International Journal of Information Acquisition
Vol. 8, No. 2 (2011) 133–152

DOI: 10.1142/S0219878911002392

World Scientific
www.worldscientific.com

AN EFFICIENT EIGEN VALUES BASED TECHNIQUE FOR ONLINE IRIS IMAGE COMPRESSION AND IDENTIFICATION

KAMTA NATH MISHRA
Department of Computer Science and Engineering
Birla Institute of Technology Ranchi — Allahabad Campus
B-7, Industrial Area, Naini, Allahabad, U.P., India
mishrakn@yahoo.com

ANUPAM AGRAWAL
Human – Computer Interaction Division
Indian Institute of Information Technology
Allahabad, U.P., India
anupam69@gmail.com

PRAKASH C. SRIVASTAVA*, VIVEK TRIPATHI† and VISHAL GUPTA‡
Department of Computer Science and Engineering
Birla Institute of Technology Ranchi — Allahabad Campus
B-7, Industrial Area, Naini, Allahabad, U.P., India
**prakash_bit123@rediffmail.com*
†*tripathivivek@live.com*
‡*vishal10.87gupta@gmail.com*

Received 8 June 2011
Accepted 29 July 2011

This paper introduces an efficient Eigen values based technique for online iris image compression and identification of a human including the case of identical twins. The iris image is extracted after removing the pupil, eye brow, skin and other noise disturbances from an actual image. The extracted iris image is divided into different blocks of size of 16×16. Now, Eigen values are calculated for each block and these Eigen values are stored in the smart card memory for further identification.

Therefore, when checking if two iris images are identical or not, all we need is to compare the stored Eigen values with online calculated Eigen values. If two iris images have the same Eigen values, this means that both iris images belong to the same person. In our research, we have concluded that iris images of different persons have different Eigen values, including the case of identical twins. We conducted experiments on CASIA and Multimedia University iris image databases and we found that our Eigen Values Based Iris

Image Identification Technique is giving 99.99% accuracy for the same image of identical twins and individuals. The implementation leads us to believe that our method is giving the best matching result in the case of identical twins and individuals. It is an efficient, secure and economically feasible approach for online personal identification.

Keywords: Eigen value; identical twins; iris image compression and identification; iris matching.

1. Introduction

Generally, the face structure or finger print or DNA sequence is used for personal identification. But these methods fail to identify a person in the case of identical twins. The DNA sequence consists of four alphabets A, C, G, T, and N where N represents unknown nucleotides (N is either A or C, or G or T). To store the DNA sequence, we need huge amount of memory. But we can store the Sort Tandem Repeat (STR) sequence of DNA in compressed format and it may be used further for criminal investigations.

Furthermore we cannot use DNA sequence for online identification of a person because there is no standard method which can be used for collecting online DNA Sample [Mishra, 2010].

Therefore, we need a technique which can identify identical twins in minimum time and it should require minimum data storage space. The iris image based identification technique can be used to identify a person including identical twins. Every person has a unique iris image, including identical twins. Therefore, the human iris image is one of the most useful prime biometric features which can be used for distinguishing people. For systems which are based on high quality images, a human iris image can offer an extra amount of unique detail. Features extracted from the human iris image can identify individuals in all cases, including genetically identical twins. A human iris image is fully formed in six months after the birth of the child and it does not change in the case of illness or pregnancy. It is very difficult for a person to modify or change his iris image through plastic surgery whereas it is highly possible for a person to modify or change his face structure and finger prints (including thumb print) through plastic surgery. The voice of a person changes in the case of illness and it can be changed or modified after performing throat surgery. Therefore, iris image based identification technique cannot be used to criminally impersonate other people, as is possible in the case of fingerprint or face structure storage techniques.

The iris image based personal identification technique can be used for the following purposes: National ID Card, Passport, Smartcard, Criminal identification, Automated Banking, Credit card, Government benefit distribution, Home security system, Confidential Database Access, and Home security System [Jain, 1999; Shih, 2010]. But, the iris image of a person takes a large amount of memory (of approximately 450KB), like fingerprint technique and human DNA sequences. There are certain hurdles which includes iris image storage space, accuracy and time requirement for identification. The accuracy includes the problems of False Rejection Rate (FRR) and False Acceptance Rate (FAR) [Doeoteo, 2006]. Time requirement means that the technique should take minimum time in identifying a right person, and it should reject the false cases of identification. Before using iris image based technique for identification, we must overcome these problems. Therefore, the iris image based identification algorithm must not take up too much memory space, and it should not take too much time for identification. It should include FAR and FRR cases. In addition, an acceptable compression speed should be achieved.

For smart card based systems, dynamic and adaptive variable-length encoding are out of question because of the complexity of these algorithms and their huge memory requirements. Thus, run length encoding and static/variable

length encoding techniques are the real alternatives for using them in smart cards. In this paper, we have proposed an Iris Image Compression and Identification technique called Iris Image Compression and Identification System [IICIS]. In this method, iris image is extracted from an eye image.

The extracted iris image will consist of a fixed number of pixels. Each pixel will consist of three colors: Red (R), Green (G) and Blue (B). These pixels will be represented by an integer number. After extracting iris image, we divide the iris ring into different blocks of size 16×16. The last few blocks of the iris image may not have the size of 16×16. Therefore, we need to convert them into the form of 16×16 matrices by multiplying them with an identity matrix. Now, for these matrices, we need to calculate the Eigen values which are enough to identify a person. Each 16×16 matrix will have 16 Eigen values. The Eigen values will be floating point numbers. These floating point numbers (Eigen values) will be stored in the server database for further identification and comparison. Therefore, online iris image compression and identification of a person will have the comparison of stored Eigen values with online calculated Eigen values of an iris image. This online Eigen values based compression and identification method will take minimum comparison time in comparison to other techniques. Our Iris Image Compression and Identification Technique is based on the Eigen values which are obtained from the iris ring of an image. A number of obstacles arise when trying to construct unique irises for images. If we need all the images in the form of pixels, then colored images can be represented in different color spaces.

2. Related Work

The shape DNA based method was used and deployed to compress the DNA sequences. Since the spectrum of the Laplace-Beltrami operator contains intrinsic shape information, it is therefore called "Shape DNA" [Reuter, 2009]. We have seen that this shape-DNA can be used (like DNA test) to identify objects in practical applications. Identical twins exist with different shape but exactly the same shape-DNA. Therefore, shape-DNA based method alone cannot be used for personal identification of human. A variant of Laplace operator has been used for mesh partitioning and compression. This technique is known as spectral compression. Details can be found in Kami [2000] and Ma *et al.* [2002]. Furthermore, the Laplace operator is used for dimensionality reduction of high dimensional data space [Belkin, 2003]. Belkin and Niyogi used the Eigen values of manifold defined by points in a given featured space. The Belkin and Niyogi method is useful if there is a set of features present which will be reduced for some reasons. The Belkin and Niyogi method is classified as Multi Dimensional Scaling method. Although it is implemented differently from the classical approaches, its performance is similar to the performance of classical approaches [Wildes, 1997]. The iris image has been a feature of choice for many biometric systems. The basic idea about iris and its compression was given by Flom and Safir in the year 1987 [Folm, 1987]. Since then, so many techniques and software systems have been developed. One of the most advanced and applied iris system was developed by Daugman [Daugman, 1999, 2007]. The use of 2D Gabor wavelets has demonstrated low probability of false positives at the cost of relatively high frequency of false negatives in numeral trials. Similar technology has been developed by performing pattern matching via Gabor filter-bank after combining the Hough transform for iris localization [Tisse, 2003; Ma, 2002]. Finally, many systems have been developed which are based on wavelets, and winner selection methods. But it is little difficult to compare the performance of existing techniques because of late availability of CASIA benchmark database [CASIA, 2010]. Iris compression for Cryptographically Secure Personal Identification has been developed recently which is based on Fourier Miller Transform [Daniel, 2004]. If we are using iris image based techniques for online identification of a person for a highly secured zone including the case of nuclear reactor, then the person who wants to access the nuclear reactor of an atomic research centre will insert his smart card and then he will

have to stand in front of the camera of Iris Image Compression and Identification System (IICIS).

The camera will take the iris image of the person. Now the skin, eyebrow, pupil, and other noise disturbances will be separated from the iris image. After separating these noise disturbances, we will get the iris in its purest and actual format. This iris image of the person will be compared with stored Eigen values of the iris images of the smart card. Here, we need a fast iris image data comparison and its retrieval technique. At the time of online identity verification of a person, the system should accurately verify the identity of the person within the shortest time. To minimize the identification time, we need to store the iris image in compressed format. That is, in the form Eigen values. Thus, we need to compress the iris image such that it should take minimum comparison and transmission time. The compressed iris image data will be stored in the form of Eigen values in the smartcard. The employee of the nuclear reactor will insert his smart card in the system and the system will compare the online calculated Eigen value of iris image of the person with the stored Eigen values of the smartcard. If online calculated Eigen values are same as the stored Eigen values, the system may display a special beep which confirms that the person is an authentic person and the person will be permitted to enter the nuclear reactor. If the Eigen values stored in the smartcard are not matching with the online calculated Eigen values, then it means that the person is not an authentic person and the person will not be permitted to access the nuclear reactor. This will be considered as the compressed format of iris image. At the time of designing the smartcard, we will have to store the employee code as the person's ID in the smartcard and this employee code of the smart card will first be matched with employee code stored in the server. Once the validity of smart card is checked, then the Eigen values stored in the smart will be compared with the online calculated Eigen values. Smaller numbers of papers deal specifically for determining the parts of the iris region that are occluded by eyelids and eyelashes. Occlusions due to eyelashes are sometimes referred to as "noise" [Bowyer, 2008].

