

# Indexing Biometric Databases using Pyramid Technique

Amit Mhatre, Sharat Chikkerur and Venu Govindaraju

Center for Unified Biometrics and Sensors (CUBS),  
University at Buffalo, New York, U.S.A  
<http://www.cubs.buffalo.edu>  
{ajmhatre,ssc5,govind}@buffalo.edu

**Abstract.** Biometric identification has emerged as a reliable means of controlling access to both physical and virtual spaces. In spite of the rapid proliferation of large-scale databases, the research has thus far been focused only on accuracy within small databases. However, as the size of the database increases, not only does the response time deteriorate, but so does the accuracy of the system. Thus for larger applications it is essential to prune the database to a smaller fraction which would not only ensure higher speeds, but also aid in achieving higher accuracy. Unlike structured information such as text or numeric data that can be sorted, biometric data does not have a natural sorting order making indexing the biometric database a challenging problem. In this paper we show the efficacy of indexing hand geometry biometric using the Pyramid Technique, to reduce the search space to just 8.86% of the entire database, while maintaining a 0% FRR.

## 1 Introduction

In an increasingly digital world, reliable personal authentication is an important human computer interface activity. Biometrics such as fingerprints, face and voice verification are gaining industrial, government and citizen acceptance. The US-VISIT program uses biometric systems to enforce homeland and border security. Governments around the world are adopting biometric authentication to implement National ID and voter registration schemes [1]. FBI maintains national criminal and civilian biometric databases for law enforcement. In spite of the rapid proliferation of large-scale databases, the research community has thus far focused only on accuracy with small databases while neglecting the scalability and speed issues important to large database applications. If biometric systems have to truly replace the existing authentication systems, the response time, search and retrieval efficiency become important factors in addition to accuracy.

Traditional databases index the records in an alphabetical or numeric order for efficient retrieval. In biometric templates, there is no natural order by which one can sort the biometric records, making indexing a challenging problem. We propose guiding the search in biometric databases by using the Pyramid technique for indexing biometric databases.

## 2 Problem Definition

It has been shown that the number of false-positives in a biometric identification system grows geometrically with the size of the database [2]. If FRR and FAR indicate the false accept and reject rates during verification, then  $FRR_N$  and  $FAR_N$ , the rates of false rejects and accepts in the identification mode are given by

$$FAR_N = 1 - (1 - FAR)^N \quad (1)$$

$$\approx N \times FAR \quad (2)$$

$$FRR_N = FRR \quad (3)$$

$$\text{No. of false accepts} = N \times (FAR_N) \approx N^2 \times FAR \quad (4)$$

There are two approaches in which we can attempt to reduce the error of such an identification system: (i) By reducing the FAR of the matching algorithm (ii) By reducing the search space (N) during identification. The FAR of a modality is limited by the recognition algorithm and cannot be reduced indefinitely. Thus, the accuracy and speed of a biometric identification system can be improved by reducing the number of records against which matching is performed. The effects of reducing the search space during identification are obtained by mathematical analysis. Assume that we are able to reduce the search space to a fraction ( $P_{SYS}$ ) of the entire database. Then the resulting FAR, FRR values and total number of false accepts is given by

$$FAR_{P_{SYS}N} = 1 - (1 - FAR)^{P_{SYS} \times N} \quad (5)$$

$$\approx (P_{SYS} \times N) \times FAR \quad (6)$$

$$FRR_{P_{SYS}N} = FRR \quad (7)$$

To reduce the search space, we require a certain classification, partitioning or indexing of the database. There exist well-established procedures such as Henry classification system [3] to classify fingerprint records based on the ridge patterns such as ‘whorl’, ‘loop’, ‘arches’ and ‘tented arch’. However, the problem with the existing Henry Classification system is that it is often found that the distribution of the fingerprints within each class is not uniform. It has been observed [4] that 2 of the classes constitute nearly 65% of the population. Also, binning a fingerprint into one of these classes is a non-trivial task with the best classification system having FRR of 5%. We would want to have a 0% FRR for an identification task, to prevent imposters sneaking through the system. Besides, apart from fingerprint, no such natural classification exists for the other biometrics.

Thus in presence of such a scenario, it is imperative that we go in for a sophisticated technique of partitioning or indexing the database to reduce the search space for not only improving the accuracy of the system, but also to ensure an efficient deployment of biometrics for real-time applications.

