An efficient technique for indexing multimodal biometric databases

Umarani Jayaraman*, Surya Prakash and Phalguni Gupta

Department of Computer Science Engineering,
Indian Institute of Technology Kanpur,
Kanpur 208016, India
Fax: +91(512) 259 7586, 259 0725
Email: umarani@cse.iitk.ac.in
Email: psurya@cse.iitk.ac.in
Email: pg@cse.iitk.ac.in
*Corresponding author

Abstract: This paper proposes an efficient indexing technique which can be used in an identification system with large multimodal biometric database. In this technique, multi-dimensional feature vectors of each trait (iris, signature, ear and face) are normalised and projected to a lower dimensional feature space. The reduced feature vectors are fused at feature level and used to index the database by forming Kd-tree. The performance of the proposed technique is also analysed with the feature vectors of all traits by first fusing them and projecting the fused feature vector to a lower dimensional space, and using it for indexing. Performance is also compared with the indexing based on score-level fusion. The experiment is performed on a multimodal database consisting of 5400 images of 150 subjects (i.e. nine images per subject, per trait). Out of the nine, eight images are used for training and one is used for testing. Our experiment shows that the proposed technique significantly reduces the data retrieval time along with possible error rates.

Keywords: indexing; feature-level fusion; score-level fusion; Kd-tree; multi-dimensional data structure.


Biographical notes: Umarani Jayaraman is currently a PhD scholar in the Department of Computer Science and Engineering, IIT Kanpur, India. She received her MTech degree in Computer Science and Engineering from Arulmigu Meenakshi Amman Engineering College, Tamil Nadu, India. Her research interest includes image processing, computer vision and biometrics.

Surya Prakash is currently a PhD scholar in the Department of Computer Science and Engineering, IIT Kanpur, India. He received his MS degree in Computer Science and Engineering from IIT Madras, India. His research interest includes image processing, computer vision and biometrics.

Phalguni Gupta, since 1987, is associated with Department of Computer Science and Engineering, IIT Kanpur, India, where currently he is a Professor. He received his PhD degree in Computer Science and Engineering from...
1 Introduction

Biometrics has become a very essential component in almost all security aspects. Biometric system provides an automated method to verify or to identify an individual based on unique behavioural or physiological characteristics. Physiological characteristics are related to the shape of the body. Biometric traits such as face, fingerprint, iris, hand geometry fall under this category. Behavioural characteristics are related to the behaviour of a person. Signature, voice, character strokes, etc., are some of the traits which fall under this category. A biometric system consists of biometric readers, or sensors; feature extractor to compute salient attributes from the input template; and feature matcher for comparing two sets of biometric features. An authentication system consists of two subsystems; one for enrolment while other for authentication. During enrolment, biometric measurements are captured from a subject and relevant information is stored in the database. The task of the authentication module is to recognize a subject as a later stage, and is either identification of one person among many, or verification that a person’s biometric matches a claimed identity (Jain et al., 2008). Let $T_i$ be the feature vector of $i$th template in the database of $d$-dimension and is defined as follows:

$$T_i = [f_1, \ldots, f_d]$$

If the query image $Q$ with feature vector of $d$-dimension is defined as $Q = [q_1, \ldots, q_d]$, then $\forall j$, $q_j$ may not be same as $f_j$, where $f_j$ and $q_j$ are the $j$th feature values of $T_i$ and $Q$, respectively. For a given query template $Q$, the problem of identification system is to find the $n$ nearest neighbours in the database consisting of $N$ templates. To make the system more powerful and fast, a feature matcher should search for the templates with some pruning technique. Traditional databases index the records in an alphabetical or numeric order for efficient retrieval. In biometric database, there is no natural order by which one can keep the biometric templates. Since the feature vector generated for every templates are known a priori, these vectors can be arranged in such a way that an efficient searching algorithm can be used.

In the literature, there are few techniques available to index the biometric templates. These techniques either cluster the multi-dimensional templates and uses cluster information for searching or map the multi-dimensional feature vector to a scalar value (Mhatre et al., 2005) and use an existing data structure to index it. The study of reducing the search space of biometric databases is already made by Center for Unified Biometrics and Sensors (CUBS) group. Their effort is based on the binning/clustering technique. The effort of binning was performed in Mhatre et al. (2001) and it was demonstrated that the search space could be reduced to approximately 5% of the original database. The data was clustered using 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 needs re-partition the entire database on addition of new samples to it. Thus binning/clustering systems are useful

IIT Kharagpur, India, in 1986. His research interest includes sequential algorithms, parallel algorithms, online algorithms, image processing and biometrics. In the last 26 years, he has published over 150 peer-reviewed journals and conference papers. He is also the co-author of six books. He is a recipient of 2007 IBM Faculty Award.
only in cases of static databases. In Mhatre et al. (2005), CUBS group made another effort to index biometrics database using pyramid technique. Pyramid technique partitions a $d$-dimensional universe into $2^d$ pyramids that meet at the centre of the universe. Then, every pyramid is divided into several slices parallel to the basis of the pyramid. The $d$-dimensional vectors representing points in space are approximated by one-dimensional quantities, called pyramid values, which are indexed by a B+ tree. However, the pyramid technique is ineffective in situations when the data points fall on or near the boundaries of the original space. Another problem in this technique is the increased storage overhead due to the need to keep the index of both the pyramid values and the original points.