3. Theoretical Background

In this section, we will explain the theoretical and mathematical background required for online iris image compression and identification. A method for uniquely identifying a particular human being by biometric analysis of iris of the eye, comprises of the following steps [Matey, 2010]:

- Acquire an image of an eye of the human to be identified.
- Isolate and define the iris of the eye within the image.
- Analyze the iris to generate a presenting iris code.
- Compare the mentioned presented code with a previously generated reference iris code to generate a measure of similarity between the mentioned presented iris code and the mentioned reference code.
- Convert the (as mentioned) similarity measure into a decision that (as mentioned) iris codes either do or do not arise from the same iris.
- Calculate a confidence level for the decision.

In our Iris Image Compression and Identification method, online Iris image compression technique has been divided into five phases. In the first phase (introduction phase), we have described the basics of iris image and its colors. In the second phase (preprocessing phase), we have converted the iris image into binary image for its processing. In the third phase (radii and centre estimation phase), we have estimated the centre and radii of the iris image. In the fourth phase (segmentation phase), we have removed the noise disturbances from an iris image and then divided it into different blocks of same size. In the fifth phase (processing phase), we have converted each block of the iris image into different matrices of size 16×16. For each matrix, Eigen values are calculated and these Eigen values are used for further human identification.

3.1. *Introduction phase*

Every digital image is made up of small dots known as pixels. Each pixel is in turn made up of three different colors R, G, and B (Red, Green

and Blue). Each color forms a different layer of the image, which, when superimposed on one another, gives rise to the complete image. Digitally, each layer can be represented as a two dimensional matrix, where each element of the matrix ranges from 0 to 255. Each component (R, G and B) of the image is represented by a single two-dimensional matrix in digital format, which converts it into three-dimensional structures. For our purpose, we just need one dimension of the image, and preferably, we will work with the Green component of the image in matrix format although we can work with either green or red or blue component of the image. Here, our research work can be divided into several stages. The first stage consists of obtaining the actual image. That is, extracting a single dimension of the image and then performing other operations. This stage is called the pre-processing phase.

Now we need to calculate the centre, inner radii and outer radii of the actual iris ring. This stage will be called the Estimating Radii and Centre of Iris Ring. Once we complete this stage, then we need to extract the iris ring and resize the image, depending upon the outer radius. This phase will be called the Segmentation phase. After the segmentation phase, we need to divide the actual image matrix obtained from the Segmentation phase into smaller matrices and calculate the Eigen value for each of the smaller part. This phase will be called the processing phase.

3.2. *Pre-processing phase*

This is the initial phase of the procedure which is applied for an input image of iris to obtain the result in compressed format.

In this phase, we obtain a process-able format of the image from the actual raw image. The raw image will be obtained from a high-resolution digital camera. As mentioned in the introduction section, an image consists of three two-dimensional matrices corresponding to Red, Green, and Blue components of the image.

Here, we have used MATLAB version 7 to carry out all our experimentation. MATLAB reads an image as three-dimensional structure of numbers, where each component of the third dimension represents a layer of numerical values corresponding to R, G and B. In MATLAB, this task is performed by the following statement:

$$\mathrm{g} = \mathrm{img}(:, :, 2).$$

Here, 'g' is the variable which holds the second layer of the image and it corresponds to the Green color. 'img' is the actual three dimensional image in the form of a three dimensional matrix. (, : and ; are MATLAB operators). A Binary image is the image whose elements are either zero or one. Here, in our case, if an element in the green component is equal to zero, then it will be taken as zero and if it is greater than zero, then it will be taken as one in the corresponding Binary image. Once we have obtained the green component, then we convert this component into the corresponding binary image. A green component of iris image looks like the following figure (Fig. 2). In order to convert a green image into black and white image, we have used an inbuilt function of MATLAB. The statement accomplishing this task is as follows:

$$\mathrm{gbw} = \mathrm{im2bw}(\mathrm{g}).$$

Here, 'gbw' is the variable which holds the Binary Image and 'im2bw' is the function which converts an image to binary image. Figure 3 is

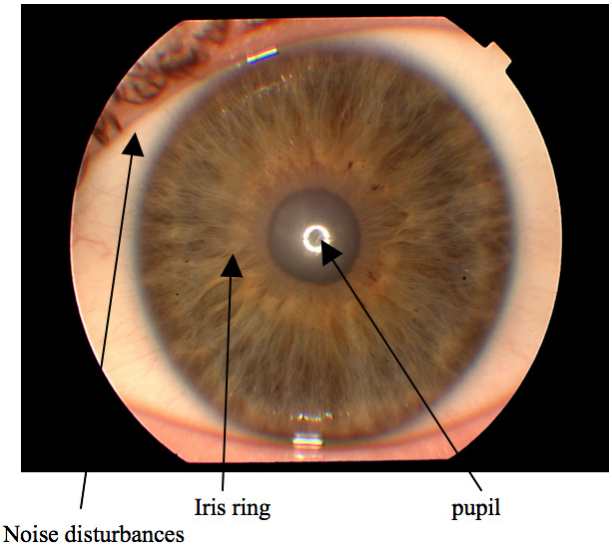

Fig. 1. A true color iris image.

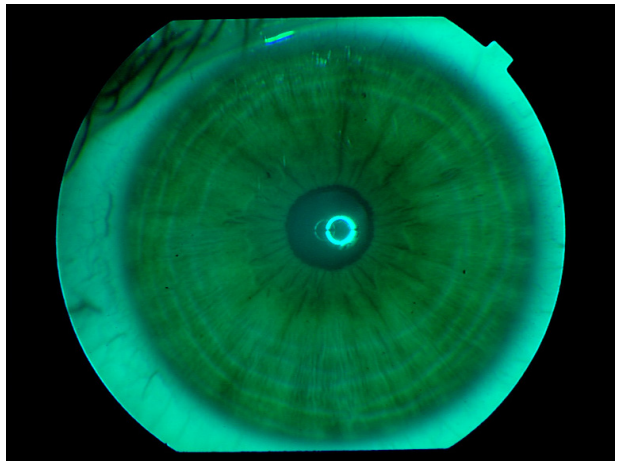

Fig. 2. Green component of image.

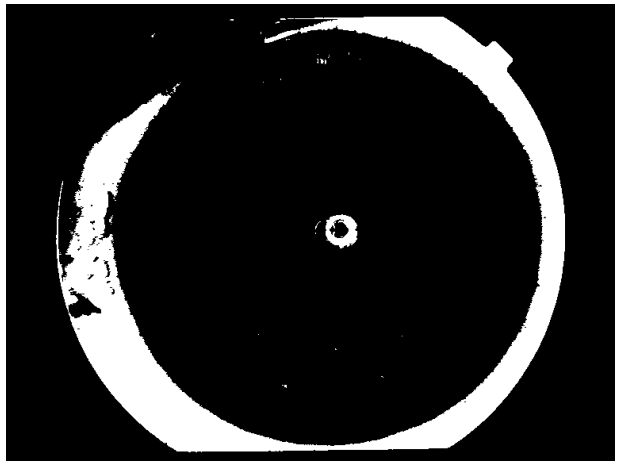

Fig. 3. Image converted to two colors (Black and White).

showing the black and white representation of Fig. 2.

Once the image has been converted into Binary Image, then this pre-processing phase will be completed and the output of this phase i.e., 'gbw' form of the Binary image will be used as the input to the next phase.

3.3. *Estimating radii and centre of iris ring*

In this phase, we will calculate the parameters which are defining the iris ring. The Iris Ring can be thought of as a portion of the image which is obtained after the removal of pupil and the portions of the image which is lying outside of the outer boundary of iris.

The iris ring and its other components can be easily visualized by Fig. 1. In this figure, all the portions of the image are shown. Here, two concentric black circles show the iris ring which is to be extracted. In order to extract this ring from a raw iris image, we need to calculate following three parameters:

(1) The coordinates of the Centre of iris image (g, h).
(2) The radius of the inner circle of the iris ring (ri).
(3) The radius of the outer circle of the iris ring (ro).

After calculating these parameters, we will proceed with the extraction of the iris image. In order to calculate these three parameters (gh, ri, & ro) we have adapted a separate method which has been implemented in MATLAB.

In this procedure, the output image 'gbw' obtained in the previous phase is taken as the input and the three parameters (namely 'gh', 'ri', and 'ro') are obtained as the output. These output variables will be used as input parameters to the next phase.

3.4. *Segmentation phase*

In this phase, the actual extraction of the iris ring is to be carried out, depending upon the parameters obtained in the previous phase. This phase proceeds in two parts.

3.4.1. *Removing noise disturbances and extracting iris ring*

In this part the unwanted portions of the image are removed and only the iris ring is retained. This is called the extraction of iris ring. We have written a customized MATLAB function in order to accomplish this task. The name of the function is 'extractIris( )'. This function takes the parameters obtained in the segmentation phase and the green component of actual iris image as the input and gives another image of

the same dimensions as the actual image, containing only the iris ring as the output. The image containing the iris ring is the same size as the actual image. Therefore, it contains some unwanted space in it. Hence, we need to go to the second part of the segmentation phase. For purpose if reference, we call this output image as 'gRing'.

3.4.2. *Resizing iris image*

In this part, we remove the extra space of the iris ring of Part 1. In other words, we can say that we now resize the 'gRing' image according to the radius of the iris ring in order to remove the unwanted space. For this purpose, we have developed a MATLAB function named 'resizeM', which resizes the image according to the radius of the iris ring. This function takes the image 'gRing' and the outer radius of the iris ring 'ro' as input. The returned output is the resized image. The variable name used for resized image in our research work is 'reszgRing'. This output will be the actual resized image of the iris ring. The output of Part 2 will be used as input to the next phase. The resized image obtained after reducing the circumference will be represented as Fig. 4.

Once the image is resized, we can proceed to the next phase called processing phase.

