2. Literature Review

1. Cervical Cancer Detection and Classification using Texture Analysis, 2016

This paper presented on classification by using an Artificial Neural Network to identify the normal and abnormal tumor images with Fourier transform and Gaussian low pass filter. Magnetic Resonance Imaging (MRI) is a widely-used method of high quality medical imaging. The soft tissue contrast and non-invasiveness are the important advantages of MRIs. The radiologist examined the MRI to identify the presence of tumor abnormal tissues. The shortage of radiologist, the time conception and the misinterpretations caused by viewing the MRI through the naked eye called out for an automated system to analyze and classify medical images. This paper proposes a classification model with MR. Features are then extracted by first order statistical features and second order grey level co-occurrence matrix (GLCM) [30]. Finally the support vector machine (SVM) classifier will predict the treatment outcome of the patient is cured or relapsed. Magnetic Resonance Images (MRI) of locally advanced cervical cancer patients are taken for this work. The block diagram of the proposed method is shown in fig.

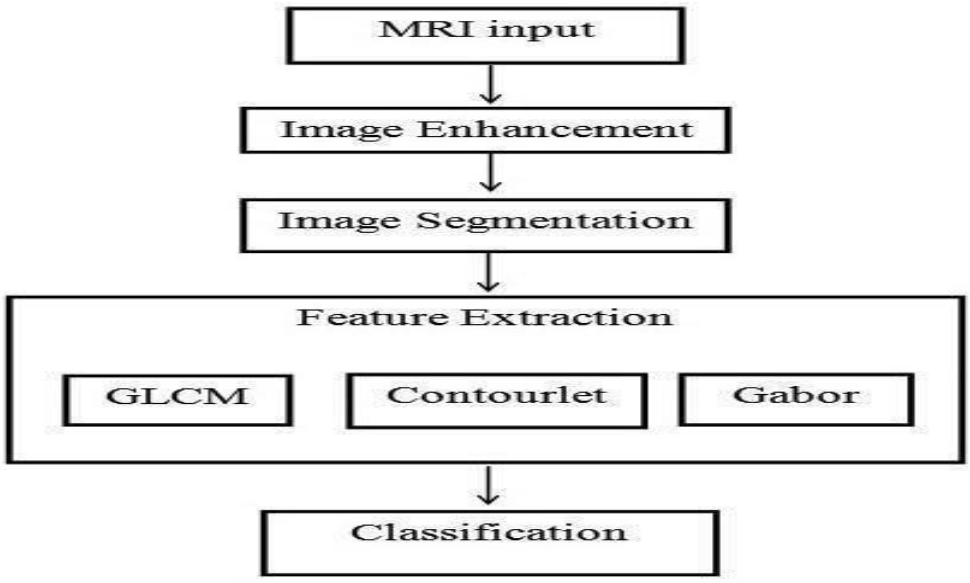

Fig2.1: Block diagram of proposed system[31]

These are preprocessed using contrast enhancement and segmentation techniques along with the morphological operations such as erosion and dilation. Features are then extracted by first order statistical features and second order grey level co-occurrence matrix (GLCM). Finally the SVM classifier will predict the treatment outcome of the patient is cured or relapsed. Support Vector Machine (SVM) is a regression prediction and classification tool that uses machine learning theory for maximizing predictive accuracy while avoiding over-fit to the data automatically. We have successfully built the prototype of classification using texture analysis and SVM [32] with the help of MATLAB. The dicom images are collected from various hospitals, Govt, Medical College and MES Medical College, Kerala. 24 images are tested for the comparative analysis with clinical factors. To confirm the clinical result expert radiologist were consulted. Three dataset are used for the examination. All the images are scaled to grey images and reconstructed for the processing using preprocessing techniques.

Advantages

- It is better to predict the treatment volume according to the staging will help the radiologist for a better treatment planning.
- It is easily implement and It is mainly help for the cancer patients.

Disadvantages

- It is not safe for the patient to take MRI test each time he/she visits the hospital to check health.

2. Biometrics in Human Identification System, (February 2016)

Biometrics is physical or behavior characteristics that can be used for human identification. They propose the ear as a biometric and investigate it with both 2D and 3D data. The ICP-based algorithm also demonstrates good scalability with size of dataset. These results are encouraging in that they suggest a strong potential for 3D ear shape as a biometric. Multibiometric 2D and 3D ear recognition are also explored. The proposed automatic ear detection method will integrate with the current system, and the performance will be evaluated with the original one**.** As the mentioned before, many research studies have

proposed the ear as a biometric[33]. Researchers have suggested that the shape and appearance of the human ear is unique to each individual and relatively unchanging during the lifetime of an adult. Bhanu and Chen presented a 3D ear recognition method using a local surface shape descriptor.

The local surface patches are defined by the feature point and its neighbors, and the patch descriptor consists of its centroid, 2D histogram [34] and surface type. There are four majors' steps in the method feature point extraction, local surface description, O line model building and recognition. The "Pre computed Voxel Closest Neighbors" strategy to improve the speed of the original ICP algorithm. This technique is aimed at a particular application in human identification. Different voxel sizes are examined, and the performance and running time are compared with the results from the original ICP algorithm[35]. The experimental results verify the expected feature of the approach.

The performance of biometric system is tested usually in terms of False Rejection Rate (FRR), False Acceptance Rate (FAR), and Failure to Enroll Rate (FER), Enrollment Time, and Verification Time. The false acceptance rate is most important when security is a priority whereas low false rejection rates are favored when convenience is the priority.

Advantage

- This method have been implemented in C++, and incorporated into the ICP matching program researcher compares the performance, space and running time between the original algorithm and the new incorporated ICP matching.

Disadvantage

- Researcher only address the problem using 3D ear data, it would be interesting to investigate whether the proposed fast ICP-based method is efficient in other applications

3. Local Binary Patterns and Its Application to Facial Image Analysis, july2016.

This paper presents a comprehensive survey of LBP methodology including several more recent variations. As a typical application of the LBP approach, LBP-based facial image analysis is extensively reviewed, while its successful extensions in dealing with various tasks of facial image analysis. The original LBP[36] operator labels the pixels of an image with decimal numbers, called Local Binary Patterns or LBP codes, which encode the local structure around each pixel. Each pixel is compared with its eight neighbors in a 3x3 neighborhood by subtracting the center pixel value. The resulting strictly negative values are encoded with 0 and the others with 1. The derived binary numbers are referred to as Local Binary Patterns or LBP codes. Machine-based face recognition [37] involves two crucial aspects, i.e. facial representation and classifier design. Facial representation consists in deriving a set of relevant features from original images for describing faces, in order to facilitate effective machine-based recognition.

Machine-based face recognition involves two crucial aspects, i.e. facial representation and classifier design. Facial representation consists in deriving a set of relevant features from original images for describing faces, in order to facilitate effective machine-based recognition. Good facial features are desired to have the following properties first, they can tolerate within-class variations while discriminate different classes well; second, they can be easily extracted from the raw images to allow fast processing and finally they lie in a space with low dimensionality to avoid computationally expensive classifiers. Since it was introduced for face representation LBP has proved to be an efficient descriptor for facial image analysis as it fulfills the above criteria quite well, and recent years have witnessed increasing interest in LBP features for facial representation. In this section, we first present LBP-based facial description, and then review existing works on different tasks including face detection, face recognition, facial expression analysis, demographic classification, and other applications[38].

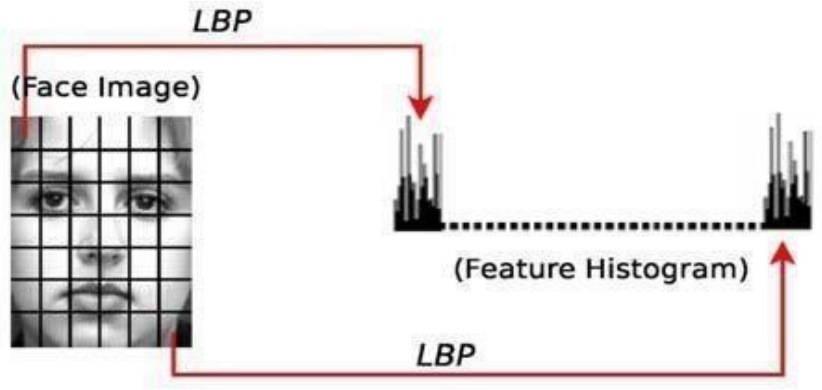

Fig2.2: Generate Histogram for face image[40]

To obtain a small set of the most discriminative LBP-based features for better performance and dimensionality reduction, LBP-based representations are associated with some popular techniques of feature selection schemes for reducing the feature length LBP codes, including rule based strategy, boosting and subspace learning. In addition, some open questions for sub-region based LBP description, e.g. facial description, concern the relevant number of components and the corresponding neighborhood of a certain LBP operator for the best analysis result. Although these questions have been discussed in several works and even with machine learning techniques [41], these conclusions drawn so far have always been dependent on the datasets used and on some given parameters.

Advantage

- LBP has been successfully used for many different image analysis tasks, such as facial image analysis, biomedical image analysis, aerial image analysis, motion analysis, and image and video retrieval.

Disadvantage

- One limitation of the basic LBP operator is that its small 3x3 neighborhood cannot capture dominant features with large scale structures.
- Our results clearly show that facial images can be seen as a composition of micro patterns such as flat areas, spots, lines and edges which can be well described by LBP.

4. A Face Recognition Technique using Local Binary Pattern Method (March 2016)

LBP is really a very powerful method to explain the texture and model of a digital image. Therefore it was ideal for feature extraction in face recognition systems. A face image is first split into small regions that LBP histograms are extracted and then concatenated in to a single feature vector. This vector forms an efficient representation of the face area and can be used to measure similarities between images in this paper, they proposed facial representation predicated on statistical local features, Local Binary Patterns, for facial expression recognition. Various machine learning methods are systematically examined on several databases. LBP features are effective and efficient for facial expression recognition. LBP features were proposed originally for texture analysis, and recently have now been introduced to represent faces in facial images analysis [42]. The most crucial properties of LBP features are their tolerance against illumination changes and their computational simplicity. We examine different machine learning methods.

Face recognition has become a well-known topic in various applications like security, surveillance etc. In LBP the entire image is split into equal sized blocks. Then local binary pattern is computed for each pixel in most blocks by comparing the guts pixel with neighboring pixels.

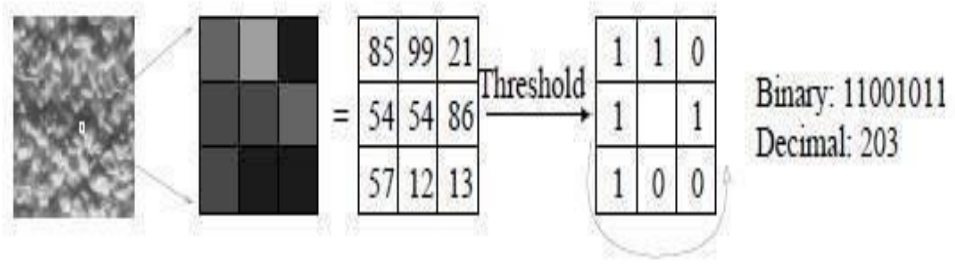

Fig2.3: The basic LBP operator[43].

The LBP operator has been widely used in different applications such as texture classification, image retrieval etc. Before our study, it was not obvious to imagine that such texture operator might be useful in representing also facial images. Our results clearly show

that facial images can be seen as a composition of micro patterns such as flat areas, spots, lines and edges which can be well described by LBP. The texture description of a single region describes the appearance of the region and the combination of all region descriptions encodes the global geometry of the face[44].

Advantages

- Major advantage in this paper, a survey on local binary pattern for face recognition has been studied.
- Facial representation predicated on statistical local features, Local Binary Patterns, for facial expression recognition [45]. Various machine learning methods are systematically examined on several databases.

Disadvantages

- In LBP the entire image is split into equal sized blocks. Then local binary pattern is computed for each pixel in most blocks by comparing the guts pixel with neighboring pixels.
- The looks feature needs to be extracted on either the whole face image or specific regions in a facial image. The face is certainly one of the main features of the human beings and usually used as identification

5. Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions, august 2015.

Dynamic texture is an extension of texture to the temporal domain. Description and recognition of dynamic textures have attracted growing attention. In this paper, a novel approach for recognizing dynamic textures is proposed and its simplifications and extensions to facial image analysis are also considered. First, the textures are modeled with volume local binary patterns (VLBP), which are an extension of the LBP operator[46] widely used in ordinary texture analysis, combining motion and appearance. To make the approach computationally simple and easy to extend, only the co-occurrences on three orthogonal planes (LBP-TOP) are then considered. A block-based method is also proposed to deal with

specific dynamic events, such as facial expressions, in which local information and its spatial locations should also be taken into account. In experiments with two dynamic texture databases, DynTex and MIT, both the VLBP and LBPTOP [47] clearly outperformed the earlier approaches. The proposed block-based method was evaluated with the Cohn-Kanade facial expression database with excellent results. Advantages of our approach include local processing, robustness to monotonic gray-scale changes and simple computation.

As our approach involves only local processing, we are allowed to take a more general view of dynamic texture recognition, extending it to specific dynamic events such as facial expressions. A block-based approach combining pixel-level, region-level and volume-level features is proposed for dealing with such nontraditional dynamic textures in which local information and its spatial locations should also be taken into account. This will make our approach a highly valuable tool for many potential computer vision applications. For example, the human face plays a significant role in verbal and non-verbal communication. Fully automatic and real time facial expression recognition could find many applications, for instance, in human-computer interaction[48], biometrics, telecommunications and psychological research. Most of the research on facial expression recognition has been based on static images. Some research on using facial dynamics has also been carried out however; reliable segmentation of lips and other moving facial parts in natural environments has proved to be a major problem. Our approach is completely different, avoiding error prone segmentation.

Local texture descriptors have gained increasing attention in facial image analysis due to their robustness to challenges such as pose and illumination changes. Recently, Alone et al. proposed a novel facial representation for face recognition from static images based on LBP features. In this approach, the face image is divided into several regions (blocks) from which the LBP features are extracted and concatenated into an enhanced feature vector. This approach is proving to be a growing success. It has been adopted and further developed by many research groups, and has been successfully used for face recognition, face detection [49] and facial expression recognition [50]. All of these have applied LBP based descriptors only for static images, i.e. they do not utilize temporal information as proposed in this paper.

A block-based approach for combining pixel level, region-level and temporal information is proposed. Facial expression recognition is used as a case study, but a similar approach could be used for recognizing other specific dynamic events such as faces from video, for example. The goal of facial expression recognition is to determine the emotional state of the face, for example, happiness, sadness, surprise, neutral, anger, fear, and disgust, regardless of the identity of the face.

