2007

Indexing techniques for fingerprint and iris databases

Rajiv Mukherjee
West Virginia University

Indexing Techniques for Fingerprint and Iris Databases

by

Rajiv Mukherjee

Thesis submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Electrical Engineering

Arun Ross, Ph.D., Chair
Xin Li, Ph.D.
Donald Adjeroh, Ph.D.

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2007

Keywords: Biometrics, Fingerprint, Iris, Indexing, Minutiae Triplets, Ridge Curve, Texture Analysis, Classification, Filtering, Clustering, Multimodal

Abstract

Indexing Techniques for Fingerprint and Iris Databases

by

Rajiv Mukherjee
Master of Science in Electrical Engineering

West Virginia University

Arun Ross, Ph.D., Chair

This thesis addresses the problem of biometric indexing in the context of fingerprint and iris databases. In large scale authentication system, the goal is to determine the identity of a subject from a large set of identities. Indexing is a technique to reduce the number of candidate identities to be considered by the identification algorithm. The fingerprint indexing technique (for closed set identification) proposed in this thesis is based on a combination of minutiae and ridge features. Experiments conducted on the FVC2002 and FVC2004 databases indicate that the inclusion of ridge features aids in enhancing indexing performance. The thesis also proposes three techniques for iris indexing (for closed set identification). The first technique is based on iriscodes. The second technique utilizes local binary patterns in the iris texture. The third technique analyzes the iris texture based on a pixel-level difference histogram. The ability to perform indexing at the texture level avoids the computational complexity involved in encoding and is, therefore, more attractive for iris indexing. Experiments on the CASIA 3.0 database suggest the potential of these schemes to index large-scale iris databases.

I dedicate my thesis to my family

Acknowledgments

My stay at West Virginia University has been one of the most gratifying experiences of my life, which I shall always cherish. My teachers, friends and colleagues have been a part of my graduate education and I would like to take this opportunity to express my gratitude towards them. First, I would like to thank my advisor - Dr. Ross, for his constant support during my Master's program. It has been an honor to have him as my research advisor. He has been instrumental in the completion of this thesis, through his stimulating suggestions and encouragement.

I would also like to thank Dr. Li and Dr. Adjeroh, my committee members, for their suggestions and guidance.

I would take this opportunity to thank my friends and fellow researchers for making my stay, at the Biometrics Systems Laboratory and at WVU, memorable. I would like to thank Chris, Nicole, Rohan, Phani, Samir, Jidnya, Simona, Nikhil, Shigefumi, Sushil, Matt, Mayank and Richa for always being there. I would also thank Baneshwar and Nitin for their constant help and support.

I will forever be indebted to my family for their unconditional love and support. I would take this opportunity to thank my mother, who will forever be with us in spirit, and my father for his constant guidance and motivation. Without them I would not have made it this far.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Biometrics [1] is playing a major role in automated personal identification systems deployed to enhance security all over the world [2]. The word biometrics is derived from the ancient Greek words ‘bios’ and ‘metron’ meaning life and measure, respectively. Thus, biometric recognition or biometrics refers to automatic recognition of individuals based on their behavioral and/or physical characteristics. Examples of physical characteristics include iris, face, fingerprints, retina, hand geometry, vascular patterns (in the palm , finger, eye etc.), while behaviorial characteristics include signature, gait, keystroke dynamics (typing). Voice is a combination of both physical and behavioral characteristics.

The biometric market segmentation by technology in January 2007, as analyzed by the International Biometric Group (IBG [2]) is shown in Figure 1.1. The report estimated the revenue of Automatic Fingerprint Identification System (AFIS) market to be highest amongst biometric technologies. It should be noted that AFIS is used in forensic applications. If non-AFIS biometric revenue is considered, fingerprint-based biometric systems continue to lead in terms of market share (38.1%), followed by face (19.4%) and iris (8.7%) based biometric systems.

The enhanced security provided by a biometric system can be attributed to the fact that the authentication system is based on pattern recognition of features unique to each individual, thus obviating the need to remember passwords or carry documents/identification (ID) cards which can easily be lost, stolen or forged. The change of focus from ‘what I remember or possess’ to ‘who I am’ has made a huge impact on improving security in applications such as

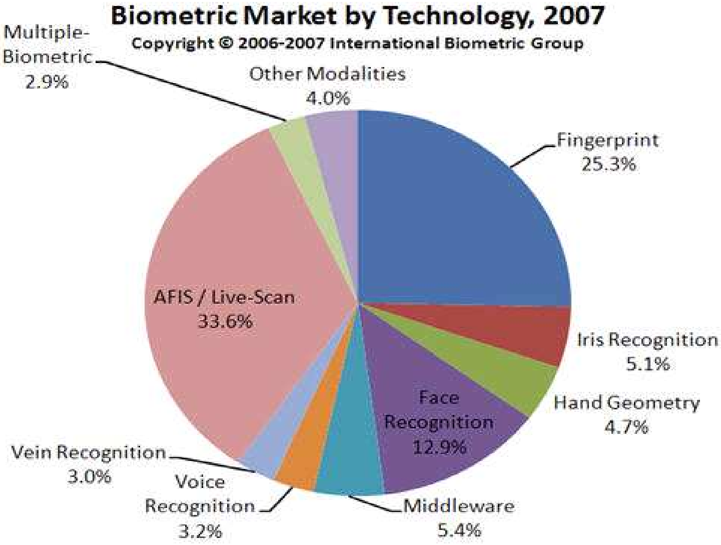


Figure 1.1: Biometrics market segmentation by technology. Source: IBG.

airport surveillance [3], corporate time keeping, home security and other personal and group ID management systems. For example, facial recognition could be used in airports as a surveillance tool to identify persons of interest (whose biometric information is available in a watch list) without the subject's cooperation or knowledge. Biometrics like fingerprint and iris can be used to perform access control for employees in the corporate sector and the resulting attendance information can be further used to perform payroll processing. Fingerprint based door locks are being used to enhance home security[1]. Hospitals are using biometric technology to strengthen system security in order to be compliant with data privacy and computer security requirements of the Health Insurance Portability and Accountability Act (HIPAA [4]).

A biometric system [5, 1] is a pattern recognition system that operates by acquiring biometric data from an individual, extracting a feature set from the data acquired, and comparing this feature set against the template set stored in the database. A biometric system consists of four main stages [1].

1. Sensor: It is an acquisition device that captures the biometric data of an individual. For example, an iris sensor images an individual's iris texture.

2. Feature extraction: It is responsible for processing the biometric data to extract a set of

[1] http://www.findbiometrics.com/press-release/3817

discriminatory features. For example, in a hand based biometric system the geometric properties of the hand image are extracted.

3. Matcher: It compares the features extracted against stored templates in the database and generates a matching score. For example, in an iris recognition system, the IrisCodes (features extracted) are compared with IrisCode templates in the database. Often the matching stage includes a decision making stage based on the matching score. For example, a subject's claimed identity is confirmed or denied (verification) or a subject's identity is established (identification).

4. Database: It stores the biometric templates of the enrolled users. The enrollment process comprises of capturing the biometric data in digital form, checking the quality of the digital representation and if the quality meets the requirement then the features extracted are stored in the database as templates (compact representation of features extracted).

A biometric system could be either a verification system or an identification system depending on the application. A verification system compares the acquired trait with the template of the claimed identity pre-stored in the system. The verification system will either accept or reject the claimed identity. A verification system performs one-to-one matching. In contrast, an identification system identifies an individual by searching potentially, the entire template database for a match. This kind of a system performs a one-to-many matching. The identification system can either establish the person's identity with some level of accuracy or fail if the individual does not exist in the enrolled database. The enrollment, verification and identification stages of a biometric system are shown in the Figure 1.2.

In a large scale identification system, the goal is to reduce the number of candidates to be considered by the identification algorithm. The contribution of this thesis is in the design of such algorithms for fingerprint and iris. The following section describes fingerprint and iris based biometric systems in more detail.

1.2 Fingerprint as a biometric

Fingerprints have been widely used in forensic as well as commercial applications for identification as well as verification. For instance, US-VISIT (United States Visitor and Immigrant

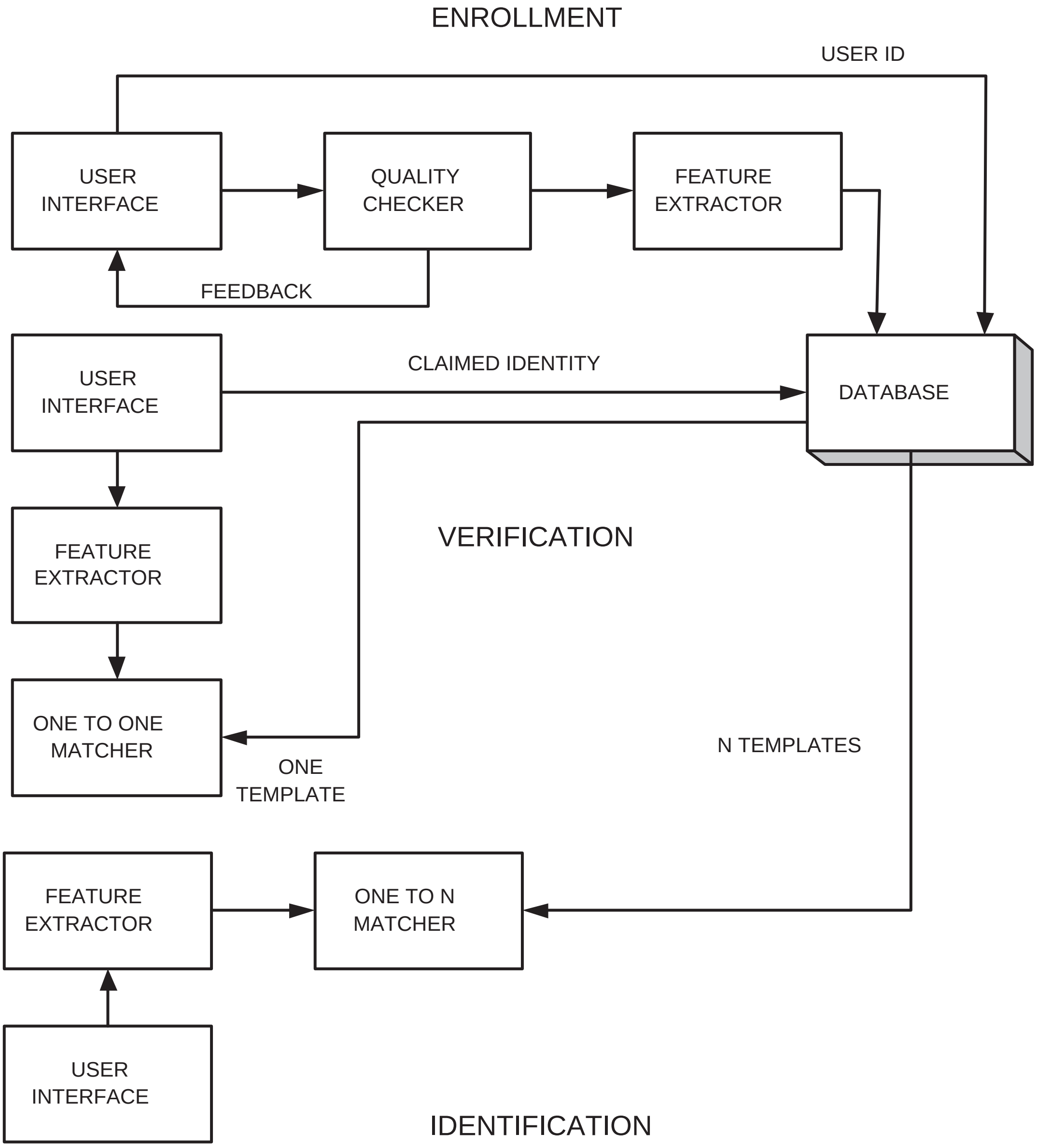


Figure 1.2: Enrollment, verification and identification stages of a biometric system.

Status Indicator Technology), a program initiated by the DHS (Department of Homeland Security), requires certain non-US citizens to have their two index fingers digitally scanned at the US port of entry. This helps to instantly check the background of the person seeking entry. Another application can be seen in the IAFIS (Integrated Automated Fingerprint Identification System). The IAFIS is a national fingerprint and criminal history system maintained by the Federal Bureau of Investigation (FBI), Criminal Justice Information Services (CJIS) Division. The IAFIS uses multiple fingerprint of a person (10 prints) as an additional information in order to allow for large scale identification invlolving millions of fingerprints [6].

There are several advantages and disadvantages associated with using fingerprint as a biometric.

The advantages can be listed as:

1. Fingerprints are considered to be unique to an individual. This uniqueness allows it to be used as a biometric trait.

2. Most people are familiar with the use of fingerprint for identification and access control. Therefore, it is accepted as a technology.

3. Unlike other biometric scanners (e.g., iris and retina scanners), fingerprints scanners are easy to use and relatively non-intrusive.

4. Most fingerprint scanners today are inexpensive and easy to install. Fingerprint scanners can be very small (e.g., swipe sensors incorporated in laptops, cell phones etc.).

The disadvantages include the following:

1. Fingerprint often have a negative connotation attached to them because they have been traditionally been used for criminal investigations.

2. The quality of fingerprint may suffer due to weather conditions, thus degrading recognition accuracy.

3. Fingerprint spoofing [7] is a major problem in unsupervised, fingerprint based biometric systems. ‘Liveness detection’ [8] is used to differentiate a true fingerprint from a fake one.

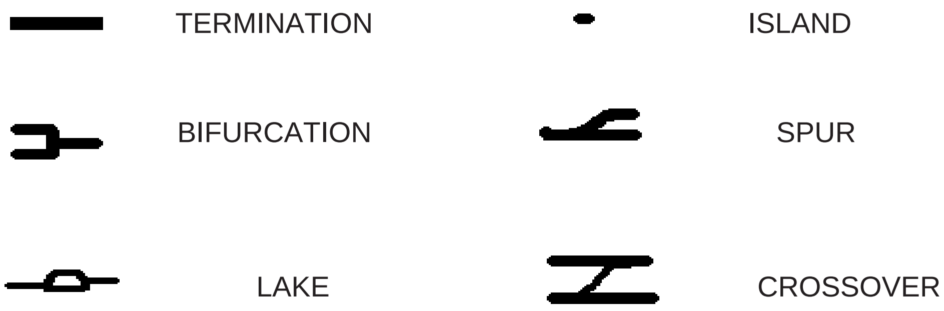


Figure 1.3: Some common minutiae types.

1.3 Fingerprint Representation

Fingerprints are oriented texture patterns present on the surface of the finger consisting of interweaved ridges and valleys. At about seven months of prenatal development, fingerprints are fully formed [5]. The finger ridge configuration of the individual do not naturally change. However cuts and bruises affect the ridge pattern. In context of digital images of fingerprints, the dark areas called ridges and the bright areas called valleys are the most important characteristics of the fingerprint structure. The ridge lines have a high curvature in certain regions when the fingerprint image is analyzed at a global level, lending the ridge lines a distinct shape. These regions are called singularities [5]. Such singular regions can be classified into 'loop', 'delta' and 'whorl'. Most fingerprint matching techniques align two fingerprints on the basis of a registration point called the 'core', which corresponds to the center point of the north most 'loop' type singularity. For fingerprints that do not contain 'loop' singularities, defining the core becomes difficult. In such cases, the 'core' is associated with point of maximum ridge line curvature. Unfortunately, due to image acquisition issues and large intra class variability of fingerprints, it is difficult to define the 'core' reliably.

When the fingerprint is analyzed at the local level, minutiae (small details) can be found in the fingerprint pattern. In 1892, Sir Francis Galton introduced the minutiae features for fingerprint matching. Minutia describes the discontinuity in the ridges (e.g., termination, bifurcation, crossover, spurs etc.). Some common minutiae types are shown in Figure 1.3. However, only a few of these minutiae types are used in practice due to practical difficulty in identifying the minutia type reliably. For instance, the FBI minutia model consists of only terminations and bifurcations [5]. In this model, each minutia is denoted by its location in the spatial domain, and

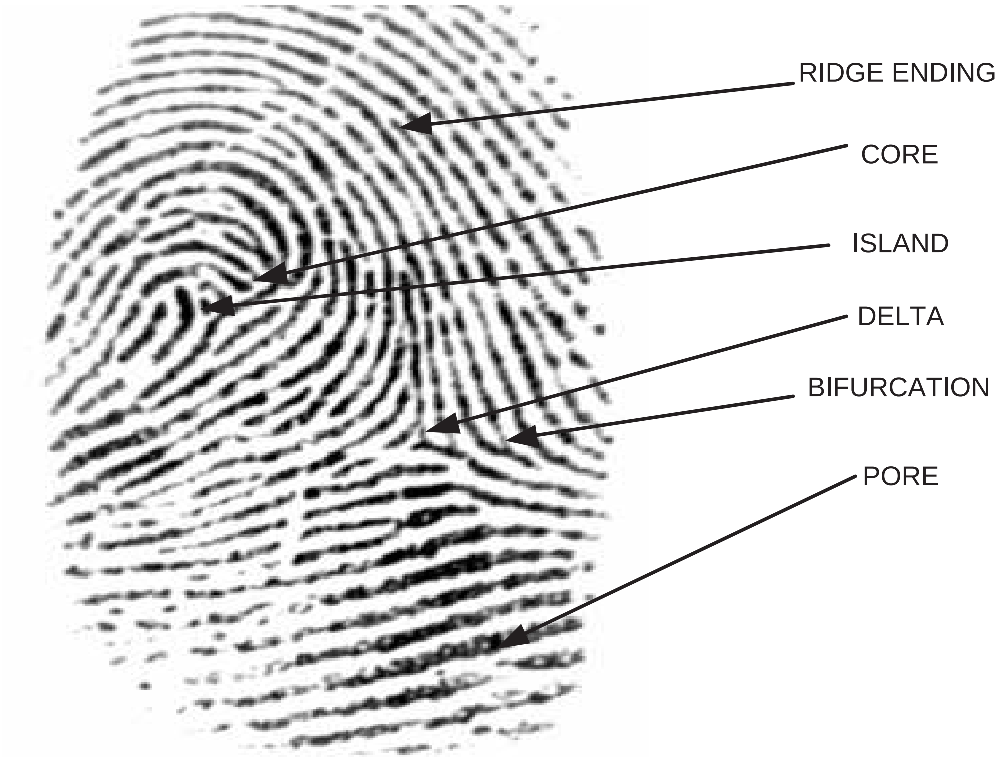


Figure 1.4: Fingerprint characteristics.

the angle between the horizontal axis and the tangent to the ridge line at the minutia location.

If a fingerprint image is acquired at higher resolution, it is possible to capture the 'sweat pores' present on the ridge lines. These pores have highly distinctive features like number, location, shape, etc. but the ability to extract such information is dependent on the availability of high resolution scanners and good quality fingerprint images. Figure 1.4 shows some important characteristics of fingerprints.

On the basis of the extracted features from fingerprints, fingerprint matching can be categorized into three types namely correlation based matching, ridge feature based matching and minutiae based matching [5].

1. **Correlation based matching**: The fingerprint images are superimposed on each other and the correlation between the corresponding pixel intensities is computed for different alignments.

2. **Minutiae based matching**: This is the most popular technique whereby minutiae points are extracted from the two fingerprints to be matched and their location and ridge orienta-

tions are stored. The matching process comprises of determining the alignment between the template and input minutiae set that results in the maximum number of minutiae pairings. For low quality fingerprint images the minutiae extraction process can be difficult.

3. **Ridge feature based matching**: Since pixel intensities and minutiae locations are features of the ridge pattern, they can be considered to be sub-categories of the ridge features. In ridge feature based matching, the texture information, local orientation, frequency and ridge pattern are used to match two fingerprints.

1.4 Iris as a biometric

The iris is considered to be a reliable biometric for human identification for several reasons. These can be listed as follows [9]:

1. The iris is an internal organ of the eye. It is protected from external wear and tear by the cornea, which is a highly transparent membrane. This makes it more reliable than fingerprints which is more susceptible to change after an extended period of manual labor.
2. Like fingerprints, iris texture generation is a chaotic process [10] that takes place during the embryonic gestation period. The human iris structure begins to form in the third month of gestation and is formed fully by the eighth month. However the color of the eye begins to form during the first year of birth. The advantage of the chaotic structure of the iris is that even genetically identical monozygotic twins have unique iris textures, though their 'DNA fingerprint' [11] is the same.
3. The iris texture is believed to have long term stability, though some medical conditions or procedures can affect the shape and color of the iris.
4. The non-contact acquisition system used for iris recognition makes it more acceptable than fingerprints which mostly use touch based sensors.
5. The geometric configuration of the iris is controlled by the sphincter and the dilator muscles. Therefore the iris shape is predictable.

However, there are certain issues which hinder the use of iris as a biometric. These include the following:

1. The iris acquisition process requires the subject's cooperation, and it is difficult to capture useful iris data if the subject is not keeping his head still and looking into the camera.

2. Since iris recognition is based on digital image acquisition, it is susceptible to failure-to-enroll (FTE) due to poor image quality.

3. Another interesting issue related to iris in particular is the 'liveness detection' [12, 8, 13] or the 'live tissue verification' [14]. This is a concern not so much in supervised iris acquisition but mainly in unsupervised iris acquisition.

1.5 Structure of the iris

Iris [15] is the most visible part of the human eye. The iris is characterized by pigmented fibrovascular tissue known as stroma. The stroma consists of sphincter and dilator muscles, responsible for the contraction and dilation of the pupil that controls the amount of light entering the pupil. The outer boundary of the iris, attached to the sclera and the anterior ciliary body, is known as the root. The iris and the ciliary body are together called the anterior uvea. The iris can be divided into two regions, namely the pupillary zone and the ciliary zone. The pupillary zone is the inner region that forms the boundary of the pupil. The rest of the iris forms the ciliary zone. The region separating the pupillary zone from the ciliary zone is called the collarette and it is region where the sphincter and dilator muscles overlap. The anatomy of the iris is shown in Figure 1.5. The anterior layer of the iris has interesting features like pupillary ruff, contraction furrows and crypts [15]. The posterior layer features include the circular contraction fold, radial contraction fold and structural fold.

Anterior Layer Features

1. **Pupillary ruff**: It is a series of ridges in the pupillary region formed due to the continuation of pigmented epithelium from the posterior surface.

2. **Contraction furrows**: They are a series of circular folds formed between the iris origin and the collarette. The formation of these folds is due to the change in iris surface caused by contraction and dilation.

3. **Crypts**: There are two places where crypts are observed in the iris. The Crypts of Fuchs are crater like openings found on either side of the collarette, that allows iris tissues to be

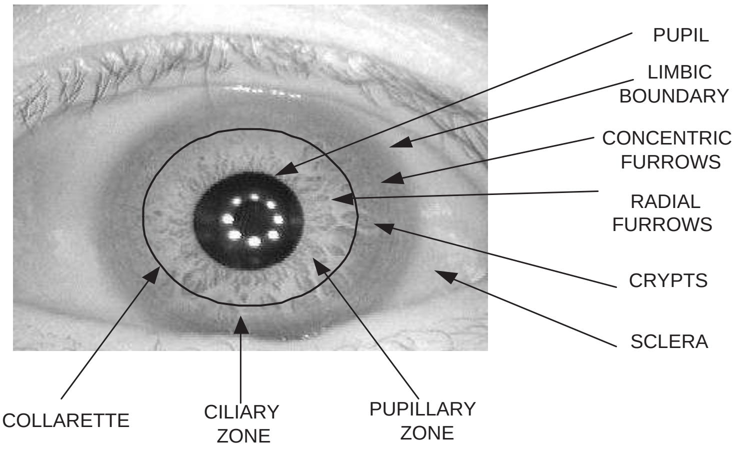


Figure 1.5: Anatomy of the iris. The white blobs on the pupil are specular reflections due to imaging device.

bathed in aqueous humour (thick watery substance present in the eye). Crypts are also observed at the base of the iris, close to the outer boundary of the ciliary region.

Posterior Layer Features

1. **Circular contraction fold**: They are circular ridge patterns over the posterior surface.
2. **Radial Contraction folds of Schwalbe**: They are radial folds observed from the pupillary boundary to the collarette.
3. **Structural folds of Schwalbe**: They are broader and more widely spaced radial folds.