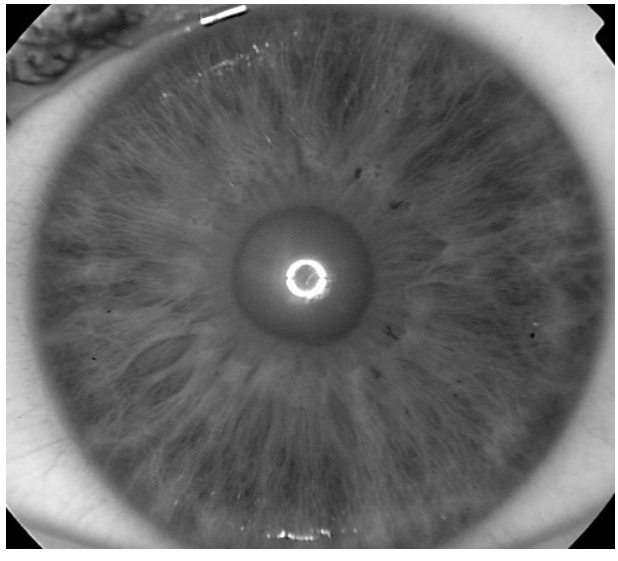

Fig. 4. Resized image after reducing the circumference of iris.

3.5. ***Processing phase***

In this phase, the actual compression of the iris image is carried out. This phase is carried out in two steps.

Step 1: *Converting iris ring into smaller matrices*

We have explained that all these images will be in the form of two-dimensional matrices. Hence, the output image of the previous phase, i.e., 'reszgRing' will be in the form of a two-dimensional matrix. Now, we divide this bigger matrix into smaller 16×16 matrices. The input image should be a square matrix. If it is not, then we make it square either by adding or removing columns or rows. This task of dividing the matrix is carried out by an indigenous MATLAB function named 'divMat( )'. This function takes the resized image, i.e., 'reszgRing' and the dimension of the smaller matrices as input and returns the divided smaller sized matrices in another three dimensional matrix. The smaller matrices obtained in this stage are used as the input to next stage.

Step 2: *Calculating the Eigen values*

In this stage, we calculate the Eigen values of the smaller matrices and store them in a linear array. The task of calculating Eigen values is carried out by our indigenous MATLAB function 'calEig'. In this method, we use characteristic equation of a matrix for calculating Eigen values. The characteristic equation is given by following mathematical function:

$$|\mathrm{A} - \mathrm{CI}| = 0. \tag{1}$$

In Eq. (1) mentioned above, 'A' is the matrix whose Eigen value is to be calculated, 'C' is a numeric constant and 'I' is an identity matrix whose order is same as that of 'A'. Here, the determinant of 'A–CI' results in a polynomial in 'C' whose degree is the same as the order of the matrix 'A', say P(C). Let the order of the matrix 'A' be 'nxn'. Hence, the degree of the polynomial P(C) will be 'n'. We will have 'n' solutions of the polynomial P(C). Each of these 'n' solutions is called the Eigen value of the matrix 'A'. These Eigen values will be considered as the compressed format of the iris image

and these Eigen values can be further used for comparison and identification of a person.

4. Mathematical Model and Steps of IICIS

We have designed the structure of Iris Image Compression and Identification System (IICIS) mathematically in this phase. The mathematical structure is the abstract representation of IICIS which specifies the following:

(i) The inputs and outputs of IICIS at each state of processing.
(ii) The number of discrete states/stages encountered from start to the end of processing.
(iii) The interconnection between inputs, outputs and processing states.
(iv) The flow and data manipulation of different processing states of IICIS.

4.1. *Definition of IICIS*

The structure of IICIS can be defined by using the following six tuples:

$$\text{IICIS} = \{\text{Q}, \Sigma, \sigma, \delta, \lambda, \text{q}_0\}. \tag{2}$$

The symbols of Eq. (2) above are defined as follows:

Q: finite, non-empty set of states.
Σ: finite, non-empty set of input matrices.
σ: finite, non-empty set of output matrices.
δ: state transition function, which brings IICIS to the next state. The next state depends on the previous state. The transition function will be

$$\delta : \Sigma^* \times Q \to Q.$$

λ: output function. The output depends on the state and the input elements. The output function can be defined as:

$$\lambda : \Sigma^* \times Q \to \sigma^*.$$

Here, we should note that input and output may consist of multiple elements of Σ and σ respectively.

q_0: the initial or starting state. The IICIS will start from this state, which is an element of Q, i.e., $\text{q}_0 \in Q$.

Now, we consider each state as a single processing unit where the actual processing on the input data/matrix is carried out. The output is produced in the form of output matrix, once the processing task is completed.

4.2. *Description of IICIS components*

The detailed description of all the components of IICIS is explained as follow:

(i) *Set of states* (Q)

Each state of IICIS can be considered as a discrete unit of processing in which the inputs and outputs are identified clearly. The processing carried out at each state can be represented by a method which will take an input and return the output.

(ii) *The input/output sets* (Σ, σ)

The input alphabet set Σ is the set of all those elements which are given as input to IICIS at various states. Mathematically, $\Sigma = \{x \mid x$ is an alphabet taken as input by a state of IICIS$\}$

The elements of Σ are matrices of order $i \times j \times k$.

$$\Sigma = \{\text{x} \mid \text{x}_{i\times j\times k} \text{ is a matrix of order } i \times j \times k\}$$

Let $\text{A} \in \Sigma$,
$\Rightarrow \text{A}_{1\times1\times1}$ is the simplest element in Σ, which is a real number.

If
$\text{A} \in \Sigma$, then $[\text{A}] = \text{a}_{l,m,n} = \text{a}(l, m, n)\ \forall\, l, m, n \in \{\mathbb{Z}^+ - 0\}$.
Here, $\text{a}_{l,m,n}$ is an element of matrix A belonging to the lth row, mth column and nth layer. The following conditions must hold for element 'a':

- $0 < \text{a}_{l,m,n}$
- $\text{a} \in \mathbb{Z}^+$

The output alphabet set σ is the set of all those elements which are obtained as output after using various states of IICIS. The following are true for σ:

$$\Sigma = \{x | x_{i\times j\times k} \text{ is a three dimensional matrix}\}$$

If A $\in \Sigma$, then elements of matrix A can be denoted by $a_{i\times j\times k}$. Matrix A can be denoted by [A]

$$\Rightarrow [\mathrm{A}] = \mathrm{a}_{i\times j\times k} \quad \forall \mathrm{i},\ \mathrm{j},\ \mathrm{k}\ \in \{Z^+ - 0\}$$

i, j, k represents row, column and plane respectively.

The simplest element of σ will be $\mathrm{A}_{1\times 1\times 1}$ which is a single element of $\{\mathbb{Z}^+ - 0\}$.

(iii) *The state transition function* (δ)

The state transition function directs the flow of control from one state to another. δ also determines the input required to move from one state to another and the output generated while transition is taking place. δ can be defined as follows:

$$\delta : \Sigma^* \times Q \rightarrow Q. \tag{3}$$

In Eq. (3), Σ^* is the set of all strings formed by the elements of Σ.

(iv) *The output function* (λ)

This function displays the output generated at each state. The definition of λ is as follow:

$$\lambda : \Sigma^* \times Q \rightarrow \sigma^* \mid \lambda : \Sigma^2 \times Q \rightarrow \sigma^2 \tag{4}$$

In Eq. (4), $\Sigma^2 = \{\mathrm{x} \mid \mathrm{x} \leq 2 \text{ and } |\mathrm{x}| > 0\}$, and $\sigma^2 = \{\mathrm{y} \mid \mathrm{y} \leq 2 \text{ and } |y| > 0\}$.

The actual values of each tuple for the equations above will be represented by following equation above (Eq. (5)):

$$\begin{aligned} Q &= \{\mathrm{q}_0, \mathrm{q}_1, \mathrm{q}_2, \mathrm{q}_3, \mathrm{q}_4, \mathrm{q}_5, \mathrm{q}_6, \mathrm{q}_7\}, \\ \Sigma &= \{\mathrm{I}, \mathrm{I_G}, \mathrm{I_{GBW}}, \mathrm{I_{RSZ}}, \mathrm{I_{EXT}}, \mathrm{C}, \mathrm{D}, \mathrm{I_{EIG}}\}, \\ \sigma &= \{\mathrm{I_G}, \mathrm{I_{GBW}}, \mathrm{I_{RSZ}}, \mathrm{I_{EXT}}, \mathrm{C}, \mathrm{D}, \mathrm{I_{EIG}}, \mathrm{M}\}, \\ \delta &= \delta\colon \Sigma^2 \times Q \rightarrow Q, \\ \lambda &= \lambda : \Sigma^2 \times Q \rightarrow \sigma^2. \end{aligned} \tag{5}$$

4.3. *Significance of mathematical model with the actual machine*

The processing tasks carried out by IICIS are divided into six algorithms. Each of these algorithms can be represented by the states of IICIS.

Now, if a $\in \Sigma^2$ and q $\in Q$, then δ (q, a) shows the processing carried out on 'a' $\in \Sigma^2$, at state q $\in Q$. Here q represents an algorithm.

Basically, δ maps an algorithm and an input symbol of Σ^2 onto the next algorithm, which is to be executed. Similarly, the output function σ maps an input and an algorithm to an output symbol.