A support vector machine (SVM) classifier was selected since it is well founded in statistical learning theory and has been successfully applied to various object detection tasks in computer vision. Since SVM [46] is only used for separating two sets of points, the six-expression classification problem is decomposed into 15 two class problems (happiness-surprise, anger-fear, sadness-disgust, etc.), then a voting scheme is used to accomplish recognition. Sometimes more than one class gets the highest number of votes. In this case, 1-NN template matching is applied to these classes to reach the final result. This means that in training, the spatiotemporal LBP histograms of face sequences belonging to a given class are averaged to generate a histogram [47] template for that class. In recognition, a nearest-neighbor classifier is adopted, i.e. the VLBP [48] or LBP-TOP histogram of the input sequence sample s is classified to the nearest class template n: $L(s, n) < L(s, c)$ for all $c \neq n$ (c and n are the indices of the expression classes having the same highest number of votes in the SVM classifier. The below figure has (a) Block volumes; (b) LBP features from three orthogonal planes; (c) Concatenated features for one block volume with the appearance and motion.

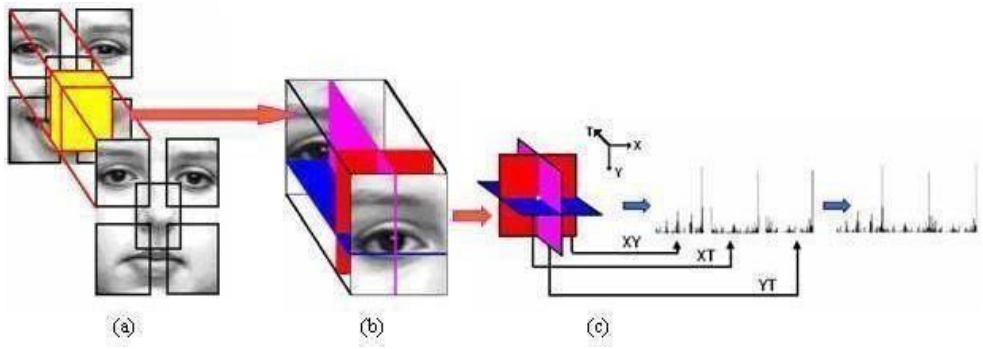

Fig2.4. Features in each block volume[48].

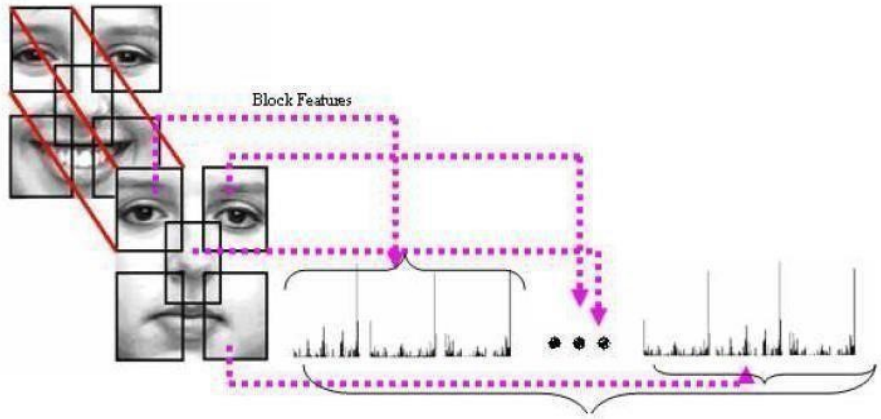

Fig2.5: Facial expression representation[49].

Advantage

- In the proposed VLBP, the parameter P determines the number of features, but also means losing more information. When the number of neighboring points increases, the number of patterns for basic VLBP will become very large.
- It is easily implement.

Disadvantage

- Furthermore, no gray-scale normalization is needed prior to applying our operators to the face images.

6. **Access Control System with High Level Security Using Fingerprints, sep 2016.**

Biometric based applications guarantee for resolving numerous security hazards. As a method of preserving of privacy and the security of sensitive information, biometrics has been studied and used for the past few decades. Fingerprint is one of the most widely used biometrics. A number of fingerprint verification approaches have been proposed until now. However, fingerprint images acquired using current fingerprint input devices that have small field of view are from just very limited areas of whole fingertips [50]. Therefore, essential information required to distinguish fingerprints could be missed, or extracted falsely. The limited and somewhat distorted information are detected from them, which might reduce the accuracy of fingerprint verification systems. In the systems that verify the identity of two

finger prints using fingerprint features, it is critical to extract the correct feature information. In order to deal with these problems, compensation of imperfect information can be performed using multiple impressions of enrollee's fingerprints. Experiments using FVC 2002 databases show that the enrollment using multiple impressions improves the performance of the whole fingerprint verification system.

A measurable biological or behavioral characteristic, which reliably distinguishes one person from another, used to recognize the identity, or verify the claimed identity, of an enrollee'. The fingerprint is one of widely used biometrics [51] satisfying uniqueness and permanency. Thus a number of fingerprint verification approaches have been proposed until now. Jain et al. Presented a minutiae-based verification, which aligns minutiae using Hough transform and performs minutiae matching by bounding box. Ross et al. proposed hybrid matching method of local-based matching and global-based matching to enhance the performance. Pan et al. proposed an alignment algorithm using limited processing power and memory space to be executed in a smart card, and showed the possibility of match-on-card. In the match-on-card system, as entire verification operation is executed on the smart card, the system doesn't have to maintain central database and the biometric template is prevented from being streamed out of the smart card. Therefore, it can prevent biometric templates from being misused by the fraud.

It is widely known that the fingerprint is unique, and invariant with aging, which implies that user authentication, can be relied on the comparing two fingerprints. In general, a professional fingerprint examiner relies on details of ridge structures of the fingerprint in order to make fingerprint identifications [52]. And the structural features are composed of the points where ridges end or bifurcate, that are called minutiae. Shows small part of an enlarged fingerprint image and two types of minutiae pointed by square marker and circle marker. The minutia marked by square is bifurcation, and that by circle is ending point, and the branch from minutia represents the direction of the minutiae. Usually, each minutia is described by the position in the coordinate, the direction it flows and the type, whether it is ridge ending or bifurcation.

Advantages

- The fingerprint is one of widely used biometrics satisfying uniqueness and permanency.
- As automatic fingerprint identification and authentication systems rely on representing the two most prominent minutiae, i.e., bifurcation and ridge ending, a reliable minutiae extraction algorithm is critical to the performance of the system.

Disadvantage

- Although the performance of matching adopting the proposed enrollment was improved considerably, the result is not satisfied yet.

7. Multimodal Biometrics: Issues in Design and Testing, April 2017.

This paper was proposed by the Robert Snelick, Mike Indovina in april 2017. Experimental studies show that multimodal biometric systems for small-scale populations perform better than single-mode biometric systems. We examine if such techniques scale to larger populations, introduce a methodology to test the performance of such systems, and assess the feasibility of using commercial off the- shelf (COTS) products to construct deployable multimodal biometric systems [53]. A key aspect of our approach is to leverage confidence level scores from preexisting single mode data. An example presents a multi modal biometrics system analysis that explores various normalization and fusion techniques for face and fingerprint classifiers. This multimodal analysis uses a population of about 1000 subjects, a number ten-times larger than seen in any previously reported study. Experimental results combining face and fingerprint biometric classifiers reveal significant performance improvement over single-mode biometric systems. Single-mode biometric solutions have limitations in terms of accuracy, enrollment rates, and susceptibility to spoofing. A recent report by the National Institute of Standards and Technology (NIST) [53] to the United States Congress concluded that approximately two percent of the population does not have a legible fingerprint and therefore cannot be enrolled into a fingerprint biometrics system. The report recommends a system employing dual biometrics in a layered approach. Combining multiple

sources of evidence improves performances, as demonstrated in several small-scale experimental studies performed in academia.

This framework allows a system designer to model hypothetical multimodal biometric systems that can vary the biometric indicator, matching algorithm, normalization and fusion Techniques, and sample databases (e.g., the subject population or environmental conditions can be varied). Given this framework, systems can be built to optimally suit a particular application. Computes the ROC curves for our study. Shows a ROC curve for the simple sum fusion rule with various normalization techniques. Clearly the use of these fusion and normalization techniques enhances the performance significantly over the single-modal face or fingerprint classifiers. For example, at a FAR of 0.1% the simple sum fusion with the min-max normalization has a GAR of 94.9%, which is considerably better than that of face, 75.3%, and fingerprint, 83.0%. Also, using any of the normalization techniques in lieu of not normalizing the data proves beneficial. The simplest normalization technique, the min max, yields the best performance. The selection of tolerable error rates, and in both single-modal and multimodal biometric systems, implementers are forced to make a trade-off between usability and security. Implementers produce ROC curves for their systems from their own test data based on these guidelines. Operators use these ROC curves to determine the FAR of the security level needed for their application [54]. The mapping table, from step 8 of our testing methodology, is used to determine the threshold value corresponding to that FAR. This mapping is usually done via an implementer provided utility, which may need to use extrapolation to determine certain values.

Thus, future plans include expanding the test databases to attain these larger sizes. In addition, to assess the feasibility of such systems for large-scale deployments, we will perform these tests using COTS products.

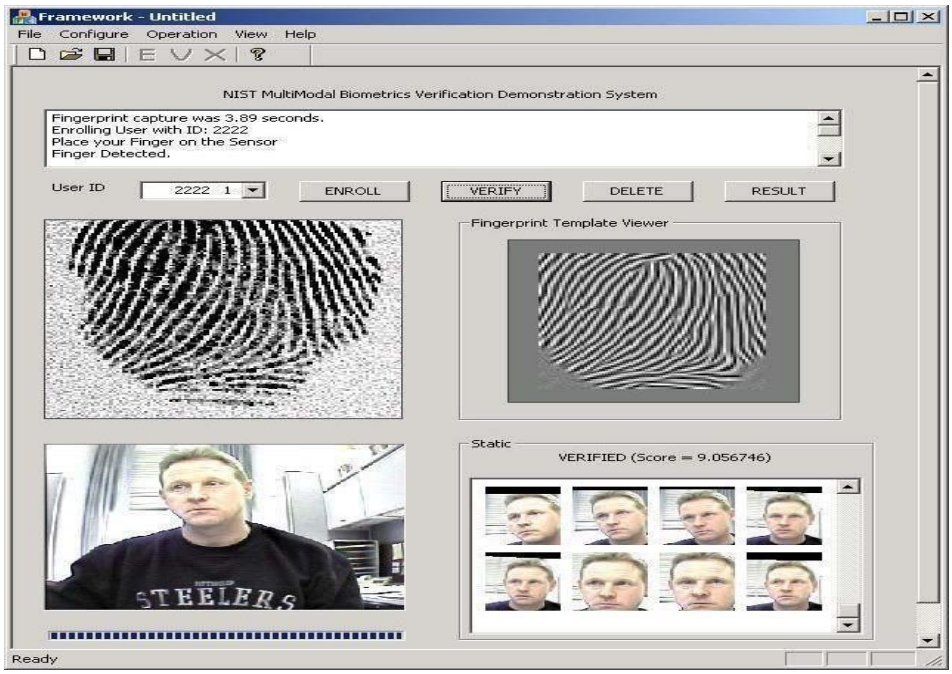

Fig2.6: Prototype Multimodal Biometric System.

Advantages

- Experimental results combining face and fingerprint biometric classifiers reveal significant performance improvement over single-mode biometric systems.
- This multimodal analysis uses a population of about 1000 subjects, a number ten-times larger than seen in any previously reported study.

Disadvantage

- Limitations upon deployment of multimodal systems include lack of a common testing framework and the absence of tools to evaluate and build such systems.

8. Information Fusion in Biometrics, October 2016

This paper was proposed by the Arun Ross, Anil Jain in October 2016. User verification systems that use a single biometric indicator often have to contend with noisy sensor data, restricted degrees of freedom and unacceptable error rates. Attempting to improve the performance of individual matchers in such situations may not prove to be active because of these inherent problems. Multimodal biometric systems [55] seek to alleviate

some of these drawbacks by providing multiple evidences of the same identity. These systems also help achieve an in-crease in performance that may not be possible by using a single biometric indicator. This paper addresses the problem of information fusion in verification systems. Experimental results on combining three biometric modalities (face, fingerprint and hand geometry) are also presented. The performance of a biometric system is largely elected by the reliability of the sensor used and the degrees of freedom covered by the features extracted from the sensed signal. Further, if the biometric trait being sensed or measured is noisy (a fingerprint with a scar or a voice altered by a cold, for example), the resultant confidence score (or matching score) computed by the matching module may not be reliable. Simply put, the matching score generated by a noisy input has a large variance. This problem can be alleviated by installing multiple sensors that capture different biometric traits. Such systems, known as multimodal biometric systems, are expected to be more reliable due to the presence of multiple pieces of evidence.

These systems are able to meet the stringent performance requirements imposed by various applications. Moreover, it will be extremely difficult for an intruder to violate the integrity of a system requiring multiple biometric indicators. However, an integration scheme is required to fuse the information churned out by the individual modalities. In this work we address the problem of information fusion by rest building a multimodal biometric system and then devising various schemes to integrate these modalities [56]. The proposed system uses the fingerprint, face, and hand geometry features of an individual for verification purposes.

Matching Module, DM: Decision Module. Fusion in the context of biometrics can take the following forms: (i) Single biometric multiple classier fusion, where multiple classifiers on a single biometric indicator are combined. (ii) Single biometric multiple matcher fusion, where scores generated by multiple matching strategies (on the same representation) are combined. (iii) Multiple biometric fusion, where multiple biometrics are utilized. Normalization typically involves mapping the scores obtained from multiple domains into a common framework [57] before combining them. This could be viewed as a two-step process in which the distributions of scores for each domain is rest estimated using

robust statistical techniques and these distributions are then scaled or translated into a common domain.

Face verification

Grayscale images of a subject's face were obtained using a Panasonic video camera. The Eigen face approach was used to extract features from the face image [58]. In this approach a set of orthonormal vectors (or images) that span a lower dimensional subspace is computed using the principal component analysis (PCA) technique.

Fingerprint Verification

Fingerprint images were acquired using a Digital Biometrics [59] sensor at a resolution of 500 dpi. The features correspond to the position and orientation of certain critical points, known as minute, that are present in every finger print Hand.

Geometry Verification

Images of a subject's right hand were captured using a Pulnix TMC-7EX camera. The feature extraction system computes 14 feature values comprising of the lengths of the fingers, widths of the fingers and widths of the palm at various locations of the hand.