In Umarani et al. (2008a), an indexing technique based on the modified B+ tree has been proposed. This technique uses the multi-dimensional feature vector to index the database by forming B+ tree with variation in the structure called Modified B+ tree. In the modified B+ tree, instead of feature values, range of feature values is stored as keys. As a result, the height of the B+ tree has been significantly reduced and it leads to compare less number of nodes in the B+ tree. For feature values of $d$-dimensions, modified B+ is formed and it is used for indexing the biometric templates. Another indexing approach is proposed in Umarani et al. (2008b), which is based on the feature-level fusion using Kd-tree. Kd-tree stores actual data (original points) based on the ith discriminator as a key value in a node, and hence there is no storage overhead as like pyramid technique. In addition since Kd-tree is based on space-partitioning indexing technique, there is no overlapping between nodes as like bounding region-based indexing techniques such as R-tree, R*-tree, M-tree and X-tree (Guttman, 1984; Samet, 1990).

This paper proposes an efficient indexing technique for multimodal databases which makes use of Kd-tree and feature-level fusion. The technique, first, normalises and projects multi-dimensional feature vectors of each trait (iris, signature, ear and face) to a lower dimensional feature space and then fuses the reduced feature vectors of all traits to index the database by forming Kd-tree.

Rest of the paper is organised as follows. Section 2 briefly discusses Haar wavelet transform, dimension reduction technique and Kd-tree which serve as the foundation for further discussion. Section 3 describes the feature extraction algorithms for iris, signature, ear and face biometric traits. In Section 4, proposed indexing techniques have been discussed. Results are analysed in Section 5. Conclusion is given in Section 6.

## 2 Preliminaries

This section discusses some of the basics techniques which are required in developing proposed indexing technique. Subsection 2.1 presents Haar wavelet transform (Gonzalez and Woods, 2001). In Subsection 2.2, dimensionality reduction technique that has been used to reduce the dimension of the feature vectors is described. Subsection 2.3 describes the Kd-tree data structure which is used to index the databases by forming Kd-tree.

### 2.1 Haar wavelet transform

This section briefly describes Haar wavelet transform which is used in feature extraction in the present study. A wavelet transform analyses a signal based on its content in different frequency ranges and defined using a scaling and a mother wavelet function.
These two functions generate a family of functions that can be used to break up or reconstruct a signal. Haar wavelet transform is the simplest, though powerful transform. The Haar scaling function $\phi(x)$ has values one for closed interval of time $x$ from 0 to 1 and zero for other interval of time. Scaling function $\phi(x)$ is formulated as follows:

$$\phi(x) = \begin{cases} 1, & \text{if } 0 \leq x \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

Haar mother wavelet function is $\psi(x)$ is a discontinuous function and resemble a step function. It takes value 1 for $x$ greater than or equal to 1 and less than 1/2. $\psi(x)$ also takes value -1 for $x$ greater than or equal to 1/2 and less than 1. $\psi(x)$ takes 0 value for other values of $x$. $\psi(x)$ is discontinuous at time equal to 0, 1/2 and 1. $\psi(x)$ is formally defined as follows:

$$\psi(x) = \begin{cases} 1, & \forall x \in [0, \frac{1}{2}) \\ -1, & \forall x \in [\frac{1}{2}, 1) \\ 0, & \text{otherwise} \end{cases}$$

Figure 1 shows the Haar scaling and mother wavelet functions. A standard decomposition of an image (two-dimensional signal) is easily done by first performing a one-dimensional transformation on each row followed by one-dimensional transformation on each column. The outcome of this decomposition is four types of wavelet coefficients: approximation, vertical, horizontal and diagonal. Approximation coefficient may be further decomposed to get the next level of decomposition.

2.2 Dimension reduction technique

Suppose, we have a dataset $T$ consisting of $\{t_i\}_{i=1}^N$ training samples. Each sample is described by a set of features $F(d = |F|)$, so there are $N$ samples described by $d$-dimensional feature vector each. This can be represented by feature object matrix $T_{d \times N}$, where each column represents a sample. The objective of dimension reduction is to reduce the data into another set of features $F'$, where $k = |F'|$ and $k < d$; $T_{d \times N}$ is reduced to
Typically, this is a linear transformation \( T' = WT \) that reduces dataset \( T \) to \( T' \) in \( k \)-dimension, where \( W = W_{k \times d} \) is the transformation technique. Principal Component Analysis (PCA), which is the dominant dimension reduction technique, transforms the data into a reduced space that captures most of the variance in the data. PCA reduces dimension of a dataset by retaining those characteristics of the dataset that contribute most to its variance. It can be done by keeping lower order principal components and ignoring higher order ones. Such low-order components often contain the most important aspects of the data.