The description of δ and λ functions will be as follows:

$$\begin{aligned} \delta(\mathrm{q}_0, \{\mathrm{I}\}) &= \mathrm{q}_1, \lambda(\mathrm{q}_0, \{\mathrm{I}\}) \\ &= \{\mathrm{I_{GBW}}, \mathrm{I_G}\}, \\ \delta(\mathrm{q}_1, \{\mathrm{I_{GBW}}\}) &= \mathrm{q}_2, \lambda(\mathrm{q}_1, \{\mathrm{I_{GBW}}\}) = \mathrm{C}, \\ \delta(\mathrm{q}_2, \{\mathrm{I_G}, \mathrm{C}\}) &= \mathrm{q}_3, \lambda(\mathrm{q}_2, \{\mathrm{I_G}, \mathrm{C}\}) = \{\mathrm{I_{RSZ}}\}, \\ \delta(\mathrm{q}_3, \{\mathrm{I_{RSZ}}, \mathrm{C}\}) &= \mathrm{q}_4, \lambda(\mathrm{q}_3, \{\mathrm{I_{RSZ}}, \mathrm{C}\}) \\ &= \{\mathrm{I_{EXT}}\}, \\ \delta(\mathrm{q}_4, \{\mathrm{I_{EXT}}\}) &= \mathrm{q}_5, \lambda(\mathrm{q}_4, \{\mathrm{I_{EXT}}\}) = \mathrm{D}. \\ \delta(\mathrm{q}_5, \{\mathrm{D}\}) &= \mathrm{q}_6, \lambda(\mathrm{q}_5, \{\mathrm{D}\}) = \{\mathrm{I_{EIG}}\}. \\ \delta(\mathrm{q}_6, \{\mathrm{I_{EIG}}, \mathrm{I_{EIG}}\}) &= \mathrm{q}_7, \lambda(\mathrm{q}_6, \{\mathrm{I_{EIG}}, \mathrm{I_{EIG}}\}) \\ &= \{\mathrm{M}\}. \end{aligned} \tag{6}$$

In Eq. (6), I is a True color image of three dimension, $\mathrm{I_G}$ is green component of image, $\mathrm{I_{GBW}}$ is black and white conversion of green image, $\mathrm{I_{RSZ}}$ is resized image, $\mathrm{I_{EXT}}$ is extracted image having iris ring only, C is the matrix having coordinates of centre and approximate radius of iris, D is a three-dimensional matrix with size of $16 \times 16 \times$ m having size of 16×16 divided matrix of extracted image $\mathrm{I_{EIG}}$ is a two-dimensional matrix with size of $16 \times$ m, having Eigen values of all m number of 16×16 matrices, and M is a matrix containing match result of the two images.

4.4. *Steps of IICIS*

The Iris Image Compression and Identification System (IICIS) uses six steps for completing the task of online iris image compression and identification. We have designed these steps in the form of different programmes using MATLAB 7. The steps of IICIS are as follow:

Step 1: *Getting the centre and radius of an iris image*

This step reads the main m $\times$ *n* image and performs the analysis of this image to gauge the

actual centre of iris according to the image pattern. It takes an arbitrary centre on the image and then traces the row from centre of image to both sides for reaching the iris border. After that, it fixes the centre on repeated examination of the border and calculates the radius of the iris image.

The input to Step 1 will be a m × n × 1 image, midpoint of the width of the image (gs), midpoint of the height of the image (hs), and the limit of continuous white pixels (lt).

The algorithm will have the following steps:

```
Algorithm getCircle( )
{
1. convert image[ ][ ] =
MGTOBLACKWHITE(image[ ][ ]);
2. d = GETDIMENSIONS(image[ ][ ]);
3. g = gs;
4. h = hs;
5. height = d(1)/2;
6. width = d(2)/2;
7. new_matrix rads[d(2)][1] = 0;
8. wquad1=0,wquad2=0,wquad3=0,wquad4=0;
9. bquad1=0,bquad2=0,bquad3=0,bquad4=0;
10. limit=lt;
11. FOR i = 0 TO height-1 STEP 1
        11.1. new_matrix rowX[] =
        image[h-1][:];
        11.2. new_matrix rowNX[] =
        image[h+1][:];
        11.3. FOR j = 0 TO width-1 STEP 1
        11.3.1. IF(wquad1<limit)
        11.3.1.1. IF (rowX[g+j] == 0)
        11.3.1.1.1. bquad1= bquad1+1
        +wquad1;
        11.3.1.1.2. wquad1=0;
        11.3.1.2. ELSE IF (rowX[g+j]==1)
        11.3.1.2.1. wquad1=wquad1+1;
        11.3.2. IF (wquad2<limit)
        11.3.2.1. IF (rowX[g-j]==0)
        11.3.2.1.1. bquad2=bquad2+1+
        wquad2;
        11.3.2.1.2. wquad2=0;
        11.3.2.2. ELSE IF (rowX[g-j]==1)
        11.3.2.2.1. wquad2=wquad2+1;
        11.3.3. IF (wquad1>=limit AND
        wquad2>=limit)
        11.3.3.1. rads[i+1]=(bquad1+
        bquad2)/2;
        11.3.3.2. BREAK;
        11.4. FOR j = 0 TO width-1 STEP 1
        11.4.1. IF (wquad3<limit)
        11.4.1.1. IF (rowNX[g-j]==0)
        11.4.1.1.1. bquad3=bquad3+1+
        wquad3;
        11.4.1.1.2. wquad3=0;
        11.4.1.2. ELSE IF (rowNX[g-j]==1)
        11.4.1.2.1. wquad3=wquad3+1;
        11.4.2. IF (wquad4<limit)
        11.4.2.1. IF (rowNX[g+j]==0)
        11.4.2.1.1. bquad4=bquad4+1+
        wquad4;
        11.4.2.1.2. wquad4=0;
        11.4.2.2. ELSE IF (rowNX[g+j]==1)
        11.4.2.2.1. wquad4=wquad4+1;
        11.4.3. IF (wquad3>=limit AND
        wquad4>=limit)
        11.4.3.1. rads[i+2]=(bquad3+
        bquad4)/2;
        11.4.3.2. BREAK;
    12. RETURN rads;
    }
```

Step 2: *Resize the image according to the iris ring*

This step accepts the image, radius, and centre of iris image as inputs and plots circle around the iris ring [World Wide Iris Database, 2010]. Once the circle is plotted, it crops the image from all four sides such that the image border touches the iris ring from its outer circle on all the four sides.

The inputs to step-2 will be a m × n × 1 image[ ] [ ], outer radius of the iris (rad), centre of the iris (h, g) and it will return the image size (ims). The resizecen() function will have the following steps:

```
Algorithm resizecen( )
{
1. d = GETDIMENSION(image[ ][ ]);
2. x = h, y = g;
3. j = 0;
4. FOR i = y-rad TO y+rad
        4.1. j = j+1;
        4.2. IF (i < d(2) AND i > 0)
          4.2.1. new_matrix im[:][j] =
          image[:][i];
5. j = 0;
```

```
6. FOR i = x-rad TO x+rad
        6.1. j = j+1;
        6.2. IF (i < d(2) AND i > 0)
        6.2.1. new_matrix ims[:][j] = im[i][:];
7. RETURN ims;
}
```

Step 3: *Extracting the ring iris from the main image*

This step takes the m × n × 1 image matrix, i.e., the green image, inner radius, outer radius of iris, and it produces an image of the same size which consists of only the iris ring. In this step, all the noise disturbances outside of the iris ring and inside of the iris ring are removed [Daugman, 1999, 2004].

The algorithm will have the following steps:

```
Algorithm extractIris( )
{
1. d = GETDIMENSIONS(image[ ][ ]);
2. x = ROUND(d(1)/2 );
3. y = ROUND(d(2)/2 );
4. peri = 2 * 3.14 * r;
5. new_matrix ims[d(1),d(2)];
6. FOR i = 0 TO 2*PI STEP 1/r1
      6.1. IF( (r1*sin(i) + x) > 0
      AND (r1*cos(i) + y) > 0)
        6.1.1. xi = ABS(r1*sin(i) + x);
        6.1.2. yi = ABS(r1*cos(i) + y);
        6.1.3. IF (i>=0 AND i<=(PI/2))
        6.1.3.1. ims[xi:d(1)][yi:d(2)]=im[xi:d(1)]
[yi:d(2)]
        6.1.4. ELSE IF (i> PI/2 AND i<=PI)
        6.1.4.1. ims[xi:d(1)][1:yi] =
im[xi : d(1)][1 : yi];
        6.1.5. ELSE IF (i>PI AND i<=
3*PI/2)
        6.1.5.1. ims[1 : xi][1 : yi] = im[1 : xi][1 :
yi];
        6.1.6. ELSE IF (i> 2*PI/3 AND i<=
2*PI)
        6.1.6.1. ims[1 : xi][yi : d(2)] = im[1 : xi][yi:
d(2)];
7. FOR i = 0 TO 2*PI STEP 1/r
        7.1. IF ((r*sin(i) + x) > 0 AND (r*cos(i)
+ y) > 0)
        7.1.1. xi = ABS(r*sin(i) + x);
        7.1.2. yi = ABS(r*cos(i) + y);
        7.1.3. IF (i>=0 AND i<=(PI/2))
        7.1.3.1. ims[xi : d(1)][yi : d(2)] = 0;
        7.1.4. ELSE IF (i> PI/2 AND i<=PI)
        7.1.4.1. ims[xi : d(1)][1 : yi] = 0;
        7.1.5. ELSE IF (i>PI AND i<=
3*PI/2)
        7.1.5.1. ims[1 : xi][1 : yi] = 0;
        7.1.6. ELSE IF (i> 2*PI/3 AND i<=
2*PI)
        7.1.6.1. ims[1 : xi][yi : d(2)] = 0;
8. RETURN ims;
}
```

Step 4: *Dividing iris ring into square matrices*

This step reads the iris image having the resized iris ring and then it divides the iris ring into smaller square matrices. The dimensions of the new matrices are also specified in step 4.

The input to step 4 will be the resized image and dimensions of the new smaller matrices (dim). This algorithm will have the following steps:

```
Algorithm divMat( )
{
    1. FOR i = dim TO image_width STEP
       dim
         1.1. FOR j = dim TO image_height
         STEP dim
         1.1.1. x[:][:] = rszimg[i-(dim-1):i]
         [j-(dim-1):j];
         1.1.2. IF (ANYELEMENT(x) > 0)
         1.1.2.1. k = k + 1;
         1.1.2.2. NEWIMAGE imr[:][:][k] = x;
    2. RETURN imr[ ][ ][ ];
}
```

Step 5: *Calculating Eigen values*

This step takes the pixel matrix as input and calculates the Eigen values for each matrix. These Eigen values will be stored in another matrix. The Eigen value calculation matrix will have following steps:

```
Algorithm calEig( )
{
1. d = GETDIMENSION(mat[ ][ ][ ]);
2. accumulator j = 1;
3. FOR i = 1 TO d STEP 1
        3.1. z = EIGEN(mat[:][:][i]);
        3.2. new_matrix modz[] =
        ABSOLUTE(z);
```

3.3. new_matrix egmat[1:16][i] = modz;
4. RETURN egmat;
}

Example: If the actual image matrix (544×544) is divided into smaller matrices of the order 16×16, then the matrix (mat) will contain all the 16×16 matrices in the three dimensions as $16 \times 16 \times 34$.