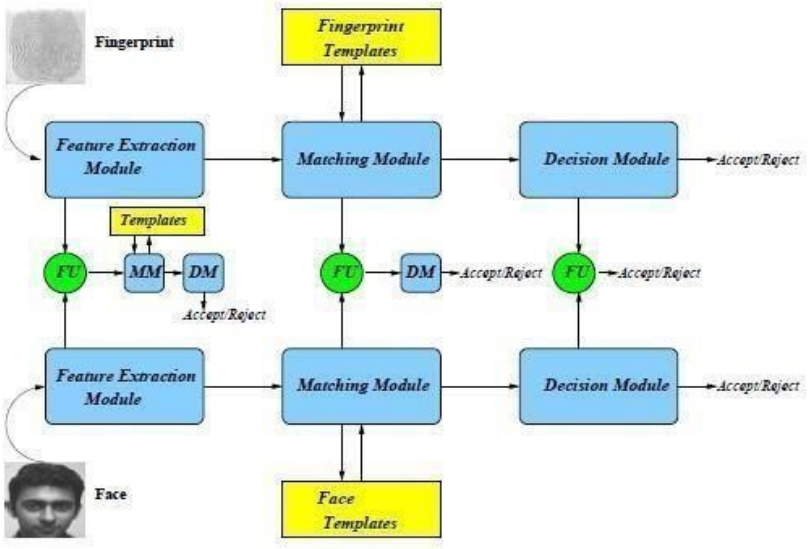

Fig2.7: A bimodal biometric system showing the three levels of fusion

Advantages

- All the three fusion schemes (at the confidence level) considered here provide better verification performance than the individual biometrics.
- The performance of a biometric system is largely affected by the reliability of the sensor used and the degrees of freedom covered by the features extracted from the sensed signal.

Disadvantage

- This problem can be alleviated by installing multiple sensors that capture different biometric traits. Such systems, known as multimodal biometric systems.

9. Texture Classification Using Hierarchical Discriminant Analysis january2016.

A**s** the representative of the linear discriminant analysis, the Fisher method is most widely used in practice and it is very effective in two-class classification. However, when it is expanded to multi-class classification problem, the precision of its discrimination may become worse. One of the main reasons is an Occurrence **of** overlapped distributions on a discriminant space built by Fisher criterion[60]. In order to take such overlap among classes into consideration, our approach builds a new discriminant space with hierarchical tree structure for overlapped classes. In this paper, we propose a new hierarchical discriminant analysis for texture classification. We can divide a discriminant space into subspace by recursively grouping overlapped classes. In the experiment, texture images of many classes are classified based on the proposed method, and we show the outstanding result compared with the conventional method. Texture analysis **is** an important and useful area in computer vision and pattern recognition. Image texture is constituted by various colors and luminance, so that its model can be represented **as** a mathematical function with variation of pixel intensities. Many methods such as statistical, geometrical, model based and signal processing methods have been devised to extract the local or global features from texture images. The effectiveness of the extracted feature set depends on how well patterns from different classes can be separated on the feature space.

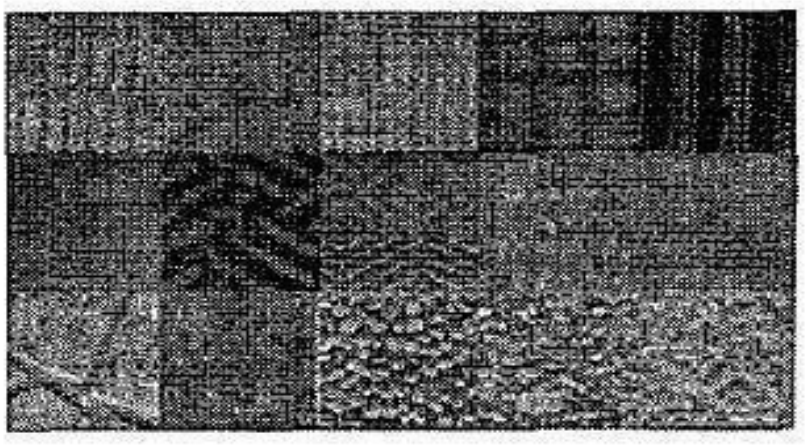

Fig2.8: Example of texture image in the Outex13 database

In this paper, we propose a new discriminating method to classify texture images using hierarchical discriminant analysis. The proposed discriminant analysis with hierarchical tree structure is similar to divisive hierarchical clustering techniques 131. We start by placing all classes into a single group on an initial discriminant space determined by Fisher method and subdivide the group recursively. In order to divide the initial discriminant space into subspaces, we need to decide on a parameter. This parameter represents distance how close the distributions of classes are on the Fisher discriminant space. With the setting of given parameter, we can evaluate a degree of overlap among classes. In this paper, we propose a new discriminating method to classify texture images using hierarchical discriminant analysis. The proposed discriminant analysis with hierarchical tree structure is similar to divisive hierarchical clustering techniques. We start by placing all classes into a single group on an initial discriminant space determined by Fisher method and subdivide the group recursively. In order to divide the initial discriminant space into subspaces, we need to decide on a parameter. This parameter represents distance how close the distributions of classes are on the Fisher discriminant space. With the setting of given parameter, we can evaluate a degree of overlap among classes. To verify the effectiveness of the proposed method, some experiments were performed on several texture database for texture classification. We use the training and test data **f**rom texture database 'Outexl3' ***(68*** class), 'CTC06**,** and mixed one 'Outex13+CTC06' **(122** class). The Outer is a public available

framework for experimental evaluation of texture classification. The Outer database contains 08 distinct texture with 10 images per texture (128 x 128 pixels). The collection of surface textures exhibits well defined in the Outex as shown in Fig. and it contains considerably similar texture images**.**

Advantage

- They propose a new multi-class discriminating method by hierarchical discriminant tree. We can obtain hierarchical discriminant space by recursively grouping the overlapped classes.
- In the experiment, texture images of many classes are classified based on the proposed method, and we show the outstanding result compared with the conventional method.

Disadvantage

- The local autocorrelation function can be used **to** assess the amount of regularity as well as the fineness coarseness of texture image.

10. Human Ear Recognition Using Geometrical Features Extraction, July 2015.

This paper was proposed by Asmaa Sabet Anwara,d, Kareem Kamal A.Ghany. The biometrics recognition has been paid more attention by people with the advancement of technology nowadays. The human ear is a perfect source of data for passive person identification. Ear seems to be a good candidate solution since ear is visible, their images are easy to take and structure of ear does not change radically over time. Ear satisfies biometric characteristic (universality, distinctiveness, permanence and collectability). In this paper we presented a new algorithm for ear recognition based on geometrical features extraction like (shape, mean, centroid and Euclidean distance between pixels). Firstly, we made a pre-processing phase by making all images have the same size. Then we used the snake model to detect the ear, and we applied median filter to remove noise, also we converted the images to binary format. After that we used canny edge and made some enhancement on the image, largest boundary is calculated and distance matrix is created then we extracted the image

features. Finally, the extracted features were classified by using nearest neighbor with absolute error distance. This method is invariant to scaling, translation and rotation. Biometric has lately been receiving attention in popular media. Biometric deals with identification of individuals based on their physiological or behavioral characteristics. It is widely believed that biometric will become a significant component of the identification technology**.**

Biometrics can be divided into two classes physiological and behavioral. Physiological which are based upon measurements of external physical traits, such as weight, height, body shape, face shape, hand shape, skin color, texture, odor, hair color, retina, iris, Deoxyribonucleic Acid, fingerprint and ear shape. Behavioral which usually measure learned behaviors, such as gait, body posture, speech, handwriting, keyboard typing pattern, heartbeat, respiration pattern, and eye blinking pattern. Biometric system can be used in two modes: verification or identification. Identification involves comparing the information against templates corresponding to all users in the database. This take much time because this depending on size of database. Verification involves comparison with only those templates corresponding to the claimed identity. This not takes much time because this compares one to one. This implies that identification and verification are two problems that should be dealt separately Traditional methods for personal identification are based on what is the person know like PIN's, passwords, identification cards, and specific keys. These methods have a lot of disadvantages like hard to remember, easy to lose, lack of security, cards and keys are often stolen and passwords can be cracked. Because of disadvantage of traditional methods for identification we preferred to use biometrics. Biometrics has lately been receiving attention in popular media. Biometric deals with identification of individuals based on their physiological or behavioral characteristics. The proposed approach consists of: Pre Processing [61], ear detection, edge detection, post processing, feature extraction and finally classification.

They proposed a new algorithm for ear recognition based on geometrical features extraction. Seven values are extracted as feature vector which are mean of ear image, centroid of x coordinate, centroid of y coordinate, four different distances from matrix which contain

Euclidean distance between every pixels in image. We tried to increase the distance values were taken to increase the feature vector which will be more representative. We not effect on the run time because the feature vector is still small but representative. K-nearest neighbor used for classification because this classifier gives higher accuracy.

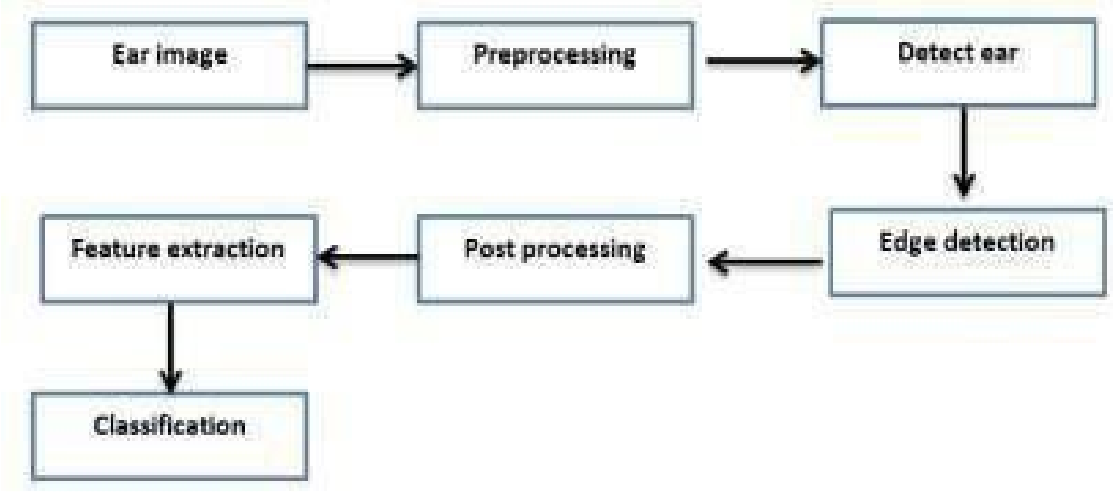

Fig2.9: Proposed Model Block Diagram

Advantages

- The experimental results showed that the proposed approach gives better results and obtained over all accuracy almost 98%.
- The proposed approach consists of: Pre-Processing, ear detection, edge detection, post processing, feature extraction and finally classification.

Disadvantage

- Researcher only address the problem using 3D ear data, it would be interesting to investigate whether the proposed fast ICP-based method is efficient in other applications.

3. REQUIREMENTS ANALYSIS

3.1 Functional Requirements

3.1.1 Software Requirements

OpenCV

OpenCV was started at Intel in 1999 by Gary Bradsky and the first release came out in 2000. Vadim Pisarevsky joined Gary Bradsky to manage Intel's Russian software OpenCV team. In 2005, OpenCV was used on Stanley, the vehicle who won 2005 DARPA Grand Challenge. Later its active development continued under the support of Willow Garage, with Gary Bradsky and Vadim Pisarevsky leading the project. Right now, OpenCV supports a lot of algorithms related to Computer Vision and Machine Learning and it is expanding day-byday. Currently OpenCV supports a wide variety of programming languages like C++, Python, Java etc and is available on different platforms including Windows, Linux, OS X, Android, iOS etc. Also, interfaces based on CUDA and OpenCL are also under active development for high-speed GPU operations. OpenCV-Python is the Python API of OpenCV. It combines the best qualities of OpenCV C++ API and Python language.

OpenCV-Python

Python is a general purpose programming language started by Guido van Rossum, which became very popular in short time mainly because of its simplicity and code readability. It enables the programmer to express his ideas in fewer lines of code without reducing any readability. Compared to other languages like C/C++, Python is slower. But another important feature of Python is that it can be easily extended with C/C++. This feature helps us to write computationally intensive codes in C/C++ and create a Python wrapper for it so that we can use these wrappers as Python modules. This gives us two advantages: first, our code is as fast as original C/C++ code (since it is the actual C++ code working in background) and second, it is very easy to code in Python. This is how OpenCV-Python works, it is a Python wrapper around original C++ implementation. And the support of Numpy makes the task more easier. Numpy is a highly optimized library for numerical operations. It gives a MATLAB-style syntax. All the OpenCV array structures are converted to-and-from Numpy arrays. So whatever operations you can do in Numpy, you can combine it with OpenCV,

which increases number of weapons in your arsenal. Besides that, several other libraries like SciPy, Matplotlib which supports Numpy can be used with this. So OpenCV Python is an appropriate tool for fast prototyping of computer vision problems.

NumPy

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

At the core of the NumPy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. There are several important differences between NumPy arrays and the standard Python sequences

- NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of an ndarray will create a new array and delete the original.
- The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of different sized elements.
- NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python's built-in sequences.
- A growing plethora of scientific and mathematical Python-based packages are using NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays. In other words, in order to efficiently use much (perhaps even most) of today's scientific/mathematical Python-based software, just knowing how to use Python's

built-in sequence types is insufficient - one also needs to know how to use NumPy arrays.

Tkinter

The Tkinter module ("Tk interface") is the standard Python interface to the Tk GUI toolkit from Scriptics (formerly developed by Sun Labs). Both Tk and Tkinter are available on most Unix platforms, as well as on Windows and Macintosh systems. Starting with the 8.0 release, Tk offers native look and feel on all platforms.

Tkinter consists of a number of modules. The Tk interface is provided by a binary extension module named _tkinter. This module contains the low-level interface to Tk, and should never be used directly by application programmers. It is usually a shared library (or DLL), but might in some cases be statically linked with the Python interpreter.

The public interface is provided through a number of Python modules. The most important interface module is the Tkinter module itself. To use Tkinter, all you need to do is to import the Tkinter module:

import Tkinter

Or, more often:

from Tkinter import *

The Tkinter module only exports widget classes and associated constants, so you can safely use the from-in form in most cases. If you prefer not to, but still want to save some typing, you can use import-as:

import Tkinter as Tk

SciPy

SciPy is an open source Python library used for scientific computing and technical computing. SciPy contains modules for optimization, linear algebra, integration,

interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering.

SciPy builds on the NumPy array object and is part of the NumPy stack which includes tools like Matplotlib, pandas and SymPy. There is an expanding set of scientific computing libraries that are being added to the NumPy stack every day. This NumPy stack has similar users to other applications such as MATLAB, GNU Octave, and Scilab. The NumPy stack is also sometimes referred to as the SciPy stack.

SciPy is also a family of conferences for users and developers of these tools: SciPy (in the United States), EuroSciPy (in Europe) and SciPy.in (in India). Enthought originated the SciPy conference in the United States and continues to sponsor many of the international conferences as well as host the SciPy website.

The SciPy library is currently distributed under the BSD license, and its development is sponsored and supported by an open community of developers. It is also supported by Numfocus which is a community foundation for supporting reproducible and accessible science.