**Figure 2** PCA in 2D, the variance in the \( f'_1 \) direction is maximum. (a) Solid lines: the original basis; dashed lines: the PCA basis; (b) the projection (1D reconstruction) of the data using the first principal component (see online version for colours)

PCA can be mathematically defined (Jolliffe, 2002) as an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by any projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate and so on. PCA is theoretically an optimum transformation for a given data in least square terms. Dimensionality reduction in PCA is accomplished by computing a set of eigenvalues of total scatter matrix \( S_t \) of the data samples defined as:

\[
S_t = \sum_{i=1}^{N} (t_i - m)(t_i - m)^T
\]

where \( m \) is the mean value of the sample set \( T \). For dimensionality reduction, \( k \) (where \( k < d \)) eigenvectors \( U = [u_1, ..., u_k] \) corresponding to first \( k \) largest eigenvalues of \( S_t \) are selected. Reduced dimension training samples \( T' = [t'_1, ..., t'_k] \) can be obtained by the transformation \( T' = U^T T \). Now, when a probe template \( t_p \) is presented for identification/verification, it is projected on \( U \) to obtain a reduced dimension vector \( t'_p = U^T t_p \). Geometric interpretation of PCA is shown in Figure 2. To see how the data is spread, we encapsulate the dataset inside an ellipse and take a look at the major and
minor axes that form the vectors $f_1'$ and $f_2'$. These are the principal component axes, i.e. the base vectors that are ordered by the variance of the data. PCA finds these vectors and gives a $[f_1', f_2'] \rightarrow [f_i']$ transformation. While this example is for 2D, PCA works for multi-dimensional data, and it is generally used with high dimensionality problems.

2.3 Kd-tree data structure

The proposed indexing technique is based on the Kd-tree data structure (Bentley, 1975; Bentley, 1990; Samet, 1990). This section discusses the salient features of Kd-tree. It is a binary tree that represents a hierarchical subdivision of space using splitting planes that are orthogonal to the coordinates axes. Kd-tree is a space-partitioning data structure for organising points in a $k$-dimensional space. Any application in which features are generated as multi-dimensional is a potential application for Kd-tree. Structure of a node in Kd-tree is given in Figure 3. Each node in Kd-tree consists of five fields. Node contains two pointers known as LLINK and RLINK, which pointing to left subtree and right subtree, respectively, if exists. Otherwise, it points to null. The field VAL is an array of length $k$ containing real feature vector. The INFO field contains descriptive information about the node.

![Figure 3 Structure of Kd-tree node](image)

The $DISC$ field is a discriminator, which is an integer between 1 and $k$, both inclusive. In general, for any node $P$ in the Kd-tree, let $i = DISC(P)$ and is defined as $\text{level}(P) \mod k$. Then for any node $L$ in $\text{LLINK}(P)$, $L.\text{VAL}[i] < P.\text{VAL}[i]$; likewise, for any node $R$ in $\text{RLINK}(P)$, $R.\text{VAL}[i] \geq P.\text{VAL}[i]$. All nodes on any given levels of the tree have the same discriminator. The root node has discriminator 1, and its two sons have discriminator 2, and so on to the $k$th level on which the discriminator is $k$. Again the $(k + 1)$th level has discriminator 1, and the cycle repeats; In general, next discriminator, denoted as $\text{NEXTDISC}$, is a function defined as follows:

$$\text{NEXTDISC}(i) = (i + 1) \mod k. \quad (5)$$

Number of nodes in the Kd-tree is same as the number of templates in the input file to be inserted in the tree. It is already mentioned that $k$ is the dimensionality of the template.

In order to insert a node $P$ having the data into the Kd-tree, it starts searching from the root of the Kd-tree and finds its appropriate position where the node can be inserted. Bentley (1975) shows that the average cost of inserting and searching a node in Kd-tree consisting of $N$ nodes is $O(\log_2 N)$. 


Furthermore, in order to perform a range search for a given query template $Q$ with a distance $r$, it determines all templates $T$ having Euclidean distance from $Q$ less than or equal to $r$. The average cost to perform a range search in Kd-tree consisting of $N$ nodes is $O(k.\sqrt[N]{N})$ (Bentley and Friedman, 1979).

3 Feature extraction

Feature extraction is a very important part of any pattern recognition application and deals with the representation of large set of data using some of its salient features. A template in the proposed indexing system is represented by four real feature vectors $T_I$, $T_S$, $T_E$ and $T_F$ for iris, signature, ear and face, respectively. Feature extraction for the signature is done using parameter extraction algorithm (Kaewkongka et al., 1999; Gonzalez and Woods, 2001; Ooi et al., 2007), while other feature vectors are extracted using discrete Haar wavelet transform. The following subsections describe the process of feature extraction for each trait.

3.1 Iris feature extraction

Iris features are extracted by localising the inner and outer iris boundaries (Wildes, 1997; Daugman, 2004). The proposed strategy uses circular Hough transformation (Chen and Chung, 2001; He and Shi, 2005) for inner iris boundary localisation in the iris image. Once the inner iris boundary (which is also the boundary of the pupil) is computed, outer iris is determined using intensity variation approach (Ma et al., 2004). The annular portion of iris after localisation is transformed into rectangular block to take into consideration the possibility of pupil dilation. This transformed block is used for feature extraction using discrete Haar wavelet transform. Haar wavelet operates on data by calculating the sums and differences of adjacent values.