### 3 Previous Work

Classifying fingerprints into the Henry classes has been tried by Jain et al in [5], yielding a system with 12.4% FRR. A similar work by Ratha et al in [6] yielded a FRR of 10% with search space pruned to 25% of the original database. For a real-time application such as on an airport, with a rejection rate of 10%, 1 out of every 10 persons would be falsely rejected and would have to be manually inspected. In an experiment conducted by Cappelli et al [4] on NIST Special Database – 4, it was shown that the distribution of Fingerprint population was non-uniform with 2 of the 5 Henry classes they considered holding nearly 65% of the population. Also, such natural classifications are not evident in other biometrics such as hand geometry.

The study of effect of binning was performed in [8] and it was demonstrated that the search space could be reduced to approximately 5% of the original database while keeping the FRR at 0% by using a parallel combination of hand geometry and signature biometrics. The data was clustered using the K-means clustering algorithm. The test template is then associated with the clusters it is closest to, with the new search space being the templates within these closest clusters. However, the binning approach has the shortcoming of having to re-partition the entire database on addition of new samples to it, which is an extremely time intensive process. Thus binning/clustering systems are useful only in cases of non-changing or static databases.

### 4 Indexing Techniques

Indexing the database implies a logical partitioning of the data space. In this approach of search-space reduction we make use of a Tree structure for organizing the data, such that each of the leaf nodes store one or more biometric templates. Thus given a test template, only the templates having similar index values would be considered as the search space, or in the case of range search, only the fraction of the database lying within the range of the indexes of the test template would be the new search space. The reduced search space could then be passed on for actual identification using a slower but a more reliable biometric such as fingerprint. With Biometric data being inherently multi-dimensional, it is indispensable that the indexing technique support multi-dimensional data. Besides supporting high-dimensional data, there are several additional requirements that the indexing technique must satisfy for lending itself to the biometric domain, such as the following:

1. Tree should be approximately balanced

A balanced Tree will ensure that the time needed to reach to every leaf node is nearly the same among all the possible paths from the root to the leaf.

2. Tree should be dynamic

The Tree structure used must be dynamic with respect to frequent insertions and not so frequent deletions. Since we refer to an online real-time application, such as an airport deployment, the number of new enrollments into the database can be expected

to be ever increasing. The Tree should not lose its original properties of being approximately height balanced or being non-skewed, with these online operations of insertions and deletions.

### 3. Tree should support range queries

Since the feature values measured for a biometric template is dependent on the data acquisition device and the feature extraction algorithm, we would want to provide a tolerance to each of the features measured, based on the intra-user variance observed for the feature. Thus in effect we would place the template into a bounding box, such that the extent along each dimension signifies the tolerance we provide for that dimension/feature. In such cases, the effective search space would be the candidates that lie within this hyper-rectangle. Indexing techniques for biometrics must hence be able to support range query searches.

### 4. Tree should not have overlapping intermediate nodes

Overlapping intermediate nodes imply that while parsing through the tree to reach a certain leaf node, we might have to check different paths and thus search in several different leaf nodes.

### 5. Scalability

Since we consider real-time applications, we can expect the database size to scale to millions of records. The indexing technique we use should be able to handle such a large amount of high-dimensional data.

Several methods such as KD Trees, R Trees and its variants, X Trees and the Pyramid technique were studied for this purpose.

#### KD Trees [9]

It is one of the most popular multi-dimensional indexing methods. KD Trees are a form of a Binary search tree, which is formed by a recursive sub-division of the universe by using a  $(d-1)$  dimensional hyper-plane at each node, where  $d$  is the dimension of the data.

However, KD Trees does not lend to our biometric indexing problem since the tree structure depends heavily upon the order of insertion of data. Its structure is not consistent and may result into a skewed structure with a certain order of insertion. Besides, it is also not a very dynamic structure, since frequent insertions may skew the tree. Deletions also pose a non-trivial problem of reorganization of the entire tree.

#### R Trees [10]

It makes use of the concept of bounding hyper-rectangles, where each leaf node is a bounding rectangle and encloses all the child nodes it contains within it. It is similar to a b-tree by keeping each of its non-root nodes at least half-full. It results in a height-balanced tree in which all the leaf nodes are at the same level.

The R Trees present us with obvious problem of having overlapping intermediate nodes, which poses the problem of having to search through multiple paths. It has been proved experimentally that with increasing dimensionality the problem of over-

lap only worsens [14]. Thus with our 24 dimension hand geometry data, using R Trees for indexing is not a viable option.

#### R\* Trees [11]

R\* Trees are similar to R Trees, however they try to go beyond the R-Tree heuristic of trying to minimize the area of each rectangle, by optimizing the area of overlap and consider other factors such as the margin of each rectangle. However, since the overlaps in intermediate nodes still exists the problem of following multiple paths would be significant in the high dimensional biometric data.