Step 6: *Comparing two iris images using the Eigen values*

This step compares the Eigen values of the stored image with the online calculated Eigen values. The algorithm for the comparison of the stored Eigen values of an iris image with its online calculated Eigen values will be as follows:

```
Algorithm compareEig( )
{
1. eigmat1=CONVERT2DMATTO1DMAT
        (eigmat1);
2. eigmat2=CONVERT2DMATTO1DMAT
        (eigmat2);
3. hit=0,mis=0, found=0,tot=0;
4. d1=GETDIMENSION(eigmat1);
5. d2=GETDIMENSION(eigmat2);
6. FOR i=1 TO d1(2) STEP 1
      6.1. found = 0;
      6.2. FOR j=1 TO d2(2) STEP 1
        6.2.1.IF (eigmat1 [i]==eigmat2[i][j])
          6.2.1.1. found=1;
          6.2.1.2. BREAK;
      6.3. IF (found == 1)
        6.3.1. hit=hit+1;
      6.4. ELSE
        6.4.1. mis=mis+1;
      6.5. tot=tot+1;
7. hitper=((hit/tot)*100);
misper=((mis/tot)*100);
8. comparison.hitPer=hitper;
9. comparison.misPer=misper;
10. comparison.eigValHit=hit;
11. comparison.eigValMis=mis;
12. comparison.totEigVals=tot;
13. RETURN comparison;
}
```

Steps 1 to 6 represent a system called Iris Image Compression and Identification System (IICIS) which computes online Eigen values of an iris image of a person and compares these Eigen values with the stored Eigen values. In the next phase (Experimental Analysis of Results phase), we will analyze the performance of our algorithms for CASIA database.

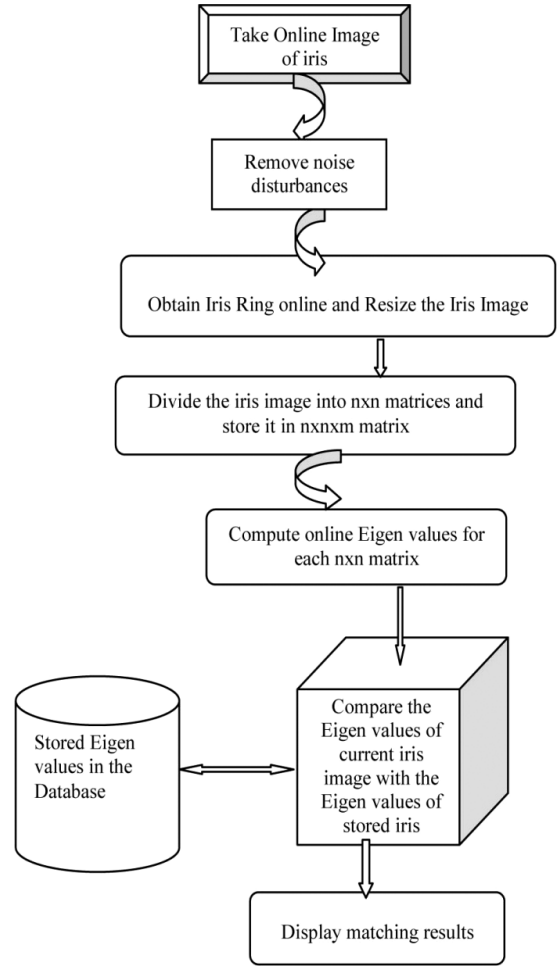

Fig. 5. Online Identification of Iris Image using Eigen values.

The complete procedure for online identification of a person using Eigen values can be explained by the flowchart as shown in Fig. 5. The flowchart of Fig. 5 shows the steps of comparing online calculated Eigen values of an iris image with the stored Eigen values. The performance of IICIS for an individual and identical twins are explained in the result analysis phase.

5. Experimental Analysis of Results with Actual Data

In this section, we have displayed the results obtained at the various steps with the execution

Table 1. Sample images and its size in KBs.

S. No.	Image	Dimensions	Size (in KB)
1	p1le1	576 × 768	448
2	p2re2	576 × 768	468
3	p3le1	576 × 768	456

of our procedure mentioned in IICIS phase. In order to verify our procedure, we have used three sample images obtained from three different individuals of CASIA database. The details of these images are as mentioned in Table 1.

We started with a consideration of the raw image. The raw image is a true color image which is taken with the help of a high-resolution camera. It is the image represented as a combination of three primary colors: **R**ed, **G**reen and **B**lue, i.e., RGB. A computing device interprets this image as a three-dimensional matrix, with each dimension representing one of the three primary color components R, G or B. A pixel comprises of some combination of values of these three primary color components [Iridian Technologies, 2009]. The raw images of three samples listed in Table 1 are shown in Fig. 6.

After obtaining the original true color image in the three-dimensional matrix format, one of the three-dimensions is extracted and we proceeded with the noise disturbance removal, as

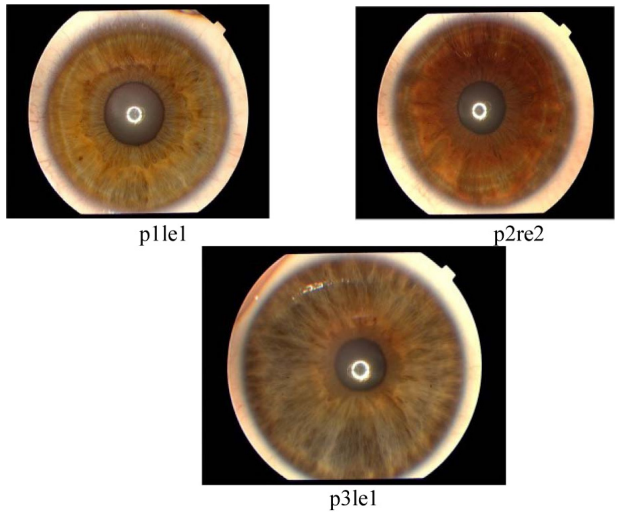

Fig. 6. The raw images of three samples.

explained in the theoretical background. After the removal of noise disturbances, we calculated the centre and radii of the iris ring. The results obtained while calculating the centre and radii of the iris ring for sample images of Table 1 are shown in Table 2.

Now, we divide the resultant image matrix into smaller matrices of dimensions 16×16. The segmented iris ring obtained after removing the pupil using our method will be like Fig. 7 [Balaji, 2006]. The pixels representing Image_Mat_1, Image_Mat_2, and Image_Mat_3 are shown in Fig. 8.

After dividing the actual iris ring into smaller matrices of size 16×16, we calculate the Eigen values. For each 16×16 matrix, we got 16 Eigen values. These Eigen values are floating

Table 2. Centre and radius of image sample.

S. No.	Image	Centre G	Centre H	Radius (in pixels)	Time taken (in ms)
1	p1le1	389	293	278	1.478
2	p2re2	387	280	274	0.796
3	p3le1	439	314	280	1.418

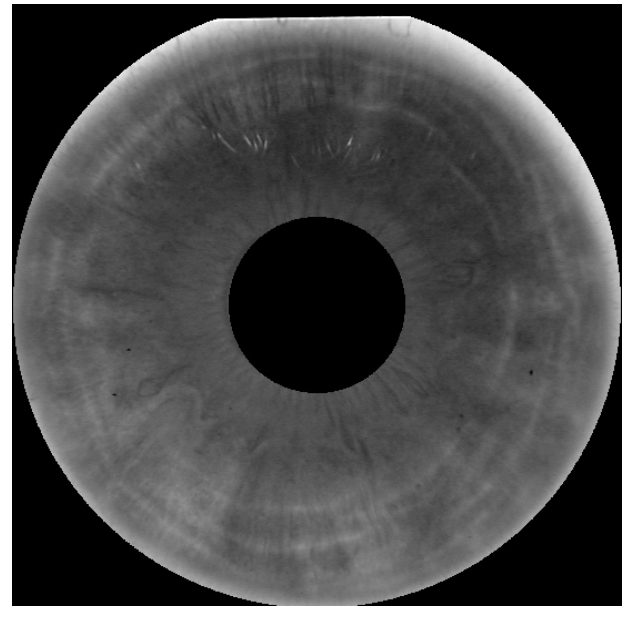

Fig. 7. Segmented Iris Ring obtained after removing pupil.