Figure 4 Four levels discrete Haar wavelet transform on iris strip
It operates first on adjacent horizontal values and then on adjacent vertical values. The decomposition is applied up to four levels on transformed rectangular iris block, as shown in Figure 4. A $d$-dimensional real feature vector $T_i$ is obtained from the fourth-level decomposition and is given by:

$$T_i = [i_1, ..., i_d].$$

### 3.2 Signature feature extraction

Signature feature extraction proceeds as follows. The signature image is scanned and cropped (approximately $300 \times 150$ pixels) in the required format. The image is first passed to a noise removal module which performs median filtering on the input image and returns noise-free image in the form of a 2D integer array of same size as the input image. The image extraction module is called to extract the binary, High Pressure Region (HPR) and thinned images from the array. After binary image extraction, the horizontal and vertical projections of the image are used to determine the area of interest from the image. To remove the extra blank spaces around the image, a normalisation procedure is used. This normalisation process is performed over the HPR and thinned images as well. Finally, noise-free images are used in the feature extraction. Signature features can be classified into two types: global and local features. Global features are characteristics that identify or describe the signature as a whole and local features are confined to a limited portion (e.g. a grid) of the signature (Kaewkongka et al., 1999; Ooi et al., 2007). Local features are the same as that of the global features except that they are calculated for each of the number of grids that the signature has been divided to. Examples of global features include width and height of individual signature components (as shown in Figure 5), width to height ratio, total area of black pixels in the binary and HPR images, horizontal and vertical projections of signature images, baseline, baseline shift, relative position of global baseline and centre of gravity with respect to width of the signature, number of cross and edge points, slant, run lengths, etc. These parameters are extracted from the signature template and are stored as feature vector of $d$-dimensions as follows:

$$T_s = [s_1, ..., s_d].$$

### 3.3 Ear feature extraction

The given input side face image is first resized to a standard size of $240 \times 340$ pixels, and the ear part is cropped from it. Discrete Haar wavelet decomposition is applied to the cropped ear image to extract the wavelet Approximation (CA), Diagonal (CD), Horizontal (CH) and Vertical (CV) coefficients. Figure 6 shows an example for the two
levels of Haar wavelet decomposition applied on the cropped input ear image. The decomposition is applied up to the fifth level on cropped ear image and a $d$-dimensional real feature vector $T_E$ is obtained using the detailed coefficients of the fifth level, and stored as follows:

$$T_E = [e_1, \ldots, e_d].$$

**Figure 6** Two levels of Haar wavelet transform applied on ear image: (a) input ear image, (b) diagonal coefficient, (c) horizontal coefficient and (d) vertical coefficient (see online version for colours)

(a) (b) (c) (d)

### 3.4 Face feature extraction

Face feature extraction procedure is similar to that of ear. The given input face image is resized to a standard size of $640 \times 480$ pixels and face part of the image is cropped leaving the background. Haar wavelet is used for extracting the features from the detected face image. The detected input image is decomposed up to the required level giving an Approximation (CA), Vertical (CV), Horizontal (CH) and Diagonal (CD) coefficients, using the discrete Haar wavelet transformation. Figure 7 shows an example of the two levels of Haar wavelet decomposition, applied on the detected face image and the corresponding diagonal, horizontal and vertical coefficients. A $d$-dimensional real feature vector $T_F$ is obtained using the detailed coefficients of the fifth-level decomposition and is given as:

$$T_F = [f_1, \ldots, f_d].$$

**Figure 7** Two levels of Haar wavelet transform applied on face image: (a) input face image, (b) diagonal coefficient, (c) horizontal coefficient and (d) vertical coefficient (see online version for colours)

(a) (b) (c) (d)
4 Proposed indexing techniques

This section presents proposed indexing techniques which are based on Kd-tree, feature-level fusion and score-level fusion. A feature normalisation procedure has been used to normalise the feature vectors before using them for indexing. In the first technique, normalised feature vectors of each trait are indexed using separate Kd-tree and final result set is computed by fusing the individual result sets generated by index structure of each trait applying score-level fusion. In the second technique, normalised feature vectors of all traits are first fused and then projected to a lower dimensional space. Projected vector is used for indexing. Third technique first projects the normalised feature vectors of each trait to a lower dimensional space, and then computes a fused feature vector using the projected vector of individual trait. Fused vector is further used for indexing by forming Kd-tree. Performances of these techniques are discussed in Section 5.

4.1 Feature normalisation

Let there be two vectors \( X = [x_1, x_2, \ldots, x_n] \) and \( Y = [y_1, y_2, \ldots, y_n] \) need to be combined to form a combined feature vector. The individual feature values of these feature vectors may exhibit significant difference in range as well as form (i.e. distribution). So there is a need of feature normalisation to modify the location (mean) and scale (variance) of the feature values via a transformation function in order to map them into a common domain. Adopting an appropriate normalisation scheme also helps to address the problem of outliers in feature values. In our study, we use min–max technique for feature normalisation. Let \( x \) and \( x' \) denote feature vectors before and after normalisation, respectively. The min–max technique of normalisation computes \( x' \) from \( x \) as follows:

\[
x' = \frac{x - \min(F_x)}{\max(F_x) - \min(F_x)}
\]

where \( F_x \) is the function which generates \( x \), and \( \min(F_x) \) and \( \max(F_x) \) represent the minimum and maximum values of \( F_x \) for all possible \( x \) values, respectively. The min–max technique is effective when the minimum and the maximum values of the component feature values are known beforehand. Normalising the feature values results in modified feature vectors \( X' = [x'_1, x'_2, \ldots, x'_n] \) and \( Y' = [y'_1, y'_2, \ldots, y'_n] \). Effectively, these feature values fall in the range of 0 and 1.