#### R+ Trees [12]

R+ Tree is similar in structure to the R Trees, however they completely avoid the overlaps in intermediate nodes. However they result in nodes with poor space utilization and longer tree structures. Also, R Tree and all of its variants are insertion order dependent structures. Thus the use of R+ Trees for indexing biometric databases is rejected too.

#### X Trees [13]

They are based on the R\* Trees. They circumvent the problem of overlapping intermediate nodes by introducing 'super nodes' having greater capacity than the other normal nodes. Besides being insertion-order dependent, they have a problem faced by most high-dimensional indexing techniques in terms of trying to use the 50% quantile when splitting a page to fulfill storage requirements. Such techniques result in pages that have an access probability close to 100% as pointed out by [14].

#### Pyramid Technique [14]

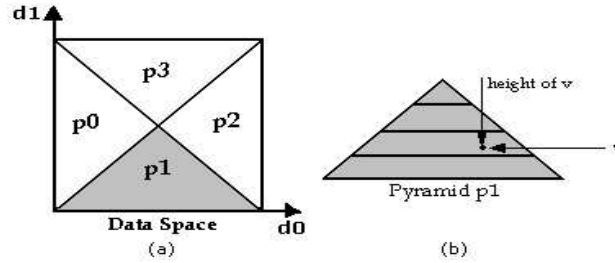
This currently seems to be the single most feasible indexing technique for indexing biometric databases. The concept is based on spatial-hashing the high-dimensional data into a single value. This single value can then be indexed very effectively using the popular B+ Trees.

The pyramid-technique was especially designed for working in higher-dimensional spaces and has been successfully tested on a 100-dimensional data set in [14]. The technique results in a tree structure that is not only insertion order invariant, but also height balanced and dynamic to the frequent insertions and deletions.

## 5 Methodology

The Pyramid technique requires normalizing the data values to lie between 0 and 1 to further divide the  $d$  dimensional data space into  $2 \times d$  number of pyramids, each having a common tip-point and a  $(d-1)$  dimensional base. For instance, for the 2-dimensional case, we have  $2 \times 2 = 4$  pyramids, as shown in the figure-1 below, with each of those pyramids (p0. .p3) having a common tip-point and a 1-dimensional base. Since we work with 24-dimensional hand geometry data, the number of pyramids formed are  $2 \times 24 = 48$ . To enroll a person in the database, we obtain 5 training sam-

ples of the hand geometry template. We average out the features in the 5 training templates and index only the mean-template of each person.



**Figure 1 a. Partitioning 2-d space into 4 Pyramids b. Height of point v in pyramid P1 [14]**

The pyramids are numbered in a logical manner, such that a pyramid is numbered ‘i’ if all the points within the pyramid are farthest from the tip-point along dimension i than any other dimension. Further, we check if the points within the pyramid have their  $i^{\text{th}}$  coordinate less than or equal to 0.5, in which case, the pyramid is labeled as i, else it is labeled as (i+d), d being the dimensionality of data. For instance, in the figure-1 shown above, all points within pyramids p1 and p3 are farthest from the tip-point along dimension d1 than along dimension d0 and points in p3 have d1 values greater than 0.5.

The height of a point inside pyramid ‘i’ is defined as the distance from the tip-point in the  $(i \bmod d)$  dimension. For instance, all points lying within pyramids p1 and p3 above, have the height defined as the distance from the tip-point along dimension d1.

The (height + pyramid number) forms the key (pyramid value) for every template and is used as the indexing key in the B+ Tree. On receiving an input template for the identification task, we bound this template by a bounding box to allow a level of tolerance for the feature values of the template by determining the lower and upper bounds for each feature as follows:

$$\text{Lowerbound}_i = F_i - \text{tol} \times \text{avgstd}_i, \quad \text{Upperbound}_i = F_i + \text{tol} \times \text{avgstd}_i$$

$F_i$ : Value of feature-i of the test template

$\text{avgstd}_i$ : average intra-user standard deviation for feature i

$\text{tol}$ : Tolerance-scale factor,  $\text{tol} \times \text{avgstd}_i$  determines the tolerance for each feature.

On performing similar operation on each of the dimensions, we obtain a hyper-rectangle for the test template. To find the candidate templates, we first determine the pyramids that this bounding box intersects. For every intersecting pyramid, we obtain the templates that lie within the range of the bounding box within the pyramid, by performing a range query using the method given in [14]. These templates form our candidate set for 1:1 matching to be performed by the final stage using another more reliable biometric such as fingerprint.