1.6 Iris Recognition System

The goal of an iris recognition system is to extract, represent and compare textural information present on the iris surface [16]. The main modules of such a system are segmentation, enhancement, feature extraction and matching. During enrollment the features are extracted and stored in the database as templates. The authentication phase encompasses image preprocessing followed by feature extraction for a given iris image. This feature set is then compared with the templates in the database in order to perform identification or verification of an individual's identity. Depending on the technique used to represent iris texture information, iris recognition algorithms [16] can be categorized into a) appearance based [17, 18], b) filter

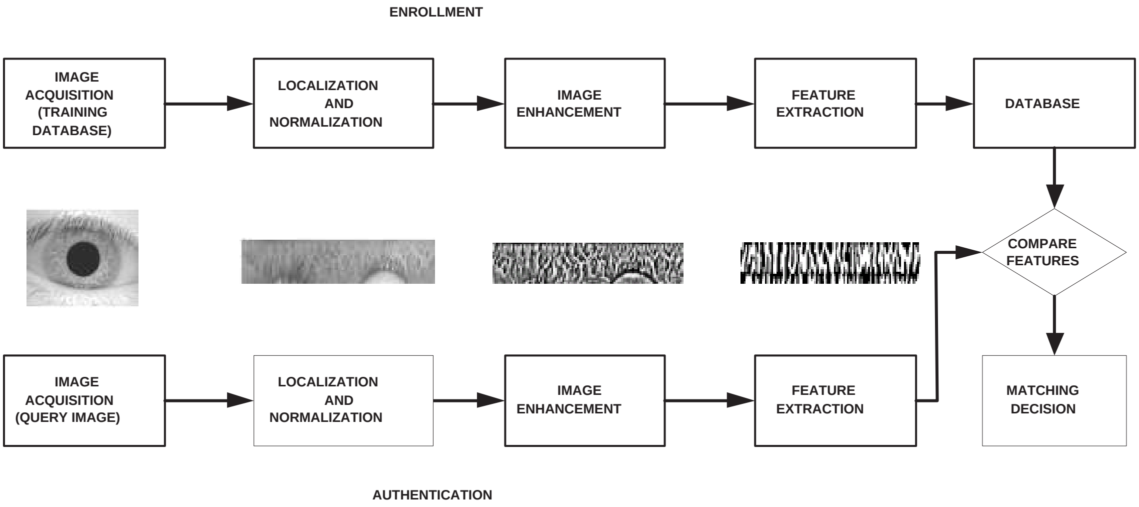


Figure 1.6: A typical iris recognition system

based [9, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 17] and c) feature based techniques [35, 36, 37]. The first iris recognition algorithm proposed by Daugman used a texture based technique to encode iris into a 256 byte IrisCode using a multi scale 2D Gabor-Wavelet transform.

1.7 Challenges faced by biometric systems

Biometric systems face certain limitations which must be considered before any large scale deployment of such systems.

Some of the challenges are listed below:

1. **Noisy data**: A poorly illuminated face image, fingerprint image with scar are examples of noisy data. Noisy data could be the result of defective or ill maintained sensors (e.g., dirt accumulation on a fingerprint sensor) or undesirable ambient conditions. Nosy biometric data may lead to a genuine user being incorrectly rejected [6].

2. **Intra-class variations**: Intra-class variations is the changes in the biometric trait over time (e.g., facial changes over the years), variations in psychological conditions of the individual (e.g., changes in behavioral characteristics like voice, hand writing etc.) due to stress etc. In order to combat the degrading effect of intra class variation, multiple

templates can be stored for each individual and these templates can be updated from time to time [6].

3. **Inter-class similarities**: The overlap of feature space corresponding to multiple classes[2] contribute towards inter-class similarities. In a large scale identification system, the inter-class similarities will lead to increase in false match rate of the system [6].

4. **Lack of universality**: Sometimes it is difficult to capture meaningful biometric data for certain group of users. For example, a fingerprint based biometric system may extract spurious minutiae points from a fingerprint due to its poor quality of ridges. The lack of universality of the biometric trait leads to increase in Faliure To Enroll (FTE) rate. This problem can be solved by using multibiometric systems [6].

5. **Interoperability issues**: The underlying assumption of most biometric systems is that the biometric data to be compared comes from the same sensor. Therefore, such comparisons cannot be done reliably when using different sensors due to changes in image resolution, sensor technology and sensing area etc. By addressing the interoperability [38] issue in feature extraction and matching stages of the biometric system, this problem can be handled better.

6. **Spoofing**: Spoofing [7, 12]a biometric system includes manipulating one's biometric trait in order to avoid authentication and creating physical replicas of original biometric traits in order to falsely take the identity of another person. In order to combat the problem of spoofing for physical traits such as fingerprints and iris, 'liveness detection' [8, 13] schemes have been proposed.

1.8 Problem Statement

In large scale authentication systems like ABIS (Automatic Biometric Identification System) and BATS (Biometric Automated Toolset System), the goal is to determine the identity of a subject from a large set of users already enrolled in the biometric database. The user accessing the authentication system in order to be identified is known as the query user. There are three ways to approach the identification problem. The first is the naive method of performing

[2] The class could be an individual.

one to one matching of the biometric trait of the query user with all the users enrolled in the system. This method is not feasible for large databases due to the enormous time complexity associated with it. The second method is traditional classification (see Figure 2.1) where the database is divided into a small number of classes (not necessarily mutually exclusive). The biometric attribute of the query user is assigned to one (or more) of these classes and matching is performed with all the users in the assigned class until a matching criteria is satisfied. The third technique is to considerably reduce the number of candidate hypotheses for matching. The indexing scheme (see Figure 1.8) assigns an index value to the biometric trait of the query user and matching is performed with respect to users in the database with comparable index values. Fingerprints and iris are being widely used in large scale authentication systems, making them attractive for indexing research. The contribution of this thesis is in designing novel indexing techniques for fingerprint and iris.

It should be noted that indexing can be done using soft biometric traits (like gender, ethnicity, age, height, eye color etc.) [6, 39]. Such soft biometric traits can be extracted from the user at the time of enrollment and later used in order to reduce the number of candidates for matching. However, soft biometrics lacks the distinction and permanence to identify an individual reliably and uniquely.

1.9 Organization

In Chapter 2, the fingerprint indexing model is discussed and the indexing performance on the FVC2002 and FVC2004 databases is reported. In Chapter 3, three iris indexing techniques are proposed and their indexing performances on CASIA 3.0 are discussed. Finally in Chapter 4, the contributions of this thesis are summarized and future work in the field of biometric indexing is presented.

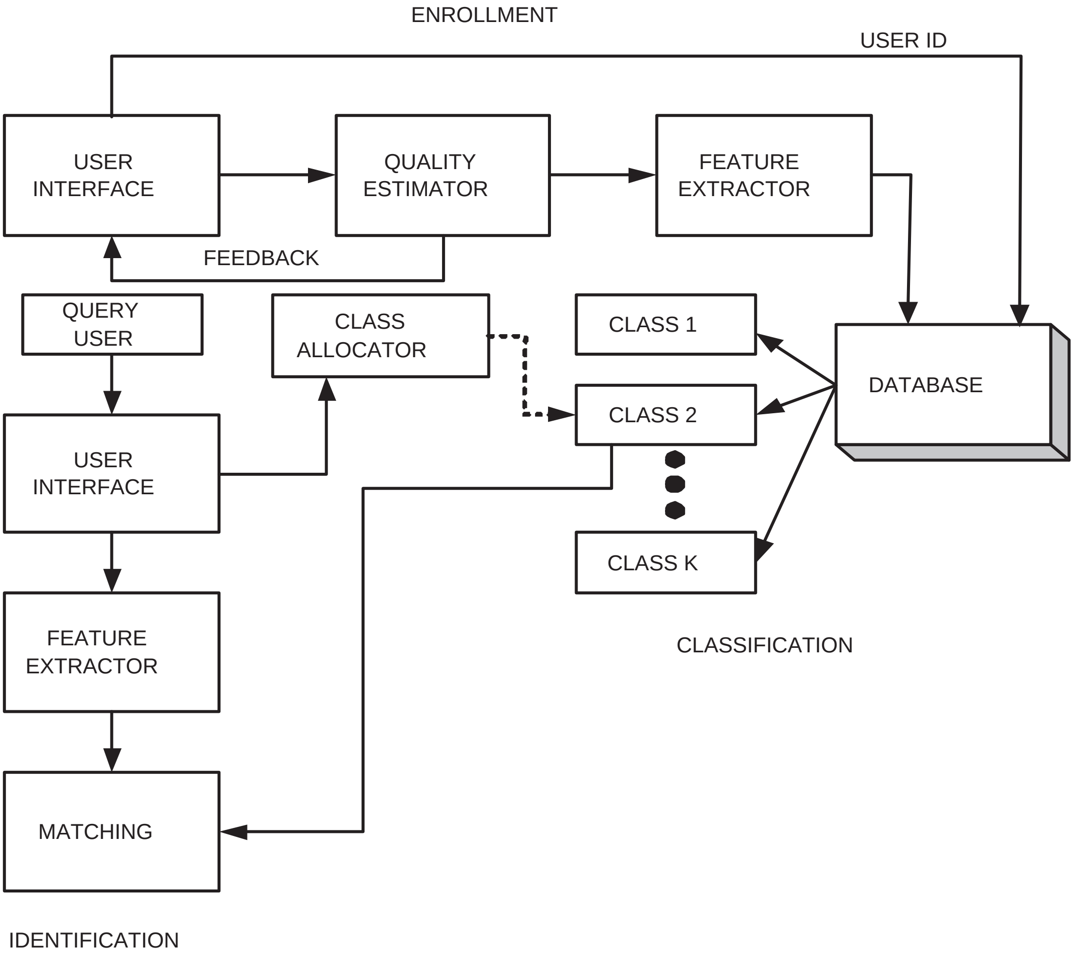


Figure 1.7: Identification using classification technique

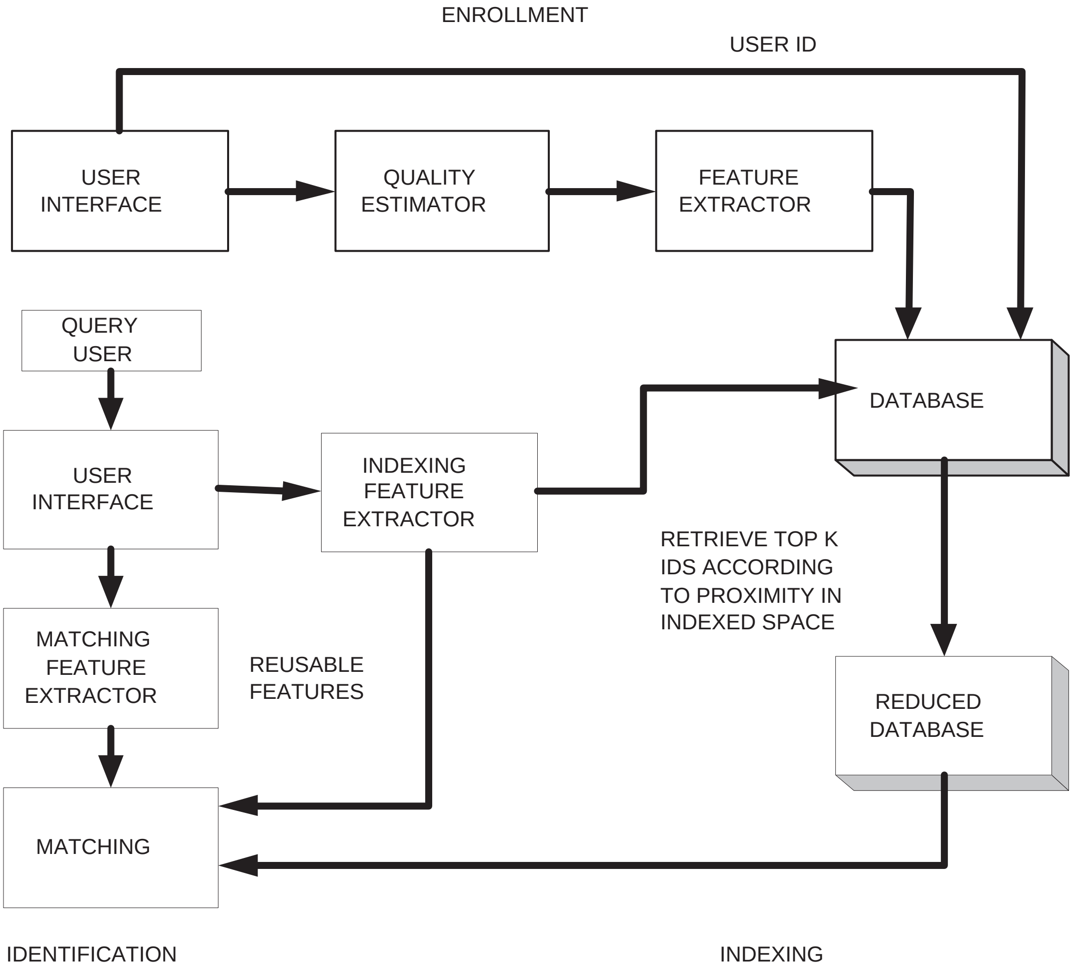


Figure 1.8: Identification using indexing technique

Chapter 2

Fingerprint Indexing

2.1 Introduction

Fingerprint-based identification systems search through a large database of fingerprint entries for possible matches based on the given query print. Each entry has an associated identity and a matching function is used in order to determine the similarity between two fingerprints. However, due to the large number of entries in the database, one to one matching of the query print with each fingerprint in the database would be computationally infeasible. Therefore, a filtering process is usually invoked in order to reduce the number of candidate hypotheses for matching operation. Filtering can be achieved by two different approaches: classification and indexing. Classification involves partitioning the database into multiple classes[1] and comparing the query print with prints belonging to the class assigned to the query print. In contrast, indexing [40] assigns an index value[2] to each fingerprint and, therefore the query print is compared with those prints in the database which have comparable index values.

Fingerprint classification schemes based on human defined categories (such as Left Loop, Right Loop, Arch, Tented Arch and Whorl. See Figure 2.1) have an inherent problem due to the small number of classes (e.g., 5 - 8) and the uneven distribution of fingerprints across these classes. Furthermore, most classification schemes [41, 42, 43, 44, 45, 46, 47] are based on the configuration of singular points (i.e., core and delta points) which may not be available in prints acquired using small-sized sensors. In this work, we focus on indexing techniques for fingerprint filtering. The proposed approach extends the indexing framework based on minutiae triplets proposed by

[1] The classes are not necessarily mutually exclusive.

[2] The index can be a vector entity.

Bhanu et al. [48], Germain et al. [49], and Bebis et al [50]. The primary contributions of this work are: (a) the inclusion of ridge curves associated with minutiae triplets in devising the indexing mechanism; and (b) demonstrating the efficacy of the indexing process across multiple sensors with comparable resolution.

2.2 Classification vs Indexing

Fingerprint classification is the problem of assigning a fingerprint to a class, generally based on global features (e.g., ridge structure and singularities), in an efficient and reliable way. However fingerprint matching is performed according to local features (e.g., minutiae). Thus, the fingerprint matcher is required to compare the query fingerprint (to be identified) to the fingerprints in its class.

2.2.1 History of Fingerprint classification

In 1823, Purkinje proposed the first fingerprint classification rule. In his classification model, fingerprints were classified according to their global ridge configuration. Nine classes were proposed, namely, transverse curve, central longitudinal stria, oblique stripe, oblique loop, almond whorl, ellipse, circle, and double whorl [5].

In 1892, Francis Galton made the first scientific study and proposed a classification model with three classes, namely, loop, arch and whorl. These classes were further divided into sub classes. Around the same time, Juan Vucetich proposed a different classification system [5].

In 1899, Edward Henry established the 'Henry system' to classify fingerprints. Fingerprint classification can be considered to be a coarse level matching of fingerprints. Though the classification technique cannot identify fingerprints uniquely, it can be used to determine non-matching fingerprints with considerable accuracy. The fingerprint classes used by most researchers are Left Loop, Right Loop, Arch, Tented Arch and Whorl. Sometimes a sixth class called Twin Loop is also used. However, because of its similarity with Whorl, it is often merged with the Whorl class.

2.2.2 Fingerprint Classification Techniques

Several different approaches have been adopted by researchers to classify fingerprints. These approaches can be divided into syntactic [41,42], structure based [43], statistical [44], singularities based and neural network based methods [45].

1. **Syntactic classifier**: Traditionally, syntactic classifier was used for fingerprint classification. In this approach, the patterns in the fingerprint image are extracted and represented as symbols. These symbols are parsed and the pattern is allotted to a class according to a predefined grammar. Due to large variations in fingerprint patterns, a complicated grammar is required.

2. **Structure based classifier**: In this approach, the directional image is first segmented into homogeneous regions using a clustering algorithm which maintains regularity in region topology by minimizing the regional variance of element direction. Next, a relational graph is created, which is invariant to translation and rotation. Thereafter, a graph matching technique is used to find out the proximity of a graph derived from the input fingerprint image, to the graphs representing the distinct classes. In geometry based classifiers, classification is based on turn detection on the traced curves generated by splines (B spline, T spline, etc.) that are used to model fingerprint ridge lines.

3. **Statistical classifier**: The basic idea behind this approach is to derive a fixed length feature vector from each fingerprint and use a statistical classifier (e.g., K-nearest neighbor) to classify it. However, training a classifier is difficult both in terms of memory and computation time when the dimensionality of the vector is high. Therefore, dimensional reduction techniques are used to reduce the dimension of the feature vector. KLT (Karhunen Loeve Transform) is a famous dimensionality reduction technique, but it can also be used for classification. The KLT guarantees a good preservation of Euclidean distance between vectors [5]. Other variants of this technique like MKLT (Multi-space KLT) have also been used in the literature.

4. **Singularities based classifier**: The location and number of singularities are used to perform classification. Sometimes additional features such as topology of the ridge curves and local orientations are used to improve performance. However, due to noise and other issues

such as partial fingerprints (where singularity points might be missing), the singularity point detection can be extremely difficult [5].

5. **Neural network based classifier**: Several multi layer perceptrons are trained to recognize fingerprints belonging to different classes. Some researchers have also proposed a multi layer self organizing map for classification [51].

Since different classifiers can misclassify different patterns, there is sufficient motivation for multiple classifier based approaches. In this approach, the complementary information offered by different classifiers can be exploited. The selection of the component classifiers can be done in a variety of ways (e.g., distinct classifiers, distinct training data, distinct features). The combination strategy could be rule-based, majority voting rule, K nearest neighbor or neural network.

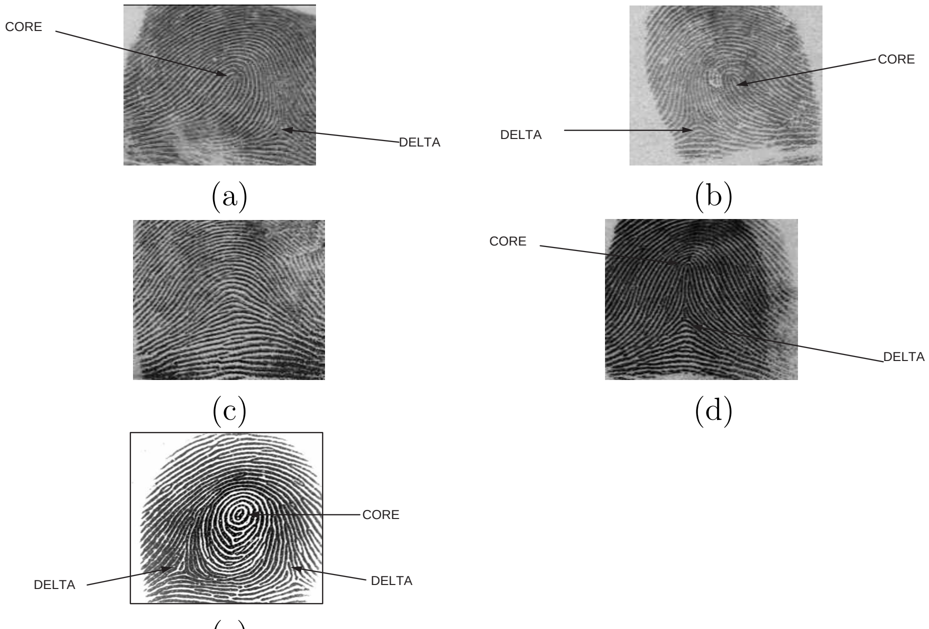


Figure 2.1: Fingerprint classification based on singularity points. (a) Left Loop. (b) Right Loop. (c) Arch. (d) Tented Arch. (e) Whorl.

2.2.3 Fingerprint Indexing Techniques

The problem with fingerprint classification is the small number of classes and the uneven distribution of fingerprints across these classes. The left loop, right loop and whorl classes contain more than 90% of the fingerprints. This does not pose much of a problem in a ten print identification system because a distinct code can be created from the knowledge of the classes of the individual fingerprints, thus reducing the number of candidate hypotheses. However, when the goal is to search a single fingerprint (and not a set of ten prints) in a large database, the classification scheme is not successful in significantly reducing the search space. The solutions suggested to address this are sub-classification and continuous classification [5].

1. **Sub-Classification**: This approach is adopted by fingerprint experts to perform manual fingerprint matching in forensic applications. However, the rules for such sub-classifications are quite complicated and are dependent on the finger (thumb, index, middle, etc.) [5]. Implementing an automated fingerprint sub-classification system is much more difficult than classifying the fingerprints into the traditional classes (left loop, right loop, whorl, twin loop, arch, tented arch). Therefore this is not practical to be used in automatic fingerprint classification systems.

2. **Continuous Classification**: This approach does not partition fingerprints into disjoint classes, but rather represents them as feature vectors, such that, similar fingerprints are mapped to close points in a high-dimensional space. Retrieval is performed by matching the query fingerprint with all fingerprints in the database whose feature vectors lie in it's vicinity.

2.2.4 Retrieval strategies

A retrieval strategy is an important consideration for an indexing or a classification scheme. For a given indexing technique, the following retrieval / search strategies [5] can be used.

1. **Search target class only**: Target class is defined as the class to which the query fingerprint has been assigned. Only fingerprints belonging to the target class are searched. The search is stopped when a matching fingerprint is found or the entire target class is visited.

2. **Search according to predefined search order**: The order in which the classes will be visited is predefined. Therefore, if the matching fingerprint is not found in the target class, other classes are searched. Potentially, the entire database may be searched using such a retrieval strategy.

3. **Search according to variable search order**: If a match is not found in the target class, then other classes are searched according to the class likelihood generated for the query fingerprint by the classifier. It should be noted that the class likelihood may be different for different fingerprints, resulting in a variable search order.

4. **Fixed radius search**: This search strategy is relevant to the continuous classification scheme for indexing. Only those fingerprints whose corresponding vectors are within a predefined radius from the vector corresponding to the query fingerprint are searched. Thus the potential candidates for matching lie within the hyper sphere whose center is defined by the query fingerprint vector. The search may be stopped as soon as a match is found, or the part of the database enclosed by the hyper sphere has been explored [5].

5. **Incremental search**: The search space is expanded in small increments if a matching fingerprint is not located within the radius specified initially.

The goal of the indexing technique is to significantly reduce the number of candidate hypotheses to be considered by the verification algorithm. In the literature there are predominantly three prominent approaches for fingerprint indexing proposed by Germain et al. [49], Bhanu et al. [48] and Bebis et. al. [50]. Germain et al. [49] use minutiae triplet features for indexing using the FLASH technique (Fast Look up Algorithm for String Homology) [52]. Bhanu et al.'s [48] technique also uses minutiae triplets. However, the features that they use are quite different from Germain et al. [49]. Moreover, they use Geometric Hashing [53]. During the feature extraction process (i.e., minutiae triplets) it is important to note that there can be significant distortion between different impressions of the same finger. These distortions include (a) translation, rotation and scaling because of downward pressure of the finger; (b) Shear transformation as the finger might exert a different shear force on the surface during each interaction; (c) Occlusion and clutter because of scar, dryness, sweat and smudge. Bhanu et al. use geometric constraints to limit the size of feature space and reduce the number of false correspondences obtained from querying the lookup table by the index. Bebis et. al. [50] use delaunaization of triangles generated using minutiae points for fingerprint indexing. All the three methods mentioned above use transformation parameter clustering [49]. The indexing techniques proposed by [49, 48, 50] have been summarized in Table 2.1.

There have been some other attempts as well. Ratha et al. [54] describes a multi level indexing approach to reduce the search space. They have implemented the search engine on Splash 2 - a field programmable gate array (FPGA) based processor to obtain near Application Specific Integrated Circuit (ASIC) level speed of matching. Liang et. al. [55] add bifurcation details as an additional feature to improve indexing performance of distorted fingerprints.

Table 2.1: Examples of three feature indexing schemes based on minutiae points.

Author	**Features used**	**Performance**
Germain et al.	Used triangulation of minutiae points. Length of each side. Local Orientations. Ridge count between two vertices.	The False Positive Rate (FPR) on a 10 person database is 9.5%. FPR on a 100 person database is 63%. With 32 disks distributed over an 8 node IBM SP2 system, could search a database of 10 million prints in 70 seconds. Disk parallelism can be used to reduce query time. Proprietary database used.
Bhanu et al.	Used triangulation of minutiae points. Maximum length of 3 sides Median and minimum angles Triangle handedness, type and direction. Ridge Count Minutiae Density	On NIST-4 database the Correct Indexing Performance (CIP) is 85.5% with verification limited to 10% of the database.
Bebis et al.	Used Delaunay triangulation of minutiae points. Ratio of minimum to maximum length Ratio of median side to maximum side. Cosine of the angle between two smallest sides.	In case of 3 imprints per person in the training set, average correct matching rate is 86.56% and the average false negative matching rate is 13.36%. Proprietary database was used.

2.3 Proposed technique

The proposed technique relies on the creation of a 9-dimensional index space model based on minutiae triplets and the associated ridge curves. The K-means clustering scheme is invoked to partition this index space into multiple clusters. Now, each fingerprint is viewed as a *collection* of points distributed in the index space with each point characterizing the 9-dimensional feature. describing a triplet and the associated ridge curves. Each of the points is assigned to one of the pre-defined clusters based on the minimum distance rule. Thus, a cluster in index space will have a listing of all fingerprint identities that have at least one point assigned to that cluster. When a query print is presented to the system, it is first decomposed into triplets and ridge curves, and the 9-dimensional collection of points is generated. Next, these points are mapped to individual clusters in the index space. A set of possible matching identities corresponding to a small number of clusters is then determined. Thus, the query print is compared against this reduced set of fingerprints in order to retrieve the best match.