4.1.1 Approach 1

Approach 1 is based on Kd-tree and score-level fusion. The match score is a measure of similarity between the input and template biometric feature vectors. When match scores produced by different biometric matchers are consolidated in order to arrive at a final recognition decision, fusion is said to be done at the match-score level. Figure 8 gives an overview of the score-level fusion used in this approach.
In this approach, data of iris, ear, face and signature is indexed separately using four Kd-trees, one for each trait. For a given query template, identification is performed using Kd-tree of individual traits (i.e iris, ear, face and signature) and the results of the individual traits are fused to get the top matches. It may be noticed that in feature-level fusion, there is only one Kd-tree formed for indexing. The indexing technique built with Kd-tree for individual traits is invoked (ear, face, iris and signature). As a result of this, we get four separate sets of matched IDs for ear, face, iris and signature, respectively. These matched IDs are fused using weighted sum rule to declare the top matches which are the nearest matched IDs. Matching score $M$ for an $ID_x$ in weighted sum rule is defined as follows:

$$M = w_E \times \text{occ}_E(x) + w_F \times \text{occ}_F(x) + w_I \times \text{occ}_I(x) + w_S \times \text{occ}_S(x)$$  \hspace{1cm} (11)$$

where $\text{occ}_E(x)$, $\text{occ}_F(x)$, $\text{occ}_I(x)$ and $\text{occ}_S(x)$ are the number of occurrences of $ID_x$ in ear, face, iris and signature result dataset, respectively, and $w_E$, $w_F$, $w_I$ and $w_S$ are respective assigned weights such that $w_E + w_F + w_I + w_S = 1$. The weights $w_E$, $w_F$, $w_I$ and $w_S$ are modelled using the accuracy of the verification system when only individual trait is considered and be defined as follows:

$$\text{Acc}_E = \frac{\text{Acc}_{E}}{\text{Acc}_{T}}, \text{Acc}_F = \frac{\text{Acc}_{F}}{\text{Acc}_{T}}, \text{Acc}_I = \frac{\text{Acc}_{I}}{\text{Acc}_{T}}, \text{Acc}_S = \frac{\text{Acc}_{S}}{\text{Acc}_{T}}$$  \hspace{1cm} (12)$$

where $\text{Acc}_{E}$, $\text{Acc}_{F}$, $\text{Acc}_{I}$ and $\text{Acc}_{S}$ are various accuracies. $\text{Acc}_{E} = 96.0$, $\text{Acc}_{F} = 95.2$, $\text{Acc}_{I} = 91.5$ and $\text{Acc}_{S} = 86.6$ are various accuracies through our experiments we have got when they are individually used in the verification system. The matching score $M$ is calculated for each of the matched IDs for an $ID_x$. These matching scores are sorted to declare the top matches which are the nearest matched IDs for an $ID_x$. 
4.1.2 Approach 2

This approach uses Kd-tree with feature-level fusion. Feature-level fusion (Ross and Govindarajan, 2005) refers to combining different feature sets extracted from multiple biometric sources. When the feature sets are homogeneous (e.g. multiple measurements of a person’s signature), a single resultant feature vector can be computed as a weighted average of the individual feature vector. When the feature sets are non-homogeneous (e.g. features of different biometric modalities like face and signature), we can concatenate them to form a single feature vector. In this approach, a multi-dimensional feature vector is obtained by fusing the normalised feature vectors of all traits (iris, signature, ear and face) and dimensionality reduction is applied on fused vector to get a new feature vector in lower dimensional space. This new vector is subsequently used for indexing in Kd-tree. Figure 9 gives the overview of this approach. Feature-level fusion and dimensionality reduction procedure used in this approach are explained below.

4.2 Feature-level fusion

The given four real feature vectors $T_I$, $T_S$, $T_E$ and $T_F$ of $d$-dimensional features for iris, signature, ear and face, respectively, are fused to form a new feature vector $T_N$ to represent an individual. The vector $T_N$ of dimensionality $4d$ \( (4d = (e_i + f_i + i_i + s_i)) \) can be generated by augmenting vectors $T_I$, $T_S$, $T_E$ and $T_F$. Vector $T_N$ (the fused feature values) of $4d$-dimension is further reduced to $k$-dimension using the PCA technique.

4.3 Dimensionality reduction using PCA and indexing

The fused feature vector $T_N$ obtained by fusing four traits is of very high dimension. Since all the features involved in this vector are not equally important, PCA is applied to reduce the dimensions and get compact representation. This reduces vector $T_N$ of $4d$-dimensions to $k$-dimensions. Algorithm 1 summarises the steps involved in this approach.