## 6. Hand Geometry Features

The algorithm to extract the feature involves the following steps: (1) Image acquisition (2) Converting to binary image (3) Contour extraction and noise elimination and (4) Feature extraction. The various geometric features that we are interested in involve the lengths and widths of the various parts of the fingers, the area under each finger. All together we have identified 24 invariant features [15] of the hand as shown below

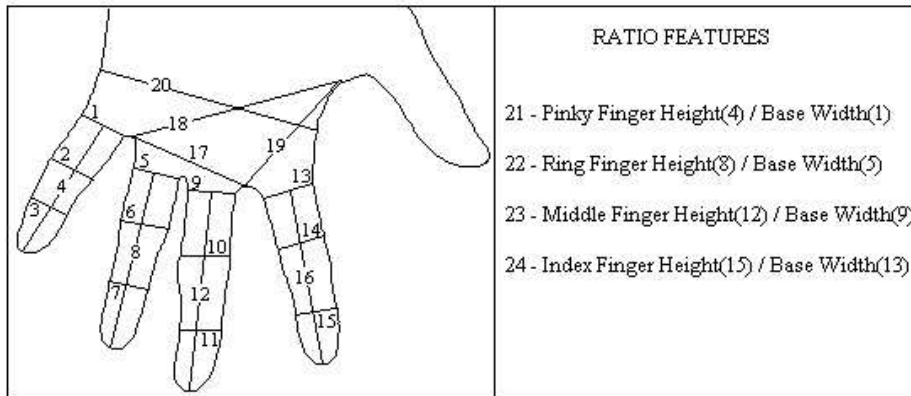


Figure 2: Features used for Hand Geometry template

## 7 Analysis

A theoretical analysis of the number of comparisons done to find all candidate templates lying within the bounding box of a given test sample yields the following:

$$\text{Number of Comparisons} = \sum_{i=1}^D (2 \times \log_2(N/m)) + \Theta(2 \times D)$$

$$\approx O(\log_2(N/m))$$

where,

- D: dimensionality of the data, 35 in our case
- N: total number of templates in the database
- m: minimum number of templates to be held per intermediate node

- $\Theta(2 \times D)$ : comparisons needed to determine the intersecting pyramids
- $\log_2(N/m)$ : the height of the B+ Tree
- $2 \times \log_2(N/m)$ : number of comparisons to get the leaf nodes representing the lower and upper bounds

Once we have obtained the leaf nodes corresponding to the bounds, we simply have to do a linear scan through the chain of the leaf nodes of the B+ Tree to get the final

candidate set for further investigation. Thus we show theoretically that the comparisons needed to determine the candidate set for the final stage, that is, the total time for pruning, is logarithmic in the total number of samples  $N$  in the database. Now we experimentally show that the size of the pruned candidate set determined by the indexing system is much smaller than the original size of the database.

## 8. Experimental Results

We evaluated the algorithm based on the fraction of the database ( $P_{SYS}$ ) that has to be searched in order to find the given user's record. The percentage was determined by running the algorithm on 5 test cases for each of the 200 users, thus a total of 1000 test cases. The fraction of the database searched  $P_{SYS}$  is the average size of the candidate set determined for the 1000 test templates.  $P_{SYS}$  can be described by the following equation

$$P_{SYS} = \frac{\sum_{T=1}^{TotalTestCases} \sum_{P=1}^{2 \times d} P_h}{TotalTestCases}$$

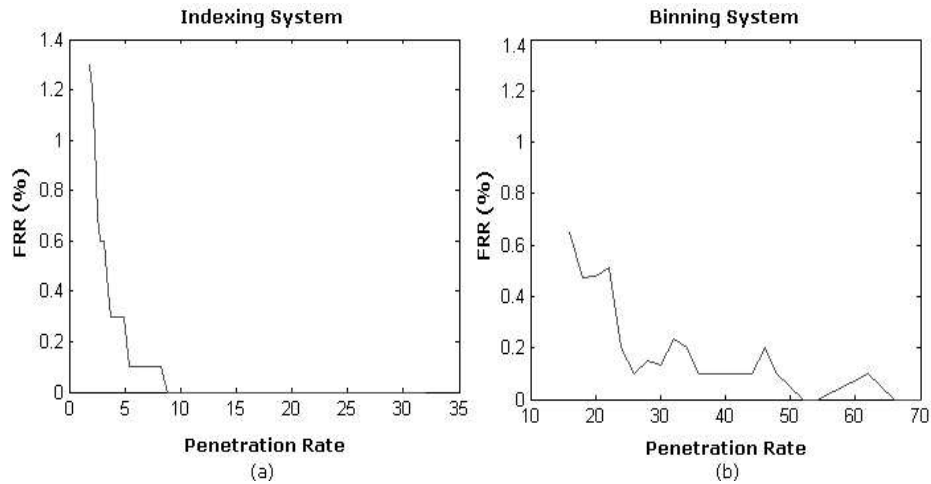
where,

P: Pyramid, which varies from 1 to  $2 \times d$ ,  $d$  = dimension of data

$P_h$ : Number of points in Pyramid P that lie within the bounding box of template T.