OS

This module provides a portable way of using operating system dependent functionality. If you just want to read or write a file see open(), if you want to manipulate paths, see the os. path module, and if you want to read all the lines in all the files on the command line see the file input module. For creating temporary files and directories see the tempfile module, and for high-level file and directory handling see the shutil module.

Notes on the availability of these functions

- The design of all built-in operating system dependent modules of Python is such that as long as the same functionality is available, it uses the same interface; for example, the function os.stat(path) returns stat information about path in the same format (which happens to have originated with the POSIX interface).

- Extensions peculiar to a particular operating system are also available through the os module, but using them is of course a threat to portability.
- An “Availability: Unix” note means that this function is commonly found on Unix systems. It does not make any claims about its existence on a specific operating system.
- If not separately noted, all functions that claim “Availability: Unix” are supported on Mac OS X, which builds on a Unix core.

Random

This module implements pseudo-random number generators for various distributions. For integers, uniform selection from a range. For sequences, uniform selection of a random element, a function to generate a random permutation of a list in-place, and a function for random sampling without replacement. On the real line, there are functions to compute uniform, normal (Gaussian), lognormal, negative exponential, gamma, and beta distributions.
For generating distributions of angles, the von Mises distribution is available.

Almost all module functions depend on the basic function random(), which generates a random float uniformly in the semi-open range [0.0, 1.0). Python uses the Mersenne Twister as the core generator. It produces 53-bit precision floats and has a period of 2**19937-1. The underlying implementation in C is both fast and threadsafe. The Mersenne Twister is one of the most extensively tested random number generators in existence. However, being completely deterministic, it is not suitable for all purposes, and is completely unsuitable for cryptographic purposes.

The functions supplied by this module are actually bound methods of a hidden instance of the random. Random class. You can instantiate your own instances of Random to get generators that don’t share state. This is especially useful for multi-threaded programs, creating a different instance of Random for each thread, and using the jumpahead() method to make it likely that the generated sequences seen by each thread don’t overlap.

Class Random can also be subclassed if you want to use a different basic generator of your own devising: in that case, override the random(), seed(), getstate(), setstate() and jumpahead() methods. Optionally, a new generator can supply a getrandbits() method — this allows randrange() to produce selections over an arbitrarily large range.

New in version 2.4: the *getrandbits()* method.

As an example of subclassing, the random module provides the WichmannHill class that implements an alternative generator in pure Python. The class provides a backward compatible way to reproduce results from earlier versions of Python, which used the Wichmann-Hill algorithm as the core generator. Note that this Wichmann-Hill generator can no longer be recommended: its period is too short by contemporary standards, and the sequence generated is known to fail some stringent randomness tests. See the references below for a recent variant that repairs these flaws.

Changed in version 2.3: MersenneTwister replaced Wichmann-Hill as the default generator.

The random module also provides the SystemRandom class which uses the system function os.urandom() to generate random numbers from sources provided by the operating system.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, the jupyter notebook, web application servers, and four graphical user interface toolkits.

Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc., with just a few lines of code. For a sampling, see the screenshots, thumbnail gallery, and examples directory

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Glob

The glob module finds all the pathnames matching a specified pattern according to the rules used by the Unix shell, although results are returned in arbitrary order. No tilde expansion is done, but *, ?, and character ranges expressed with [] will be correctly matched. This is done by using the os.listdir() and fnmatch.fnmatch() functions in concert, and not by actually invoking a subshell. Note that unlike fnmatch.fnmatch(), glob treats filenames beginning with a dot (.) as special cases. (For tilde and shell variable expansion, use os.path.expanduser() and os.path.expandvars().)

3.2 Non-functional requirements

Accuracy

Through a new shape-finding algorithm called “image ray transform,” which boasts 99.6 percent accuracy, according to a study presented at the IEEE Fourth International Conference on Biometrics Sept. 29, the outer ear may prove to be one of the most accurate and least intrusive ways to identify people.

Maintainability

Maintainability testing shall use a model of the maintainability requirements of the software/system. The maintainability testing shall be specified in terms of the effort required to effect a change under each of the following four categories:

Corrective maintenance –Correcting problems. The maintainability of a system can be measured in terms of the time taken to diagnose and fix problems identified within that system. Perfective maintenance – Enhancements: The maintainability of a system can also be measured in terms of the effort taken to make required enhancements to that system. This can be tested by recording the time taken to achieve a new piece of identifiable functionality such as a change to the database, etc. A number of similar tests should be run and an average time calculated. The outcome will be that it is possible to give an average effort required to implement specified functionality. This can be compared against a target effort and an assessment made as to whether requirements are met.

Adaptive maintenance- Adapting to changes in environment. The maintainability of a system can also be measured in terms on the effort required to make required adaptations to that system. This can be measured in the way described above for perfective maintainability testing. Preventive maintenance- Actions to reduce future maintenance costs. This refers to actions to reduce future maintenance costs.

Usage

It can be mainly used for Conference on Biometrics Sept. 29, the outer ear may prove to be one of the most accurate and least intrusive ways to identify people. Fingerprint databases of U.S. government agencies alone store the records of more than 100 million people, but prints can rub off or callous over during hard or repetitive labor. With the advent of computer vision, researchers and identification industries are seeking easier and more robust biometrics to get their hands on.

Robustness

In general, building robust systems that encompass every point of possible failure is difficult because of the vast quantity of possible inputs and input combinations. Since all inputs and input combinations would require too much time to test, developers cannot run through all cases exhaustively. Instead, the developer will try to generalize such cases.

4. SOFTWARE DESIGN

Object-oriented analysis and design is a popular technical approach for analyzing, designing an application, system, or business by applying the object-oriented paradigm and visual modelling throughout the development life cycles to faster better stakeholder communication and product quality. Initially all the requirements are to gather, After requirement gathering these requirements are analyzed for their validity and the possibility of incorporating the requirements in the system, is also studied. In the design phase, the design functions and operations are described in detail, including screen layouts, business rules, process diagrams and other documentation. The output of this stage will describe the new system as a collection of modules or subsystems.

4.1 Software process Model

Extreme Programming

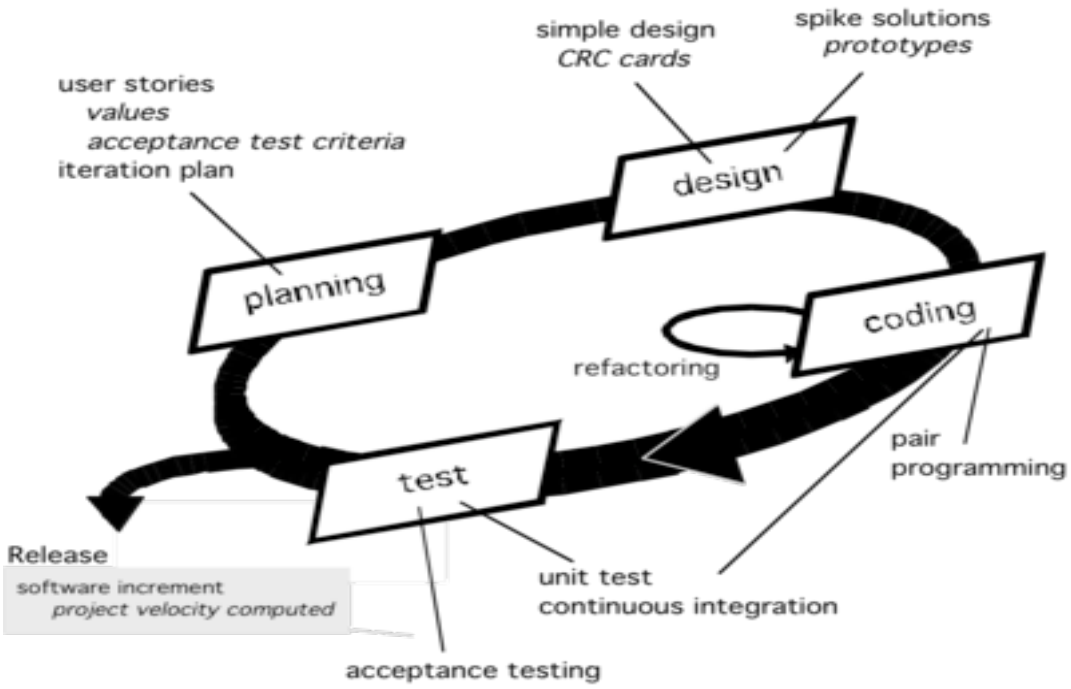

Fig 4.1: Extreme programing

It is agile software development methodology, it is also known as XP. The basic advantage is that the whole process is visible and accountable. The developers will make concrete commitments about what they will accomplish, show concrete progress in the form of deployable software, and when a milestone is reached they will describe exactly what they did and how and why that differed from the plan. The reason behind this approach is that software development is a very fluid process where requirements cannot be fully predicted from the beginning but will always change as projects move on. Hence software development needs a methodology that is capable to adapt to changing requirements at any point during the project life. The Extreme programming software development process starts with Planning, and all iterations consist of four basic phases in its life Cycle: designing, coding, testing and listening. The overriding values that drive the XP life cycle are continual communication among the team and with the client.

UML Diagrams

4.2 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different functionalities in which user involved.

In the below use case diagram there are two actors user and administrator user will first login with the system and user will capture the photo and it can be feature extraction using LBP and the user can be validate the Pearson correlation matrix and the login details will be check and authenticate the administrator and once login and it can be feature extraction using LBP.

When the user can validation using ppmc and then check the administrator and the validate data can be updated and the update the data base and check the data is valid or not if the valid data can be updated and the search result is found otherwise it can be delete the user.

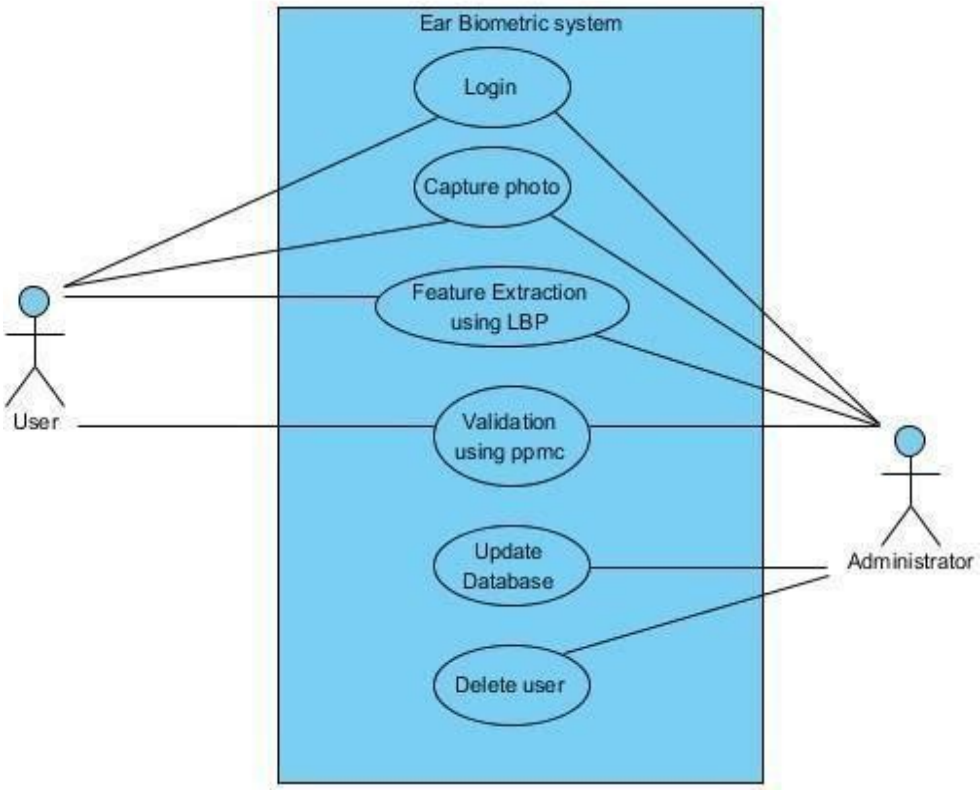

Fig 4.2: Use case diagram for human ear biometrics

4.3 Activity Diagram

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. Before drawing an activity diagram we must have a clear understanding about the elements used in activity diagram. The main element of an activity diagram is the activity itself. An activity is a function performed by the system. After identifying the activities we need to understand how they are associated with constraints and conditions.

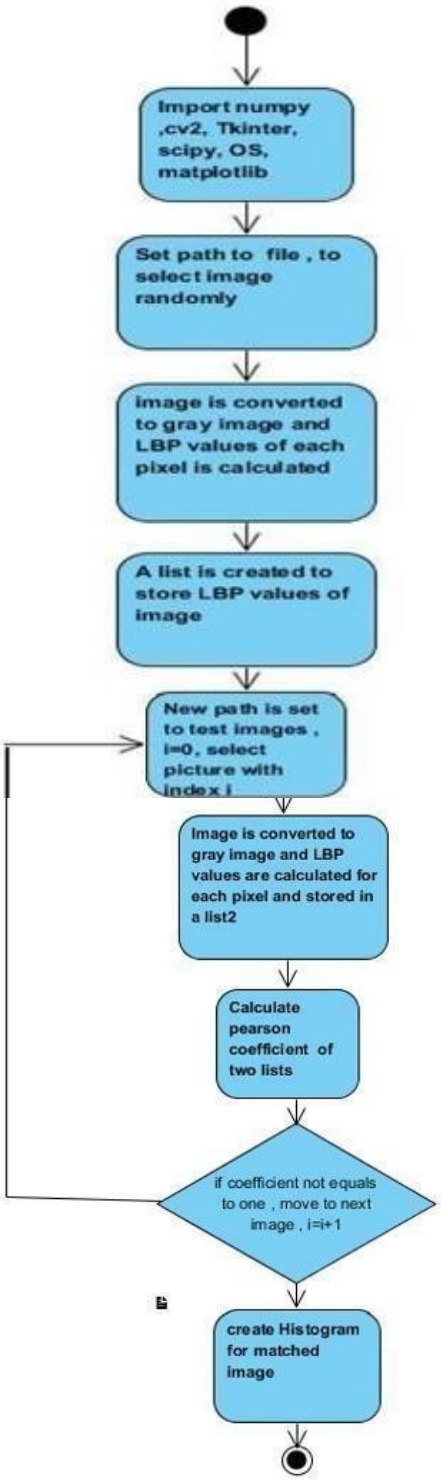

Fig4.3: Activity diagram for the ear authentication.

5. METHODOLOGY

5.1 Local Binary patterns

Local Binary Patterns[61], or LBPs for short, are a texture descriptor made popular by the work of Ojala et al. in their 2002 paper, Multiresolution Grayscale[62] and Rotation Invariant Texture Classification[63] with Local Binary Patterns (although the concept of LBPs were introduced as early as 1993).

Unlike Haralick texture features that compute a global representation of texture based on the Gray Level Co-occurrence Matrix[64], LBPs instead compute a local representation of texture. This local representation is constructed by comparing each pixel with its surrounding neighborhood of pixels.