Algorithm to generate index space model

1. Consider a training database (FVC2004 DB4 was considered for experiments). For each fingerprint in the training database, perform the following.

   a. Extract minutiae information from the fingerprint.

   b. Perform Delaunay triangulation on minutiae points.

   c. Prune any ‘skinny triangles’ thus obtained.

   d. For each triangle perform feature extraction. A nine tuple feature vector comprising of geometric and ridge based features are computed to represent the triangle in the feature space.

2. Perform unsupervised clustering (K-means) on the feature vectors representing triangles. Set the number of clusters desired. For our experiments K was set to 600.

3. The index space model stores the centroid information of K clusters.

The index space model generated can be updated either using a different training database (where the training samples are comprised of the training samples acquired during enrollment) or it can be incrementally updated as and when a query fingerprint is presented to the indexing system.

Algorithm to embed a fingerprint identity in the index space model

1. For a fingerprint

   a. Extract minutiae information from the fingerprint.

   b. Perform Delaunay triangulation on minutiae points.

   c. Prune any ‘skinny triangles’ thus obtained.

d. For each triangle perform feature extraction. A nine tuple feature vector comprising of geometric and ridge based features are computed to represent the triangle in the feature space.

e. Allot each triangle to single (or multiple) clusters depending on the distance of the feature vector representing the triangle in feature space to the cluster centroid.

f. Update the cluster to which the triangle is allotted with the user identity of the fingerprint from which the triangle was generated.

g. Each fingerprint has a multiple clusters associated with it.

2.3.1 Feature Extraction

The features are extracted by examining the structural information contained in the distribution of minutiae points using Delaunay triangulation. This allows for choosing more meaningful minutiae groups during indexing, so that structural information pertaining to a local neighborhood is preserved. The Delaunay triangulation associates a unique topological structure with the fingerprint minutiae. This process is also computationally efficient since it eliminates the need to consider all possible minutiae triplets in a fingerprint image.

Given a fingerprint image with minutiae configuration $M = \{m_1, m_2, \ldots, m_o\}$, $m_i = (x_i, y_i, \theta_i)$, where (x_i, y_i) is the location of minutia m_i and θ_i is its orientation, the process of Delaunay triangulation generates triplets of the form $t = (m_i, m_j, m_k)$, $1 \leq i, j, k \leq o$. The maximum number of triplets generated as a result of this triangulation will be $9o$+1 [56]. Two sets of features are extracted from each triplet. The first set of features correspond to the geometry of the triangle generated by the triplet, while the second set pertains to the shape of the ridges associated with the three minutiae points constituting the triplet.

Let l_1,l_2 and l_3 represent the length of the three sides of the triangle defining the triplet, such that $l_1 \leq l_2 \leq l_3$. Let α_{max} denote the maximum interior angle of the triangle. Then, the following three features are extracted based on the geometry of the triangle:

$$\begin{aligned} \alpha &= \cos(\alpha_{max}), & (2.1)\\ \beta &= \frac{p^2}{a}, & (2.2)\\ \gamma &= \frac{l_3}{l_1}. & (2.3) \end{aligned}$$

where,

$$\begin{aligned} p &= l_1 + l_2 + l_3,\\ a &= \sqrt{s(s-l_1)(s-l_2)(s-l_3)},\\ s &= \frac{p}{2}. \end{aligned}$$

The three geometry based features extracted from the triangles are invariant to rotation and scaling. The reason for choosing α as the cosine of the maximum angle rather than the angle itself is that the angle is sensitive to noise generated by the minutiae extraction algorithm. The cosine filter can filter out this noise. The angle considered for α is the maximum angle of the triangle. However, very large angles correspond to triangles whose points are almost collinear. Such 'skinny triangles' are not desirable and Delaunay triangulization tends to avoid it. Therefore, triangles whose largest angles are greater than a threshold are rejected. β denotes the shape factor of the triangle. The shape factor is invariant to rotation and translation. For instance, for an equilateral triangle, p=3l, where l_1=l_2=l_3=l and a=$\frac{\sqrt{3}*l^2}{4}$. Therefore, the shape factor for all equilateral triangles will be s=20.78. γ considers the ratio of the maximum to the minimum side. This ratio is more robust to noise compared to the lengths alone which are affected by scaling and shear.

As a final measure to eliminate skinny triangles, a quality factor $\rho = \frac{4*\sqrt{3*a}}{\Sigma_{i=1}^{3} l_i^2}$ is computed and the triangle is retained if ρ is above a certain threshold (ρ=0.6 in the experiments). The quality factor ensures that skinny triangles are eliminated. It should be noted that the orientation information of the minutiae is not used for feature extraction.

The second set of features is based on fitting a quadratic curve to the ridges associated with each triplet. For every minutiae point detected in the fingerprint, a ridge tracing algorithm is invoked that gives a set of points lying on a ridge containing the minutiae point. Since ridge tracing commences from a minutiae point, it is possible for the algorithm to proceed in more than one direction (e.g., in the case of bifurcation points). In such cases, the ridge containing the maximum number of points is selected. Each ridge curve is represented as a second order normalized curve parameterized by the coefficients p_0, p_1 and p_2, i.e., a point (x, y) on the parameterized curve satisfies $y = p_2x^2 + p_1x + p_0$. The ratio of these coefficients, viz., $\kappa = \frac{p_2}{p_1}$ and $\lambda = \frac{p_1}{p_0}$, are used as features. Since there are three ridges associated with each triplet a set of six features are obtained: $\kappa_1, \kappa_2, \kappa_3, \lambda_1, \lambda_2, \lambda_3$.

The set of geometric and ridge curve features that are extracted from a triplet arrangement of minutiae points are shown in Figure 2.2. The feature extractor stage is shown in Figure 2.3.

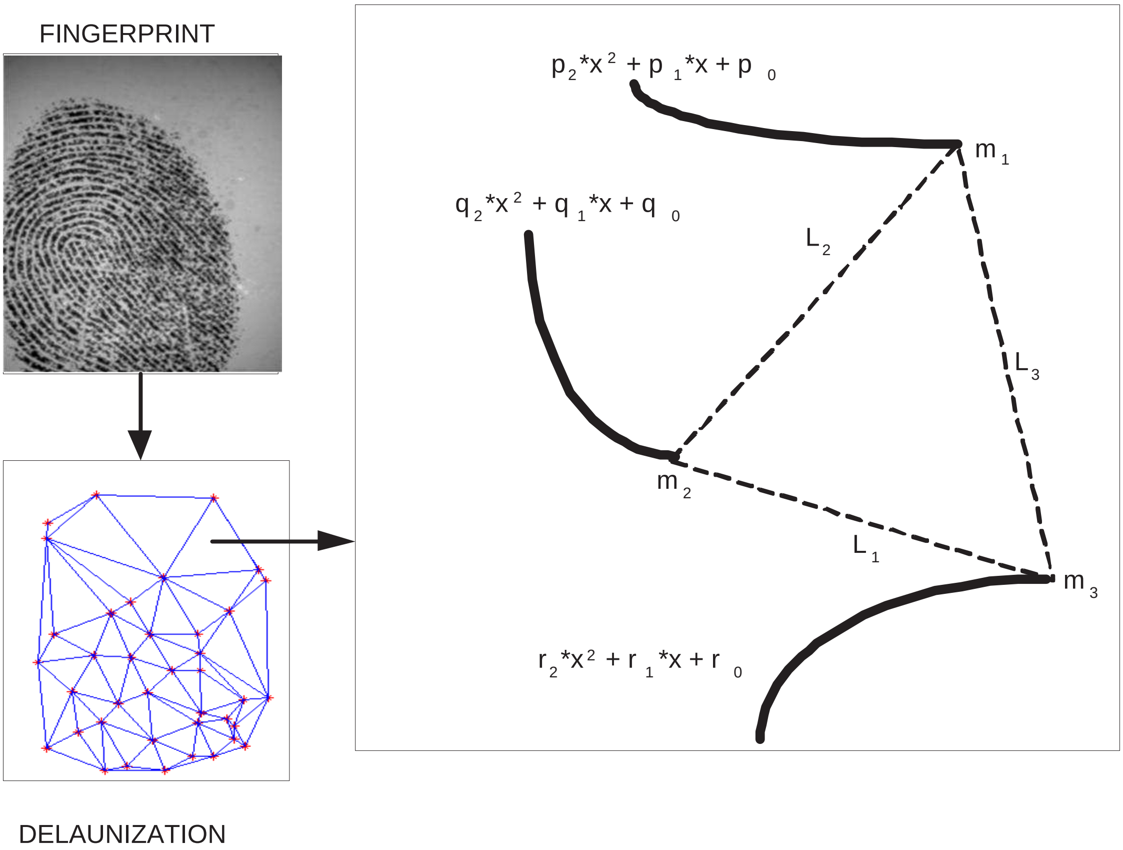


Figure 2.2: Feature extraction from triplet arrangement of minutiae points.

Besides their small number, the minutiae triangles obtained for indexing using delaunay triangulation have good discrimination power, because they satisfy the properties of delaunay triangulation [56]. Figure 2.4 shows the effect of intra-class[3] variability on delaunay triangulation of fingerprint resulting in non matching triangles due to insertion of spurious minutiae points.

[3] The Class is the same as user in this case.

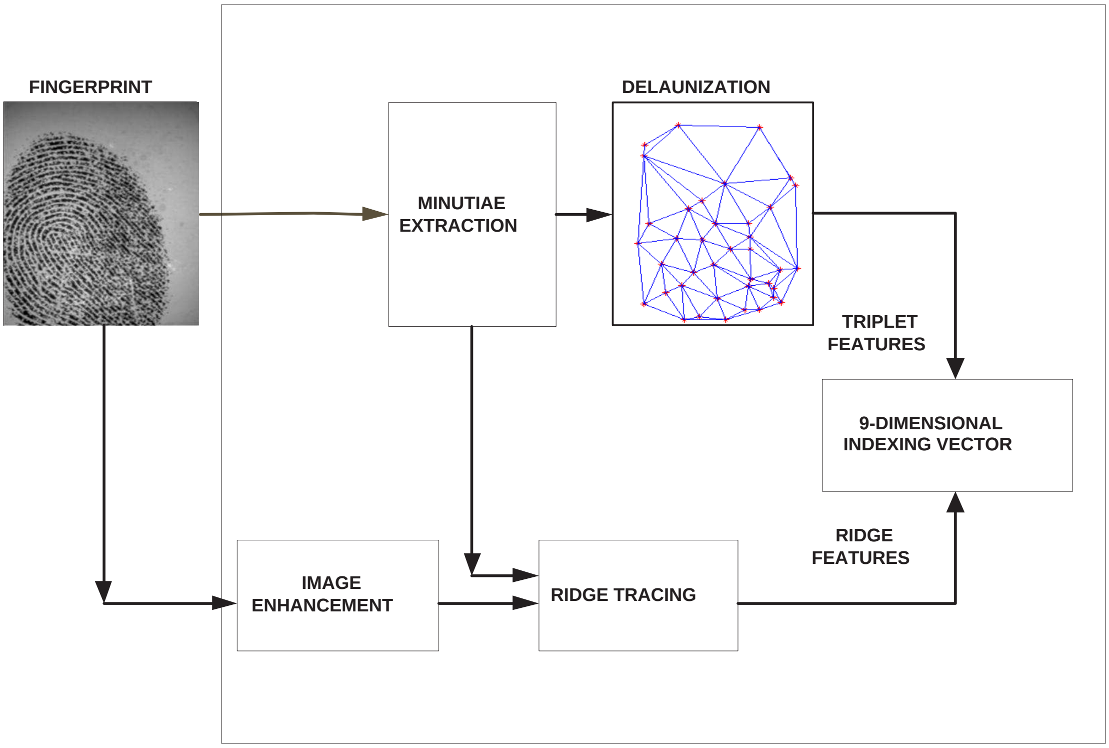


Figure 2.3: Extracting features for creating the index space model. The extracted features are a collection of 9-dimensional entities.

Let ς be a set of k points (or sites) in a 2D plane, $\varsigma = \varsigma_1, \varsigma_2, ..., \varsigma_k$. The Voronoi diagram of ς is the subdivision of the plane into k regions, one for each site in ς. The Voronoi diagram of ς, denoted by $VOR(\varsigma)$ is shown in Figure 2.5. The region of site ς is called the Voronoi cell, denoted by $V(\varsigma)$. The graph G has a node for every Voronoi cell and it has an arc between two nodes if corresponding cells share an edge. Thus G has an arc for every edge of $V(\varsigma)$. If we consider the straight line embedding of G, where the node corresponding to the voronoi cell $V(\varsigma_1)$ is the point ς_1, and the arc connecting the nodes of $V(\varsigma_1)$ and $V(\varsigma_2)$ is the segment $\overline{\varsigma_1\varsigma_2}$. This is called 'embedding the Delaunay[4] graph of ς' and is denoted by $DG(\varsigma)$. Delaunay triangulation [56] is defined to be any triangulation obtained by adding the edges to the Delaunay graph as shown in Figure 2.5. Since all faces of $DG(\varsigma)$ are convex, obtaining such a triangulation is easy. Delaunay triangulation of ς is unique if and only if $DG(\varsigma)$ is a triangulation. This uniqueness makes it ideal for indexing.

Delaunay triangulation has the following properties [50]:

1. If T is the Delaunay triangulation of ς, then the circumcircle of any triangle T does not

[4] Named after the mathematician Boris Nikolaevich Delone

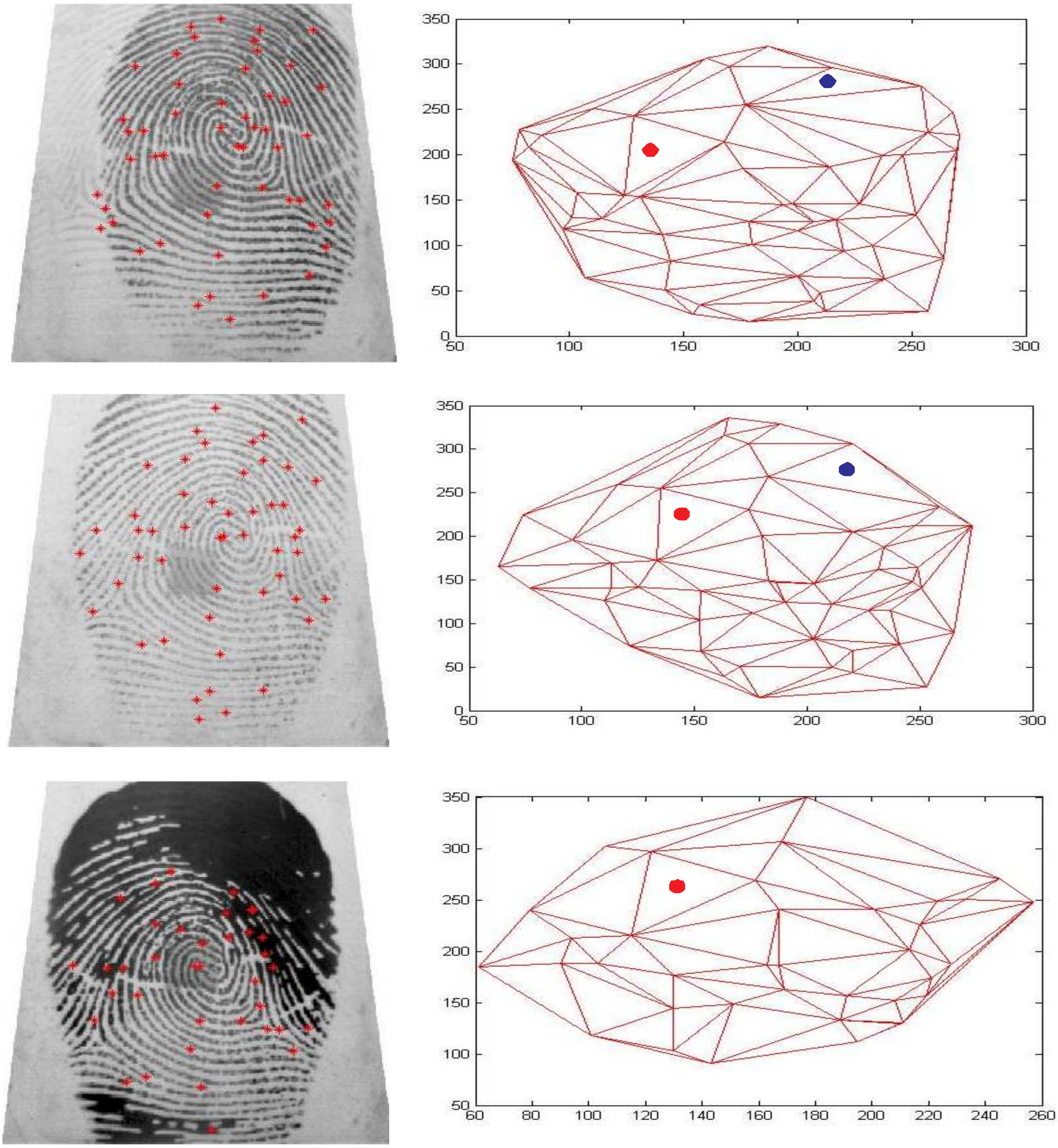


Figure 2.4: Effect of intra class variability on Delaunay triangulation of fingerprints. The red dot denotes a matching triangle in all the three prints of the subject. However, due to insertion of spurious minutiae points (in case of poor quality fingerprints), the triangle denoted by a blue dot, finds a match in two of the three prints only.

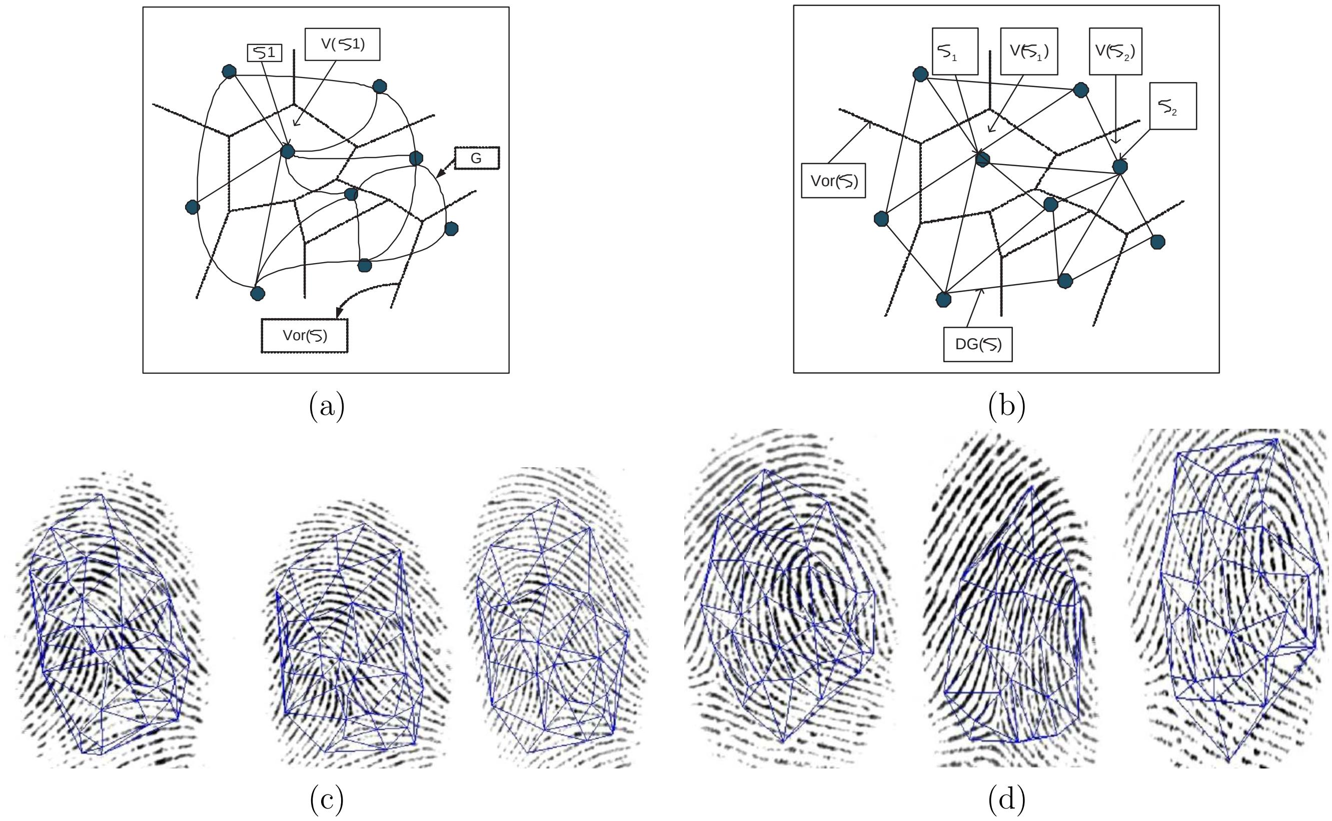
contain a point of ς inside it.

2. Delaunay triangulation maximizes the minimum angle over all triangulations of ς, therefore avoiding skinny triangles. However it does not eliminate all skinny triangles.

3. Delaunay triangulation of a set ς of k points can be computed in $O(k\ log(k))$ expected time, using $O(k)$ expected storage.

4. The upper bound on the number of triangles created by Delaunay triangulation is $9k+1$ [56].

The delaunization of minutiae points of different quality fingerprints from the same user is shown in Figure 2.5.



(a) (b)

(c) (d)

Figure 2.5: Delaunay Triangulation (a) The dual graph of $VOR(\varsigma)$. (b) The Delaunay Graph $DG(\varsigma)$. (c) Delaunization of good quality fingerprint of the same user. (d) Delaunization of varying quality fingerprint of the same user.

2.3.2 Delaunay Triangulation Algorithms

There are several Delaunay triangulation algorithms in literature. The algorithms can be divided into the following categories:

1. Point distribution algorithm.
2. Plane sweep algorithm proposed by Fortune [57].
3. Incremental algorithm proposed by Lee and Schachter [58].
4. Divide and conquer algorithm proposed by Lawson [59].

It has been shown by Shewchuk [60], that in terms of computation time, the divide and conquer algorithm is more efficient than the plane sweep algorithm, which in turn performs better than the incremental algorithm. The divide and conquer algorithm itself can be implemented in three ways, namely, the horizontal cut technique, the vertical cut technique and the alternating cut technique. The alternating cut technique partitions the vertices with alternating horizontal and vertical cuts. Alternating cuts enahance the speed of the algorithm. The Matlab implementation of Delaunay triangulation is based on a Quick Hull algorithm for convex hulls. It performs a randomized incremental algorithm and is sensitive to the number of vertices generated. The Quickhull [61] algorithm performs faster than randomized incremental algorithm and executes faster for inputs with non-extreme points. If there are n input points in R^d and v is the number of output vertices, the worst case complexity of Quickhull is $O(nlogv)$ for $d \leq 3$.

2.3.3 Creating and populating the index space model

Each fingerprint image can be represented as a set of Delaunay triangles $T = \{\vec{t_1}, \vec{t_2}, \ldots \vec{t_3}\}$ that is generated from its minutiae distribution. Each triplet, $\vec{t_i}$ is further characterized by a nonuple consisting of an agglomeration of geometric and ridge curve features, i.e., $\vec{t_i} = \{\alpha^i, \beta^i, \gamma^i, \kappa_1^i, \kappa_2^i, \kappa_3^i, \lambda_1^i, \lambda_2^i, \lambda_3^i\}$. This 9-dimensional entity can be viewed as a single point in hyperspace; thus, each fingerprint image will have a *collection* of points (pertaining to all Delaunay triplets) residing in this 9-dimensional space. Given a set of training fingerprint images, an index space model is first created by performing unsupervised clustering (K-means clustering) on the set of all 9-dimensional entities generated from these images. This results in K clusters, $c_1, c_2, \ldots c_K$ with cluster, c_j, represented by its centroid, $\vec{\mu_j}$.

When a fingerprint corresponding to an identity, y, is input to the system, it is first decomposed into its constituent triplets, $\vec{t_1}, \vec{t_2}, \ldots \vec{t_r}$, which are then mapped into the 9-dimensional index space. Each $\vec{t_i}$, $i = 1, 2, \ldots, r$, will be assigned to exactly one cluster, c_j, $j = 1, 2, \ldots, K$ according to the minimum distance rule, i.e., assign $t_i \rightarrow c_j$ and $y \rightarrow c_j$ if

$$j = \arg \min_{k=1}^{K} ||\vec{t_i} - \vec{\mu_k}||, \tag{2.4}$$

where $||.||$ is the L2 norm. This process is repeated for every print in the database. Thus, each cluster, c_j, will have a listing of all fingerprint identities, $\{y_{j,1}, y_{j,2}, \ldots y_{j,n_j}\}$, which have at least one triplet assigned to that cluster.

The procedure for index space population is shown in Figure 2.6.