**Algorithm 1: Algorithm of Approach 2**

1. Normalise the feature vectors of each trait.
2. Fuse the feature vectors at feature level.
3. Reduce the feature vectors using PCA.
4. Form the Kd-tree with the fused feature vectors for indexing.
5. Invoke indexing technique through multimodal identification system.

4.3.1 Approach 3

In Approach 2, feature vectors of individual traits are first fused and then obtained feature vector is projected to a lower dimensional space, which is further used for indexing. Other way to do the feature fusion could be as follows. Normalised feature vectors of each trait (iris, signature, ear and face) are first projected to a lower dimensional feature space and then fused to get a multi-dimensional feature vector representing all traits. Figure 10 gives the overview of this approach. Dimensionality reduction and feature-level fusion procedures used in this approach are explained below.
Figure 9  Overview of Approach 2

Figure 10  Overview of Approach 3
4.4 Dimensionality reduction using PCA

The real feature vectors \( T_I, T_S, T_E \) and \( T_F \), each of \( d \)-dimension, obtained for iris, signature, ear and face, respectively, are reduced to \( k \)-dimension using the PCA. Dimensionality reduction is applied as the feature vectors extracted from four traits are of high dimensions and all the features generated are not equally important. In this approach, the features of four traits viz. ear, face, iris and signature are reduced from \( e_{d_e}, f_{d_f}, i_{d_i} \) and \( s_{d_s} \) to \( e_{k_e}, f_{k_f}, i_{k_i} \) and \( s_{k_s} \) dimensions, respectively.

4.5 Feature-level fusion and indexing

The reduced \( k \)-dimensional features are fused to yield a new feature vector \( T_N \) to represent an individual. The vector \( T_N \) of dimensionality \( 4k \) \((4k \leq (e_{d_e} + f_{d_f} + i_{d_i} + s_{d_s}))\) is generated by augmenting vectors \( T_I, T_S, T_E \) and \( T_F \) of iris, signature, ear and face, respectively. Vector \( T_N \) (the fused feature values) are used for indexing in the Kd-tree. Algorithm 2 summarises the steps involved in this approach.

Algorithm 2: Algorithm of Approach 3

1. Normalise the feature vectors of each trait.
2. Reduce the feature vectors using PCA.
3. Fuse the feature vectors at feature level.
4. Form the Kd-tree with the fused feature vectors for indexing.
5. Invoke this indexing technique through multimodal identification system.

5 Experimental results

In order to demonstrate the indexing using Kd-tree for multimodal identification, data for four traits (iris, signature, ear and face) is obtained from the database available at Indian Institute of Technology, Kanpur (IITK). In the following subsections, first we briefly discuss the databases of various traits that we have used, and then present our results of indexing on multimodal identification using Kd-tree with feature-level fusion and score-level fusion.

5.1 Iris database

Indian Institute of Technology, Kanpur iris image database is used for the iris data. The database comprises of 1350 iris images of 150 subjects (nine images per subject) from their left eye. The images are JPEG in grey scale and are resized with 450 \( \times \) 350 pixels resolution. Out of the nine, eight images are used for training and one image is used for testing.

5.2 Signature database

Indian Institute of Technology, Kanpur signature database is used for signature data. The database consists of 1350 signature images of 150 subjects (nine images per subject).
These images are in JPEG format. In the predefined sheet, users are asked to sign their signatures in the nine boxes provided in the sheet. The sheet is scanned at 200 dpi as a grey scale image. Size of the scanned image is around $1700 \times 2300$ pixels and the size of one box is $300 \times 150$ pixels. Out of the nine, eight images are used for training and one image is used for testing. A sample scanned signature sheet is shown in Figure 11.

5.3 Ear database

Ear database consists of 1350 side images of human faces from IITK with the resolution of $240 \times 340$ pixels. The images are taken from 150 subjects (nine images per subject). These images are captured using a digital camera from a distance of 0.5 to 1 metre. Ears are cropped from these input images and used for the feature extraction.

5.4 Face database

Face database consists of 1350 images (nine images per subject) of 150 subjects from IITK. These images are captured using a digital camera from a distance of 0.5 to 1 metre. The face is localised ($120 \times 250$ pixels) from the given input image by clicking three points. Two points on left and right eyes and third point on lower lip portion. The localised face (detected face) image is used for the feature extraction.

Figure 11  Scanned signature sheet

In our experiment, dimensions of the feature vectors are found to be 64 for the ear and the face, 88 for the iris and 27 for the signature, respectively. In the following subsections, experimentation for the techniques discussed in Section 4 is presented. Effectiveness of these techniques is evaluated based on False Rejection Rate (FRR), an optimum distance $r$ and the number of IDs falls in the first match. The FRR is defined as the percentage of identification instances in which false rejection occurs.
5.5 Approach 1

Approach 1 is based on the score-level fusion technique. In it, data of iris, ear, face and signature is indexed separately using four Kd-trees, one for each trait. For a given query template, identification is performed using Kd-tree of each trait (i.e. iris, ear, face and signature) and individual results are fused to get the top matches. Following subsections analyse the performance of this approach with and without dimension reduction of the feature vectors.