Image_Mat_1 (286th matrix of p1le1)

{{73,75,81,79,76,78,80,79,79,78,75,73,79,83,81,80},
{75,76,75,74,77,82,81,81,80,78,80,83,81,81,84,84},
{76,79,79,79,79,79,80,80,79,79,79,80,82,85,88,87},
{75,77,82,84,81,81,84,81,79,81,82,83,83,87,86,85},
{80,81,79,79,78,76,82,86,81,80,83,84,82,83,85,83},
{78,80,79,79,80,79,78,81,79,81,83,81,81,81,82,84},
{78,81,81,78,77,78,77,77,80,81,78,77,77,77,76,79},
{79,83,84,83,84,81,81,81,81,81,80,81,82,80,76,77},
{84,85,85,80,80,80,80,83,81,82,85,84,83,80,80,83},
{85,85,82,80,79,82,87,84,83,87,86,81,82,85,84,84},
{85,84,83,82,80,82,83,84,84,83,83,83,83,86,87,89},
{84,81,81,80,81,80,79,84,82,83,86,86,87,88,86,86},
{82,82,79,76,79,85,88,88,87,87,91,91,89,87,87,92},
{81,78,80,79,78,77,81,85,85,86,85,86,89,89,90,94},
{82,83,83,82,81,80,82,85,87,87,87,86,86,89,92,93},
{81,82,82,82,80,83,84,82,84,85,87,86,87,88,88,88}}

Image_Mat_2 (842nd matrix of p2re2)

{{99,95,96,96,91,85,85,85,85,84,83,79,75,78,80,80},
{99,95,93,95,89,82,82,85,84,83,81,80,80,78,76,78},
{97,95,91,90,92,90,88,85,81,81,83,80,77,75,76,76},
{100,99,99,99,95,88,86,85,83,83,83,86,83,79,77,77},
{109,109,106,103,100,94,91,90,87,85,90,92,87,82,78,
72},{109,113,118,119,111,102,94,92,97,99,96,89,86,
86,85,82},{113,118,122,126,119,108,101,95,102,114,
118,101,94,90,87,88},{113,116,123,130,121,104,101,
105,111,138,155,116,92,88,87,91},{106,110,115,114,
112,106,98,103,111,136,156,128,103,92,90,95},{103,
102,104,103,101,95,93,99,107,119,132,132,107,89,87,92},
{102,102,102,97,94,95,92,90,97,103,101,105,101,92,87,
92},{100,96,98,96,90,88,90,90,92,96,95,95,95,96,95,96},
{102,101,101,99,94,94,93,88,92,94,94,94,97,99,99,97},
{102,100,100,96,98,97,97,96,97,97,96,95,94,95,99,104},
{105,102,102,102,99,102,101,97,98,103,100,98,98,101,
106,107},{108,107,104,102,101,102,105,107,114,112,
106,104,105,107,107,107}}

Image_Mat_3 (542nd matrix of p3le1)

{{71,66,67,72,72,69,72,74,71,71,73,75,76,78,76,72},
{74,70,69,75,77,72,71,72,72,72,72,72,78,81,76,70},
{73,74,76,78,78,75,74,75,74,73,71,74,82,84,79,76},
{73,75,79,82,78,73,73,72,70,68,71,74,77,78,74,73},
{72,74,74,76,76,75,79,78,72,69,70,74,80,80,78,74},
{68,73,76,75,74,72,75,77,72,71,73,75,79,82,78,73},
{72,75,73,74,76,75,77,74,73,71,72,77,80,82,79,72},
{74,79,79,74,74,75,75,75,73,74,75,77,81,79,75,73},
{71,79,79,75,74,77,76,71,74,80,79,78,79,75,72,71},
{72,75,78,75,76,79,79,74,75,71,71,73,77,79,77,74},
{70,77,77,73,74,79,79,74,71,74,77,79,84,84,78,72},
{72,73,76,74,72,74,75,74,74,75,77,78,81,79,74,73},
{77,81,79,72,75,80,76,75,77,74,73,77,83,82,73,69},
{75,73,73,76,76,78,78,75,72,72,76,78,76,73,73,72},
{79,77,79,76,78,83,82,75,69,68,73,79,81,78,72,72},
{79,77,74,73,79,87,85,75,72,70,73,81,87,79,73,73}}

Fig. 8. Pixels representing Image_Mat_1, Image_Mat_2, and Image_Mat_3.

Eigen_Values_Image_Mat_1 = {1.3136, 0.0096, 0.0096, 0.0085, 0.0059, 0.0059, 0.0040, 0.0032, 0.0038, 0.0038, 0.0056, 0.0056, 0.0045, 0.0045, 0.0022, 0.0018}

Eigen_Values_Mat_2 = {1.5543, 0.0603, 0.0293, 0.0293, 0.0184, 0.0112, 0.0099, 0.0065, 0.0065, 0.0033, 0.0033, 0.0021, 0.0020, 0.0012, 0.0012, 0.0001}

Eigen_Values_Image_Mat_3 = {1.2036, 0.0201, 0.0092, 0.0092, 0.0072, 0.0072, 0.0059, 0.0059, 0.0056, 0.0035, 0.0035, 0.0010, 0.0028, 0.0028, 0.0013, 0.0013}

Fig. 9. Eigen values for image matrices.

point numbers. The Eigen values obtained for Image_Mat_1, Image_mat_2, and Image_Mat_3 are shown in Fig. 9.

The results of calculating the Eigen values for image matrices are listed in Table 3.

In the below mentioned table (Table 3), TM represents the Total Matrices obtained for the iris images p1le1, p2re2 and p3le1, where the size of each matrix is 16×16. The result shows that we need less than 1 second of time to calculate all the Eigen values of an iris image and we need approximately 1.25 seconds to compare online calculated Eigen values with the stored Eigen values. We have stored Eigen values of an iris image for further identification of a person. The memory required for storing the Eigen values of iris images p1le1, p2re2, p3re1 and its comparison with actual memory required for storing these iris images are given in Table 4.

The robustness of comparing the two iris images will have following the three steps:

Table 3. Calculating Eigen values and its comparison with stored Eigen values for iris images.

S. No.	Image	TM	Time required for dividing image into matrices (in ms)	Eigen value calculation time (in ms)	Iris image comparison time (in ms)
1	p1le1	834	25.40	9.93	125.83
2	p2re2	870	26.0	9.68	121.49
3	p3le1	670	17.9	7.97	135.56

Table 4. Memory required in kb for storing iris images and Eigen values.

S. No.	Image	Original size of true color image (in KB)	Size after compression (in KB)	Compression ratio
1	p1le1	448	52.13	8.59
2	p2re2	468	51.06	9.16
3	p3le1	456	41.87	10.88

5.1. *Comparison of two iris images*

All the procedures above (step 1 to 6) have been carried out to accomplish a successful comparison between the already stored Eigen values of a person and the newly calculated Eigen values from the Iris image of the same person so that the identification of person can be uniquely verified. In order to carry out comparison, we took the Iris image (I1) of a person whose identity is to be verified and applied the above procedures (Step 1 to 6) for converting it into Eigen values. Let these calculated Eigen values of Image I1 be known as E1. Let the already stored Eigen values of the same person be known as E2. Now, E1 and E2 were compared to verify the identity of the person. When comparing the two sets of Eigen values E1 and E2, we compared the first Eigen value of E1 with all the Eigen values of E2, and then the second Eigen value of E1 was compared with all the Eigen values of E2. This procedure was repeated until all the Eigen values of E1; compared with the Eigen values of E2.

5.2. *Match percentage*

The number of Eigen values which are common in two iris images are known as Match Percentage. Depending on this match percentage, we can verify whether the two images (i.e., the one stored already and the one taken freshly from the camera) are exactly the same or not. We have carried out a series of Robustness Tests to get a basic idea of how the match percentage varies with the increasing amount of noise in the image. The robustness test is explained in the next phase.

6. Discussion of Robustness Testing in Iris Image Compression

The robustness testing determines the application of our IICIS for its practical use. The result analysis is incomplete without checking its robustness. Robustness means the extent to which our algorithm can withstand the corresponding changes or modifications in the input image. We have tested the robustness of the image by replacing randomly chosen pixels of the image with random values. If 'I' is the image on which the robustness test is to be carried out, then the overall test will be carried out as per the following steps:

S1: Replace n pixels of 'I' to get the new image 'I1'.
S2: Compare 'I' with 'I1'.
S3: Record the comparison results in tabular form.
S4: Repeat the steps S1, S2, and S3 for different values of n ($n = 100$, 200, 300 and 500).

The procedure which we have used to replace the pixels of an iris image randomly has the following inputs:

- ims — the image in which the pixels are to be replaced/changed. The image must be a two-dimensional image.
- The number of pixels to be replaced (n).
- The segment of the image to be altered (segment).
- The image is divided into four quarters (q — for whole image, q1 — for quadrant1, q2 — for quadrant2, q3 — for quadrant3, q4 — for quadrant4).

The output of our procedure is the altered iris image. The procedure above randomly selects 'n' pixels in the image and then replaces them with random values. The random values will be between 0 and 255. The IICIS algorithm divides actual image matrix into small 16×16 blocks and then the Eigen values of these blocks are calculated. Therefore, the distribution of these randomly modified pixels throughout the image plays an important role in calculating match percentage for two images (actual image and randomly modified image). Our analysis

is divided into two cases. These cases are as follow:

Case 1: If the changed pixels are confined to a small region of the image.
Case 2: If the changed pixels are distributed throughout the image.

In Case 1, the changes were confined only to one of the four quadrants (i.e., q1, q2, q3 or q4) of any given input image. The tests were carried out on three sample images of Table 1 and comparison results were recorded. In the first image of Table 1 (i.e., p1le1) we changed the first quadrant (q1) and kept the rest of the image unaltered. In the second image (p2re2) of Table 1, we altered the pixels of second quadrant (q2). In the next test, we changed the pixels of the third quadrant (q3) and the fourth quadrant (q4) of the third image (p3le1) and left the rest of the image unaltered. If n represents the number of pixels to be altered, then the procedure above is repeated for four different values of $n(n = 100, 200, 300, 500)$ for verifying the accuracy of our algorithm in robustness testing. This is for each time we compared the altered image with the actual image and stored result in tabular form.

In Case 2, we took the whole image as a single unit and randomly modified the pixels throughout the iris image. We again recorded the results of the comparison of the iris images of Table 1 and their corresponding modified images. The results of robustness testing of our system are shown in Table 5.

In Table 5, the total number of Eigen values for different images are different. This is due to the fact that each image belongs to a different person and these images of different persons are different in size after resizing. The reasons for getting different number of 16×16 matrices for

Table 5. Results of robustness testing.