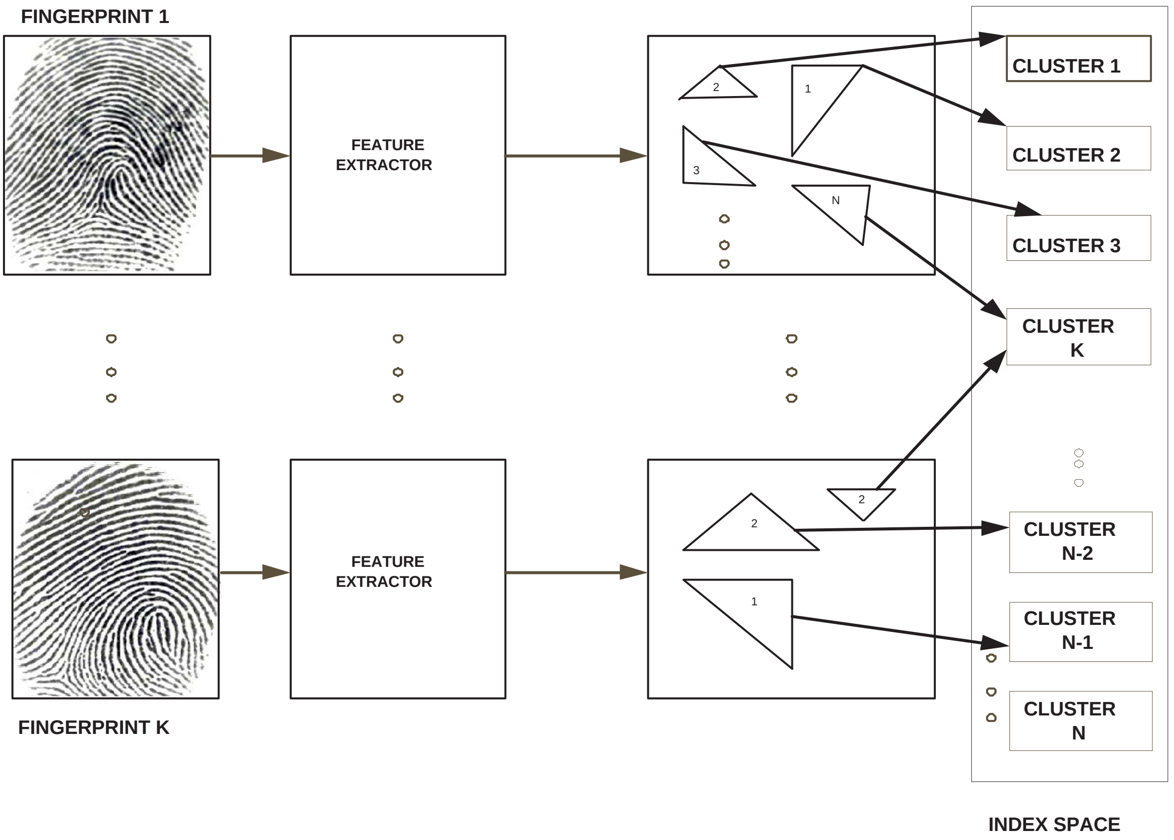


Figure 2.6: Mapping fingerprints in a database to the proposed index space by using the 9-dimensional points extracted from each image.

2.3.4 K-means Data Clustering

Clustering is the classification or partitioning of data into subsets or clusters, so that data in each subset share certain properties. Data clustering is a common statistical analysis tool used in different fields like image analysis, pattern recognition and machine learning.

If the training samples used to design a classifier are labeled by their class membership, the partitioning technique is called supervised clustering. However, if a collection of unlabeled samples have to be partitioned, the partitioning technique is called unsupervised clustering.

K-means algorithm is an unsupervised clustering algorithm to cluster objects into K partitions based on their attributes. The goal is to determine the K means of data generated from Gaussian distributions. K-means algorithm tries to minimize the total intra class variance or the squared error function E (refer to equation 2.5), where there are K clusters C_i, $i = 1, 2, ...K$ and μ_i is the centroid of all the points x_j in cluster C_i.

$$E = \Sigma_{i=1}^{K} \Sigma_{x_j \in C_i} ||\vec{x_j} - \vec{\mu_i}||. \tag{2.5}$$

K-means Algorithm

1. The input data set is partitioned into K initial sets.

2. The centroid of each set is calculated.

3. Further partitioning is performed by attributing each point to the closest centroid.

4. Centroids are recalculated for the new clusters.

5. The creation of new partitions and recalculation of centroids is continued until convergence is achieved. Convergence is achieved when the centroids are stable and do not switch clusters.

The K-means [62] is quite popular because it converges quite quickly in practice. However, the final solution is dependent on the initial set of clusters, therefore the algorithm could be run several times in order to return the best clustering of the data set. Another problem of the algorithm is that the number of clusters to be formed has to be pre-specified. This might lead to undesirable results for data sets which are not naturally clustered.

2.3.5 Fingerprint retrieval

Single target cluster: When a query print, q, is presented to the system, it is first decomposed into its constituent triplets and ridge curves. The set of 9-dimensional points, $\vec{t_1}, \vec{t_2}, \ldots \vec{t_r}$, corresponding to the extracted features are then generated. Next, each point, $\vec{t_i}$, is mapped onto a cluster, c_{π_i} ($\pi_i \in \{1, 2, \ldots, K\}$) in index space using the minimum distance rule (equation 2.4). This process identifies r (possibly) non-unique target clusters, $c_{\pi_1}, c_{\pi_2}, \ldots c_{\pi_r}$, associated with the query print. Those identities occurring frequently in the target clusters (i.e, the top-N identities) are retrieved for further matching.

Multiple target clusters: The algorithm for fingerprint retrieval can be slightly modified by associating more than one cluster (the top m nearest centroids) with every $\vec{t_i}$ of the query print, i.e., assign $\vec{t_i} \rightarrow \{c_{\pi_i,1}, c_{\pi_i,2}, \ldots c_{\pi_i,m}\}$ such that $||\vec{t_i} - \mu_{\vec{\pi_i},1}|| \leq ||\vec{t_i} - \mu_{\vec{\pi_i},2}|| \leq \ldots \leq ||\vec{t_i} - \mu_{\vec{\pi_i},m}||$ and $||\vec{t_i} - \mu_{\vec{\pi_i},k}|| \geq ||\vec{t_i} - \mu_{\vec{\pi_i},m}||$ $\forall k \notin \{1, 2, \ldots m\}$. The retrieval procedure is not affected by this modification. However, using multiple target clusters results in increased computational complexity.

The retrieval process is shown in Figure 2.7.

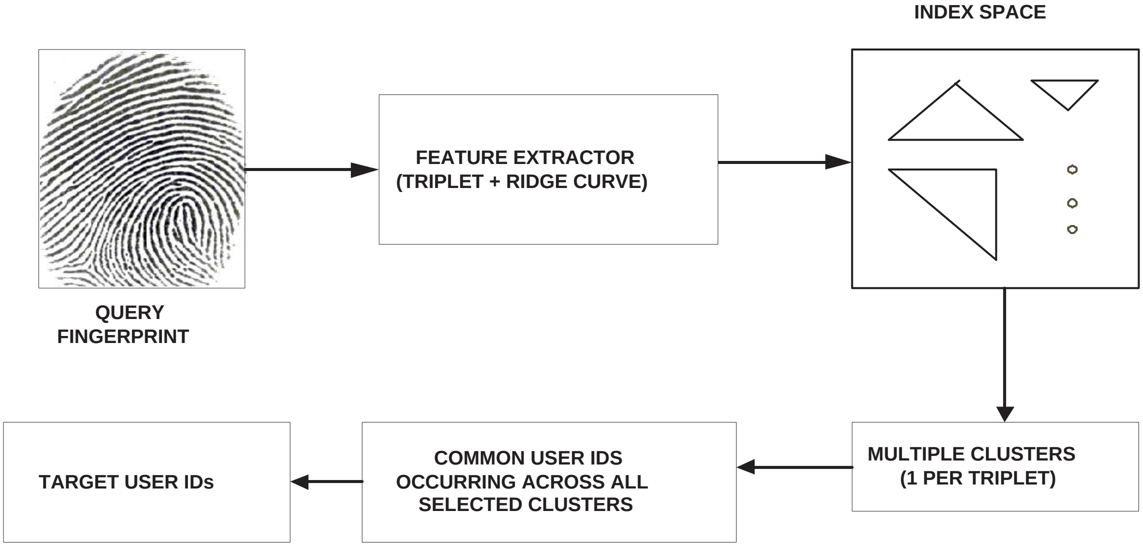


Figure 2.7: Retrieving the top few identities corresponding to the given query fingerprint.

2.4 Experimental evaluation

The performance of the proposed indexing mechanism is summarized using two measures namely the *penetration rate* and the *hit rate*. The penetration rate defines the fraction of user identities retrieved from the database upon presentation of the query print. The hit rate is defined as the probability that the correct user identity is retrieved. In the context of our experiments the following procedure was adopted to compute these measures. Suppose that a database has n fingerprints and there are s query fingerprints. For query print, q_i, we define p_i to be the minimum number of fingerprints that have to be retrieved from the database (based on the retrieval technique described in the previous section) in order to guarantee a hit. Further, without loss of generality let us assume that $p_1 \geq p_2 \geq p_3 \ldots \geq p_s$. Thus, the value $\frac{\Sigma_{i=1}^{z} p_i}{n}$ will be the penetration rate corresponding to a hit rate of $\frac{z}{s}$ since the entries, p_i, are sorted.

Experiments were conducted using the Fingerprint Verification Competition 2004 (FVC2004) database that is partitioned into four (DB1, DB2, DB3, DB4).[5] Each partition has fingerprint images acquired using a particular sensor (see Table 2.7). There are 880 images in each partition corresponding to 110 fingers (with 8 images per finger). There is no correspondence indicated between fingers across these four databases.

1. The index space was first generated using the minutiae points and ridge curves extracted from the images in the synthetic database (DB4). *Note that* true *fingerprint images are not required to create the index space model.* This is, perhaps, an interesting characteristic of the proposed approach. A total of 600 clusters (i.e., K = 600) were created in the index space. This number may be arbitrarily increased for larger databases.

2. The FVC2004 DB1 database was partitioned into two sets, DB1-S1 and DB1-S2. The first three samples of each of the 110 fingers were used to create DB1-S1; the remaining five samples of each of the 110 fingers were used to create DB1-S2. All images in DB1-S1 were then projected onto the clusters in the index space. The images in DB1-S2 were used as query prints to test the efficacy of the indexing model. The performance of the indexing model can be seen in Table 2.2.

3. The above experiment was repeated using the FVC2004 DB2 and DB3 databases also. The resulting performance can be seen in Table 2.2.

4. In order to demonstrate the significance of incorporating ridge curve features $(\kappa_1, \kappa_2, \kappa_3, \lambda_1, \lambda_2, \lambda_3)$ in addition to the geometric features of the triplet (i.e., α, β, γ), the performance with and without using the ridge curve features is presented in Table 2.3.

5. The importance of assigning a triplet to multiple target clusters is borne out in Table 2.4 where the indexing performance is observed to improve upon the consideration of multiple clusters.

[5] http://bias.csr.unibo.it/fvc2004/

In order to test the performance of the indexing space model generated using FVC2004 DB4 on other databases, experiments were conducted using the FVC2002 database that is partitioned into four (DB1, DB2, DB3, DB4) sets.[6] Each partition has fingerprint images acquired using a particular sensor (see Table 2.7). There are 880 images in each partition corresponding to 110 fingers (with 8 images per finger).

1. The FVC2002 DB1 database was partitioned into two sets, DB1-S1 and DB1-S2. The first three samples of each of the 110 fingers were used to create DB1-S1; the remaining five samples of each of the 110 fingers were used to create DB1-S2. All images in DB1-S1 were then projected onto the clusters in the index space. The images in DB1-S2 were used as query prints to test the efficacy of the indexing model. The performance of the indexing model can be seen in Table 2.5.

2. The above experiment was repeated using the FVC2002 DB2, DB3 and DB4 databases also. The resulting performance can be seen in Table 2.5.

3. The effect of considering multiple clusters is shown in Table 2.6.

   Computation Time: Creating and populating the index space model are both off-line processes. When a query print is presented for indexing, the retrieval includes feature extraction (geometric and ridge based) followed by identification of target users (for matching). In MATLAB environment (on a 2.3 GHz Intel Processor with 512MB RAM), the average retrieval time for fingerprints in FVC2002 and FVC2004 databases, is 3.4 seconds (0.7 seconds for geometric feature extraction, 2.5 seconds for ridge feature extraction and 0.2 seconds to identify the target users).

2.5 Analysis of Cluster Population in the Index Space Model

The entities of the fingerprints (triangles) are allotted to clusters. Therefore, each cluster is associated with user identities. Such cluster-user association for FVC2002 is shown in the

[6] http://bias.csr.unibo.it/fvc2002/

Table 2.2: Performance of the proposed indexing model on the DB1, DB2 and DB3 partitions of the FVC2004 database. The index space model was created using the DB4 partition in all three cases

Database	**Hit Rate (%)**	**Penetration Rate (%)** **1 Target Cluster**
FVC2004DB1	100	51.40
FVC2004DB1	95	48.75
FVC2004DB1	90	45.97
FVC2004DB1	85	43.03
FVC2004DB1	80	40.04
FVC2004DB2	100	52.00
FVC2004DB2	95	49.34
FVC2004DB2	90	46.45
FVC2004DB2	85	43.61
FVC2004DB2	80	40.79
FVC2004DB3	100	52.41
FVC2004DB3	95	49.83
FVC2004DB3	90	46.68
FVC2004DB3	85	44.43
FVC2004DB3	80	41.68

Table 2.3: Indexing performance improvement due to inclusion of ridge features.

Database	**Hit Rate (%)**	**Penetration Rate (%)** **Geometric Features**	**Penetration Rate (%)** **Geometric+Ridge Features**
FVC2004DB1	100	54.0	51.40
FVC2004DB1	95	51.43	48.75
FVC2004DB1	90	48.72	45.97
FVC2004DB1	85	45.99	43.03
FVC2004DB1	80	43.25	40.04

Table 2.4: Indexing performance when multiple target clusters are identified for each Delaunay triplet. Using more than two clusters does not seem to have any benefit.

Database	**Hit Rate (%)**	**Penetration Rate (%)** **1 Target Cluster**	**Penetration Rate (%)** **2 Target Clusters**	**Penetration Rate (%)** **3 Target Clusters**
FVC2004DB2	100	52.00	47.05	46.97
FVC2004DB2	95	49.34	44.22	44.17
FVC2004DB2	90	46.45	41.36	41.33
FVC2004DB2	85	43.61	38.54	38.53
FVC2004DB2	80	40.79	35.78	35.66

Table 2.5: Performance of the proposed indexing model on the DB1, DB2, DB3 and DB4 partitions of the FVC2002 database. The index space model was created using the FVC2004 DB4 in all four cases.

Database	**Hit Rate (%)**	**Penetration Rate (%)**
FVC2002DB1	100	47.32
FVC2002DB1	95	44.22
FVC2002DB1	90	41.11
FVC2002DB1	85	38.11
FVC2002DB1	80	35.10
FVC2002DB2	100	47.07
FVC2002DB2	95	44.28
FVC2002DB2	90	41.54
FVC2002DB2	85	38.83
FVC2002DB2	80	36.12
FVC2002DB3	100	50.27
FVC2002DB3	95	47.54
FVC2002DB3	90	44.69
FVC2002DB3	85	41.81
FVC2002DB3	80	38.93
FVC2002DB4	100	45.39
FVC2002DB4	95	42.46
FVC2002DB4	90	39.61
FVC2002DB4	85	36.79
FVC2002DB4	80	33.93

Table 2.6: Indexing performance on FVC2004 DB4 and FVC2002 DB4 when varying number of target clusters are considered.

Database	**Hit**	**Penetration**	**Penetration**	**Penetration**
	Rate (%)	**Rate (%)**	**Rate (%)**	**Rate (%)**
		1 Target Cluster	**2 Target Clusters**	**3 Target Clusters**
FVC2002DB4	100	51.46	45.39	45.41
FVC2002DB4	95	48.76	42.46	42.51
FVC2002DB4	90	45.63	39.61	39.62
FVC2002DB4	85	42.57	36.79	36.75
FVC2002DB4	80	39.53	33.93	33.95
FVC2004DB4	100	51.42	48.00	47.86
FVC2004DB4	95	48.72	45.19	45.08
FVC2004DB4	90	45.68	42.29	42.30
FVC2004DB4	85	42.60	39.54	39.54
FVC2004DB4	80	39.52	36.87	36.78

Table 2.7: Sensor Characteristics.

Database	**Sensor Type**	**Image Size**	**Resolution**
FVC2004DB1	Optical	640x480	500 dpi
FVC2004DB2	Optical	328x364	500 dpi
FVC2004DB3	Thermal Sweep	300x480	512 dpi
FVC2004DB4	Synthetic	288x384	about 500 dpi
FVC2002DB1	Optical	388x374	500 dpi
FVC2002DB2	Optical	296x560	569 dpi
FVC2002DB3	Capacitive	300x300	500 dpi
FVC2002DB4	Synthetic	288x384	about 500 dpi

Table 2.8: Indexing Performance: FVC2002

Database	**Hit Rate (%)**	**Penetration Rate (%) 1 Target Cluster**	**Penetration Rate (%) 2 Target Clusters**	**Penetration Rate (%) 3 Target Clusters**
FVC2002DB1	100	51.19	47.32	51.36
FVC2002DB1	95	48.22	44.22	48.65
FVC2002DB1	90	45.09	41.11	45.58
FVC2002DB1	85	41.92	38.11	42.84
FVC2002DB1	80	38.74	35.10	40.09
FVC2002DB2	100	48.14	47.07	46.68
FVC2002DB2	95	45.32	44.28	43.88
FVC2002DB2	90	42.40	41.54	41.08
FVC2002DB2	85	39.55	38.83	38.31
FVC2002DB2	80	36.87	36.12	35.59
FVC2002DB3	100	58.19	50.27	47.58
FVC2002DB3	95	55.86	47.54	44.85
FVC2002DB3	90	53.16	44.69	42.09
FVC2002DB3	85	50.12	41.81	39.38
FVC2002DB3	80	46.88	38.93	36.66
FVC2002DB4	100	51.46	45.39	45.41
FVC2002DB4	95	48.76	42.46	42.51
FVC2002DB4	90	45.63	39.61	39.62
FVC2002DB4	85	42.57	36.79	36.75
FVC2002DB4	80	39.53	33.93	33.95

Table 2.9: Indexing Performance: FVC2004

Database	**Hit Rate (%)**	**Penetration Rate (%) 1 Target Cluster**	**Penetration Rate (%) 2 Target Clusters**	**Penetration Rate (%) 3 Target Clusters**
FVC2004DB1	100	51.40	50.17	49.98
FVC2004DB1	95	48.75	47.52	47.34
FVC2004DB1	90	45.97	44.74	44.58
FVC2004DB1	85	43.03	41.90	41.82
FVC2004DB1	80	40.04	39.05	39.11
FVC2004DB2	100	52.00	47.05	46.97
FVC2004DB2	95	49.34	44.22	44.17
FVC2004DB2	90	46.45	41.36	41.33
FVC2004DB2	85	43.61	38.54	38.53
FVC2004DB2	80	40.79	35.78	35.66
FVC2004DB3	100	52.41	50.24	50.11
FVC2004DB3	95	49.83	47.56	47.40
FVC2004DB3	90	46.68	44.80	44.60
FVC2004DB3	85	44.43	42.04	41.87
FVC2004DB3	80	41.68	39.27	39.13
FVC2004DB4	100	51.42	48.00	47.86
FVC2004DB4	95	48.72	45.19	45.08
FVC2004DB4	90	45.68	42.29	42.30
FVC2004DB4	85	42.60	39.54	39.54
FVC2004DB4	80	39.52	36.87	36.78

Figures 2.8,2.9,2.10,2.11.On an average, the percentage of users allotted to each cluster is 11% for FVC2002 DB1, 13.6% for FVC2002 DB2, 7.2% for FVC2002 DB3 and 9% for FVC2002 DB4.

Figures 2.12,2.13,2.14,2.15 depict the cluster-user association for FVC2004 database. On an average, the percentage of users allotted to each cluster is 13.6% for FVC2004 DB1, 12.7% for FVC2004 DB2, 9% for FVC2004 DB3 and 11% for FVC2004 DB4.

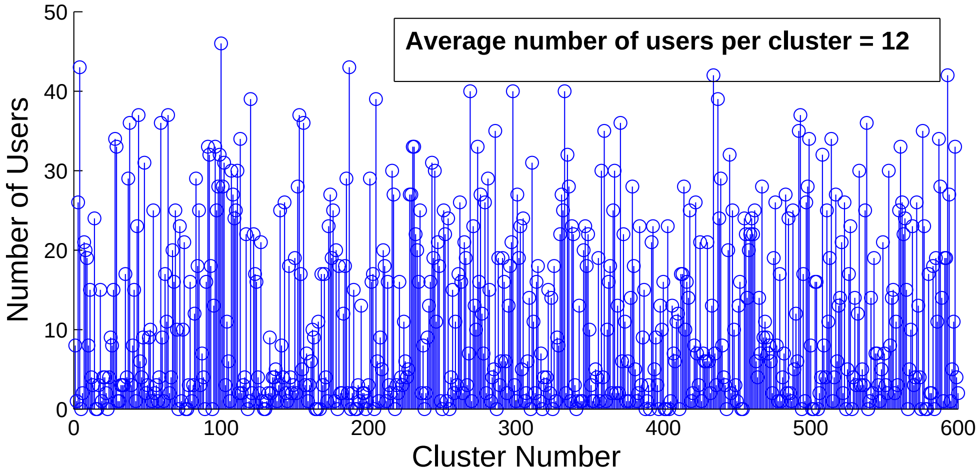


Figure 2.8: User distribution in clusters for FVC2002 DB1.

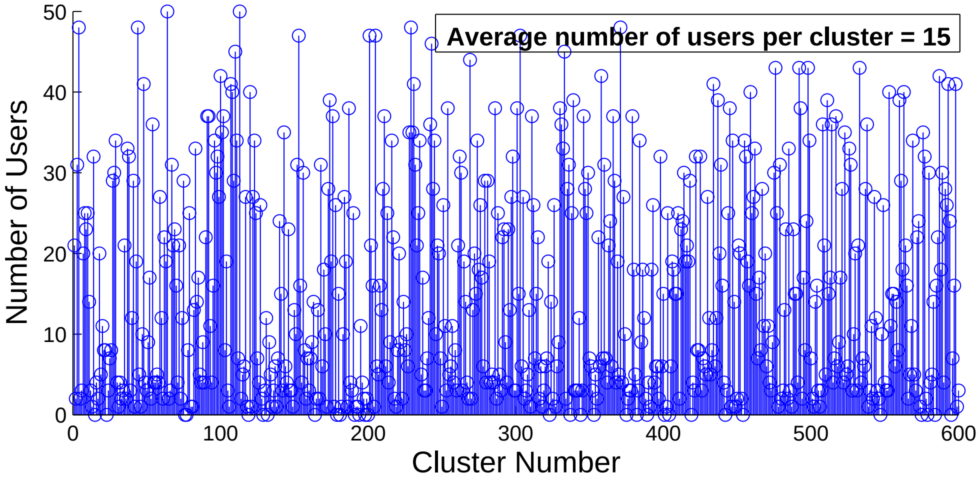


Figure 2.9: User distribution in clusters for FVC2002 DB2.

The number of cluster allotted to each user would depend on the clusters allotted to each entity (triangle) of the given user. The distributions of the number of clusters for users in the FVC2002 and FVC2004 database are shown in the Figures 2.16,2.17,2.18,2.19,2.20,2.22,2.23.

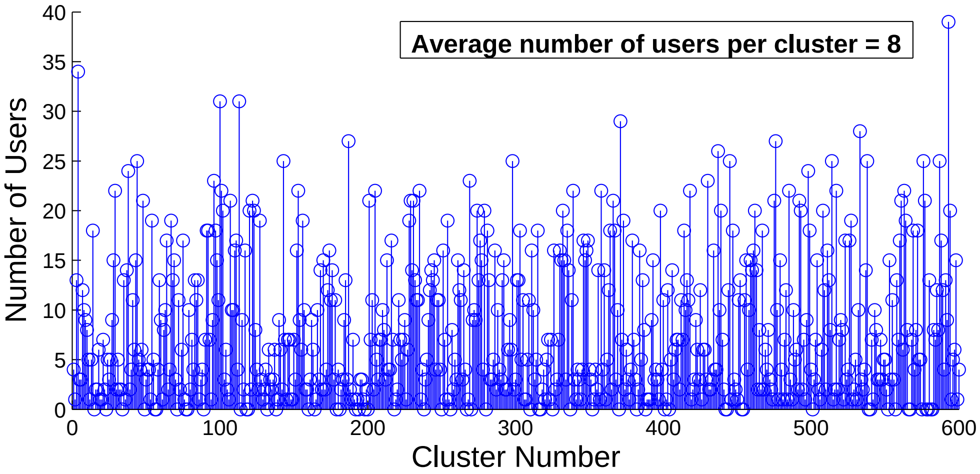


Figure 2.10: User distribution in clusters for FVC2002 DB3.