5.5.1 Without dimension reduction

In this case, the original feature vectors of each trait are used for indexing without reducing their dimensions. The indexing technique built with Kd-tree for individual trait (ear, face, iris and signature) is invoked. The distance \( r \) for each tree is varied and set of matched IDs are computed for each \( r \). Distance value \( r \), which results minimum FRR in the identification system, is chosen as an optimum \( r \). Scores obtained from each of these traits are fused to get the final set of matched IDs, using the technique discussed in Subsection 4.1. In our experiment, for distance values \( r_e = 32, r_f = 25, r_i = 1300 \) and \( r_s = 30 \), we get the optimum values of FRR, where \( r_e, r_f, r_i \) and \( r_s \) are the distances of ear, face, iris and signature, respectively.

5.5.2 With dimension reduction

In this case, dimensions of the individual trait are reduced from \( d \)-dimensions to \( k \)-dimensions and then reduced feature vector is used in the Kd-tree for indexing.

### Table 1  FRR and distance \( r \) for each trait in Approach 1

<table>
<thead>
<tr>
<th></th>
<th>Iris (a)</th>
<th>Signature (b)</th>
<th>Ear (c)</th>
<th>Face (d)</th>
<th>(a + b + c + d)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRR</td>
<td>r</td>
<td>FRR</td>
<td>r</td>
<td>FRR</td>
<td>r</td>
</tr>
<tr>
<td>Without dimension reduction</td>
<td>5.33</td>
<td>1300</td>
<td>2</td>
<td>30</td>
<td>2.66</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.66</td>
<td>32</td>
<td>2.67</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2.66</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>With dimension reduction (( k = 10 ))</td>
<td>2</td>
<td>1200</td>
<td>2.1</td>
<td>30</td>
<td>1.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3.33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.66</td>
</tr>
</tbody>
</table>

In our experiment, each of the trait is reduced to ten dimensions and then indexing technique built with Kd-tree for individual trait is invoked. In our experiment, for the distance of \( r_e = 17, r_f = 18, r_i = 1200 \) and \( r_s = 30 \), we get the optimum FRR, where \( r_e, r_f, r_i \) and \( r_s \) are the distances of ear, face, iris and signature, respectively. Figure 12 shows the computation of distance \( r_e \), based on the minimum FRR and maximum number of IDs fall in the first match for the ear trait (for feature vector \( k = 10 \)). Table 1 shows the optimum FRR and distance \( r \) for which an optimum FRR is achieved for each trait. It is clear from the table that for reduced dimension \( k = 10 \), this indexing technique performs better. Figure 13 shows top matches vs. number of IDs underlying for both with and without dimension reduction.
Figure 12 Computation of distance $r_e$ for $k = 10$ to achieve optimum values of FRR and maximum number of IDs in first match. Circle shows the point ($r_e = 0.9$) for which optimum FRR and maximum first match IDs are obtained for ear trait ($k = 20$) (a) Distance $r_e$ (for $k = 20$) vs. FRR for ear trait (b) distance $r_e$ (for $k = 20$) vs. number of IDs fall in first match (see online version for colours)

5.6 Approach 2

Approach 2 is based on feature-level fusion technique. In this approach, the normalised feature vectors of each traits are first fused and then dimensionality reduction is performed to the fused feature vector using PCA technique. In our experiment,
64 dimensions of face and ear as well as 88 dimensions of iris and 27 dimensions of signature feature vectors (normalised) are fused to get a 243 dimensional feature vector. This fused feature vector is reduced to 40 and 20 dimensions and used to index the database by forming Kd-tree.

**Figure 13** Top matches vs. number of IDs underlying in Approach 1 (see online version for colours)

The experiment proceeds as follows. It takes the root of a Kd-tree and a query template \( Q \) as input and retrieves all sets of templates \( T \) which lies within distance \( r \) from \( Q \). Proximity is quantified between them using Euclidean distance. The distance \( r \) for the tree is varied and set of matched IDs are computed for each \( r \). Distance value \( r \), which results the minimum value of FRR and the maximum number (value) of IDs fall in the first match in the identification system, is chosen as an optimum \( r \). Set of matched IDs obtained for the optimum \( r \) are then sorted based on the their occurrences and top matched IDs are declared. In our experiment (feature vector \( k = 40 \)) for \( r = 0.9 \), we get the minimum FRR = 0.66% and maximum number of IDs in the first match (first match = 107). Figure 14 shows the computation of distance \( r \), based on the minimum FRR and maximum number of IDs fall in the first match. Table 2 shows FRR, distance \( r \) and number of IDs in first match for the reduced dimensions \( k = 40 \) and 20. Figure 15 shows the top matches vs. number of IDs underlying for the reduced dimensions \( k = 40 \) and 20. It is evident from Table 2 and Figure 15 that for \( k = 40 \), this indexing technique performs better. For \( k = 40 \), out of 150, 107 IDs fall in first match, 133 IDs fall in the top two matches and 148 IDs fall in the top seven matches with FRR = 0.66%. For \( k = 20 \), only 98 IDs fall in the first match, 116 IDs fall in the top two matches and 146 IDs fall in the top ten matches with FRR = 0.66%.
Table 2  FRR, distance $r$ and number of IDs fall in first match in Approach 2