Experiments showed that after indexing the hand geometry data, the penetration rate that it yielded was 8.86% while maintaining 0% FRR, with the *tol*, Tolerance-scale factor, equal to 3.4. Thus the search space for the identification task was reduced to 8.86% of the original size of the database without falsely rejecting even a single template. The penetration rate could be further reduced however at the expense of an increased FRR by reducing the tolerance we provide for each feature.

The effectiveness of indexing can be especially seen from the figures shown below, where the 3(a) refers to the graph of FRR versus Penetration Rate for an Indexing system and 3(b) refers to the Binning system described in section 3. The graph for the indexing system was obtained by varying the tolerance-scale factor, *tol*. The graph for the binning system was obtained by varying the number of clusters in which the database was split and averaging the FRR values for same penetration rates.



**Figure 3: Comparison of FRR v/s Penetration Rate for Indexing and Binning Systems**

The FRR monotonically decreases with increase of the penetration rate in the Indexing system. However, the FRR decreases non-monotonically in the case of a binning system, which implies that there might be a case where even after increasing the penetration, the FRR might worsen instead of improving. Also, as can be seen, the indexing system converges to 0% FRR with much lesser penetration.

## 9 Conclusion & Future Work

We have presented a framework for implementing indexing in biometric databases in our endeavor to reduce the search space for the identification tasks. We have showed that on using the Pyramid technique for indexing biometric databases, we can prune the database to 8.86% of its original size maintaining a 0% FRR. Our future work involves using a parallel combination of hand geometry and signature indexed systems, which will definitely reduce the search space further. We also intend to evaluate the proposed scheme on a very large data set, by generating synthetic data.

## 10 References

- [1] Ruud Bolle, J. H. Connell, S. Pankanti, N. K. Ratha, and A. W. Senior. Guide to Biometrics, Springer Professional Computing, ISBN: 0-387-40089-3, 2004.
- [2] D. Maio, D. Maltoni, A. K. Jain and S. Prabhakar. Handbook of Fingerprint Recognition, Springer Verlag, ISBN: 0-387-95431-7, 2003.
- [3] International Biometric Group: <http://www.biometricgroup.com/>.

- [4] Cappelli R., Maio D., Maltoni D., Nanni L. *A two-stage fingerprint classification system*. ACM SIGMM workshop on Biometrics methods and applications, 2003
- [5] Anil Jain, Sharath Pankanti. *Fingerprint Classification and Matching*. Handbook for Image and Video Processing, A. Bovik (ed.), Academic Press, April 2000
- [6]. Ratha N., Karu K., Chen S., Jain A.. *A Real-time Matching System for Large Fingerprint Databases*. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 18, No. 8, August 1996.
- [7]. Searching in high dimensional spaces. Bohm, Berchtold, Keim. ACM Computing Survey, Volume 33, September 2001
- [8] Mhatre A., Palla S., Chikkerur S. and Govindaraju V. *Efficient Search and Retrieval in Biometric Databases*. SPIE Defense and Security Symposium, 2004.
- [9] Multidimensional Binary Search Trees Used for Associative Searching. Communications of the ACM, September 1975, Volume 18, Number 9.
- [10] R-Trees: A dynamic index structure for spatial searching, ACM SIGMOD international conference on Management of data, 1984
- [11] The R\*-tree: An Efficient and Robust Access Method for Points and Rectangles. Beckmann, Kriegel, Schneider, Seeger. ACM SIGMOD, Volume 19, (June 1990)
- [12] The R+ Tree: A dynamic index for multidimensional objects. *Timos Sellis, Nick Roussopoulos and Christos Faloutsos*. VLDB 1987: 507-518
- [13] The X-tree: An Index Structure for High-Dimensional Data. Berchtold S., Keim D., Kriegel H., Proceedings of the 22nd VLDB Conference Mumbai, India, 1996
- [14] The Pyramid-Technique: Towards Breaking the Curse of Dimensionality. Berchtold, Böhm, Kriegel Proc. Int. Conf. on Management of Data, ACM SIGMOD, Seattle, Washington, 1998
- [15] Pavan Rudravaram. Prototype pegless hand geometry verification. Technical report, Center for Unified Biometrics and Sensors, University at Buffalo, 2004