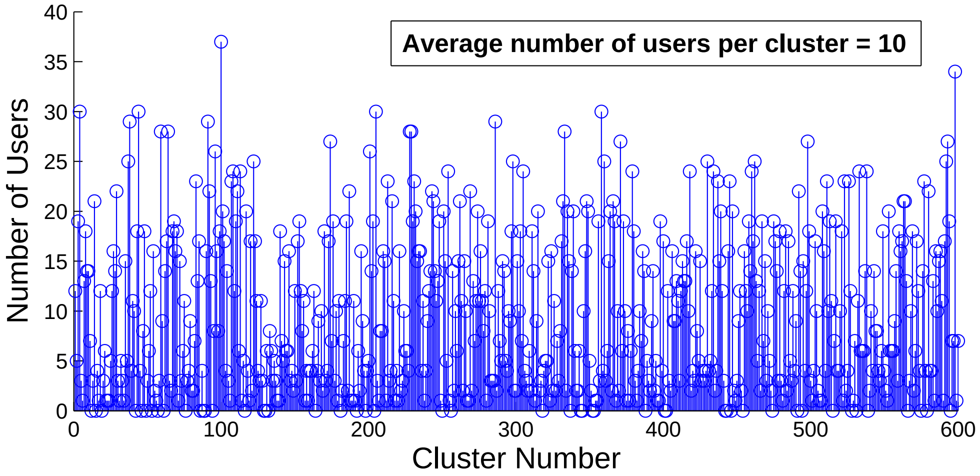


Figure 2.11: User distribution in clusters for FVC2002 DB4.

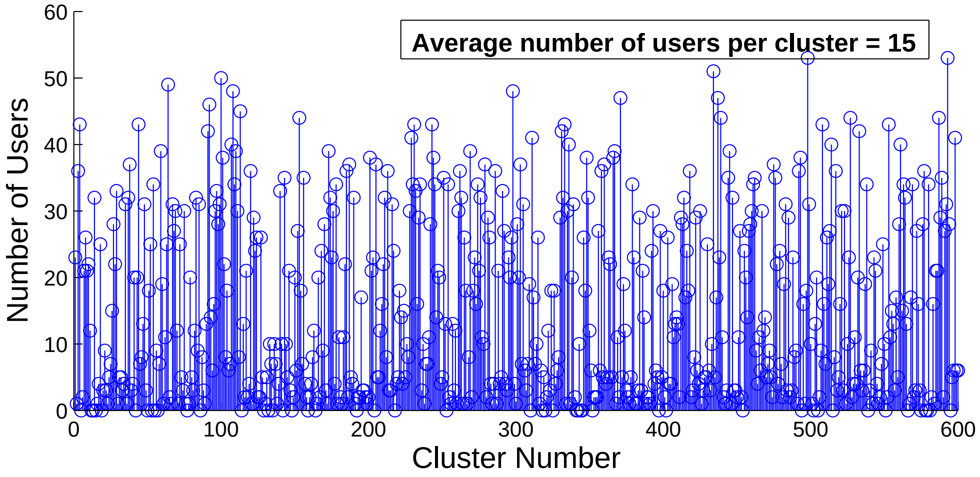


Figure 2.12: User distribution in clusters for FVC2004 DB1.

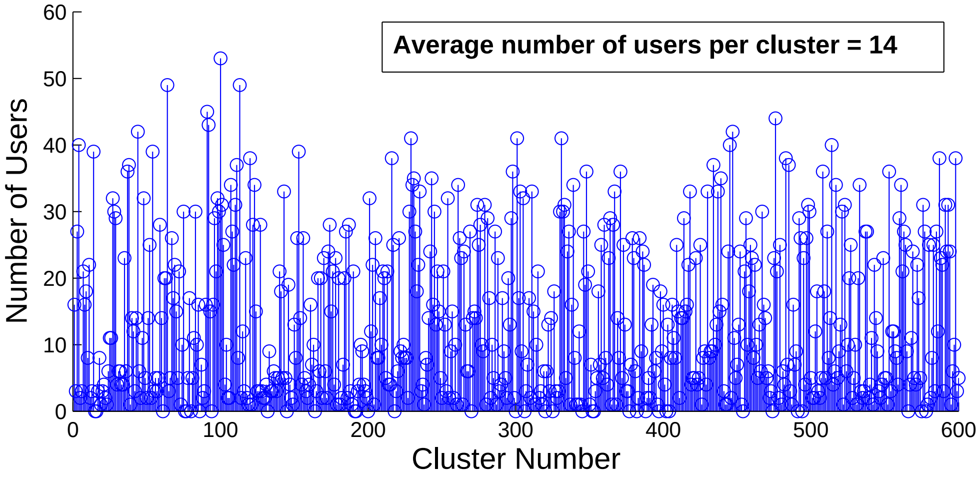


Figure 2.13: User distribution in clusters for FVC2004 DB2.

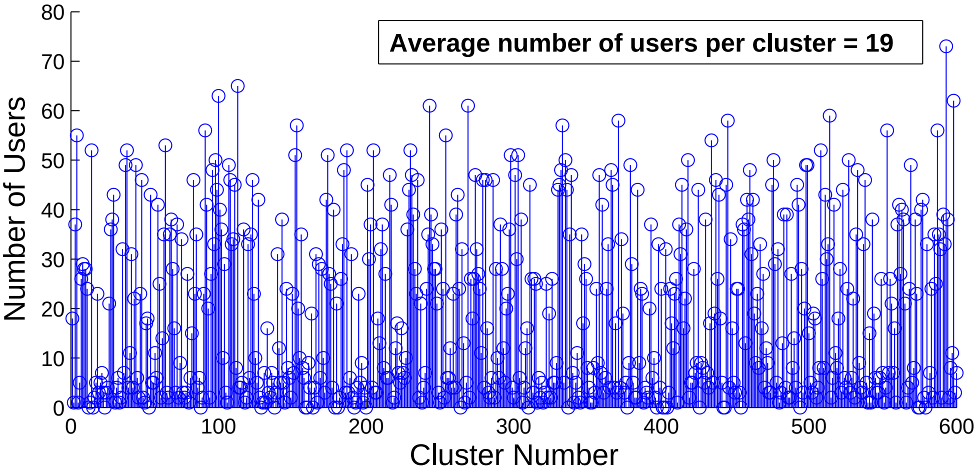


Figure 2.14: User distribution in clusters for FVC2004 DB3.

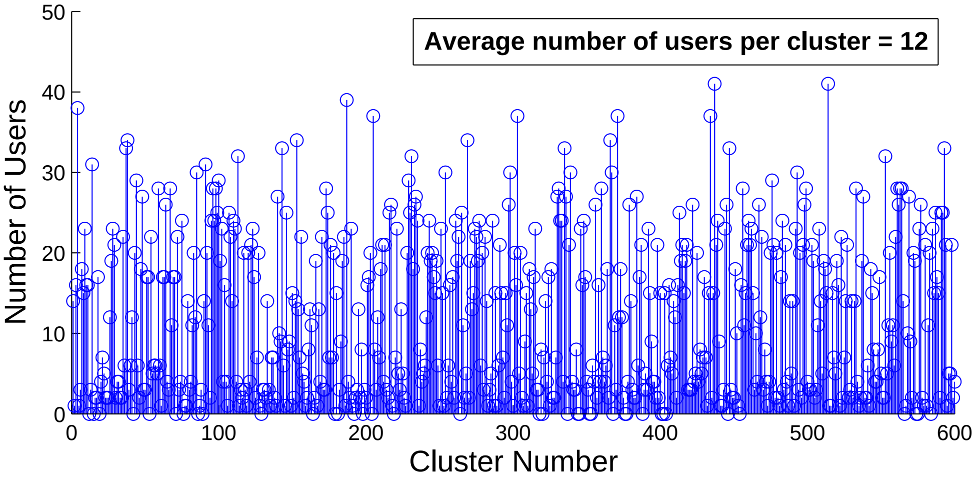


Figure 2.15: User distribution in clusters for FVC2004 DB4.

On an average, the percentage of distinct clusters containing each user is 12.5% for FVC2002 DB1, 14.2% for FVC2002 DB2, 7.3% for FVC2002 DB3, 8.6% for FVC2002 DB4, 14% for FVC2004 DB1, 12.5% for FVC2004 DB2, 18.5% for FVC2004 DB3 and 11.3% for FVC2004 DB4.

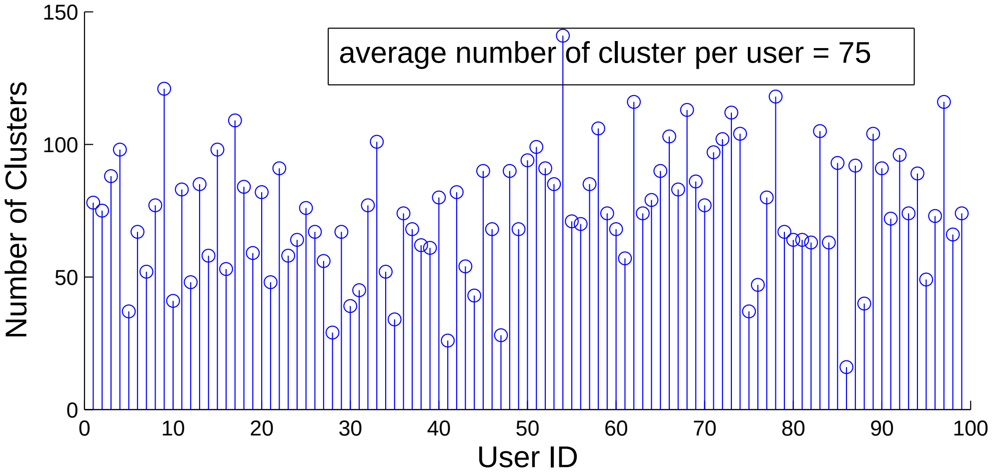


Figure 2.16: Variation in clusters allotted for users in FVC2002 DB1.

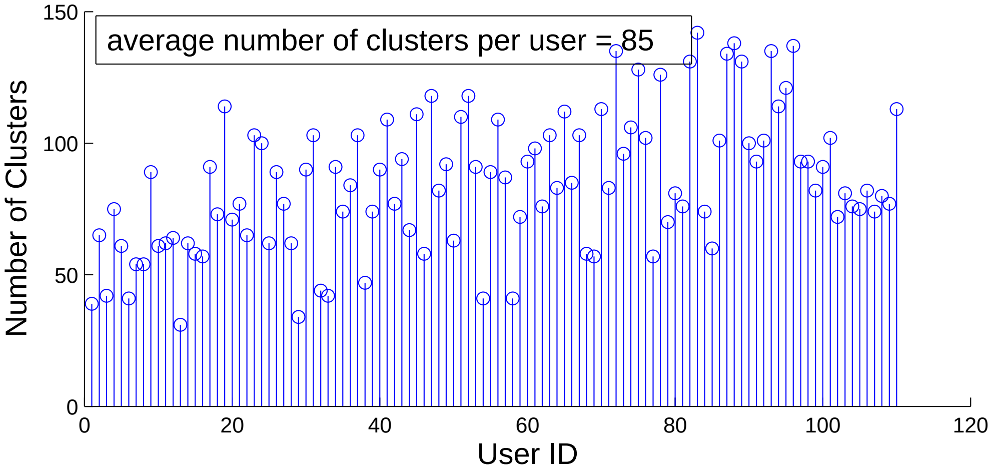


Figure 2.17: Variation in clusters allotted for users in FVC2002 DB2.

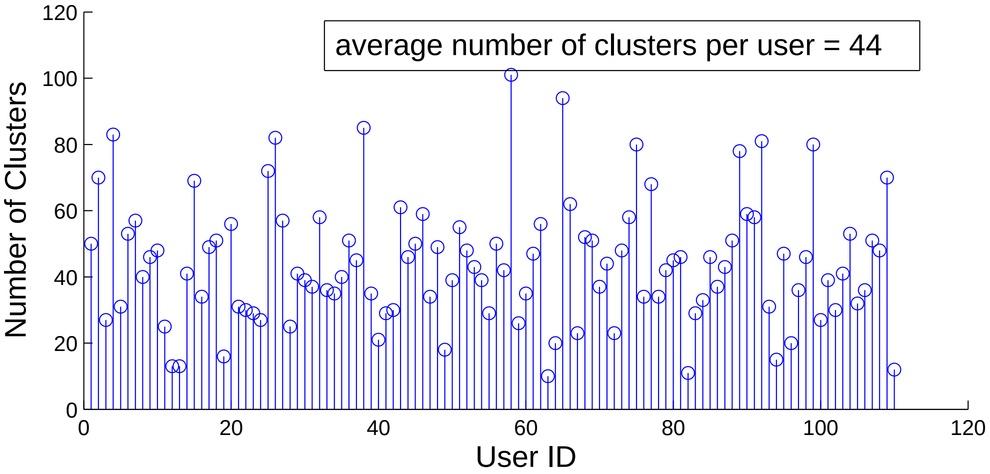


Figure 2.18: Variation in clusters allotted for users in FVC2002 DB3.

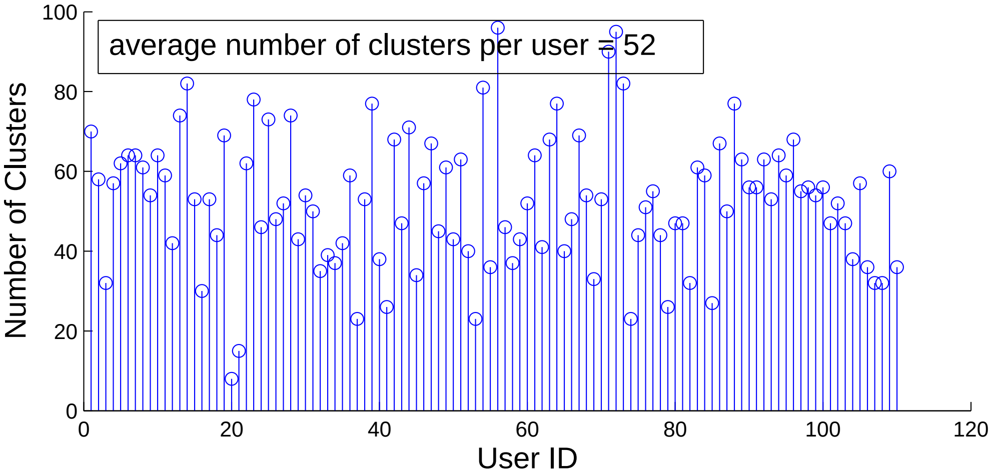


Figure 2.19: Variation in clusters allotted for users in FVC2002 DB4.

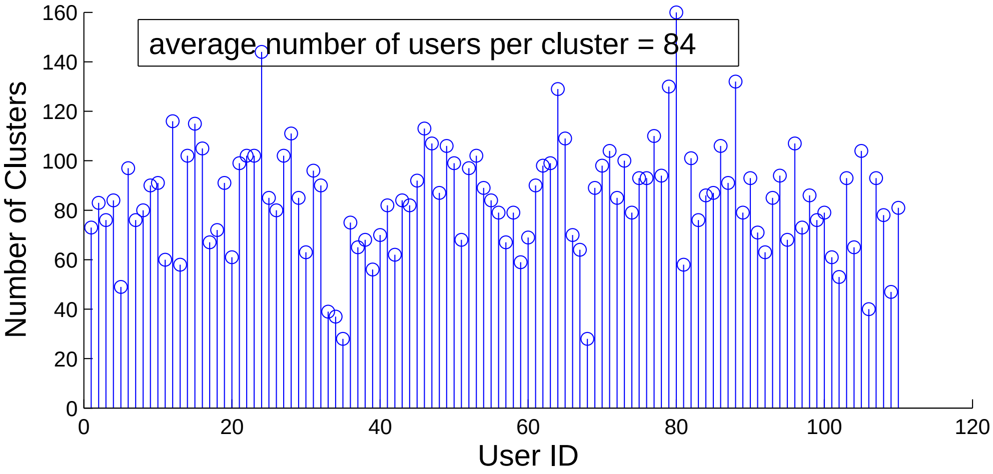


Figure 2.20: Variation in clusters allotted for users in FVC2004 DB1.

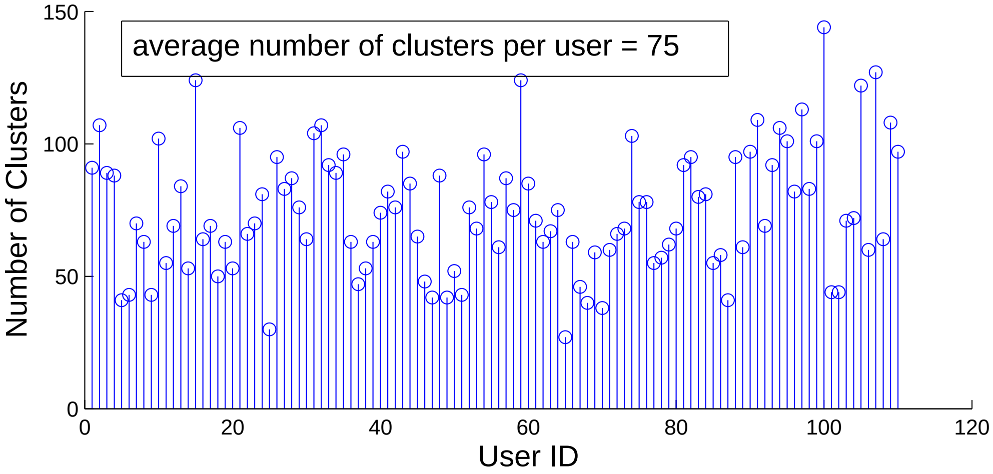


Figure 2.21: Variation in clusters allotted for users in FVC2004 DB2.

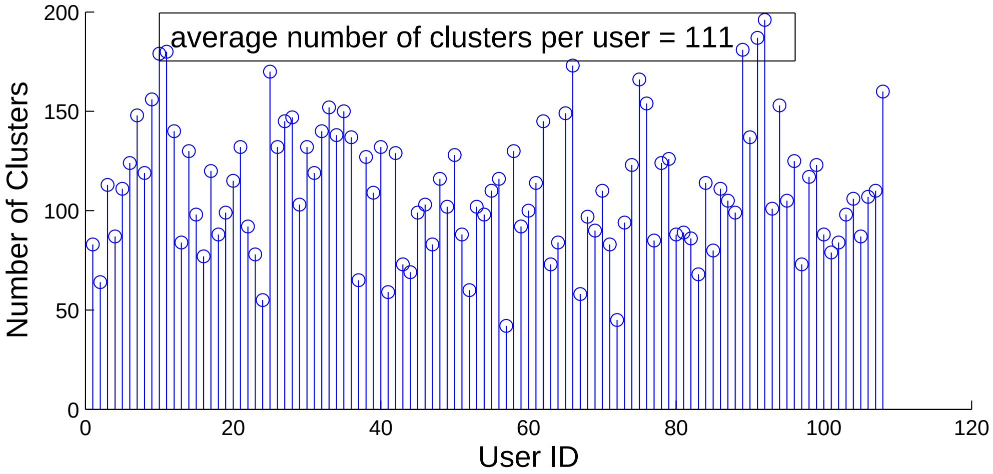


Figure 2.22: Variation in clusters allotted for users in FVC2004 DB3.

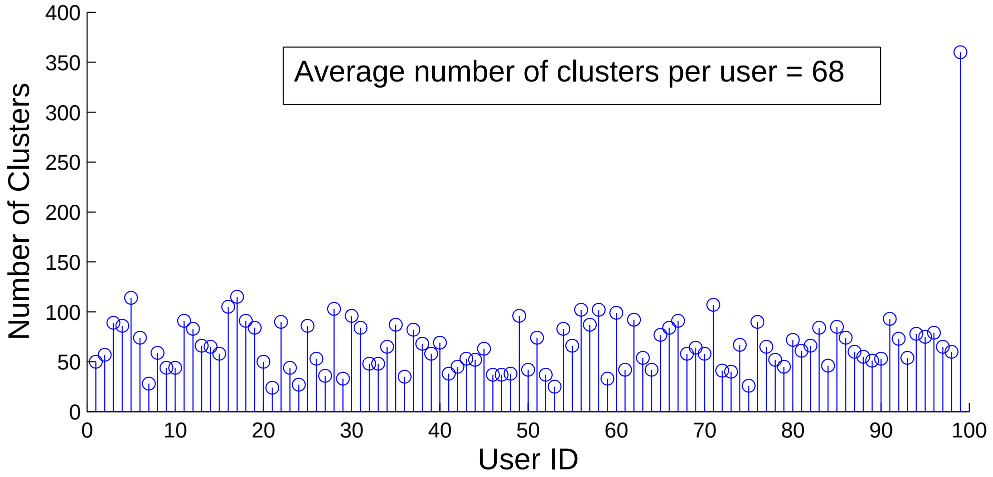


Figure 2.23: Variation in clusters allotted for users in FVC2004 DB4.

2.6 Summary and Future Work

The purpose of this paper was to highlight the improvement in indexing performance whilst augmenting the minutiae-triplet based features with the associated ridge curve information. Since the features used for indexing can be computed efficiently and do not depend on singularities in the fingerprint, the proposed indexing scheme is robust to noise and distortion. Furthermore, the ability to use the index space model designed using a synthetic database for indexing images captured using different imaging devices having different characteristics underscores the significance of the proposed technique. However, it should be noted that the resolutions of these sensing devices are comparable. If they were not, the ridge based feature extraction would require a scaling factor. It will be interesting to observe the performance of this scheme across databases where the users remain the same but the imaging device characteristic changes. Future work might include analyzing the improvement in the efficiency of the indexing scheme by using parallel computing. The effect of varying the total number of clusters on the indexing performance could also be analyzed. In order to obtain meaningful data regarding the computation time and real-time performance, the indexing algorithm could be implemented on a field programmable gate array (FPGA) based array processor [54].

Chapter 3

Iris Indexing

3.1 Introduction

Iris is considered to be a highly reliable biometric for personal identification. The iris is an internal organ of the eye and is located behind the cornea and in front of the lens. It is the annular region between the pupil (inner boundary) and the sclera (outer boundary). This region is rich in texture and has several features such as crypts, furrows, freckles, corona, stripes, moles etc. Thus iris texture is chaotic and unique to each individual.

The goal of an iris recognition system is to extract features representing the textural information present in the iris and thus authenticate (verify or identify) an individual, on the basis of such features. The main modules of such a system are segmentation, enhancement, feature extraction and matching. During enrollment the features are extracted and stored in the database as templates. The authentication phase encompasses image preprocessing followed by feature extraction for a given iris image. This feature set is then compared with the templates in the database in order to perform identification or verification of an individual's identity.

An iris indexing technique would reduce the number of candidate hypotheses to be considered by the iris matching algorithm. Currently, the iris algorithms perform exhaustive matching based on the hamming distance between IrisCodes [9], which is the encoded form of the features extracted from the iris texture. Analyzing the iris texture itself would be useful in exploiting the individual properties of the user, that is crucial for designing user-specific matchers and iris indexing for large databases.

This chapter investigates three iris indexing techniques. The first is based on the Principal

Component Analysis (PCA) of IrisCode [9] which is a feature vector extracted from the iris texture. The last two techniques examines the textural content of the image. The second technique is based on the Local Binary Pattern (LBP) analysis of iris texture. The third technique is based on Signed Pixel Level Difference Histogram of the raw pixel intensities.

Most iris recognition algorithms effectively extract relevant features from iris and use them for identification/verification. However, none of these algorithms *explicitly* characterize the iris texture. The iris texture analysis problem can be handled in a better way if we answer two fundamental questions : What is texture (texture definition) and how is that texture distributed (texture location)? Traditionally, Markov Random Field (MRF) models of images have used predefined pairs of interacting neighbors and therefore, is not the ideal choice for iris which has non-uniform texture. In the proposed work, a statistical analysis of the Signed Pixel Level Difference Histogram (SPLDH) of block pairs is considered to define texture, and multiple block-wise[1] interaction is used to define the significant texture location. It should be noted that texture properties also vary with the resolution of the acquired image: global properties are more prominent at lower resolutions and local properties are more prominent at higher resolutions.

3.2 Iris Indexing

The problem of automatic iris identification involves comparing a query iris image, q, with iris entries, $D = \{d_1, d_2, d_3, ...d_n\}$, in a database in order to determine the identity y of the iris. Each entry d_j, $j = 1, 2, \ldots n$, is assumed to be associated with an identity, y_j and, hence, $y = y_k$ where $k = \arg\max_{j=1}^{n}\{S(q, d_j)\}$ and S is the matching function that assesses the similarity between two irides. The computational complexity of the identification process is primarily dictated by the number of entries, $\mid D \mid= n$, in the database. In order to reduce the number of matching operations, a filtering procedure is usually invoked to identify a subset R of irides ($R \subset D$) such that $|R| = m << n$. Filtering can be accomplished using two distinct approaches: classification and indexing.

Most iris recognition algorithms used today are able to perform fast one to many matching. Therefore, the need to delve into iris classification or indexing may not be immediately appealing. With increasing database size and non-ideal iris processing (which takes more time), the increase

[1] Block is a localized group of pixels

in computation cost could be shared by additional hardware to facilitate the parallel search for a given query iris image. However, hardware alone cannot solve the problem. The search can be guided more effectively by employing iris classification or indexing techniques. With the widespread acceptance of iris recognition as a reliable biometric, the future may see many iris recognition systems deployed all over the world implementing different iris recognition algorithms and using different acquisition devices. If the need arises to search for identities amongst such a large and diverse collection of irides, an indexing scheme becomes imperative.

