentPane()
{

```

```

return panel;
}

```

```

public void init()
{

```

```

//

```

```

tf=new JTextField(10);

```

```

// set attributes

```

```

panel=new JPanel();

```

```

panel.add(tf);

```

```

btn1=new JButton("Insert");

```

```

btn1.addActionListener(this);

```

```

panel.add(btn1);

```

```

// btn2=new JButton("Delete");

```

```

// btn2.addActionListener(this);

```

```

// btn3=new JButton("Exit");

```

```

// btn3.addActionListener(this);

```


RELOAD THE IMAGE

```
import javax.swing.*;
import javax.swing.JTextField;
import java.awt.event.*;
import java.awt.*;
import java.io.*;

public class insert1 extends JFrame implements ActionListener
{
    JTextField t1;
    Image img;
    JButton getPictureButton;
    JButton button1;
    public Container contentPane;
    public JPanel panel;
    public insert1()
    {
        t1=new JTextField(10 );
        t1.setEditable(true);
        panel = new JPanel();
        panel.add(t1);
        button1 = new JButton("submit");
        button1.addActionListener(this);
        panel.add(button1);
        contentPane = getContentPane();
        contentPane.add(panel);
        setVisible (true);
    }
}
```

```

    }

    public void actionPerformed(ActionEvent e)
    {
        if(e.getSource() == button1)
        {
            insert2 in =new insert2();
            System.out.println(t1.getText());
            in.readFile(t1.getText());
        }
    }
}

```

Linked file insert2()

```

import java.io.*;
import javax.swing.*;
import javax.swing.JTextField;
import java.awt.event.*;
import java.awt.*;
import java.io.*;

class insert2 extends JFrame implements ActionListener{
    JTextField t1;
    Image img;
    JButton button1;
    JButton getPictureButton;
    public Container contentPane;
    public JPanel panel;
    public insert2()
    {

```

```

    }
    public void readFile(String s)
    {
        try{
            InputStream f1=new FileInputStream(s);
            OutputStream f2=new FileOutputStream("aa1.jpg");
            t1=new JTextField("writing",10 );
            t1.setEditable(true);
            panel = new JPanel();
            panel.add(t1);
            button1 = new JButton("submit");
            button1.addActionListener(this);
            panel.add(button1);
            contentPane = getContentPane();
            contentPane.add(panel);
            setVisible (true);
            byte b[]=new byte[1];
            while( (f1.read(b) ) == 1 )
            {
                f2.write(b);
                System.out.println(b+"\n");
            }
            f2.close();
            t1.setText("completed");
        }
        catch(Exception e)
        {
            System.out.println("Exception due to " +e);
        }
    }
    public static void main(String args[])

```

```

{
insert2 i=new insert2();
i.readFile("aa.jpg");
}

public void actionPerformed(ActionEvent e)
{
if(e.getSource() ==button1)
{
// insert3 in =new insert3();
// in.init();
}
}
}

```

BIBLIOGRAPHY

1. Nalin k. Ratha's, Shaoyun Chen and Anil.K.Jain's Real-time Matching System for Large Fingerprint Databases. [2003]
2. Shegliny Yang and Ingrid M.Verbauwhede. A Secure Fingerprint Matching Technique [2003].
3. X.Tong, J.huang, X.tang and D.shi Fingerprint minutiae matching using adjacent feature vector [2005].
4. Ning Liu, Yilong yin, Hongwei Zhang A Fingerprint Matching Algorithm Based on Delaunay Triangulation Net [2002].
- 5.Xuwen Qin liLi, Shufang Tian A New Image Matching Algorithm With Greedy Algorithm For Remote Sensing Imagery.
- 6.Sarat C.Dass , Anil K.Jain Fingerprint- Based Recognition .[10-26-06].
7. Andres Almansa ,Laurnet Cohen 's- Fingerprint Image Matching By Minimization Of A Thin Plate Energy Using A two-step Algorithm With Auxiliary Variables.
- 8.Yong fang Zhu, Sarat C.Dass and Anil K.Jain –statistical model for assessing the individuality of the Fingerprints.
- 9.Virginia Espinosa-Dur6 (Polytechnic University of California)-Minutiae Detection Algorithm For Fingerprint Recognition.
- 10.Abdullah Bal , AedM, El-baba and Mohammad.S.alam-Improved Fingerprint Identification with supervised filtering Enhancement.[2005].
- 11.Corealtion based matching algorithm –presentations.
- 12.Black book of JAVA programming .
- 13.Java- how to program by H.M.Deitel, P.J.Deitel.
- 14.The complete reference JAVA2 by Herbert schildt.
- 15.Fundamentals of Database System by Elmarsi,Navathe,Somayajulu,Gupta
16. www.wikipedia.com.
- 17.www.soruceforge.net
- 18.www.biometrics.cse.msu.edu/fingerprint.html
- 19 ece.uprm.edu/crc/crc2004/papers/AlfredoLopez.pdf