<table>
<thead>
<tr>
<th>Iris + signature + ear + face</th>
<th>$FRR$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k = 40$</td>
<td>0.66</td>
<td>0.90</td>
</tr>
<tr>
<td>$k = 20$</td>
<td>0.66</td>
<td>0.75</td>
</tr>
</tbody>
</table>

Figure 14  Computation of distance $r$ to achieve optimum values of FRR and maximum number of IDs in first match, circle shows the point ($r = 0.9$) for which optimum FRR and maximum first match IDs are obtained: (a) distance $r$ vs. FRR and (b) distance $r$ vs. number of IDs fall in first match (see online version for colours)
5.7 Approach 3

Approach 3 is also based on feature-level fusion technique. In this approach, normalised feature vectors of each trait are first projected to a lower dimensional space using PCA, and then fused to form a single multi-dimensional vector. In our experiment, 64 dimensions of face and ear as well as 88 dimension of iris and 27 dimension of signature feature vectors are all reduced to ten dimensions (by considering top ten eigenvalues), respectively. These reduced ten dimensions from each of the traits are fused together, resulting a 40-dimensional feature vector \( T_n \). This feature vector is used to index the database by forming Kd-tree. The experiment procedure is same as discussed in Approach 2.

In our experiment (feature vector \( k = 40 \)) for \( r = 0.67 \), we get the minimum \( FRR = 0.66\% \) and maximum number of IDs in the first match (first match = 146). Figure 16 shows the computation of distance \( r \), based on the minimum FRR and maximum number of IDs fall in the first match. Table 3 shows FRR, distance \( r \) and number of IDs in first match for the reduced dimensions \( k = 40 \) and 20. Figure 17 shows the top matches vs. number of IDs underlying for the reduced dimensions \( k = 40 \) and 20. It is evident from Table 3 and Figure 17 that for \( k = 40 \), this indexing technique performs better. For \( k = 40 \), the maximum IDs fall in the first match which is 146 out of 150, 147 IDs fall in the top two matches and 149 IDs fall in the top five matches with \( FRR = 0.66\% \). For \( k = 20 \), out of 150 total of 107 IDs fall in the first match, 137 IDs fall in the top two matches and 148 IDs fall in the top five matches with \( FRR = 0.66\% \).
Table 3  FRR, distance $r$ and number of IDs fall in first match in Approach 3

<table>
<thead>
<tr>
<th></th>
<th>FRR</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k = 40$</td>
<td>0.66</td>
<td>0.67</td>
</tr>
<tr>
<td>$k = 20$</td>
<td>0.66</td>
<td>0.60</td>
</tr>
</tbody>
</table>

Figure 16  Computation of distance $r$ to achieve optimum values of FRR and maximum number of IDs in first match, circle shows the point ($r = 0.67$) for which optimum FRR and maximum first match IDs are obtained: (a) distance $r$ vs. FRR and (b) distance $r$ vs. number of IDs fall in first match (see online version for colours)
5.8 Comparison of the proposed approaches

Biometric identification system is tested with three proposed approaches for 150 query templates. The top matches vs. number of IDs underlying is plotted in Figure 18 for the reduced dimensions of $k = 40$ (Approaches 2 and 3). In the case of Approach 1, reduced dimension is $k = 10$ for individual trait. It is found that, in Approach 1 (score-level fusion), we need to form four separate Kd-trees, one for each trait while in other approaches (Approaches 2 and 3) only a single Kd-tree is needed to index biometric templates. Hence in the case of Approaches 2 and 3, search process is more efficient as it needs to search only one tree. And further, the Approach 3 performs best among all the three approaches, because the maximum number of IDs falling in the first match with minimum FRR.

It is clear from Figure 18 that indexing based on feature-level fusion performs better than the score-level fusion. In feature-level fusion (Approach 3), out of the 150 query templates, 146 IDs fall in the first match, 147 IDs fall in the top two matches and 149 IDs fall in the top five matches with 0.66% FRR. In score-level fusion, 100 IDs fall in the first match and 149 IDs fall in the top ten matches with 0.66% FRR.
6 Conclusion

This paper has proposed an efficient indexing technique based on Kd-tree and feature-level fusion. The proposed technique (Approach 3) can be employed in a large multimodal biometric database to reduce the search space for a query template. Kd-tree has been used for efficient storage and retrieval of records having multi-dimensional feature vectors. The proposed technique (Approach 3) significantly decreases the database search time as it uses only one Kd-tree which supports range search with good pruning. Performance of this technique is compared with the variants of feature-level fusion (Approach 2) and score-level fusion (Approach 1) based indexing techniques. For experimentation, a multimodal database available at IITK consisting of 5400 images of 150 subjects (i.e. nine images per subject, per trait) is used. It is found that the proposed technique (Approach 3) outperforms the other indexing techniques to declare the top matches for any query template. In this technique, out of 150 query templates, 146 IDs fall in the first match, 147 IDs fall in the top two matches and 149 IDs fall in the top five matches with 0.66% FRR.

References

Multimodal biometric databases indexing technique