3.2.1 Approach to the problem

Most of the commercially used iris recognition systems use IrisCodes- a binary representation of information extracted from the iris. Therefore the fundamental way to approach the problem is to analyze the possibility of indexing iris on the basis of IrisCodes. However, considering the original motivation of the problem, whereby the indexing scheme should be able to handle multiple algorithms and acquisition devices, the result of the analysis on IrisCodes should be considered as a benchmark for further analysis. The next level of analysis should delve into the more fundamental iris texture. Therefore, in order to research into the problem of iris indexing, the analysis can be categorized into two parts - IrisCode analysis and iris texture analysis.

3.3 IrisCode Analysis

Before discussing the proposed technique to perform indexing based on IrisCodes. The IrisCode generation is discussed below.

3.3.1 IrisCode Generation:

Gabor filtering is an important part of IrisCode generation. Gabor wavelets are formed from two components, a complex sinusoidal carrier ($c(x, y)$ see equation 3.1) and a gaussian envelope ($g(x, y)$ see equation 3.4), where x and y are spatial coordinates. The complex carrier takes the form as shown in equation(3.1). The real part of the function is given by equation(3.2) and the imaginary part is given by equation(3.3). The parameters u_0 and v_0 represent the frequency of the horizontal and vertical sinusoid respectively. ϕ represents an arbitrary phase shift.

$$c(x,y) = e^{j(2\pi(u_0x+v_0y)+\phi)}, \tag{3.1}$$

$$Re[c(x,y)] = cos(2\pi(u_0x+v_0y)+\phi), \tag{3.2}$$

$$Im[c(x,y)] = sin(2\pi(u_0x+v_0y)+\phi), \tag{3.3}$$

The envelope is the other important component of a Gabor wavelet. The envelope has a Gaussian profile and is described in equation (3.4), where κ is a scaling constant, a and b are envelope axis scaling constants, the subscript r denotes the rotation operation, θ is the envelope rotation constant, and x_0 and y_0 are coordinates of the Gaussian envelope peak.

$$g(x,y) = \kappa(e^{-\pi(a^2(x-x_0)_r{}^2+b^2(y-y_0)_r{}^2)}), \tag{3.4}$$

$$(x-x_0)_r = (x-x_0)\cos\theta + (y-y_0)\sin\theta, \tag{3.5}$$

$$(y-y_0)_r = -(x-x_0)\sin\theta + (y-y_0)\cos\theta \tag{3.6}$$

Finally, a 2-D Gabor wavelet is derived by multiplying $c(x,y)$ by $g(x,y)$.

In the context of iris recognition, once we have the Gabor wavelet, we can extract a set of unique features [9] and store them. This set of features is known as the IrisCode. Thus when an unknown iris is presented to the iris recognition system, the IrisCodes can be compared to search for a possible match.

Gabor filters are good at detecting patterns in images, thus making them ideal for iris images which have a unique texture generated through a chaotic process. The Gabor filtering is performed on the segmented iris which is unwrapped (mapped onto cartesian coordinate). A fixed frequency 2-D Gabor filter looks for patterns in the unwrapped image. The convolution of the image with the 2-D Gabor filter generate a complex result having real and imaginary parts which are treated separately. In order to reduce storage space, quantization is performed on the real and imaginary parts. If a the resultant value is positive it is stored as 1 otherwise it is stored as 0. The resultant real and imaginary binary images are called IrisCodes. The prominence of iris texture changes with increasing distance from the pupil. Therefore a set of three Gabor filters with different scales and frequency, but with the same orientation (0°) are applied to different regions of the 'normalized' iris as shown in Figure 3.1.

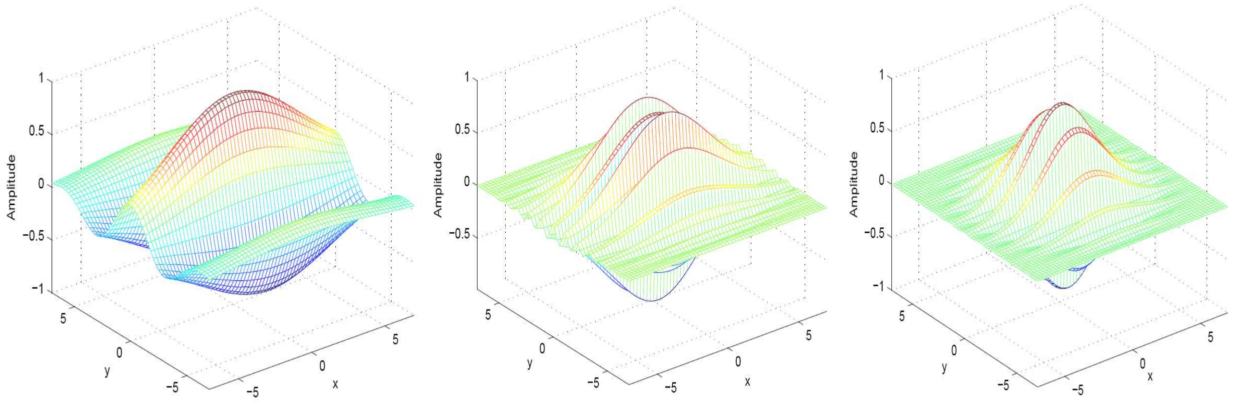


Figure 3.1: Real part of 2-D Gabor wavelet filters with different scales and frequency but same orientation (0°).

In theory, two IrisCodes independently generated from a unique iris will be the same. However due to issues in acquisition systems, illumination and head rotation (non-ideal iris) two IrisCodes of the same individual are in reality not the same. Since the iris generation process itself is believed to be random, two IrisCodes from different individuals will be statistically independent (having a hamming distance larger than a particular threshold). Therefore only IrisCodes from the same eye will fail the test of statistical independence [9]. The IrisCode generation procedure is summarized in Figure 3.2.

3.3.2 Analysis of IrisCodes

An indexing technique could be based on Principal Component Analysis (PCA) of the IrisCodes. PCA [63] or Hotelling transform is a useful statistical technique that has been adopted in the field of face recognition and image compression, in order to find patterns in data of high dimension.

PCA is a linear transformation that transforms the data to a new coordinate system such that the first or the principal component has the greatest variance by any projection of data lying on it. PCA can be used for dimensionality reduction in a dataset while retaining those characteristics of a dataset that contributes most to its variance. Thus the low order principal components are retained and the higher order ones are ignored. Unlike other linear transforms

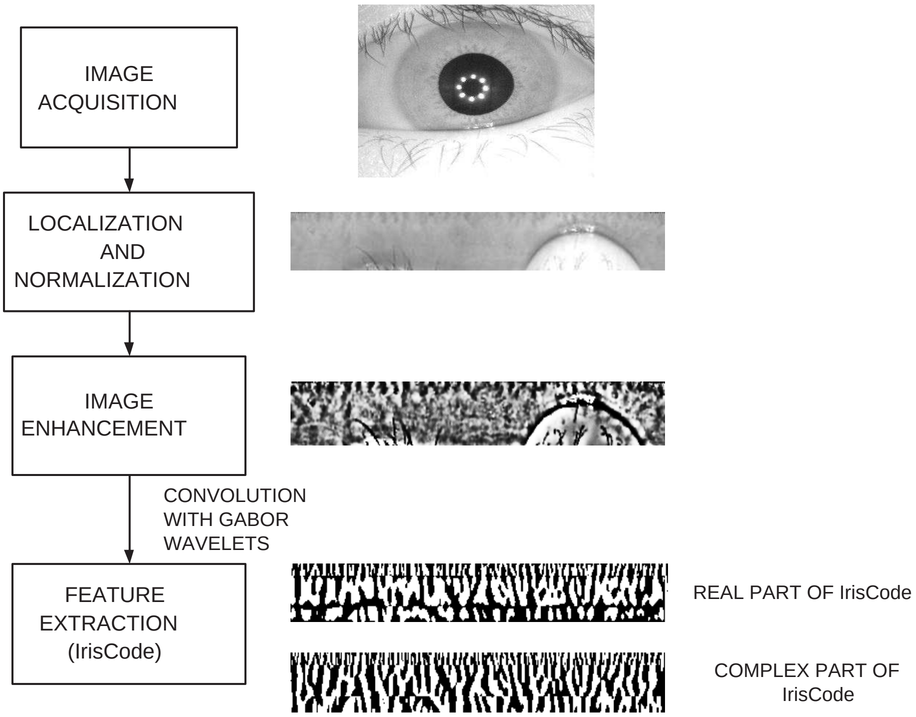


IMAGE ACQUISITION
LOCALIZATION AND NORMALIZATION
IMAGE ENHANCEMENT
CONVOLUTION WITH GABOR WAVELETS
FEATURE EXTRACTION (IrisCode)
REAL PART OF IrisCode
COMPLEX PART OF IrisCode

Figure 3.2: IrisCode generation procedure: IrisCodes are generated by convolution of Gabor wavelets with enhanced iris image. The enhancement is performed on the localized and normalized iris image.

like Discrete Cosine Transform (DCT), PCA does not have a fixed set of basis vectors. Its basis vectors depend on the dataset.

In the first proposed technique for iris indexing, PCA is performed on the IrisCodes of a training set of data. Thereafter an unsupervised clustering (K-means) is performed on the reduced dimension vector. The feature set extracted (refer to Section 3.5), number of principal components and clusters are varied and the result is reported (refer to Experimental Evaluation Section).

The motivation behind PCA analysis of IrisCodes is that, in databases where the IrisCode is already generated and stored, it will be easier to reduce the number of candidate hypothesis by clustering such features extracted from the IrisCodes.

3.4 Iris Texture Analysis

The human mind is easily able to perceive and distinguish between different types of texture. However, in the field of computer vision making a computer understand texture is still a challenging task.

A texture can be defined as a measure of the variation of a surface intensity, quantifying properties such as smoothness, coarseness and regularity. Texture is often used as a region descriptor in computer vision and image analysis.

There are certain properties that a texture can possess [64].

1. Texture is a region descriptor, it cannot be defined for a single point alone.
2. Texture is a property of the gray scale value of an image. Even if the color information is lost, the human mind can still correlate texture from a colored image to the corresponding gray scale image.
3. Any statistical tool used to describe the gray scale variation in an image can be used to describe the texture of the image.
4. Texture can be 'felt' in terms of fineness, coarseness, regularity, contrast, density, homogeneity, directionality, etc.
5. Texture properties differ at various resolutions. While lower resolutions are better for capturing global texture properties, the local texture properties are well represented at the higher resolutions.

Texture can be described using primarily three approaches - statistical, structural and spectral. Statistical techniques use statistical properties of the gray levels of the points comprising a surface. Generally such properties are computed from the gray level histogram or the gray level co-occurrence matrix of the surface. The motivation behind this approach is the statistical analysis of pixel intensities and position. Statistical techniques are well suited for micro structures. The structural approach relies on the concept of texture primitives, called 'texels' or 'textons'. The texture is described using a 'texel' vocabulary and inter 'texel' relationship. Structural techniques are well suited for macro structures. Spectral techniques are based on the properties of the Fourier spectrum. They are well suited to describe global periodicity of the gray levels of a surface by detecting high energy peaks in the spectrum.

Texture analysis has been an area of intense research for almost half a century now. However, due to problems like varying illumination, changes in surface shape and lack of homogeneity in real world textures, analyzing them has proved to be difficult. Gabor filtering is by far the most elegant texture analysis tool. Its good performance can be attributed to fact that it encompasses both local edge and spatial frequency information. However, it is affected by varying illumination conditions and is computationally expensive for large mask sizes. For the purpose of iris indexing, a texture analysis tool is required which can analyze macro as well as micro structure, thus taking advantage of both statistical and structural approaches simultaneously. Thus, the second proposed technique for iris indexing is based on local binary pattern, which serves this purpose. The third iris indexing technique is motivated by statistical analysis of pixel intensities and positions and is based on Signed Pixel Level Difference Histogram (SPLDH).

3.4.1 Local Binary Pattern (LBP) Analysis of Iris Texture:

The LBP operator was first introduced as a complementary measure for local image contrast [65]. The operator first worked with eight neighbors of a pixel, using the central pixel as a threshold. An LBP code was generated by multiplying the thresholded values with corresponding weights given to the pixels and summing up the result as shown in Figure 3.3. In the context of iris indexing, the proposed technique suggests applying the invariant LBP operator on the enhanced iris texture image to form a LBP histogram for each image. Thereafter, the proximity between LBP histograms is exploited to reduce the number of candidate hypothesis for matching. The database organization can be done on the basis of unsupervised clustering of LBP histograms in order to support the indexing technique. This technique was analyzed by varying the number of neighborhood pixels for calculating the LBP code. The experiments performed are explained in detail in Section 3.5.

3.4.2 Signed Pixel Level Difference Histogram Analysis of Iris Texture:

In context of iris indexing, the technique used for iris analysis should capture the inter-pixel relationship. Thus, the iris indexing scheme should group similar pixels and capture the distribution of such similarities across the iris texture. In order to reduce the complexity, the proposed scheme delves into evolving an iris indexing technique utilizing inter-block relationship. In order to extract pixel level difference features of blocks, a Signed Pixel Level Difference Histogram is generated. Suppose, the iris texture of a user is divided into blocks. SPLDH can be generated

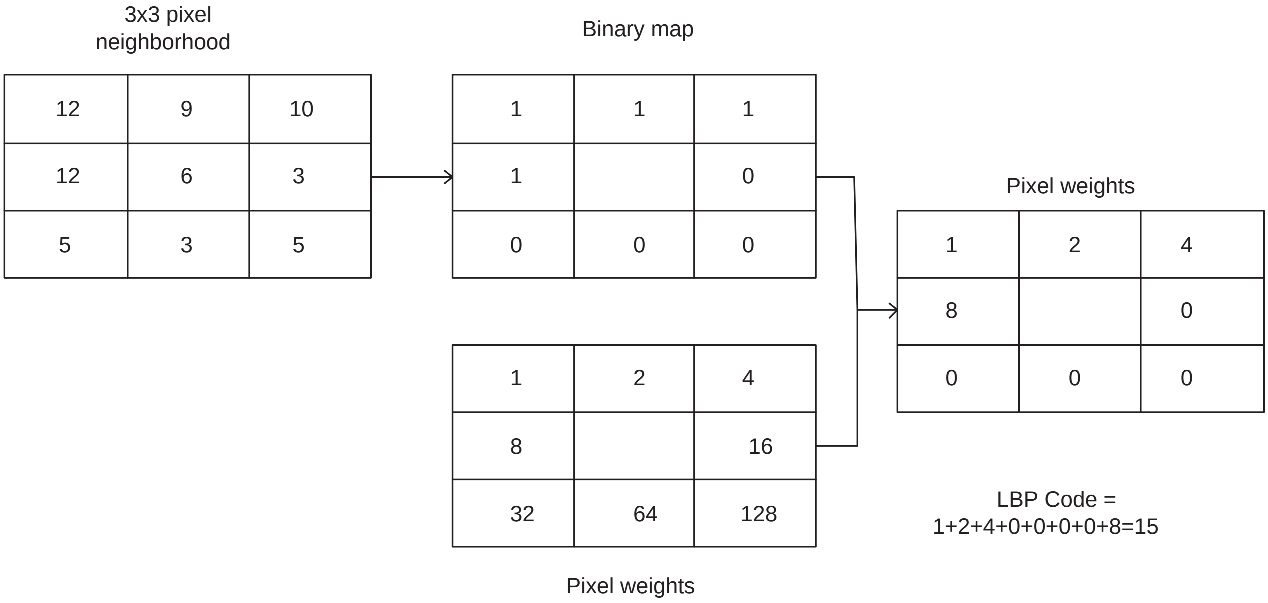


Figure 3.3: LBP basic operator

from two such blocks within the iris texture - first block being the one that is being analyzed and the second block is a potential candidate for having textural similarity to the first block. The SPLDH is computed by taking the histogram of the signed differences of the corresponding pixel positions of the two blocks. In order to limit the number of bins in the histogram, the pixel intensity differences are limited from $-Q_{max}$ to $+Q_{max}$. Finally, the entropy of the histogram is calculated in order to determine the importance of the second block with respect to the first block. The absolute position of the second block is stored in order to assign an index to the first block. N such block positions can be identified in order to form an index. Thus, the inter block relationship is analyzed both in the vicinity of a current block (local inter block relationship) and also at a distance (global inter block relationship). The experiment performed and the organization of the database utilizing the block based indexing scheme is given in the next section.

3.5 Experimental Evaluation

This section presents the details regarding the database used, experiments performed and results observed.

3.5.1 Database used

CASIA (Chinese Academy of Sciences' Institute of Automation) Iris Image Database V3.0 [66] was used for experimental evaluation of the iris indexing techniques proposed. CASIA-IrisV3 is divided into `CASIA-IrisV3-Interval`, `CASIA-IrisV3-Lamp` and `CASIA-IrisV3-Twins`. For the experimental evaluation of the indexing technique, the `CASIA-IrisV3-Interval` was used because this database contained images captured in two sessions, with at least one month interval. This database contained images with resolution 320×280 pixels captured from 249 subjects. Both the left and right eye images of a subject are present in the database. However all the users do not have same number of image samples per eye. Since the left and right eyes are assumed to be independent, they can be treated as distinct users. Only those users are selected which have at least six sample images per eye, in order to have three images for training and three images for testing. Thus, 143 users corresponding to the left eye and 139 users corresponding to the right eye, were identified. The training and test set each contain (randomly chosen) three samples per user. Figure 3.4 shows a few images present in the `CASIA-IrisV3-Interval` database.

3.5.2 Analysis of IrisCodes

The training dataset for analysis contains IrisCodes generated from the training images. Five experiments are done to analyze the indexing performance based on IrisCodes.

Experiment 1a: PCA is performed on the IrisCode image and dimensionality reduction is achieved according to the number of significant eigen vectors. An unsupervised clustering (K-means) is applied on the reduced dimension set and the training dataset is split into K classes. Each class has user numbers associated with it. For a given query image Q, the IrisCode is generated and PCA is performed to obtain the reduced dimension feature set. Thereafter, Q is allotted to a class according to its Euclidian distance from the centroid of the clusters generated from the training set. The classification result is reported by varying the eigen vectors and number of clusters. The result is presented in Table 3.1.

PCA Algorithm

The goal of PCA is to reduce the dimensionality of a given data set X of dimension M to a new data set Y of dimension L, such that $L << M$.

1. Suppose X is arranged as a set of N data vectors $(\vec{X_1}, \vec{X_2} \ldots \vec{X_N})$, where each observation

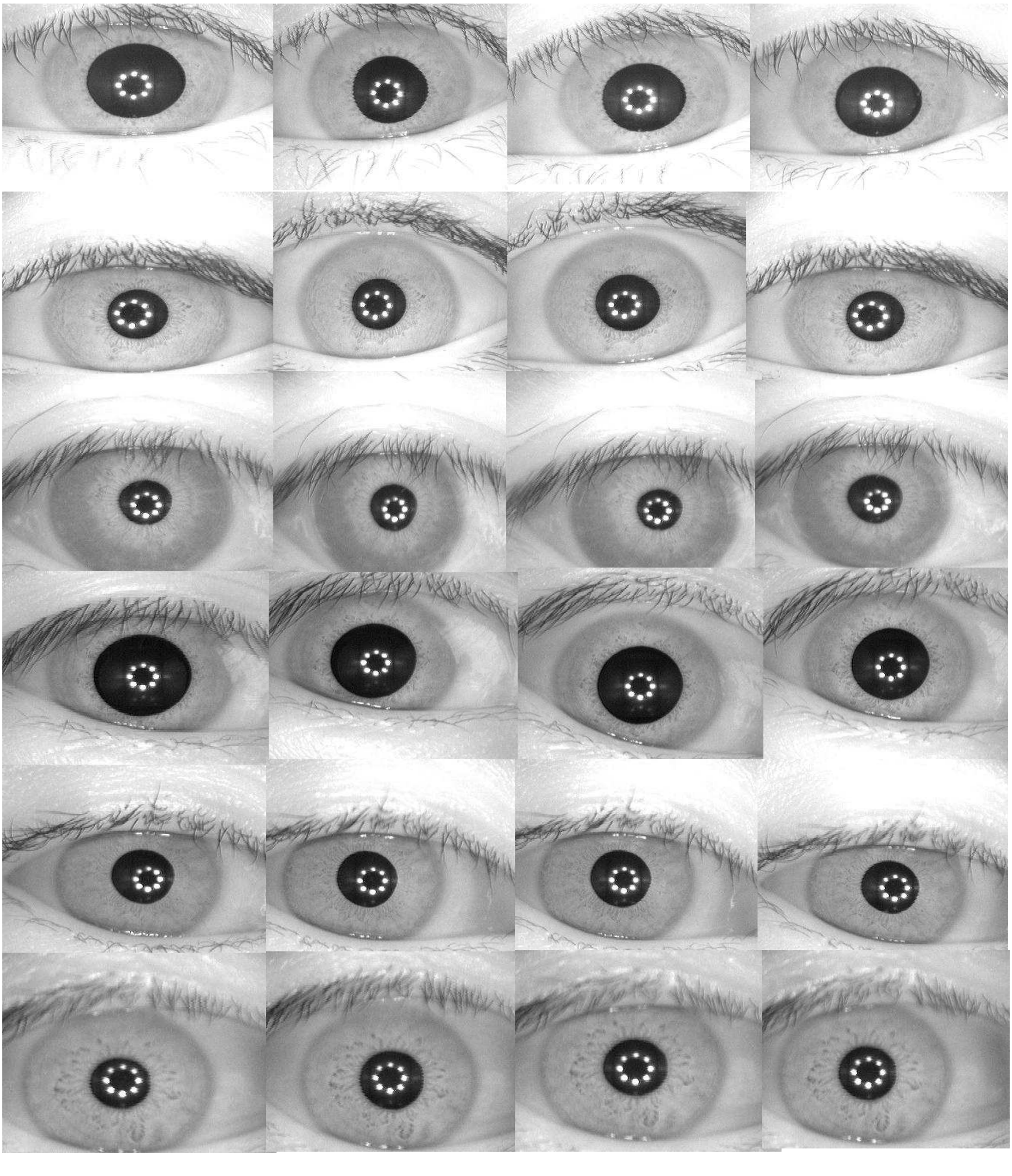


Figure 3.4: A snapshot of images present in the CASIA-IrisV3-Interval database. Each row contains multiple captures of the iris of a user.

has M variables. X can be written as column vectors $(\vec{X_1}, \vec{X_2} \dots \vec{X_N})$ where each column has M rows. Thus X is of the dimension $M \times N$.

2. Calculate the empirical mean along each dimension $m = 1 \dots M$. The empirical mean vector $\vec{u}$ is given by $u[m] = \Sigma_{n=1}^{\vec{N}} X[\vec{m}, n]/N$.

3. Store the mean subtracted data in $\vec{B}$ of dimensions $M \times N$. $\vec{B} = \vec{X} - \vec{u}.h$, where h is an identity matrix of size $1 \times N$.

4. Calculate the $M \times M$ covariance matrix $\vec{C}$ from the outer product of matrix $\vec{B}$ with itself.

5. Find the eigenvectors matrix $\vec{V}$ which diagonalizes the covariance matrix $\vec{C}$, such that $\vec{V}^{-1}\vec{C}\vec{V} = \vec{D}$, where $\vec{D}$ is the eigenvalues matrix of $\vec{C}$.

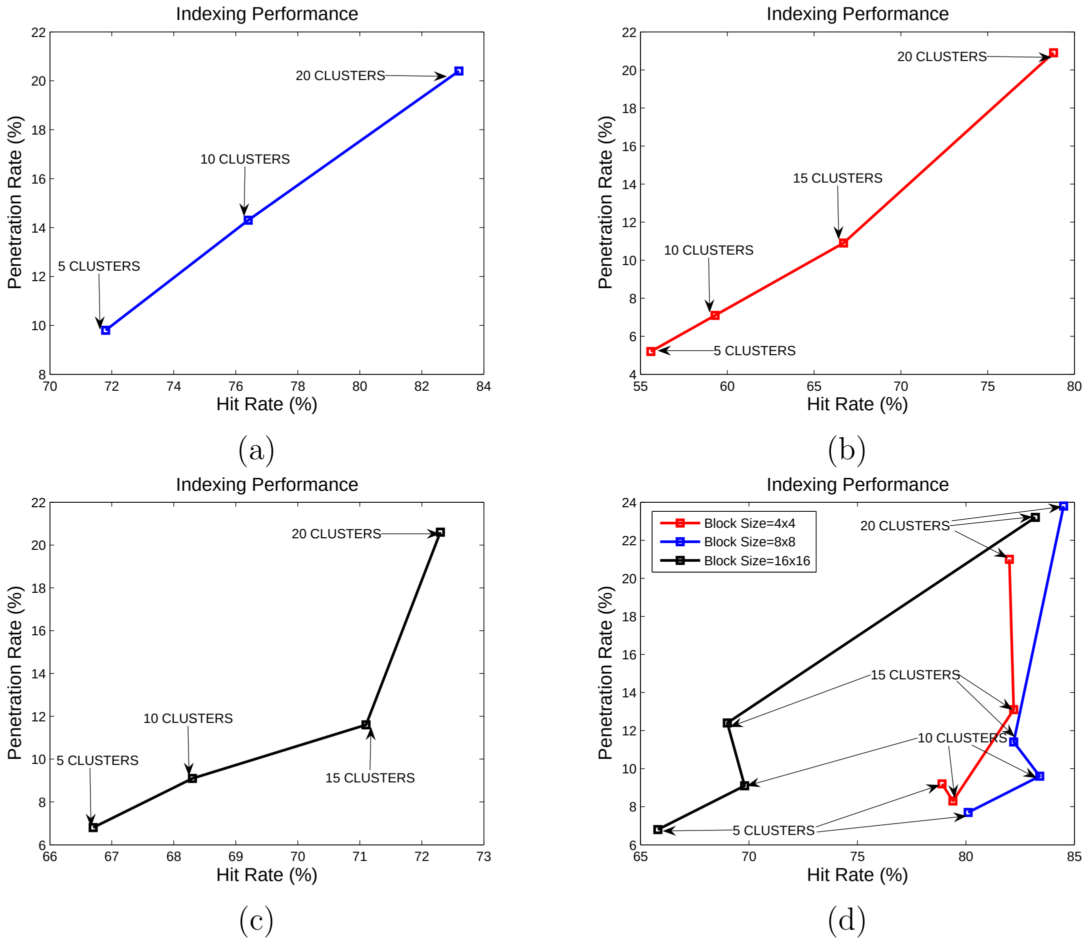
6. Rearrange the eigenvectors by sorting the columns of the eigenvector matrix $\vec{V}$ and eigenvalue matrix $\vec{D}$ in order of decreasing eigenvalue.