S. No.	Image	Quadrant/ portion of image	Total Eigen values	No. of pixels modified	No. of Eigen values affected by pixel modification	% of Eigen values affected	% of Eigen values unaffected	Hit percent
1	P1le1	Q1	13344	100	719	5.38	94.61	94.61
2	P1le1	Q1	13344	200	1136	8.51	91.48	91.48
3	P1le1	Q1	13344	300	1605	12.02	87.97	87.97
4	P1le1	Q1	13344	500	2285	17.12	82.87	82.87
5	P2re2	Q2	13072	100	560	4.28	95.71	95.71
6	P2re2	Q2	13072	200	1104	8.44	91.55	91.55
7	P2re2	Q2	13072	300	1656	12.66	87.33	87.33
8	P2re2	Q2	13072	500	2315	17.70	82.29	82.29
9	P3le1	Q3	12960	100	608	4.69	95.30	95.30
10	P3le1	Q3	12960	200	1024	7.90	92.09	92.09
11	P3le1	Q3	12960	300	1407	10.85	89.14	89.14
12	P3le1	Q3	12960	500	2016	15.55	84.44	84.44
13	P3le1	Q4	12960	100	784	6.04	93.95	93.95
14	P3le1	Q4	12960	200	1276	9.84	90.15	90.15
15	P3le1	Q4	12960	300	1769	13.64	86.35	86.35
16	P3le1	Q4	12960	500	2428	18.73	81.26	81.26
17	P1le1	Q	13344	100	800	5.99	94.00	94.00
18	P1le1	Q	13344	200	1328	9.95	90.04	90.04
19	P1le1	Q	13344	300	2127	15.93	84.06	84.06
20	P1le1	Q	13344	500	3235	24.24	75.75	75.75
21	P2re2	Q	13072	100	640	4.89	95.10	95.10
22	P2re2	Q	13072	200	1417	10.83	89.16	89.16
23	P2re2	Q	13072	300	2138	16.35	83.64	83.64
24	P2re2	Q	13072	500	3111	23.79	76.20	76.20
25	P3le1	Q	12960	100	659	5.08	94.91	94.91
26	P3le1	Q	12960	200	1356	10.46	89.53	89.53
27	P3le1	Q	12960	300	1809	13.95	86.04	86.04

two different people's iris images after dividing it into matrices are as follows:

- The iris size of an eye is different for different persons. In our experiments, we found that generally, the outer radius of iris ring for CASIA and multimedia university databases lie in the range of 270 pixels to 290 pixels.
- The sizes of the pupils of eyes of different persons are different, and we are discarding the pupil during the extraction phase. Therefore, we are getting different number of 16×16 matrices for two different people's iris images. But, for two different iris images of the same, we are getting the same number of 16×16 matrices.
- The size of pupil is inversely proportional to the number of matrices. Since the number of 16×16 matrices varies for different iris images, we get different number of Eigen values for different persons' iris images.

Finally, we have represented the observations in the form of bar graphs. In these graphs, the number of modified pixels is represented by X-axis and the percentage of matched Eigen values are represented by Y-axis. We have drawn a separate graph for each test image. Case 1 testing contributed to four graphs and Case 2 contributed to three graphs. These graphs are represented by Figs. 10–16.

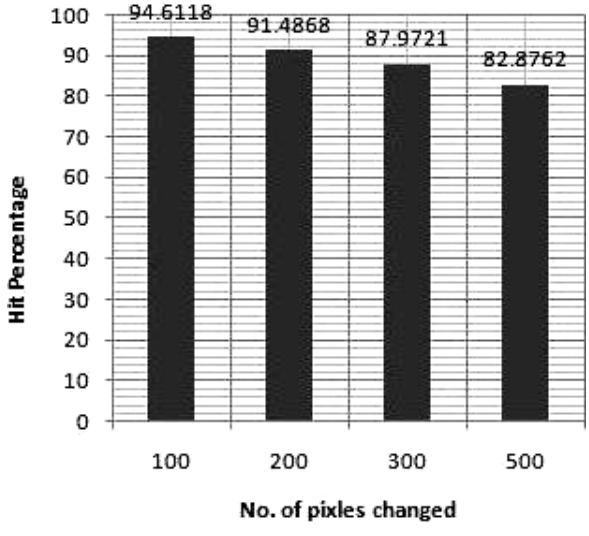

Fig. 10. Robustness testing results obtained after altering the pixels of Q1 for image p1le1.

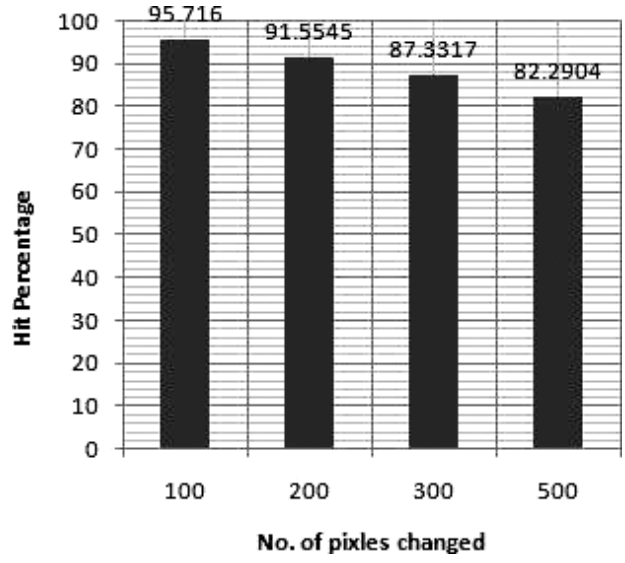

Fig. 11. Robustness testing results obtained after altering the pixels of Q2 for image p2re2.

Based on the above mentioned bar graphs (Figs. 10–16), we have concluded that the match percentage is inversely proportional to n (the number of pixels modified). Figures 10–16 show that when n was 0, the match percentage was 100%. But, when n became 50, the match percentage started decreasing and finally, it became zero. While testing our IICIS software, we have

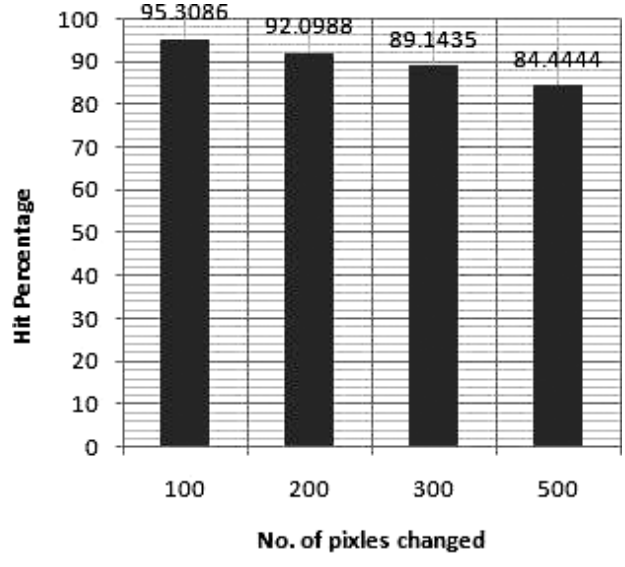

Fig. 12. Robustness testing results obtained after altering the pixels of Q3 for image p3le1.

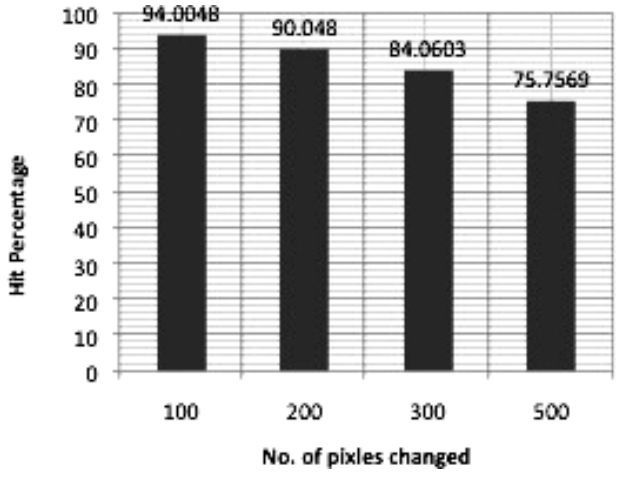

Fig. 13. Robustness testing results obtained after altering the pixels of Q4 for image p3le1.

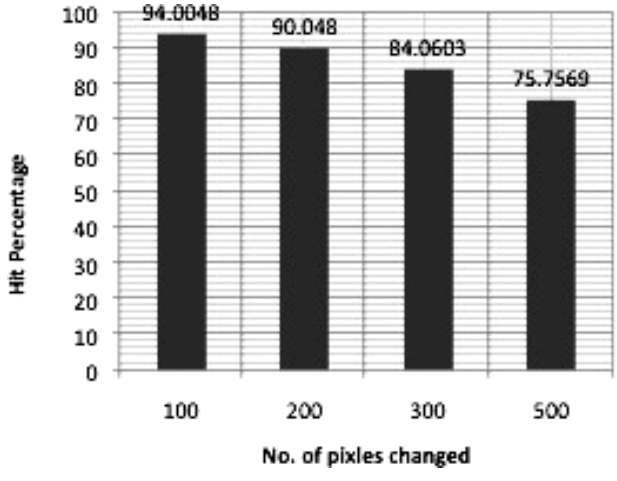

Fig. 14. Robustness testing results obtained after altering the pixels of complete image for image p1le1.

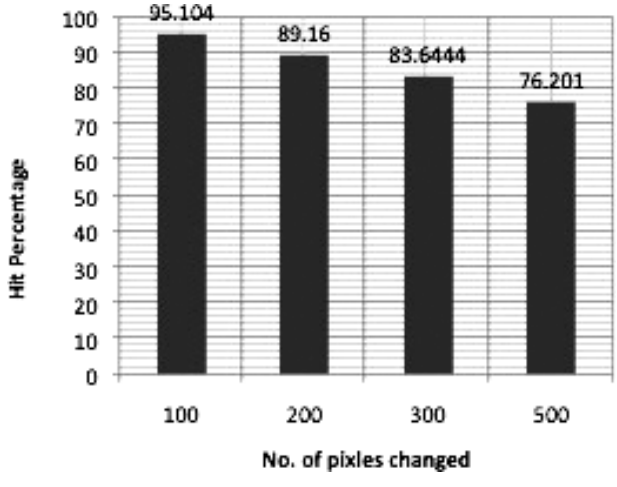

Fig. 15. Robustness testing results obtained after altering the pixels of complete image for image p2re2.