` The first step in constructing the LBP texture descriptor is to convert the image to grayscale. For each pixel in the grayscale image, we select a neighborhood of size *r* surrounding the center pixel. A LBP value is then calculated for this center pixel and stored in the output 2D array with the same width and height as the input image.

For example, let's take a look at the original LBP descriptor which operates on a fixed *3 x 3* neighborhood of pixels just like this:

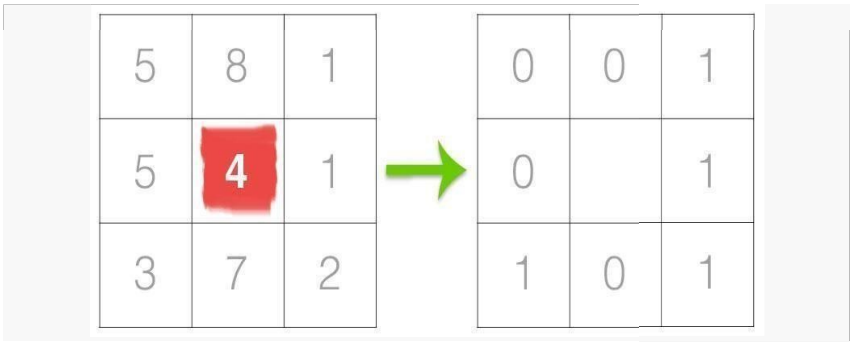

Fig 5.1: The first step in constructing a LBP is to take the 8 pixel neighborhood surrounding a center pixel and threshold it to construct a set of 8 binary digits.

In the above figure we take the center pixel (highlighted in red) and threshold it against its neighborhood of 8 pixels. If the intensity of the center pixel is greater-than-or-equal to its neighbor, then we set the value to *1*; otherwise, we set it to *0*. With 8 surrounding pixels, we have a total of *2 ^ 8 = 256* possible combinations of LBP codes.

From there, we need to calculate the LBP value for the center pixel. We can start from any neighboring pixel and work our way clockwise or counter-clockwise, but our ordering must be kept *consistent* for all pixels in our image and all images in our dataset. Given a *3 x 3* neighborhood, we thus have 8 neighbors that we must perform a binary test on. The results of this binary test are stored in an 8-bit array[65], which we then convert to decimal, like this:

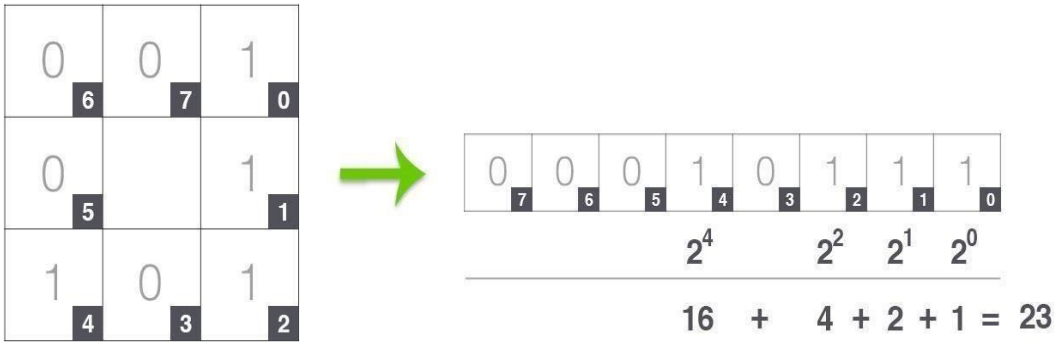

Fig5.2: Taking the 8-bit binary neighborhood of the center pixel and converting it into a decimal representation.

In this example we start at the top-right point and work our way **clockwise** accumulating the binary string as we go along. We can then convert this binary string to decimal, yielding a value of 23. This value is stored in the output LBP 2D array, which we can then visualize below:

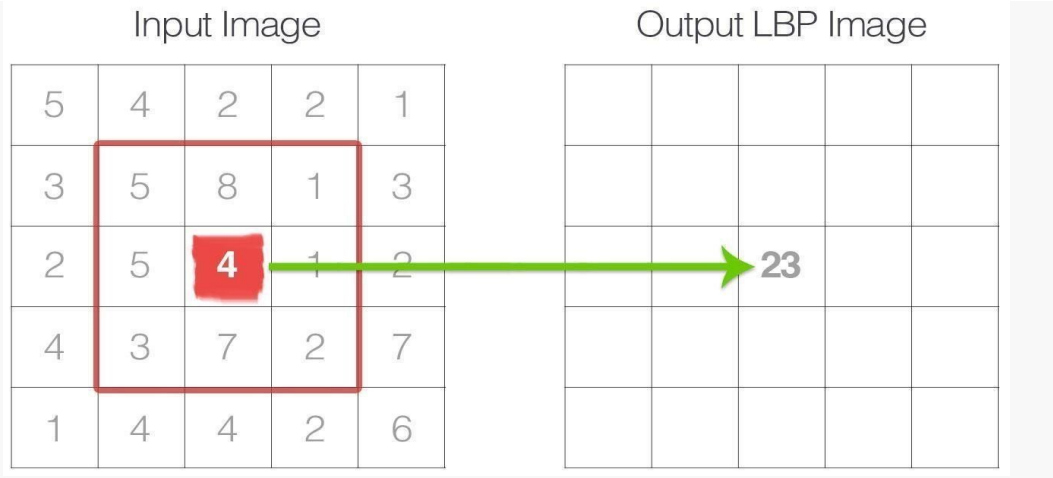

Fig5.3: The calculated LBP value is then stored in an output array with the same width and height as the original image.

This process of thresholding, accumulating binary strings, and storing the output decimal value in the LBP array is then repeated for each pixel in the input image.

Here is an example of computing and visualizing a full LBP 2D array:

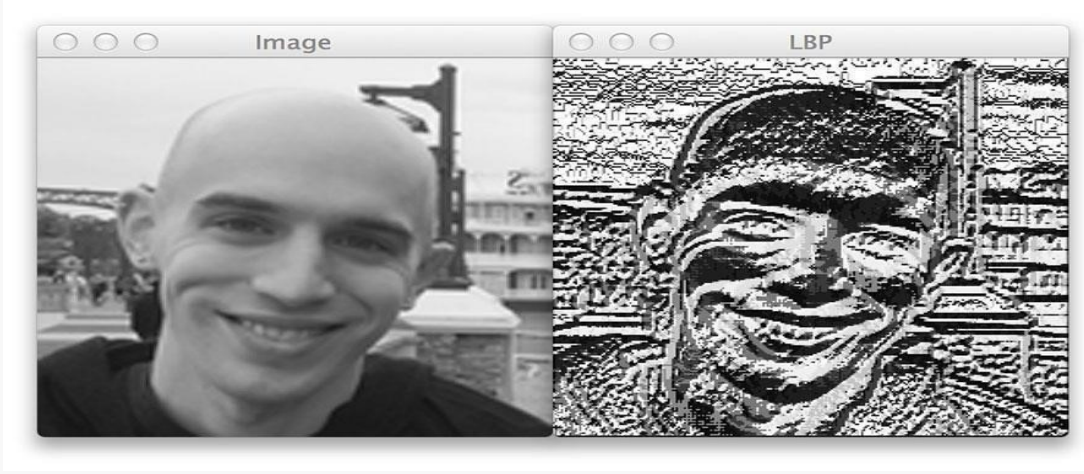

Fig5.4: An example of computing the LBP representation *(right)* from the original input image *(left)*.

The last step is to compute a histogram over the output LBP array. Since a *3 x 3* neighborhood has *2 ^ 8 = 256* possible patterns, our LBP 2D array thus has *a* minimum value of 0 and a maximum value of 255, allowing us to construct a 256-bin histogram of LBP codes as our final feature vector:

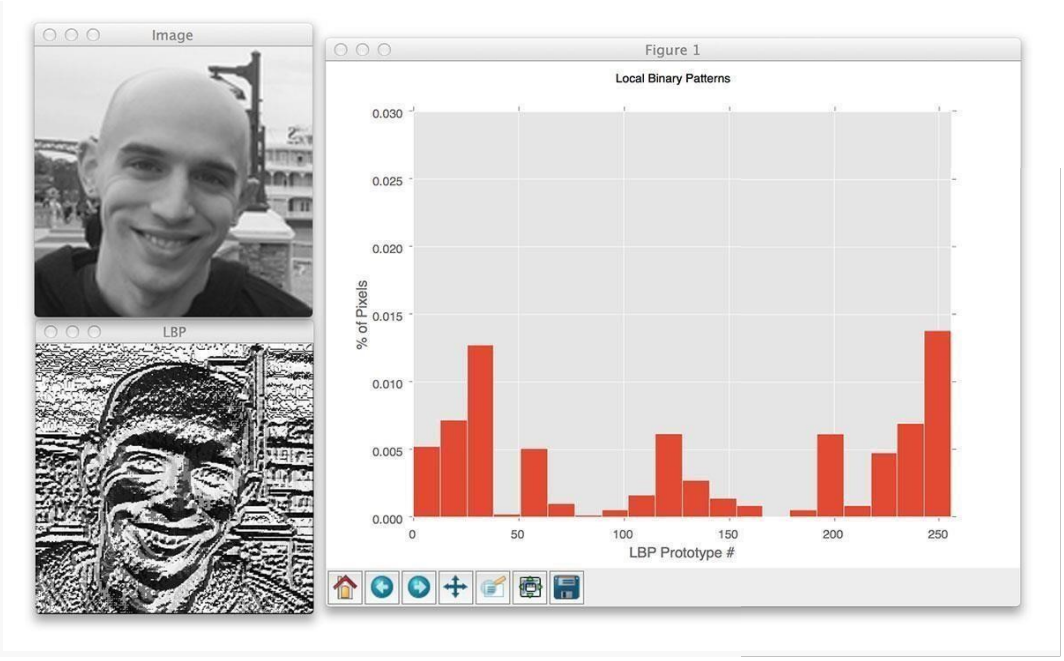

Fig5. 5: Finally, we can compute a histogram that tabulates the number of times each LBP pattern occurs. We can treat this histogram as our feature vector.

A primary benefit of this original LBP implementation is that we can capture extremely fine-grained details in the image. However, being able to capture details at such a small scale is also the biggest drawback to the algorithm — we cannot capture details at varying scales, only the fixed *3 x 3* scale.

To handle this, an extension to the original LBP implementation was proposed by Ojala et al. to handle variable neighborhood sizes. To account for variable neighborhood sizes, two parameters were introduced:

1. The number of points *p* in a circularly symmetric neighborhood to consider (thus removing relying on a square neighborhood).
2. The radius of the circle *r*, which allows us to account for different scales.

Below follows a visualization of these parameters.

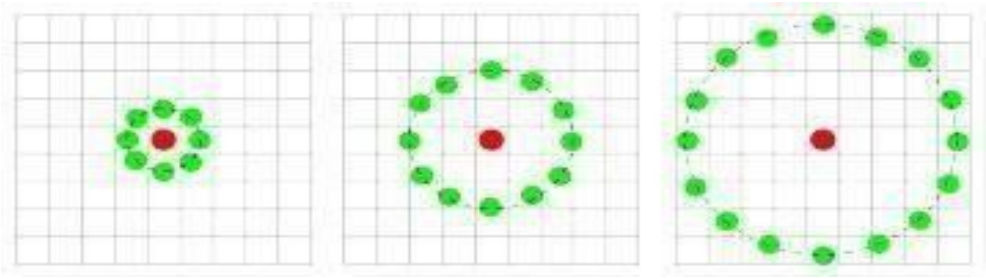

Fig5.6: Three neighborhood examples with varying *p* and *r* used to construct Local Binary Patterns.

Lastly, it's important that we consider the concept of LBP ***uniformity***. A LBP is considered to be uniform if it has ***at most*** two *0-1* or *1-0* transitions. For example, the pattern 00001000 (2 transitions) and 10000000 (1 transition) are both considered to be **uniform patterns** since they contain at most two *0-1* and *1-0* transitions. The pattern 01010010) on the other hand is *not* considered a uniform pattern since it has six *0-1* or *1-0* transitions.

The number of uniform prototypes in a Local Binary Pattern is completely dependent on the number of points *p*. As the value of *p* increases, so will the dimensionality of your resulting histogram. Please refer to the original Ojala et al. paper for the full explanation on deriving the number of patterns and uniform patterns based on this value. However, for the time being simply keep in mind that given the number of points *p* in the LBP there are $p + 1$ **uniform patterns[**66**]**. The final dimensionality of the histogram is thus $p + 2$, where the added entry tabulates all patterns that are ***not uniform***.

So why are uniform LBP patterns so interesting? Simply put: they add an extra level of rotation and grayscale invariance, hence they are commonly used when extracting LBP feature vectors from images.

5.2 Comparing the patterns using Pearson coefficient and checking their Validity

Pearson's correlation coefficient

Pearson's correlation coefficient[67] is a statistical measure of the strength of a linear relationship between paired data. In a sample it is denoted by r and is by design constrained as follows Is a measure of the linear dependence (correlation)[68] between two variables X and Y.? It has a value between +1 and −1 inclusive, where 1 is total positive linear correlation, 0 is no linear correlation, and −1 is total negative linear correlation.

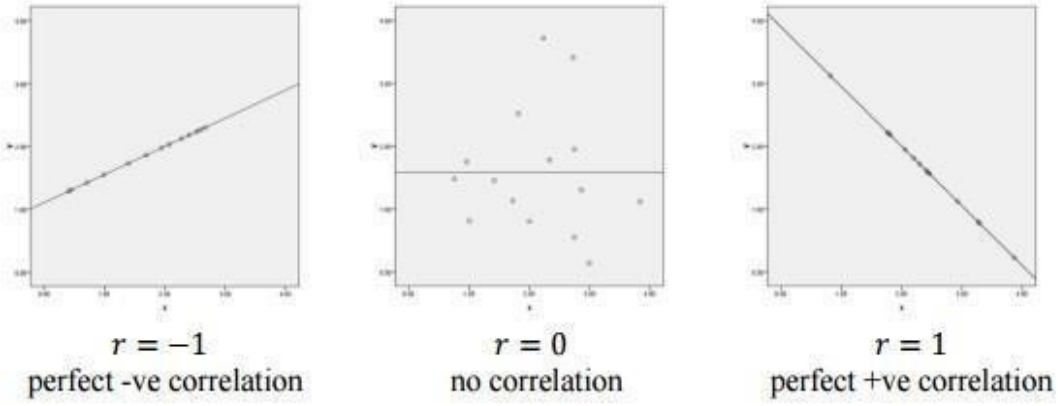

Fig:5.7 correlation

$$r = \frac{\sum XY - \frac{(\sum X)(\sum Y)}{n}}{\sqrt{\left(\sum X^2 - \frac{(\sum X)^2}{n}\right)\left(\sum Y^2 - \frac{(\sum Y)^2}{n}\right)}}$$

Fig 5.8: pearson coefficient formula

Let us take two sample data sets i.e.