7. Select the top L eigenvectors to form the basis of the data. L could be either predefined or it could be found from the cumulative energy g. The cumulative energy for the m^th eigenvector is given as $g(m) = \Sigma_{q=1}^{m} D(p, q)$, where $p = q$ and $m = 1 \dots m$.

Table 3.1: Experiment 1a: PCA Analysis on entire IrisCode by varying the number of principal axis components (eigen vectors) and number of clusters

Number of Eigen Vectors	**Number of Clusters (K)**	**Hit Rate(%)**	**Penetration(%)**
40	5	82.9	32.2
80	5	84.3	39.6
120	5	80.6	23.8
40	10	74.5	18.1
80	10	78.1	19.6
120	10	77.4	16.3
40	20	71.8	8.6
80	20	71.6	13.4
120	20	68.9	11.0

Experiment 1b: Instead of varying the number of principal components, the 92% rule is used and the result is reported in Figure 3.5. It was observed that, 92% rule performed slightly



(a)

(b)

(c)

(d)

Figure 3.5: Indexing performance using IrisCodes (a) PCA analysis on IrisCode according to 92% rule. (b) Analysis of row based features extracted from the IrisCode considering only the target cluster. (c) Analysis of column based features extracted from the IrisCode considering only the target cluster. (d) PCA analysis of block based features extracted from the IrisCode.

better. The 92% rule suggests that the smallest value of L (see 3.5.2) is chosen such that the cumulative energy $g(m = L) \geq 92\%$.

Experiment 2: The feature extracted is the mean of each row in the IrisCode present in the training database. Rows of the IrisCode correspond to concentric circles in the iris. The feature vector size is limited by the number of rows in the IrisCode. Finally, unsupervised clustering (K-means) is performed on the extracted feature vectors, and the training dataset is split into K classes. For a given query image, the IrisCode is generated and the row based feature vector is extracted. The query image is then allotted a class according to its feature vector's euclidian distance from the cluster centroid. The user numbers associated with the cluster represent the reduced candidate set for matching. The indexing result using the row based features of the IrisCode is reported in Figure 3.5 . The classification accuracy can be improved by extending the search to the nearest neighboring class at the cost of increased penetration into the database.

Experiment 3: The feature extracted is the mean of each column in the IrisCode present in the training database. Columns of the IrisCode correspond to radial direction in the iris. The rest of the procedure is the same as in experiment 2. It is observed that row based features perform better than column based features.

Experiment 4: Each IrisCode in the training database is split into blocks of size m by m, where $m = 4, 8, 16$. The first order statistic (mean) computed for each block serves as a feature. The feature vector is extracted from all the IrisCode in the training database and unsupervised clustering (K-means) is performed on the feature vectors, and the training database is split into K classes. The classification performance is reported by varying values of K and m. The results for $m = 4, 8, 16$ with a single target cluster are shown in Figure 3.5. The results indicate that $m = 4$ and $m = 8$ perform better than $m = 16$. This can be attributed to the fact that as the block size increases the local information is lost.

3.5.3 Local Binary Pattern Analysis of Iris Texture

The training database for LBP analysis contains 3 normalized iris images per user from the database.

Experiment 5: Each image in the training database is divided into blocks of size m by m. A histogram is computed for all the blocks in the given image. The number of bins (b) in the histogram depends on the radius r, considered to generate the LBP. For example, for

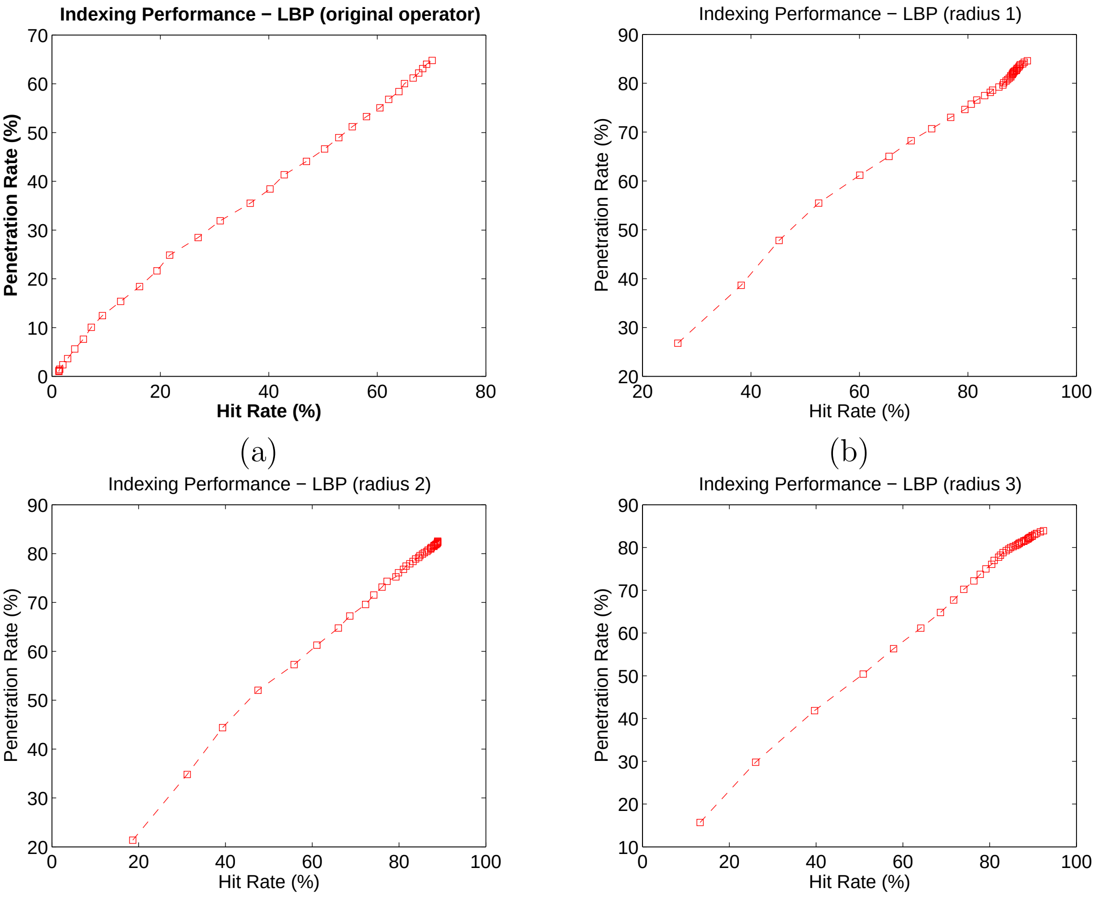
$r = 1$, $m = 3$ and $b = 10$, for $r = 2$, $m = 5$ and $b = 18$, for $r = 3$, $m = 7$ and $b = 24$. For a given r, the histogram bin heights for each image in the training database forms the feature vector. Unsupervised clustering (K-means) is performed on the feature vectors representing the training images, and an index space model is created which holds the centroid information of the clusters. For a given query image Q, a similar histogram is generated and the proximity of the histogram is computed with respect to the cluster centroids (generated in the index space) using Kullback Leibler distance (KL) distance. The KL distance is a natural distance function from a probability distribution p to a probability distribution q. For discrete probability distributions, $p = p_1, p_2, \dots p_n$ and $q = q_1, q_2, \dots q_n$, the KL distance is defined to be $KL(p,q) = \Sigma_{i=1}^{n} p_i \log_2 \frac{p_i}{q_i}$ The users whose KL distance lies below the threshold α are considered for matching. The accuracy of this scheme and the fraction of database penetrated by varying the size of the neighborhood (considered to generate LBP) are shown in Figure 3.6.

3.5.4 SPLDH Analysis of Iris Texture

Experiment 6: For a given segmented and normalized iris image I_i of size m_i by n_i in the training database, a cropped part of fixed size say $k \times l$ is extracted, where $1 \leq i \leq N_{tr}$ and N_{tr} is the total number of iris images in the training database. Let the cropped iris image be denoted by I_{ci}. Divide I_{ci} into blocks of size 4×4. Assuming k and l to be multiples of 4, the number of blocks in the cropped image is b_{tot}, where $b_{tot} = (k_i * l_i)/16$.

The next step involves finding the SPLDH for each block in I_{ci} corresponding to all other blocks in I_{ci}. The histogram is calculated for differences ranging from $-Q_{max}$ to $+Q_{max}$, where Q_{max} defines the maximum value of pixel level difference defined by the indexing scheme. In the experiments performed $Q_{max} = 15$. Different features can be extracted from the SPLDH (e.g., entropy, contrast, homogeneity etc.). In the experiments performed, entropy was used as the feature extracted from SPLDH. Let the feature extracted be represented by μ_b, where $1 \leq b \leq b_{tot}$. Thereafter the block positions are ranked according to ascending or descending order of the features (as desirable for a given feature). Finally, each block in I_{ci} corresponds to the top λ block positions. These block positions represent where the important texture information lies in the image. This procedure is carried out for all the blocks in the image I_{ci} to generate the 'intra block statistic', and for all the images in the training database.

For a given segmented, enhanced and cropped query iris image Q of size $k \times l$, the above



(a) (b)

(c) (d)

Figure 3.6: Local Binary Pattern analysis. (a) Indexing performance using original LBP operator. (b) Indexing performance using LBP operator considering pixels at radius 1. (c) Indexing performance using LBP operator considering pixels at radius 2. (d) Indexing performance using LBP operator considering pixels at radius 3.

procedure is repeated to generate the ’intra block statistic’.

In order to account for head movement while presenting the iris to the system, inherent noise present in the system (e.g.,sensor noise, modeling error etc.) and external noise (e.g. varying illumination), the top λ block positions corresponding to each block in the iris image is allowed a tolerance zone of a 8×8, i.e., the block position could move by one position in the north, south, east west, north-east, south-east, north-west or south-west direction.

Database Management and Retrieval:

Since the indexing scheme is based on block statistics, therefore the database management is done in such a away so as to incorporate indexing at the block level. Finally, the indexing results at the block level are construed. Accordingly a reduced number of users are identified for matching. This section explains the database management and the retrieval scheme using a simple illustration.

In order to explain the retrieval scheme, let us assume that each image can be divided into 16 blocks of equal width and height. Therefore, using the notations introduced in the previous section, $b_{tot} = 16$. For each of these blocks, the SPLDH is computed, features are extracted and on the basis of these features, the top λ block positions are identified. For sake of explanation, let us assume $\lambda = 2$. Then for each block an index BI_τ can be defined, where τ denotes the absolute block position in the image and $BI_\tau = p_1, .., p_\lambda$. Thus in the example being discussed, $BI_\tau = p_1, p_2$. For each block position, there could be a maximum of eight neighborhood positions which have to be considered in order to incorporate the ‘tolerance zone’. For instance, in case of $b_{tot} = 16$, the ‘tolerance zone’ of block position 1 includes block positions $2, 5, 6$. Similarly, the ’tolerance zone’ of block position 11 includes $6, 7, 8, 10, 12, 14, 15, 16$. Let us assume that for a given block in the image, the block index is $1, 2$, i.e. $p_1 = 1$ and $p_2 = 2$. The first level represented by squares (ranging from 1 to 16) in the Figure 3.8, denote p_1. According to the value of p_1, the first node represented by p_1 is further split into all possible block positions according to the ‘tolerance zone’. In the Figure 3.8, such possible block positions are represented by pentagons at level 1. Therefore, when p_1 is 1, the retrieval process will go into the node 1 at level 1 first, and then it will move into the sub node 1 (shown as hexagons in level 1). This level 1 sub node is further split according to p_2, into level 2 nodes. The level 2 node is represented by pentagons with index values β_1, β_2, where β_1 represents the level 1 node according to which splitting took place and β_2 represents the block positions represented by block positions in the tolerance zone of p_2. Since there are only two levels in the given example, the second level sub-nodes can also be called leaves. The leaves store information regarding user-ID's having the block index corresponding to index representing the leaf. In this example since the first level node splitting takes place according to $\beta_1 = 1$, therefore the second level splitting results in $\beta_2 = 2, 3, 5, 6, 7$. Therefore, for a two level indexing example, for a block with index $BI_\tau = 1, 2$ will look into all nodes with indexes θ_1, θ_2, where $\theta_1 = 1, 2, 5, 6$ and $\theta_2 = 1, 2, 3, 5, 6, 7$. Thus a total of 24 sub leaves have to be searched out of a possible $b_{tot}^2 = 16^2 = 256$ possible leaves. The information regarding the user ID's held by the leaf can be updated either using an incremental procedure, where there is no training data to begin with or there could be prior information generated from the training database regarding user id's embedded in the leaf nodes. The retrieval scheme is shown in Figure 3.8. An example of possible traversal path in the tree for a given block index $1, 2$ is demonstrated by the shaded blocks. After having visited all the possible leaves according to the block index, a list of user numbers is obtained corresponding to potential candidates for matching. A voting scheme can be employed to find out which users have a higher probability of a match.

The retrieval process is an online process. For a given query image the complexity of the algorithms depends on number of blocks in the iris image (b_{tot}) and the number of levels in the index (λ). The computational complexity of searching for the block positions and the corresponding prospective candidates for matching, in the data structure is $O(b_{tot} * \lambda * t_i)$, where t_i is the number of block positions in the tolerance zone of the block being analyzed.

It is observed that on an average 2.3% of the users in the CAISA-3 database occupy each leaf of the indexing data structure (tree). In the experiment performed $b_{tot} = 96$ and $\lambda = 2$.

In the experiment performed on CASIA-3 database, it was found that a hit rate of 84% is achieved by looking into 30% of the CASIA-3 database.

3.6 Analysis of iris indexing techniques

The first technique analyzes the IrisCode and performs unsupervised clustering on features extracted from the IrisCode. The hit rate and the penetration rate reported for a fixed number of cluster specified for unsupervised clustering is the average of all the hit rates and their

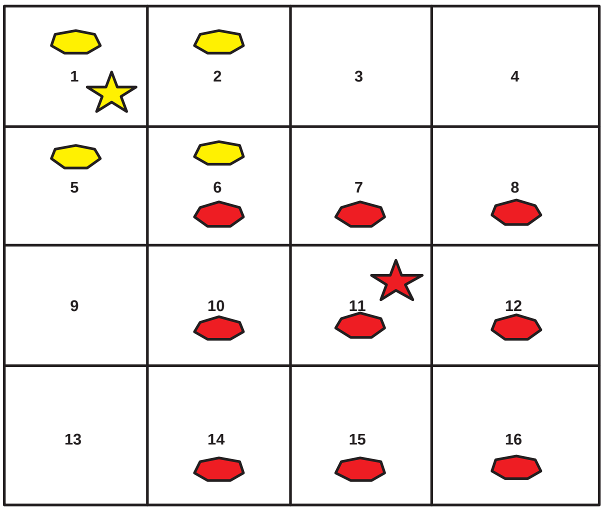


Figure 3.7: Block tolerance zone for a given block, marked by star.

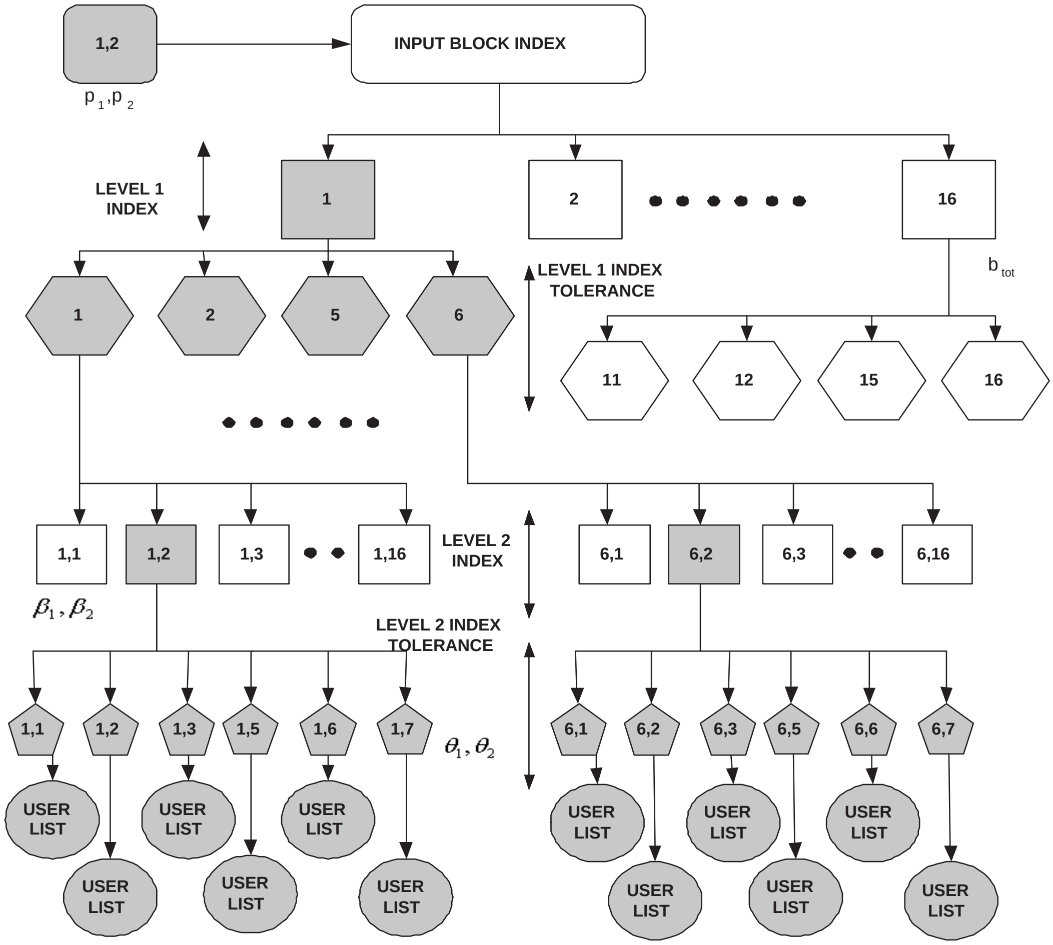


Figure 3.8: Block based indexing retrieval scheme for a given input block index

corresponding penetration rates observed. In the PCA based indexing of IrisCode the average penetration for a 80% hit rate is 17%, whereas for the indexing based on column-based statistics of the IrisCode, the average penetration rate for 80% hit rate is approximately 21%. However, when the indexing is based on the block based statistics of the iris code with block size=8x8, the average penetration for 80% hit rate is only 8%. Thus, it is observed that block based indexing has higher merit over other indexing techniques considered for the IrisCodes. However, the iris code analysis assumes that for each iris in the database encoding has been performed and an iris code is available. The first iris texture based indexing scheme uses block based Local Binary Pattern (LBP) in order to perform indexing. However it is observed that for a hit rate of 80%, the average penetration rate is around 70%, suggesting that block based LBP does not represent the iris texture well to be considered as a representative feature for indexing. Moreover since block based LBP represents the local contrast, it is implied from its poor performance that the texture based indexing technique has to be not just block based but should also inherently capture the interblock relationship, thus resulting in the third technique - indexing using block based SPLDH. It was observed that for a hit rate of 84%, the average penetration rate is 30%. Though this technique does not perform as well as the indexing technique using block based features of IrisCode, the performance is far better than the indexing technique based on LBP.

3.7 Summary and Future Work

Three iris indexing algorithms have been proposed based on analysis of IrisCode, LBP analysis of iris texture and SPLDH analysis of iris texture. The ability of the SPLDH based algorithm to skip the iris enhancement and encoding stages for iris indexing, along with the resulting improvement in performance over the LBP based algorithm makes it attractive for future research. However, it should be noted that the LBP based algorithm performs poorly in its basic implementation. Other versions of LBP, might lead to better indexing performance [65]. Since the indexing scheme is block based, there is an inherent sense of parallelism, where each block can be indexed independently and the result consolidated thereafter. In addition, since the retrieval scheme is based on block positions, the tree like structure for database management is fixed, thus making incremental addition of new users to the tree simpler. The future work may include analysis of features other than entropy from the SPLDH. Since this is the first time iris indexing is being addressed, this work attempts to highlight a new direction of research that would be helpful for large scale authentication systems in the future.

Chapter 4

Summary and Future Work

In this thesis, indexing techniques have been proposed for fingerprint and iris. While the fingerprint indexing technique was based on Delaunay triangulation of minutiae points, the iris indexing techniques were based on the analysis of IrisCodes and iris texture.

4.1 Fingerprint Indexing

In the proposed model for fingerprint indexing, it was demonstrated that augmenting ridge based features with minutiae based features enhances fingerprint indexing performance. The proposed model was tested on FVC2002 and FVC2004 databases. The sensors used in these databases have comparable resolution. Therefore, the impact of ridge based features on the indexing performance in databases where images have been acquired using sensors of different resolution, is not known and will constitute future research. It will also be interesting to analyze the indexing performance where the database contains multiple samples of a user from different types of sensors. The simple ratios of coefficients in ridge based features improves the performance of indexing when used in conjunction with minutiae information. However, such ridge based features do not perform well when used alone. Thus, the minutiae based features can be considered as the most discriminating features, whereas the ridge based features further enhance the discriminating capability of the minutiae based features.

The index space model generated has 600 cluster definitions (the value of K is 600 in the K-means algorithm). In the scheme is proposed, the number of clusters is fixed. It should be noted that the $K = 600$ was chosen after analyzing some other values(e.g.; $K = 100$, 300 and

900. It was observed that $K = 600$ gives better performance over other values. The effect of varying the number of clusters, on the indexing performance might constitute future work.

4.2 Iris Indexing

The iris indexing, based on Signed Pixel level Difference Histogram (SPLDH) analysis of iris texture, gave interesting results. The ability of the scheme to identify relevant texture and extract information regarding distribution of such texture lends to the block based indexing. The inter-block interaction is used to define an index for the iris database. It is easy to add, remove or update the information using the block based data structure. However, maintaining such a data structure might become cumbersome for large databases. The block based model proposed tries to look at iris indexing from the texture point of view. Though, the present iris matching techniques are fast and accurate, they typically use IrisCodes for matching. This means that all images in the database have to undergo localization, enhancement, unwrapping and encoding. However with the proposed scheme, the localized and normalized iris image itself can be used for indexing. In future, other indexing techniques should be researched in order to index unsegmented iris, since segmentation is computationally intensive. In case, the iris recognition system extracts colored iris images, color can be used as a feature for iris indexing. It should be noted that the performance of the iris indexing scheme proposed is dependent on the localization algorithm. The indexing performance would be low for iris image which do not have prominent iris texture. It will be interesting to observe the performance of the indexing scheme when images of high resolutions are available.

It should be noted that in both fingerprint and iris indexing schemes, the matcher is given a list of possible candidates. The features extracted by the matcher or the order in which the matcher should use the list of candidates can be confirmed only when the matcher is integrated with the indexing model. In future, research into such integration methodologies will result in more accurate matchers.

Currently, the indexing scheme considers a closed set identification; the users whose samples have been used to generate the training database, comprise the test database as well (with different samples). Thus, a novel user (whose samples do not exist in the training database), should be ideally rejected. However, the indexing schemes proposed are so designed, that the

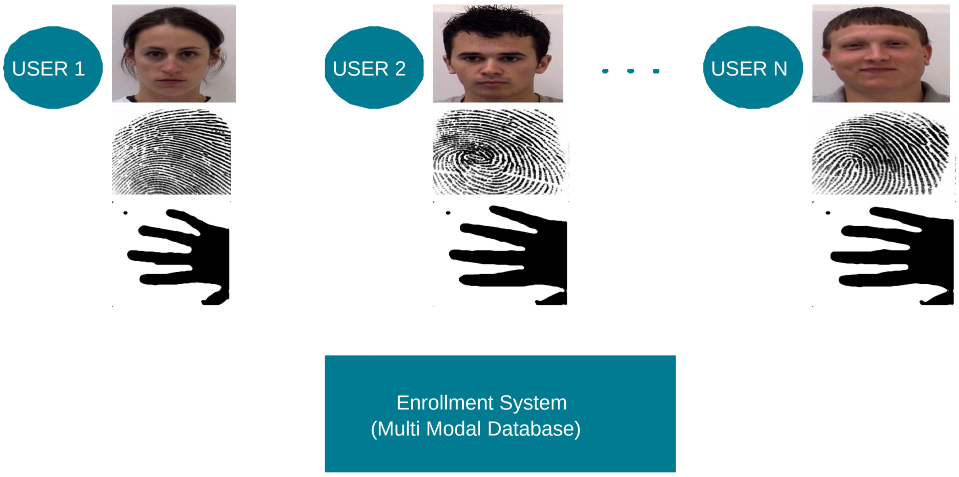


Figure 4.1: Multimodal enrollment process.

novel user will cause the indexing technique to produce a list of target users. In order to handle open set identification, the indexing model should incorporate the facility to reject fingerprints. This can be done depending on the application which the indexing is supposed to serve.