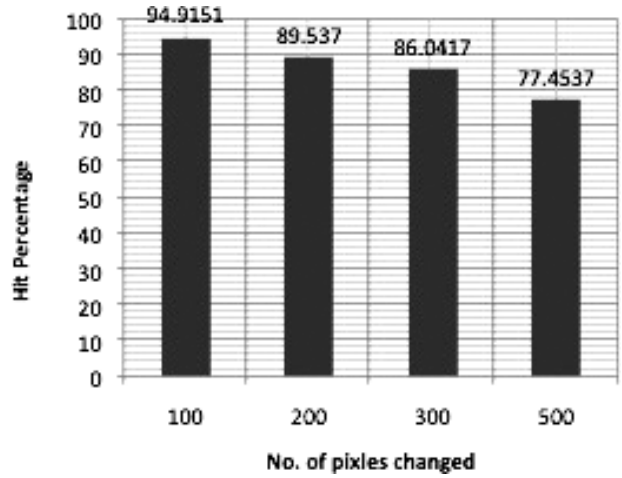

Fig. 16. Robustness testing results obtained after altering the pixels of complete image for image p3le1.

assumed a few constraints. These constraints are:

- The distance of the iris image from the camera will remain constant while repeating the procedure.
- All the iris images are taken from the same angle.
- The intensity of light does not change drastically while taking the iris image.

7. Conclusion and Future Work

In this paper, we have introduced a new method called IICIS. This method can identify an individual and identical twins with an accuracy of 99.99%. It means that for every 10,000 persons, our IICIS will correctly identify 9999 persons. Our algorithm has used six steps for personal identification of an individual or identical twins. Using these steps, we have tested the iris images of individuals and identical twins for CASIA database. In our testing, we found that our algorithm is able to store an iris image of 450KB in the form of Eigen values which uses approximately 50KB of memory. These Eigen values which were stored in 50KB memory were sufficient to identify a person or identical twins.

To check the robustness of our algorithm, we have altered 100, 200, 300, and 500 pixels of the iris image randomly and then compared the Eigen values of the altered iris image with the

Eigen values of the actual iris image. In this comparison, we have observed the following results:

Result 1: If maximum altered pixels are lying in the first few 16×16 matrices, we found that the altered pixels are not affecting the identity of a person and by more than 99.50% the Eigen values of the altered iris image are the same as the Eigen values of the actual iris image.

Result 2: If all the altered pixels are scattered such that each pixel is in a separate 16×16 matrix, our IICIS software has given promising result. In the worst case scenario, when the Eigen values of these 100, 200, 300, and 500 pixels of the altered iris image were not same as the Eigen values of the actual image. In this case we also found more than 90% of the Eigen values of the altered iris image and the actual iris image are the same.

Result 3: If a few of the altered pixels are lying in the first few 16×16 matrices and others are scattered in other matrices, we found that approximately 97% Eigen values of the altered image are exactly the same as the Eigen values of the actual image.

The researchers can develop a technique for human identification which can include the cases of these constraints. In our algorithm, we have converted the iris image in the form of Eigen values for further human identification. These Eigen values are sufficient for personal identification of an individual and identical twins. But we could not get the actual image from the Eigen values. The researchers can develop a method for obtaining actual iris image from the Eigen values.

Acknowledgment

The researchers are thankful to Prof. B. N. Giri (Director, Birla Institute of Technology Ranchi, India - Allahabad Campus) for his kind support and able guidance. The researchers are thankful to Dr. Ajay Chakravorty (Vice Chancellor, Birla Institute of Technology Ranchi, India) for creating academic and research environment at the institute. The research team is thankful to Dr. Edries Abdelhadi Ali (Assistant Professor, Computer Science Department, Sebha University, Libya) for helping the team in getting this new idea of using Eigen values based technique for online iris image compression and identification of a person.

References

Belkin, M. and Niyogi, P. [2003] "Laplace eigenmaps for dimensionality reduction and data representation," *Neural Network Compute* **15**, 1373–1396.

Bowyer, K. W., Hollingsworth, K. P. and Flynn, P. J. [2008] "Image understanding for iris biometrics: A survey," *Computer Vision and Image Understanding* **110**, 281–307.

CASIA [2010] "Iris image database — iris recognition research group national laboratory of pattern recognition (NLPR)," *Institute of Automation. Chinese Academy of Sciences*. http://www.sinobiometrics.com

Daugman, J. [1999] *Recognizing Persons by Their Iris Patterns. Biometrics: Personal Identification in Networked Society* (Kluwer Academic Publishers).

Daugman, J. [2004] "How iris recognition works," *IEEE Transactions On Circuits and Systems for Video Technology* **14**(1), 21–30.

Daugman, J. [2007] "New methods in Iris recognition," *IEEE Transactions on Systems, Man, Cybernetics* **37**(5), 1167–1175.

Doroteo, T. and Toledano [2006] "Usability evaluation of multi-modal biometric verification systems," *Interacting with Computers* **18**, 1101–1122.

Folm, L. and Safir, A. [1987] "Iris recognition system," *US Patent No. US4641349*.

Ganeshan, B., Theckedath, D., Young, R. and Chatwin, C. [2006] "Biometric iris recognition system using a fast and robust iris localization and alignment procedure," *Optics and Lasers in Engineering* **44**, *Elsevier*, 1–24.

Iridian Technologies [2009], http://www.iriscan.com.

Jain, A., Bolle, R. and Pankanti, S. [1999] *Biometrics: Personal Identification in Network Society* (Springer Publisher), pp. 369–384.

Karni, Z. and Gotsman, C. [2000] *Special Compression of Mesh Geometry* (Computer Graphics Proceedings, ACM Press, New York), pp. 279–286.

Ma, L., Yunhong, W. and Tieniu, T. [2002] "Iris recognition using circular symmetric filters," in *Proceedings of International Conference on Pattern Recognition* **2**, 414–417.

Mattey, J. R., Broussard, R. and Kernnell, L. [2010] "Iris image segmentation and sub-optimal images," *Image and Vision Computing* **28**(1), 215–222.

Mishra, K. N., Agrawal, A., Srivastav, P. C. and Hadi, E. A. [2010] "An efficient horizontal and vertical method for online DNA sequence compression and identification," *International Journal of Computer Applications, Foundation of Computer Science* **3**(1), 39–45.

Reuter, M., Franz-Erich, W. and Peinecke, N. [2006] "Laplace-Beltrami spectra as shape DNA of surface and solids," *The Journal of Computer Aided Design* **38**, *Elsevier*, 342–366.

SchonBerg, D. and Kirovski, D. [2004] "Iris compression for cryptographically secure person identification," *IEEE Computer Society, Proceedings of the Data Compression Conference*, 459–468.

Shih, S.-W., Chen, W.S., Lin, L.Y. and Lio, J.C. [2002] "Automatic iris recognition technique based on divider dimension and Karhunen-Loeve transform," in *Proceedings of Conference on Computer Vision, Graphics and Image Processing* (*CVGIP*), 78–85.

Tisse, C. [2003] "Person identification technique using human iris recognition," *Journal of System Research* **4**, 67–75.

Turk, M. A. and Pentland, A. P. [1991] "Face recognition using Eigen faces," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 586–591.

Wildes, R. P. [1997] "Automated iris recognition: An emerging biometric technology," *Proceedings of IEEE* **85**(9), 1348–1363.

World Wide Iris Database [2010] www.advancedsourcecode.com/irisdatabase.asp

Biography

Kamta Nath Mishra received a Master of Technology (M.Tech., Software Systems) degree from Birla Institute of Technology and Science (BITS) Pilani, India in 2003, and Master of Computer Application (MCA) degree from Madan Mohan Malviya Engineering College Gorakhpur, U.P., India in 1996. He is an Assistant professor at Birla Institute of Technology, Mesra, Ranchi India (Allahabad Campus) and pursuing Ph.D. from the same Institute. His research interest includes Software Biometrics System Based Identification Methods.

Anupam Agrawal received his M.Tech. (CS & E) degree from IIT Madras, Chennai and Ph.D (Information Technology) degree from IIIT, Allahabad (in association with IIT, Roorkee). He carried out his Post-Doctoral Research work at the Department of Computing and Information System, University of Bedfordshire (UK). He is presently working as an Associate Professor at Indian Institute of Information Technology, Allahabad since 2006. He was previously working as Scientist 'D' at DEAL, DRDO, Govt. of India, Dehradun. His research interests include Interactive Technologies and Human Computer Interaction, Computer Vision, Pattern Recognition, Image Processing, AI and Soft Computing.

Prakash C. Srivastava received a Master degree in mathematics from University of Allahabad, India in 2001, and Ph.D. degree from University of Allahabad, India in 2009. He is an Assistant Professor at Birla Institute of Technology Ranchi, India (Allahabad Campus). His research interests include functional analysis and Software Measurement Methods.

Vivek Tripathi received a Bachelor of Science (B.Sc., Math.) degree from Andhra University Vishakhapatanam, India in 2008 and he is currently pursuing Master of Computer Applications (MCA) degree from Birla Institute of Technology, Ranchi, India (Allahabad Campus). His research interests include Image Processing and Data Analysis.

Vishal Gupta received a Bachelor of Computer Application (BCA) degree from Integral University Lucknow, India in 2009 and he is currently pursuing Master of Computer Applications (MCA) degree from Birla Institute of Technology, Ranchi, India (Allahabad Campus). His research interests include Image Processing and Data structuring.