A = {1,2,1,3,8,6,5,5} and B = {1,2,2,3,5,6,4,9}

Pearson's coefficient between A and B:

X Values	Y Values
$\sum = 31$	$\sum = 32$
Mean = 3.875	Mean = 4
$\sum(X - M_x)^2 = SS_x = 44.875$	$\sum(Y - M_y)^2 = SS_y = 48$

X and Y Combined	R Calculation
$N = 8$	$r = \sum((X - M_y)(Y - M_x)) / \sqrt{((SS_x)(SS_y))}$
$\sum(X - M_x)(Y - M_y) = 33$	$r = 33 / \sqrt{((44.875)(48))} = 0.711$

The value of R is 0.711. This is a moderate positive correlation, which means there is a tendency for high X variable scores go with high Y variable scores

5.3 Statistical significance

Significance is a statistical term that tells how sure that difference exists. We might be very sure that a relationship exists, but is it a strong, moderate, or weak relationship? After finding a significant relationship, it is important to evaluate its strength. Significant relationships can be strong or weak. Significant differences can be large or small. It just depends on your sample size.

6. CODING

```
import numpy as np            /**Importing numpy*/
import cv2,Tkinter
import scipy                  /** Importing Scipy*/
from scipy.stats.stats import pearsonr   /** importing pearson from scipy*/
import os,random
import glob
from matplotlib import pyplot as plt
from numpy import *
from PIL import Image
def thresholded(center, pixels):
    out = []
    for a in pixels:              /** loop starts and check the conditions*/
        if a >= center:
            out.append(1)
        else:
            out.append(0)
    return out


def get_pixel_else_0(l, idx, idy, default=0):   /** get the pixels of random image*/
    try:     /** Exception occurs*/
        return l[idx,idy]
    except IndexError:
        return default
path2='C:/Users/SAMBA SIVA RAO/Desktop/test'            /** Assigning the dataset path*/
#imgd =random.choice([x for x in os.listdir(path2) if os.path.isfile(os.path.join(path2,x))])
/** pick the random image and assign to the variable.*/
a=random.choice(os.listdir(path2))   /**  Pick the random image in dataset2*/
file=path2+'//'+a
```

```
imgd=cv2.imread(file,0)
transformed_imgd = cv2.imread(file, 0)    /** image transformed*/
lbp2=list()     /** list() function called*/
for x in range(0, len(imgd)):    /** loop started and compare all images with the help of pixel
values. If same image exist then exit the loop */
   for y in range(0, len(imgd[0])):
      center        = imgd[x,y]
/** getting the pixel values of each and every image and assigned to the variables  */
      top_left      = get_pixel_else_0(imgd, x-1, y-1)
      top_up        = get_pixel_else_0(imgd, x, y-1)
      top_right     = get_pixel_else_0(imgd, x+1, y-1)
      right         = get_pixel_else_0(imgd, x+1, y )
      left          = get_pixel_else_0(imgd, x-1, y )
      bottom_left   = get_pixel_else_0(imgd, x-1, y+1)
      bottom_right  = get_pixel_else_0(imgd, x+1, y+1)
      bottom_down   = get_pixel_else_0(imgd, x,   y+1 )
      values = thresholded(center, [top_left, top_up, top_right, right, bottom_right,
                                    bottom_down, bottom_left, left])

      weights = [1, 2, 4, 8, 16, 32, 64, 128]
      res = 0
      for a in range(0, len(values)):
         res += weights[a] * values[a]

      transformed_imgd.itemset((x,y), res)
   lbp2.append(res)
print "List of LBP values for test image: ",lbp2
print len(lbp2)
path='C:/Users/SAMBA SIVA RAO/Desktop/AMI Ear DB/subset-1'
i=1
```

```
for infile in glob.glob(os.path.join(path,'*.jpg')):
    print "\n\n"
    print " \t\tValidating IMAGE",i
    imgb = cv2.imread(infile,0)
    transformed_imgb = cv2.imread(infile, 0)
    lbp1=list()
    for x in range(0, len(imgb)):
        for y in range(0, len(imgb[0])):
            center        = imgb[x,y]
            top_left      = get_pixel_else_0(imgb, x-1, y-1)
            top_up        = get_pixel_else_0(imgb, x, y-1)
            top_right     = get_pixel_else_0(imgb, x+1, y-1)
            right         = get_pixel_else_0(imgb, x+1, y )
            left          = get_pixel_else_0(imgb, x-1, y )
            bottom_left   = get_pixel_else_0(imgb, x-1, y+1)
            bottom_right  = get_pixel_else_0(imgb, x+1, y+1)
            bottom_down   = get_pixel_else_0(imgb, x,   y+1 )
             values = thresholded(center, [top_left, top_up, top_right, right, bottom_right,
                                  bottom_down, bottom_left, left])
/** calculating Image Weights*/
            weights = [1, 2, 4, 8, 16, 32, 64, 128]
            res = 0
            for a in range(0, len(values)):
                res += weights[a] * values[a]
            transformed_imgb.itemset((x,y), res)
        lbp1.append(res)
    print "List of LBP values for image in file: ",lbp1    /** print the LBP values*/
    print len(lbp1)
    pc,ps=pearsonr(lbp1,lbp2)
    if pc==1.0 and ps==0.0:
```

```
        print "MATCHED!!!"   /** If the same image found the print the message*/
        print "pearson coffecient=",pc,"pearson significant=",ps
/** generating the Histograms */
        hist1 = cv2.calcHist([imgb],[0],None,[256],[0,256])
        hist2 = cv2.calcHist([imgd],[0],None,[256],[0,256])
/** print the histograms */
        cv2.imshow('back ear image', imgb)
        cv2.imshow('thresholded image of back ear', transformed_imgb)
        cv2.imshow('down ear image', imgd)
        cv2.imshow('thresholded image of down ear', transformed_imgd)
        hist,bins = np.histogram(imgb.flatten(),256,[0,256])
        plt.hist(transformed_imgb.flatten(),256,[0,256], color = 'r')
        hist,bins = np.histogram(imgd.flatten(),256,[0,256])
        plt.hist(transformed_imgd.flatten(),256,[0,256], color = 'g')
        plt.xlim([0,256])
        plt.legend(('back ear','down ear'), loc = 'upper left')
        plt.show()
       cv2.waitKey(0)   /** waiting state*/
        cv2.destroyAllWindows()  /** close all windows*/
        i=i+1
 else:
        print "pearson coffecient=",pc,"pearson significant=",ps      /** print the pearson coefficient value*/
        print "not matched,lets try next one "

        i=i+1        /** increments the I value */
print "NO matching image found in the dataset"
exit()   /** after comparing all images it exit from the dataset */
```

7. TESTING

We perform mainly two types of testing of testing here. They are

1. Validation

2. Verification

7.1 Validation

The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements. Validation Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on appropriate environment.

By taking an image form the train dataset as test image and passing it to the system we perform validation i.e, whether the system accepting proper image or not;

Let us take a test image one among the dataset

Fig 7.1 test image

Fig 7.2 train image

When the test image gets matched with the dataset image, we get a pearson zero difference.

Fig 7.3: Validating of test images

When we plot the histograms for the matched images, they both coincide and one overrides another.

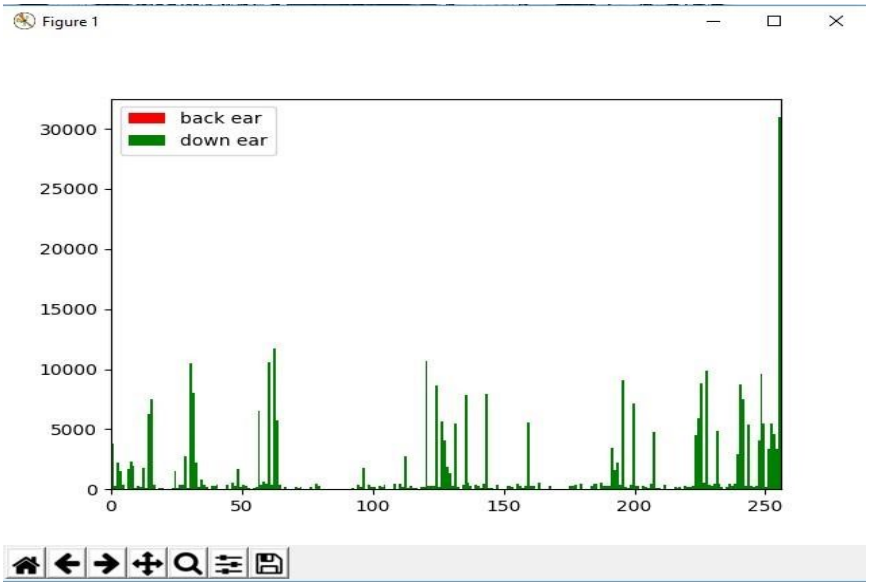

Fig 7.4: histogram of down ear

7.2 Verification Testing:

Verification is the process of evaluating work-products of a development phase to determine whether they meet the specified requirements. verification ensures that the product is built according to the requirements and design specifications. It also answers to the question, Are we building the product right?

Here we do verification by taking a test image out of dataset, to see if the system is correctly discarding it or not.

Fig 7.5: Image out of dataset

As no image from the dataset matches it, we get the output as

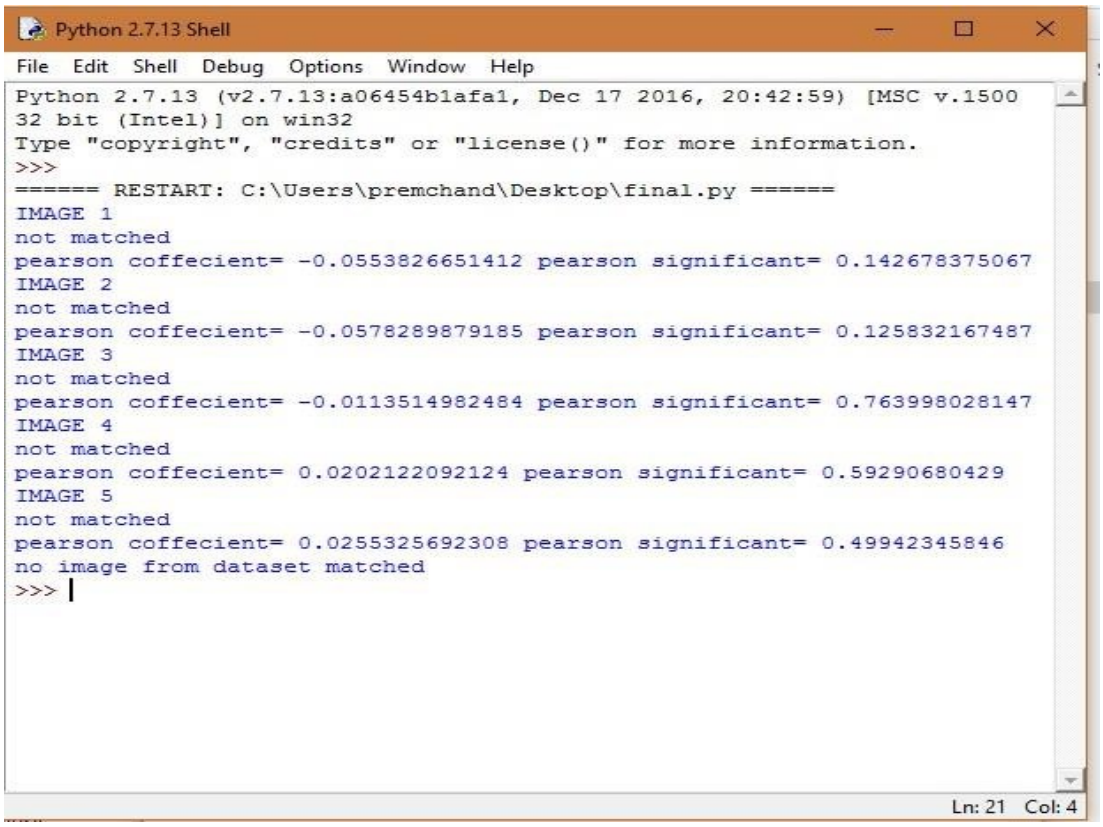

Fig 7.6: No image matched with test image

Performance Analysis:

Different metrics can be used to rate the performance of a biometric factor, solution or application. The most common performance metrics are the False Acceptance Rate FAR and the False Rejection Rate FRR.

When using a biometric application for the first time the user needs to enroll to the system. The system requests fingerprints, a voice recording or another biometric factor from the operator, this input is registered in the database as a template which is linked internally

to a user ID. The next time when the user wants to authenticate or identify himself, the biometric input is compared to the templates in the database by a matching algorithm which responds with acceptance (match) or rejection (no match).

FAR and FRR

The FAR or False Acceptance rate is the probability that the system incorrectly authorizes a non-authorized person, due to incorrectly matching the biometric input with a template. The FAR is normally expressed as a percentage, following the FAR definition this is the percentage of invalid inputs which are incorrectly accepted.

The FRR or False Rejection Rate is the probability that the system incorrectly rejects access to an authorized person, due to failing to match the biometric input with a template. The FRR is normally expressed as a percentage, following the FRR definition this is the percentage of valid inputs which are incorrectly rejected.

FAR and FRR are very much dependent on the biometric factor that is used and on the technical implementation of the biometric solution. Furthermore the FRR is strongly person dependent, a personal FRR can be determined for each individual. Take this into account when determining the FRR of a biometric solution, one person is insufficient to establish an overall FRR for a solution. Also FRR might increase due to environmental conditions or incorrect use, for example when using dirty fingers on a fingerprint reader. Mostly the FRR lowers when a user gains more experience in how to use the biometric device or software.

FAR and FRR are key metrics for biometric solutions, some biometric devices or software even allow to tune them so that the system more quickly matches or rejects. Both FRR and FAR are important, but for most applications one of them is considered most important. Two examples to illustrate this:

1. When biometrics are used for logical or physical access control, the objective of the application is to disallow access to unauthorized individuals under all circumstances. It is

clear that a very low FAR is needed for such an application, even if it comes at the price of a higher FRR.

2. When surveillance cameras are used to screen a crowd of people for missing children, the objective of the application is to identify any missing children that come up on the screen. When the identification of those children is automated using a face recognition software, this software has to be set up with a low FRR. As such a higher number of matches will be false positives, but these can be reviewed quickly by surveillance personnel.

False Accept Rate is also called False Match Rate, and False Reject Rate is sometimes referred to as False Non-Match Rate.