Analysis of the sensitivity and selectivity of the features used for indexing might comprise future work.

4.3 Multimodal Indexing

With the growing deployment of biometric systems based on fingerprint, face, iris, hand etc., there is a trend towards adopting multi-modal biometric systems. The advantage of multibiometric [6] system lies not only in the fact that decision making becomes more reliable (due to decision level or feature level fusion of multiple biometrics), but also that individuals who cannot use a particular biometric (e.g., loss of hand due to personal injuries or dry skin condition leading to unacceptable fingerprint image) can still use the system using some other biometric. In the context of large scale identification, filtering can be performed by indexing sequentially using different biometrics, thus reducing the number of candidates for matching. This section suggests a possible design for a multimodal indexing system.

Multimodal boimetric indexing would include the following steps:

1. Multimodal enrollment: In this stage, the user enrolls his biometrics into the system. For example, in a tri-modal system (see Figure 4.1) based on face, fingerprint and hand geom-

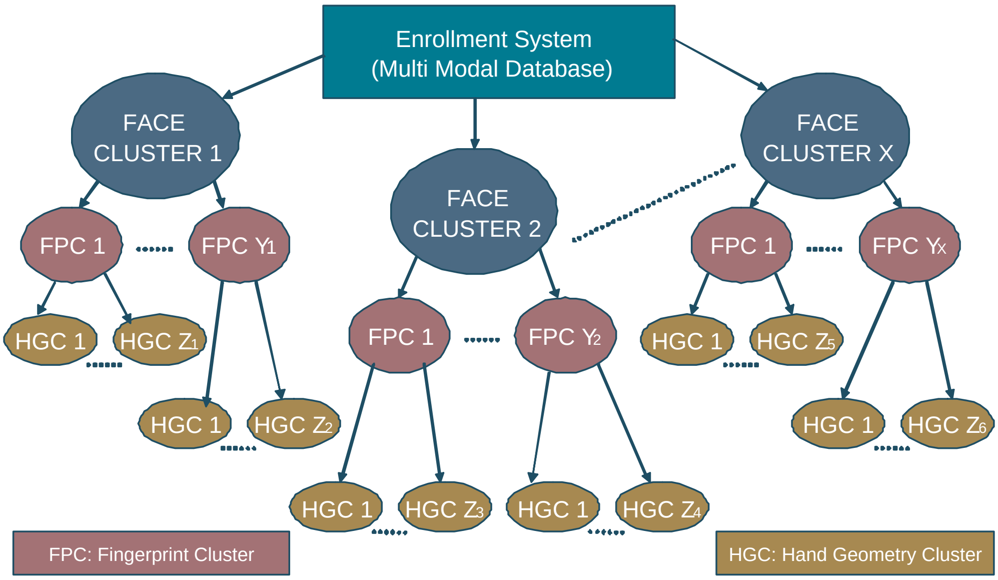


Figure 4.2: Design of the multimodal index space.

etry, the user will enroll all the three biometrics (unless a preexisting condition prevents doing so).

2. Designing the multimodal index space model: After enrollment, the user data is arranged by sequential clustering of the face, fingerprint and hand biometrics. This sequence will be dictated by how well the indexing scheme performs for individual biometrics. One possible configuration of the database is shown in the Figure 4.2.

3. Retrieval: When a query user provides his multiple biometrics to the system (face, fingerprint and hand in this case; see Figure 4.3), the indexing system searches the face cluster first and eliminates certain users based on face indexing (see figure 4.4). Thereafter, fingerprint indexing is performed on the remaining users and certain users are further eliminated from the list of prospective candidates (see figure 4.5). Finally, in the third and final level of the tri-modal indexing scheme, the hand geometry index space is analyzed in order to eliminate users. The final list of prospective candidates is shown in figure 4.6.

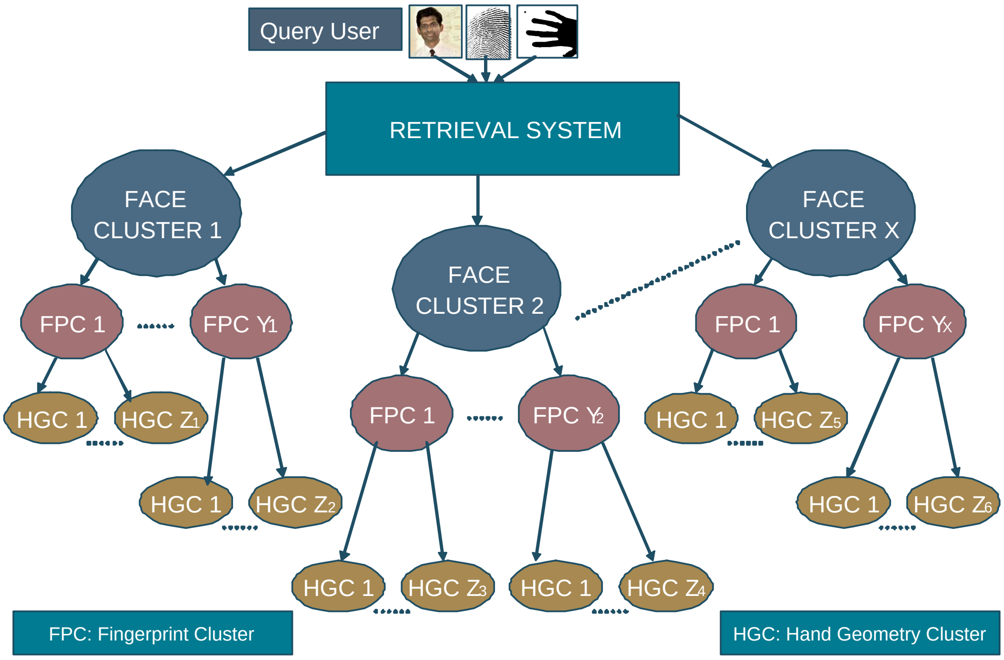


Figure 4.3: Query user presenting the biometrics to the retrieval system.

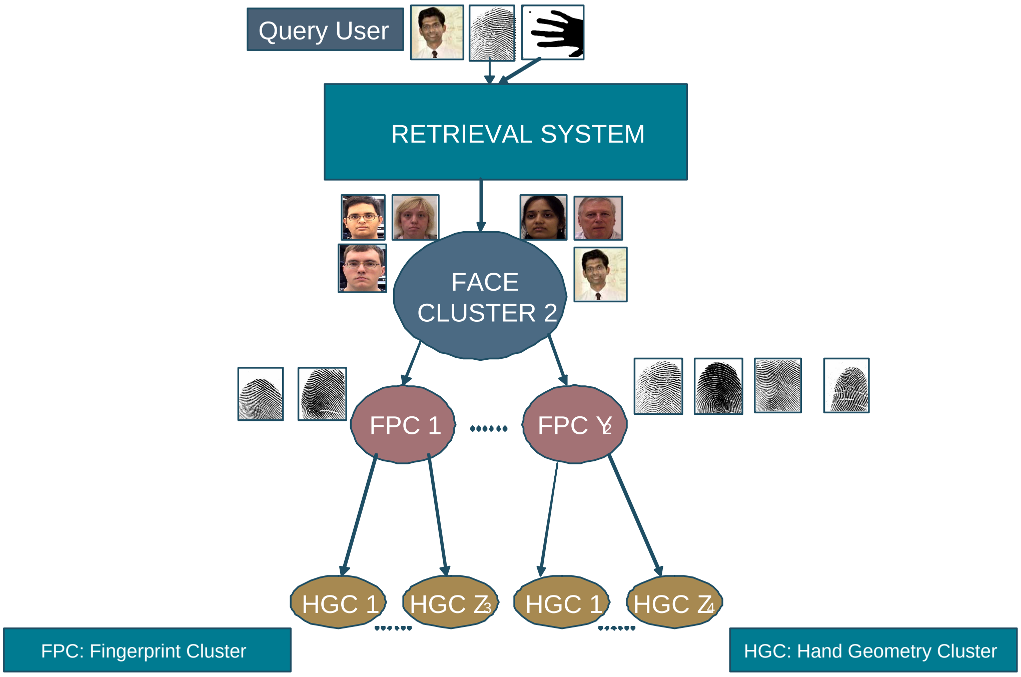


Figure 4.4: Indexing system eliminates users based on searching the face index space (clusters).

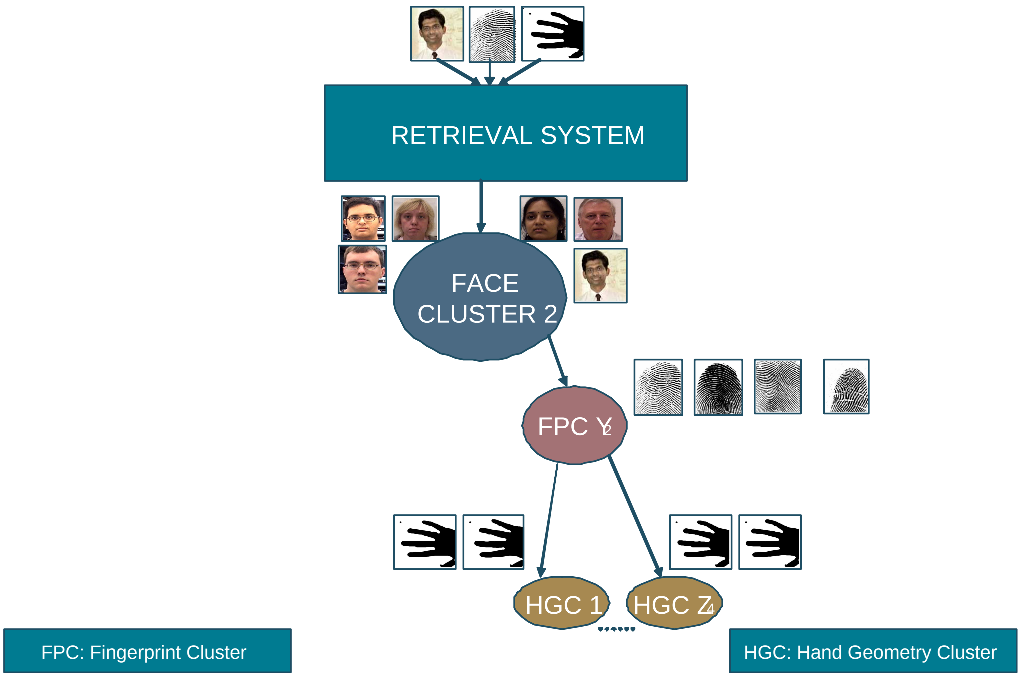


Figure 4.5: Indexing system eliminates users based on searching the fingerprint index space (clusters) of the users left after elimination based on face index space.

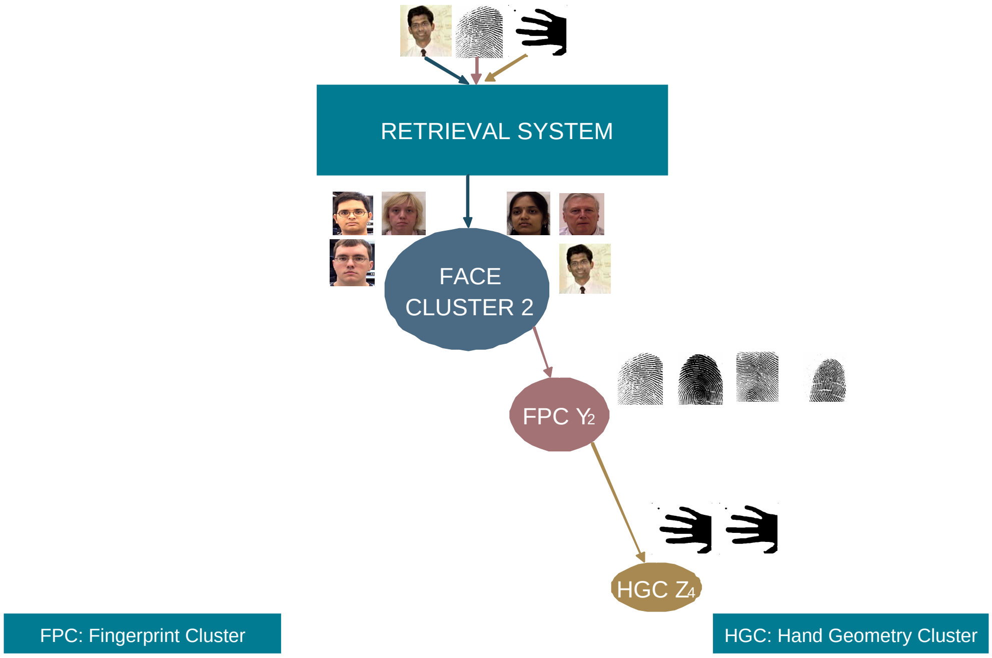


Figure 4.6: Reduced data set after eliminating users sequentially from the face, fingerprint and hand geometry index space.

References

[1] A. Jain, A. Ross, and S. Prabhakar, “An introduction to biometric recognition,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 1, pp. 4–20, January 2004.

[2] “International biometric group: Biometrics market and industry report 2007-2012.” [Online]. Available: http://www.biometricgroup.com/reports/public/market_report.html

[3] K. Rhodes, “Aviation security - challenges in using biometric technologies,” *Testimony Before the Subcommittee on Aviation, Committee on Transportation and Infrastructure, House of Representatives*, May 2004.

[4] G. Annas, “HIPAA regulations - a new era of medical record privacy,” *The New England Journal of Medicine*, vol. 348, no. 15, pp. 1486–1490, April 2003.

[5] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, *Handbook of fingerprint recognition*, 1st ed. New York, Berlin Heidelberg: Springer-Verlag, 2003.

[6] A. Ross, A. Jain, and K. Nandakumar, *Handbook of Multibiometrics.* Springer, 2006.

[7] R. Derakhshani, “Spoof-proofing fingerprint systems using evolutionary time delay neural networks,” *IEEE International Conference on Computational Intelligence for Homeland Security and Perosnal Safety*, April 2005.

[8] Y. Moon, J. Chen, K. Chan, K. So, and K. Woo, “Wavelet based fingerprint liveness detection,” *IEEE Electronic Letters*, vol. 41, no. 20, pp. 1–2, September 2005.

[9] J. Daugman, “High confidence visual recognition of persons by a test of statistical independence,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1148–1160, November 1993.

[10] ——, “Iris recognition,” *American Scientist*, vol. 89, pp. 326–333, July 2001.

[11] M. Lynch, “The similarity index and dna fingerprinting,” *Molecular Biology and Evolution*, vol. 7, no. 5, pp. 478–484, 1990.

[12] J. Daugman, “Anti-spoofing liveness detection.” [Online]. Available: http://www.cl.cam.ac.uk/~jgd1000/countermeasures.pdf

[13] A. Abhyankar and S. Schuckers, “Fingerprint liveness detection using local ridge frequencies and multiresolution texture analysis techniques,” *IEEE International Conference on Image Processing*, pp. 321–324, October 2006.

[14] S. Liu and M. Silverman, "A practical guide to biometric security technology," *IT Professional*, vol. 3, no. 1, pp. 27–32, January 2001.

[15] Wikepedia, "Iris anatomy." [Online]. Available: http://en.wikipedia.org/wiki/Iris

[16] S. Shah, "Enhanced iris recognition: Algorithms for segmentation, matching and synthesis," Master's thesis, West Virginia Univeristy, 2006.

[17] K. Bae, S. Noh, and J. Kim, "Iris feature extraction using independent component analysis," *Proceedings 4th International Conference on Audio and Video Based Biometric Person Authentication (ABVPA)*, pp. 838–844, 2003.

[18] V. Dorairaj, N. Schmid, and G. Fahmy, "Performance evaluation of iris based recognition system implementing PCA and ICA encoding techniques," *Proceedings of SPIE Conference on Biometric Technology for Human Identification III*, vol. 5779, pp. 51–58, April 2005.

[19] J. Daugman, "Complete discrete 2-d gabor transforms by neutral networks for image analysis and compression," *IEEE Transactions on Acoustic, Speech and Signal Processing*, vol. 36, no. 7, pp. 1169–1179, July 1998.

[20] ——, "Demodulation by complex-valued wavelets for stochastic pattern recognition," *International Journal of Wavelets, Multi-resolution and Information Processing*, vol. 1, no. 1, pp. 1–17, January 2003.

[21] W. Boles and B. Boashash, "A human identification technique using images of the iris and wavelet transform," *IEEE Transactions on Signal Processing*, vol. 46, no. 4, pp. 1185–1188, April 1998.

[22] R. Wildes, J. Asmuth, G. Green, S. Hsu, R. Kolczynski, J. Matey, and M. S., "A system for automated iris recognition," *Proceedings of the Second IEEE Workshop on Applications of Computer Vision*, pp. 121–128, 1994.

[23] R. Wildes, "Iris recognition: An emerging biometric technology," *Proceedings of the IEEE*, vol. 85, no. 9, pp. 1348–1363, September 1997.

[24] L. Ma, Y. Wang, and T. Tan, "Iris recognition using circular symmetric filters," *Proceedings of 16th International Conference on Pattern Recognition (ICPR)*, vol. 2, pp. 805–808, August 2002.

[25] L. Ma, T. Tan, Y. Wang, and D. Zhang, "Personal identification based on iris texture analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 12, pp. 1519–1533, December 2003.

[26] J. Huang, L. Ma, Y. Wang, and T. Tan, "Iris model based on local orientation description," *Proceedings of Asian Conference on Computer Vision*, pp. 954–959, April 2004.

[27] J. Huang, Y. Wang, T. Tan, and J. Cui, "A new iris segmentation method for recognition," *Proceedings of 17th International Conference on Pattern Recognition (ICPR)*, vol. 3, pp. 23–26, August 2004.

[28] S. Lim, K. Lee, O. Byeon, and T. Kim, "Efficient iris recognition through improvement of feature vector and classifier," *Journal of Electronics and Telecommunication Research Institute*, vol. 33, no. 2, pp. 61–70, June 2001.

[29] X. Yuan and P. Shi, "Iris feature extraction using 2d phase congruency," *Third International Conference on Information Technology and Applications (ICITA)*, vol. 33, pp. 437–441, July 2005.

[30] L. Masek, "Recognition of human iris patterns for biometric identification," *Project Report, The School of Computer Science and Software Engineering*, 2003.

[31] P. Kovesi, "Matlab functions for computer vision and image analysis." [Online]. Available: Available:http://www.cs.uwa.edu.au/~pk/Research/MatlabFns/index.html

[32] J. Thornton, M. Savvides, and B. Vijaya Kumar, "Robust iris recognition using advanced correlation techniques," *Second International Conference on Image Analysis and Recognition (ICIAR)*, vol. 3656, pp. 1098–1105, September 2005.

[33] A. Abhyankar and S. Schuckers, "Active shape model for effective iris segmentation," *Proceedings of SPIE Conference on Biometric Technology for Human Identification III*, vol. 6202, April 2006.

[34] A. Abhyankar, L. Hornak, and S. Schuckers, "Bi-orthogonal wavelet based iris recognition," *Proceedings of SPIE Conference on Biometric Technology for Human Identification III*, pp. 59–67, April 2005.

[35] L. Ma, T. Tan, and Y. Wang, "Efficient iris recognition by charecterizing key local variations," *IEEE Transactions on Image Processing*, vol. 13, no. 6, pp. 739–750, June 2004.

[36] Z. Sun, T. Tan, and X. Qiu, "Graph matching iris image blocks with local binary pattern," *International Conference on Biometric Authentication (ICBA)*, pp. 366–372, January 2006.

[37] L. Pau and P. Wang, *The Handbook of Pattern Recognition and Computer Vision*, 2nd ed. World Scientific Publishing Co., 1998.

[38] A. Jain, S. Dass, and K. Nandakumar, "Soft biometric traits for personal recognition systems," *Proceedings of International conference on Biometric Authentication*, pp. 731–738, July 2004.

[39] A. Ross and R. Nadgir, "A calibration model for fingerprint sensor interoperability," *Proceedings of SPIE Conference on Biometric Technology for Human Identification III*, vol. 62020B, pp. 1–8, April 2006.

[40] A. Ross and R. Mukherjee, "Augmenting ridge curves with minutiae triplets for fingerprint indexing," *Proceedings of SPIE Conference on Biometric Technology for Human Identification IV*, vol. 6539, April 2007.

[41] K. Rao and K. Black, "Type classification of fingerprint: A syntactic approach," *IEEE Transactions on PAMI*, pp. 223–231, 1980.

[42] B. Moayer and K. Fu, "A syntactic approach to fingerprint pattern recognition," *Pattern Recognition*, vol. 7, pp. 1–23, 1975.

[43] M. Chong, T. Ngee, L. Jun, and R. Gay, "Geometric framework for fingerprint image classification," *Pattern Recognition*, vol. 30, no. 9, pp. 1475–1488, 1997.

[44] A. Senior, "A hidden Markov model fingerprint classifier," *Asilomar Conf. Signals, Systems and Computers*, vol. 1, pp. 306–310, November 1997.

[45] A. Jain, S. Prabhakar, and S. Hong, "A multi channel approach to fingerprint classification," *IEEE Transactions on PAMI*, vol. 21, no. 4, pp. 348–359, April 1999.

[46] K. Karu and A. Jain, "Fingerprint classification," *Pattern Recognition*, vol. 29, no. 3, pp. 389–404, 1996.

[47] T. Liu, G. Zhu, C. Zhang, and P. Hao, "Fingerprint indexing based on singular points," *International Conference on Image Processing*, vol. 3, pp. 293–296, September 2005.

[48] B. Bhanu and X. Tan, "Fingerprint indexing based on novel features of minutiae triplet," *IEEE Transactions on PAMI*, vol. 25, no. 5, pp. 616–622, May 2003.

[49] R. Germain, A. Califano, and S. Colville, "Fingerprint matching using transformation parameter clustering," *IEEE Computational Science and Engineering*, vol. 4, no. 4, pp. 42–49, October 1997.

[50] G. Bebis, T. Deaconu, and M. Georgiopoulos, "Fingerprint identification using delaunay triangulation," *IEEE International Conference on Intelligence, Information, and Systems (ICIIS)*, pp. 452–459, 1999.

[51] U. Halici and G. Ongun, "Fingerprint classification through self organizing feature maps modified to treat uncertainties," *Proceedings of the IEEE*, vol. 84, no. 10, pp. 1497–1512, October 1996.

[52] A. Califano and I. Rigoutsos, "Flash: A fast look-up algorithm for string homology," *Proceedings CVPR'93*, pp. 353–359, June 1993.

[53] H. Wolfson and I. Rigoutsos, "Geometric hashing: an overview," *IEEE Computational Science and Engineering*, vol. 4, no. 4, pp. 10–21, October 1997.

[54] N. K. Ratha, K. Karu, S. Chen, and A. K. Jain, "A real-time matching system for large fingerprint databases," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 799–813, 1996.

[55] X. Liang, T. Asano, and A. Bishnu, "Distorted fingerprint indexing using minutiae detail and delaunay triangle," *3rd International Symposium on Voronoi Diagrams in Science and Engineering (ISVD'06)*, pp. 217–223, 2006.

[56] M. Berg, M. Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry (Algorithms and Application).* Berlin Heidelberg: Springer-Verlag, 1997.

[57] S. Fortune, "A sweepline algorithm for voronoi diagrams," *Algorithmica*, vol. 2, no. 2, pp. 153–174, 1987.

[58] D. Lee and B. Schachter, "Two algorithms for constructing a delaunay triangulation," *International Journal of Computer and Information Sciences*, vol. 9, no. 3, pp. 219–242, 1980.

[59] C. Lawson, *Software for C Surface Interpolation*, mathematical software iii ed. New York: Academic Press, 1977.

[60] J. Shewchuk, “Triangle: Engineering a 2d quality mesh generator and delaunay triangulator,” *Applied Computational Geometry: Towards Geometric Engineering*, vol. 1148, pp. 203–222, 1996.

[61] C. B. Barber, D. Dobkin, and H. Huhdanpaa, “The quickhull algorithm for convex hulls,” *ACM Transactions on Mathematical Software*, vol. 22, no. 4, pp. 469–483, 1996.

[62] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. Wiley-Interscience, November 2000.

[63] L. I. Smith, “A tutorial on principal component analysis,” pp. 1–27, February 2002. [Online]. Available: http://csnet.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf

[64] S. Makthal, “Analysis and synthesis of iris images,” Master’s thesis, West Virginia Univeristy, 2005.

[65] T. Maenpaa, “The local binary pattern approach to texture analysis - extensions and applications,” *Academic Dissertation, Infotech Oulu and Department of Electrical and Information Engineering, University of Oulu*, pp. 1–80, 2003.

[66] “Casia iris image database ver 3.0.” [Online]. Available: http://www.cbsr.ia.ac.cn/Databases.htm