FRR on ear data set:

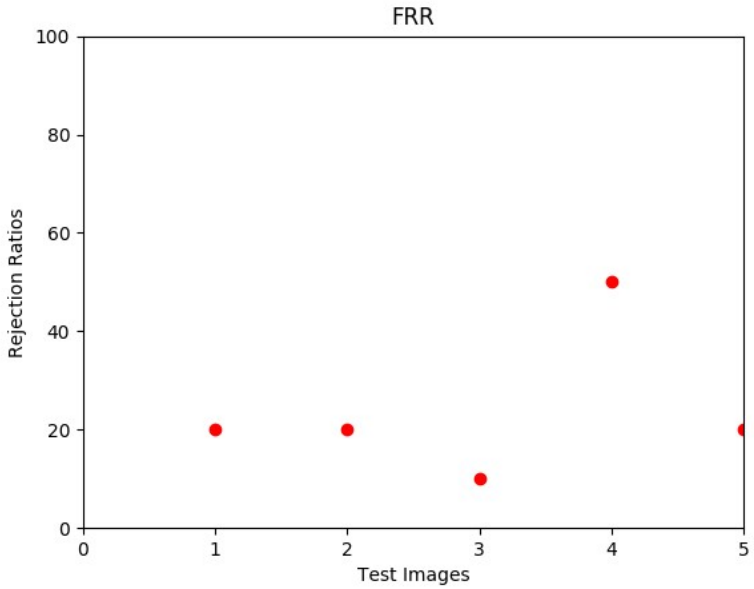

Fig 7.7: A graphical representation of FRR

If a test image belongs to any of the category of the dataset images, then it must be matched to all the images of the category. Otherwise it is said to be incorrectly rejected. Here

we calculate the False Rejection Ratios for some test images. Each test image is checked with only the members of its category and we take the count values of the number of images it matched and rejected. After that rejection ratio is calculated as the number of rejected images to total number of images tested. The ratio of each test image is appended to a list and at the end it is plotted in the form a graph with ylabel as Rejection Ratios and xlabel as Test image.

FAR on ear data set:

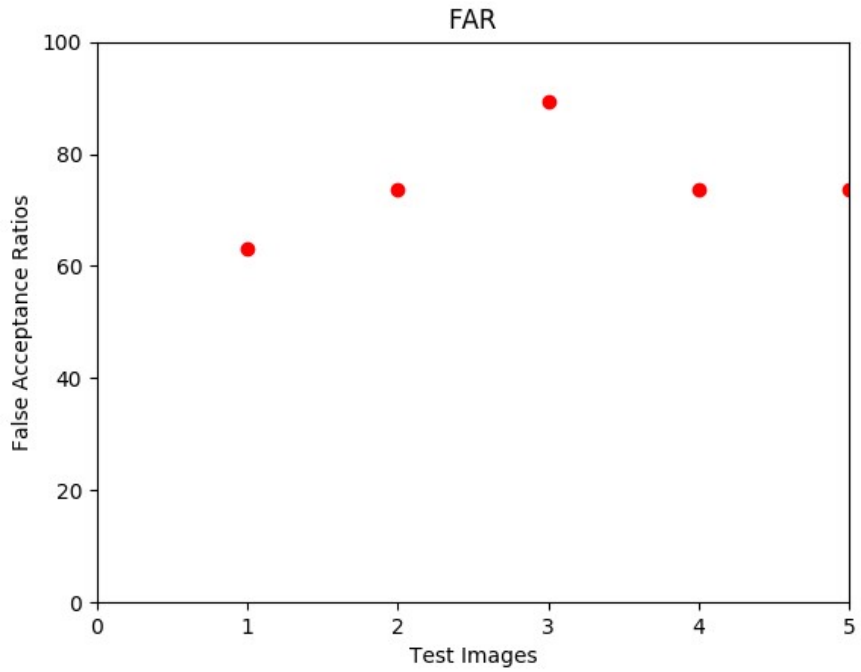

Fig 7.8: A graphical representation of FAR

If an image which does resemble any of the dataset images, is accepted by the system then it is accepted incorrectly. Let us take a group of test images which are not part of dataset. For each test image take the counts of number of images it is matched and rejected. False Acceptance Ratio(FAR) is number of images it is accepted to total number of images on which it is tested. Appending the ratio values of all the test images to a list and at the end a graph is plotted with xlabel as Test image and ylabel as the False acceptance ratios.

8. Output

Ear data set

Fig 8.1 ear data set.

Command prompt

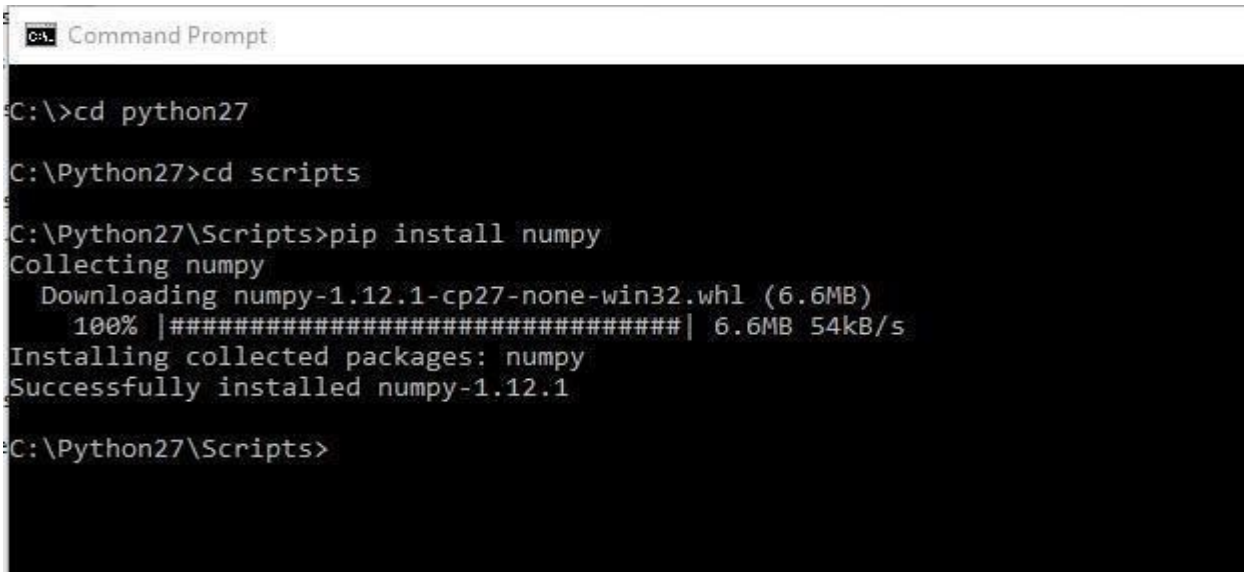

Fig 8.2 command prompt

Validating image

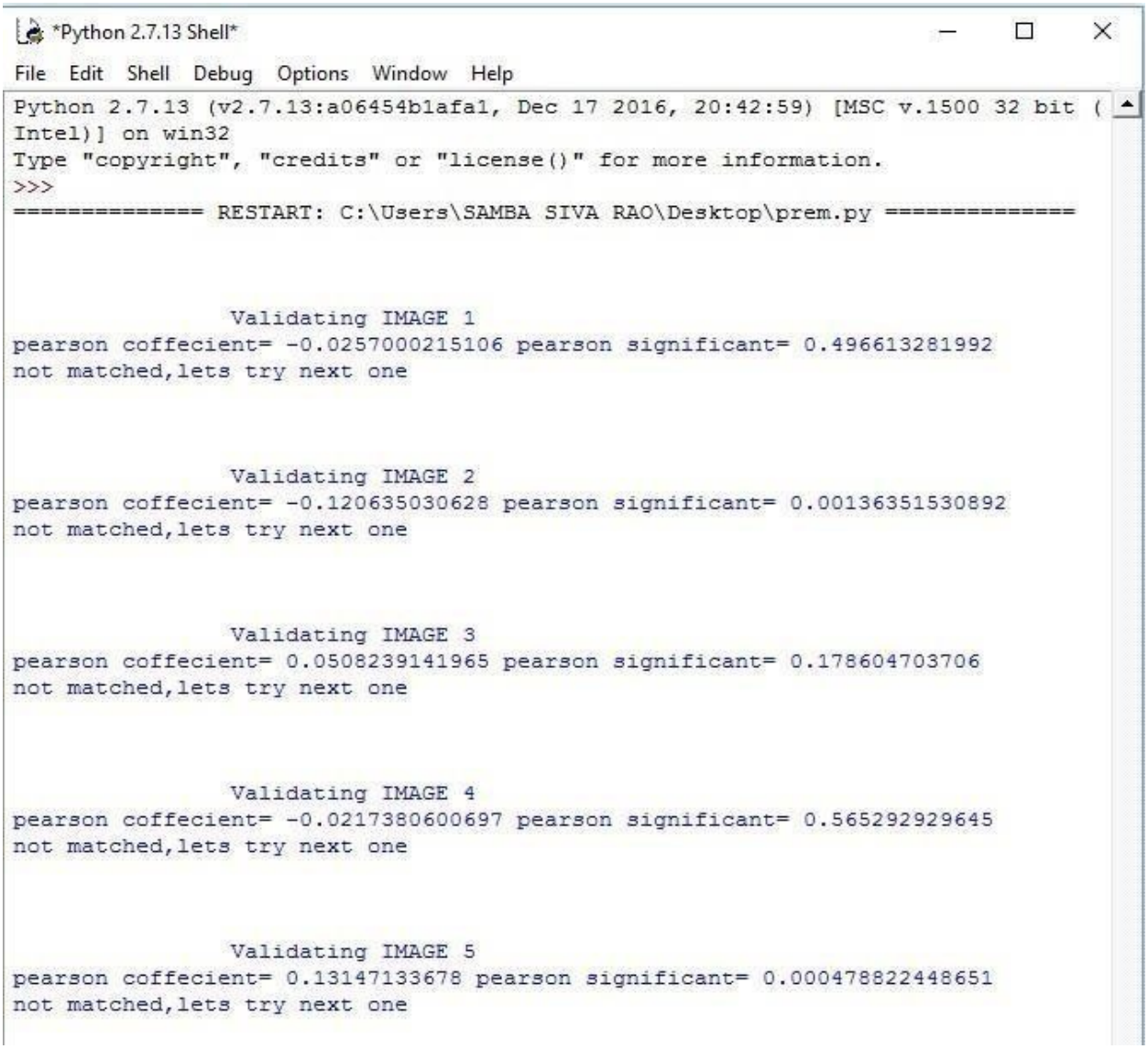

Fig 8.3 validating image

Gray scale image

Fig 8.4 gray scale image

Threshold image

Fig 8.5 threshold image

Histogram of the threshold image

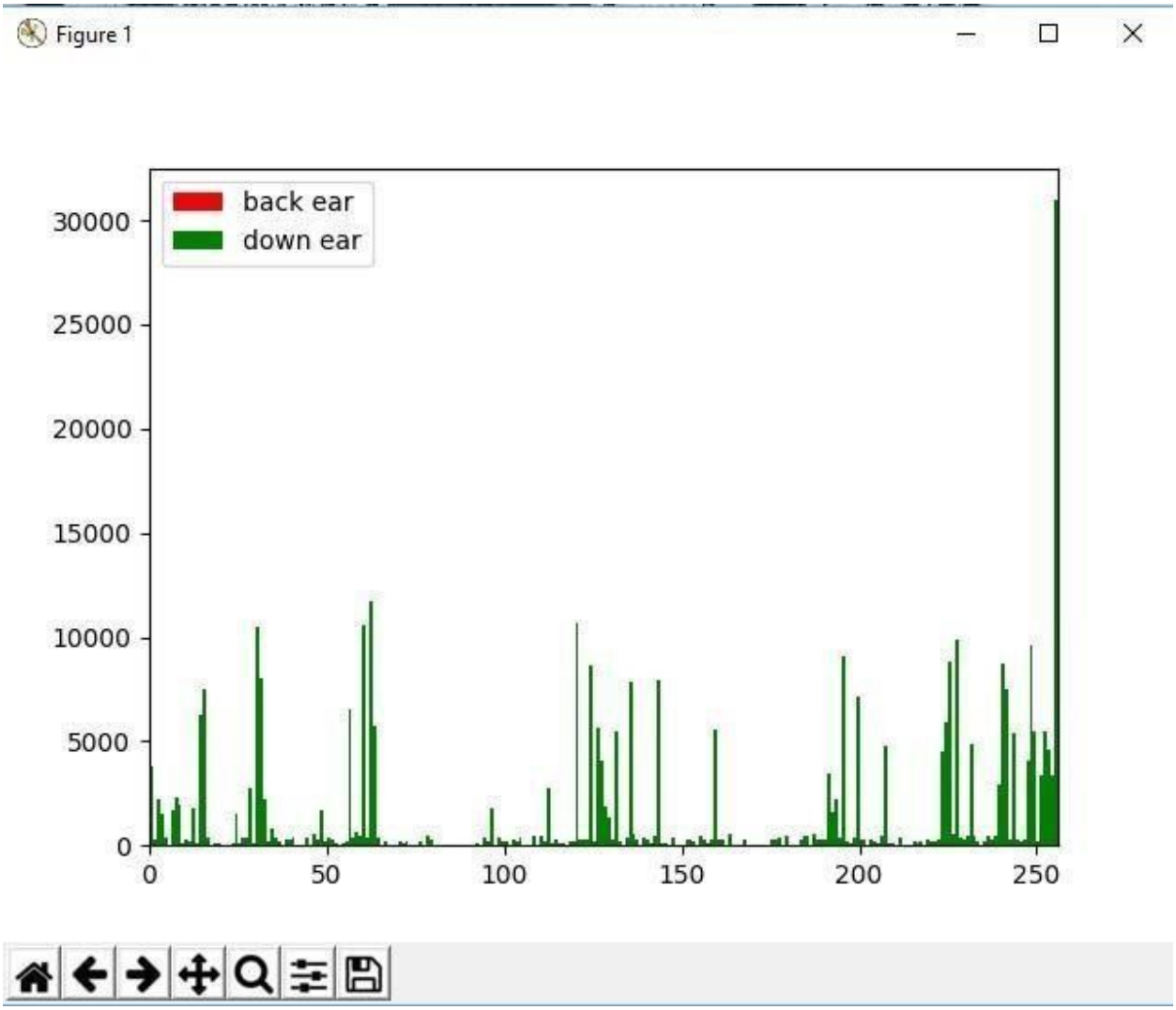

Fig 8.6 histogram of the threshold image

9. References

[1] T. Ojala and M. Pietikainen. Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns, IEEE Trans on Pattern Analysis and Machine Intelligence, Vol. 24. No.7, July, 2016.

[2] Ahonen, T., Hadid, A., Pietikäinen, M.: Face description with local binary patterns: Application to face recognition. IEEE Trans. Pattern Anal. Mach. Intell. 28(12), 2037–2041 (2006).

[3] AnaghaV.Malkapurkar, Rupali Patil, Prof. Sachin Murarka, A New Technique for LBP Method to Improve Face Recognition, International Journal of Emerging Technology and Advanced Engineering ISSN 2250-2459, Volume 1, Issue 1, November 2014.

[4] Ahonen, T., Hadid, A., Pietikäinen, M.: Face Recognition with Local Binary Patterns. In: Computer Vision, ECCV 2004 Proceedings, Lecture Notes in Computer Science 3021 (2004) 469481.

[5] Caifeng Shan, Shaogang Gong , Peter W. McOwan Facial expression recognition based on
Local Binary Patterns: A comprehensive study. Image and Vision Computing 27 (2009) 803–816

[6] C. Shan, S. Gong, P.W. McOwan, Robust facial expression recognition using local binary patterns, in: IEEE International Conference on Image Processing (ICIP), Genoa, vol. 2, 2005, pp. 370–373.

[7] Li Liu, Lingjun Zhao, Yunli Long, Gangyao Kuang, Paul Fieguth, Extended local binary patterns for texture classification : Image and Vision Computing 30 (2012) 86–99.

[8] Md. Abdur Rahim, Md. Najmul Hossain, Tanzillah Wahid & Md. Shafiul Azam: Face Recognition using Local Binary Patterns (LBP), Global Journal of Computer Science and Technology Graphics & Vision, Volume 13 Issue 4 Version 1.0 Year 2013.

[9] Bo Yang, SongcanChen, A comparative study on local binary pattern (LBP) based face recognition: LBP histogram versus LBP image: Neuro computing120(2013)365–379.

[10] Timo Ahonen, Abdenour Hadid, and Matti Pietik¨ainen, Face Recognition with Local Binary Patterns, Machine Vision Group, Pattern Analysis and Machine Intelligence 24 (2002) 971– 987.

[11] D. Chetverikov and R. P´eteri, “A Brief Survey of Dynamic Texture Description and Recognition,” Proc. Int’l Conf. Computer Recognition Systems, pp. 17-26, 2005.

[12] T. Ojala, M. Pietik¨ainen, and T. M¨aenp¨a¨a, “Multiresolution Gray Scale and Rotation Invariant Texture Analysis with Local Binary Patterns,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 971-987, 2002.

[13] M. Pantic and L.L.M. Rothkrantz, “Automatic Analysis of Facial Expressions: The State of the Art,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no.12, pp. 14241455, 2000.

[14] B. Fasel and J. Luettin, “Automatic Facial Expression Analysis: A Survey,” Pattern Recognition, vol. 36, pp. 259-275, 2003.

[15] T. Kanade, J.F. Cohn, and Y. Tian, “Comprehensive Database for Facial Expression Analysis,” Proc. Int’l Conf. Automatic Face and Gesture Recognition, pp. 46-53, 2000.

[16] X. Feng, M. Pietik¨ainen, and A. Hadid, “Facial Expression Recognition with Local Binary Patterns and Linear Programming,” Pattern Recognition and Image Analysis, vol. 15, no. 2, pp. 546-548, 2005.

[17] C. Shan, S. Gong and P.W. McOwan, “Robust Facial Expression Recognition Using Local Binary Patterns,” Proc. IEEE Int’l Conf. Image Procession, pp. 370-373, 2005.

[18] M.S. Bartlett, G. Littlewort, I. Fasel, and R. Movellan, “Real Time Face Detection and Facial Expression Recognition: Development and Application to Human Computer Interaction,” Proc. CVPR Workshop on Computer Vision and Pattern Recognition for Human-Computer Interaction, 2003.

[19] G. Littlewort, M. Bartlett, I. Fasel, J. Susskind, and J. Movellan, “Dynamics of Facial Expression Extracted Automatically from Video,” Proc. IEEE Workshop Face Processing in Video, 2004.

[20] M. Yeasin, B. Bullot, and R. Sharma, “From Facial Expression to Level of Interest: A Spatiotemporal Approach,” Proc. Conf. Computer Vision and Pattern Recognition, pp. 922927, 2004.

[21] S.P. Aleksic and K.A. Katsaggelos, “Automatic Facial Expression Recognition Using Facial Animation Parameters and Multi-stream HMMS,” IEEE Trans. Signal Processing, Supplement on Secure Media, 2005.

[22] I. Cohen, N. Sebe, A. Garg, L.S. Chen, and T.S. Huang, “Facial Expression Recognition from Video Sequences: Temporal and Static Modeling,” Computer Vision and Image Understanding, vol. 91, pp. 160- 187, 2003.

[23] R.C. Nelson and R. Polana, “Qualitative Recognition of Motion Using Temporal Texture,” Computer Vision, Graphics, and Image Processing, vol. 56, no. 1, pp. 78-99, 1992.

[24] P. Bouthemy and R. Fablet, “Motion Characterization from Temporal Co-occurrences of Local Motion-based Measures for Video Indexing,” Proc. Int’l Conf. Pattern Recognition, vol.1, pp. 905-908, 1998.

[25] R. Fablet and P. Bouthemy, “Motion Recognition Using Spatio-temporal Random Walks in Sequence of 2D Motion-related Measurements,” IEEE Int’l Conf. Image Processing, pp. 652655, 2001.

[26] R. Fablet and P. Bouthemy, “Motion Recognition Using Nonparametric Image Motion Models Estimated from Temporal and Multiscale Cooccurrence Statistics,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 25, pp. 1619-1624, 2003.

[27] Z. Lu, W. Xie, J. Pei, and J. Huang, “Dynamic Texture Recognition by Spatiotemporal Multiresolution Histogram,” Proc. IEEE Workshop Motion and Video Computing, vol. 2, pp. 241246, 2005.

[28] C.H. Peh and L.-F. Cheong, “Exploring Video Content in Extended Spatiotemporal Textures,” Proc. 1st European Workshop Content-Based Multimedia Indexing, pp. 147-153, 1999.

[29] C.H. Peh and L.-F. Cheong, “Synergizing Spatial and Temporal Texture,” IEEE Trans. Image Processing, vol. 11, pp. 1179-1191, 2002.

[30] R. P´eteri and D. Chetverikov, “Qualitative Characterization of Dynamic Textures for Video Retrieval,” Proc. Int’l Conf. Computer Vision and Graphics, 2004.

[31] S. Fazekas and D. Chetverikov, “Normal Versus Complete Flow in Dynamic Texture Recognition: A Comparative Study,” 4th Int’l Workshop Texture Analysis and Synthesis, pp. 3742, 2005.

[32] P. Saisan, G. Doretto, Y.N. Wu, and S. Soatto, “Dynamic Texture Recognition,” Proc. Conf. Computer Vision and Pattern Recognition, vol. 2, pp. 58-63, 2001.

[33] K. Fujita and S.K. Nayar, “Recognition of Dynamic Textures Using Impulse Responses of State Variables,” Proc. Third Int’l Workshop Texture Analysis and Synthesis, pp. 31-36, 2003.

[34] J.R. Smith, C.-Y. Lin, and M. Naphade, “Video Texture Indexing Using Spatiotemporal Wavelets,” Proc. IEEE Int’l Conf. Image Processing, vol. 2, pp. 437-440, 2002.

[35] K. Otsuka, T. Horikoshi, S. Suzuki, and M. Fujii, “Feature Extraction of Temporal Texture Based on Spatiotemporal Motion Trajectory,” Proc. Int’l Conf. Pattern Recognition, vol. 2, pp. 1047-1051, 1998.

[36] J. Zhong and S. Scarlaro, “Temporal Texture Recognition Model Using 3D Features,” Technical report, MIT Media Lab Perceptual Computing, 2002.

[37] G. Aggarwal, A.R. Chowdhury, and R. Chellappa, “A System Identification Approach for Video-based Face Recognition,” Proc. 17th Int’l Conf. Pattern Recognition, vol. 1, pp. 175178, 2004.

[38] G. Zhao and M. Pietik¨ainen, “Dynamic Texture Recognition Using Volume Local Binary
Patterns,” Proc. Workshop on Dynamical Vision WDV 2005/2006, LNCS 4358, pp. 165177, 2007.

[39] G. Zhao and M. Pietik¨ainen, “Local Binary Pattern Descriptors for Dynamic Texture Recognition,” Proc. Int’l Conf. Pattern Recognition, vol. 2, pp. 211-214, 2006.

[40] T. Ahonen, A. Hadid, and M.Pietik¨ainen, “Face Recognition with Local Binary Patterns,” Proc. Eighth European Conf. Computer Vision, pp. 469- 481, 2004.

[41] T. Ahonen, A. Hadid, and M.Pietik¨ainen, “Face Description with Local Binary Patterns: Application to Face Recognition,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 28, no. 12, pp. 2037-2041, 2006.

[42] J. Bassili, “Emotion Recognition: The Role of Facial Movement and the Relative Importance of Upper and Lower Areas of the Face,” Journal of Personality and Social Psychology, vol. 37, pp. 2049-2059, 1979.

[43] Chetan Ballur & Shylaja’s “Application of Local Binary Pattern and Principal Component
Analysis for Face Recognition”International Journal of Electrical, Electronics and Data communication, issn (p): 2320-2084, volume-1, issue-, july-2013.

[44] Fabrice Bourel & C.C. Chibelushi’s “Facial Expression Recognition:A Brief tutorial Overview” , School of Computing, Staffordshire University, Beaconside, Stafford ST18 0DG, ORSYP, 101 quartier Boieldieu, La Défense 8, F-92042 Paris La Défense Cedex, France.

[45] Taranpreet Singh Ruprah’s “Face Recognition Based on PCA Algorithm”, Special Issue of
International Journal of Computer Science & Informatics (IJCSI), ISSN (PRINT) : 2231– 5292, Vol.- II, Issue-1, 2.

[46] Suman Kumar Bhattacharyya & Kumar Rahul’s “Face Recognition by Linear Discriminant
Analysis” , International Journal of Communication Network Security, ISSN: 2231 – 1882, Volume-2, Issue-2, 2013.

[47] S´ebastien Marcel, Yann Rodriguez and Guillaume Heusch’s “On the Recent Use of Local
Binary Patterns for Face Authentication”, International Journal of Image and Video Processing, special issue on Facial Image Processing, Manuscript received May 1, 2006; revised October 20, 2006; accepted May 25, 2007.

[48] Rajendra A Kerkar, Yogesh V Kulkarni, “Screening for cervical cancer : an overview”, J Obstet Gynecol India Vol. 56, No. 2 : March/April 2006 , Pg 115-122.

[49] Mangal Mahajan, Rajesh Kuber, KR Chaudhari, Prashant Chaudhari, Pravin Ghadage, and Rushikesh Naik, "MR imaging of carcinoma cervix", Indian J Radiol Imaging. 2013 Jul-Sep; 23(3): 247–252.

[50] Mustafa, N., N. A. Mat Isa, M. Y. Mashor, and N. H. Othman. "New Features of Cervical Cells for Cervical Cancer Diagnostic

[51] System Using Neural Network." IJSSST, Vol. 9, No. 2, | May 2008.

[52] Das, Abhishek, Avijit Kar, and Debasis Bhattacharyya. "Preprocessing for Automating Early Detection of Cervical Cancer." In Information Visualisation (IV), 15th International Conference on, pp. 597-600. IEEE, 2011.

[53] Asselin, Marie-Claude, James PB OConnor, Ronald Boellaard,

[54] Neil A. Thacker, and Alan Jackson. "Quantifying heterogeneity in human tumours using MRI and PET." European Journal of Cancer 48, no. 4 (2012): 447-455.

[56] Turid Torheim, EirikMalinen, Knut Kvaal, Heidi Lyng, Ulf G. Indahl, Erlend F. Andersen, and Cecilia M. Futsæther,"Classification of Dynamic Contrast Enhanced MR Images of Cervical Cancers Using Texture Analysis and Support Vector Machines",IEEE Transactions on medical imaging, Vol. 33, No. 8, Aug 2014.

[57] Jagadeeswari, S.Malarkhodi, "Segmentation and Classification Using Artificial Neural Network of Cervical Cancer in Magnetic Resonance Image", International Journal of Mathematics and Computer Research, Volume 2 issue 5 May 2014.

[58] Rupinderpal Kaur, Rajneet Kaur / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 Vol. 3, Issue 2, March -April 2013, pp.1611-1614.

[59] Shih-Chia Huang, Fan-Chieh Cheng, and Yi-Sheng Chiu," Efficient Contrast Enhancement Using Adaptive Gamma Correction With Weighting Distribution" , IEEE Transactions On Image Processing, Vol. 22, No. 3, March 2013.

[60] Hetal J. Vala, Prof. Astha Baxi, "A Review on Otsu Image Segmentation Algorithm", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 2, Issue 2, February 2013, ISSN: 2278 – 1323.

[61] Robert M. Haralick, K. Shanmugam and It'shak Dinstein, Texture Features for Image Classification, IEEE Transactions on Systems, Man and Cybernetics, 3(6), 1973, 610-621.

[62] Mohanaiah, P. Sathyanarayana, L. GuruKumar, "Image Texture Feature Extraction Using GLCM Approach", International Journal of Scientific and Research Publications, Volume 3, Issue 5, May 2013.

[63] Jain, "Brain Cancer Classification Using GLCM Based Feature Extraction in Artificial Neural Network",International Journal of Computer Science & Engineering TechnologyVol. 4, Issue 7, pp. 966-970, 2013.

[64] Cristianini and J. Shawe-Taylor, An Introduction to Support Vector Machines : And Other Kernel-Based Learning Methods. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[65] Apeksha Chaudhari ,Poof. K .S.Bhagat, "An Overview of Content Based Image Categorization Using Support Vector Machine", IJISET – International Journal of Innovative Science, Engineering & Technology, Vol. 1 Issue 10, December 2014, ISSN 2348 – 7968.

[66] Bhanu and H. Chen, Human ear recognition in 3D. In Workshop on Multimodal User Authentication, pages 91–98 (2003).

[67] Burge and W. Burger, Ear biometrics. In Biometrics: Personal Identification in Networked Society, pages 273–286, Kluwer Academic (1999).

[68] Burge and W. Burger, Ear biometrics in computer vision. In 15th International Conference of Pattern Recognition, volume 2, pages 822–826 (2000).

[69] Chang, K. Bowyer and V. Barnabas, Comparison and combination of ear and face images in appearance-based biometrics. In IEEE Trans. Pattern Anal. Machine Intell. volume 25, pages 1160–1165 (2003).

[70] Chang, K. Bowyer and P. Flynn, Face recognition using 2D and 3D facial data. In Workshop on Multimodal User Authentication, pages 25–32 (2003).

[71] Iannarelli, Ear identification. In Forensic identification series, Fremont, California, Paramont Publishing Company (1989).

[72] Victor, K. Bowyer and S. Sarkar, An evaluation of face and ear biometrics. In 16th International Conference of Pattern Recognition, pages 429–432 (Aug. 2002).

[73] Pulli, Multiview registration for large data sets. In Second International Conference on 3-D Imaging and Modeling (3DIM '99), pages 160–168 (October 04-08, 1999).

[74] Huttenlocher, G. Klanderman and W. Rucklidge, Comparing images using the hausdorff distance. In IEEE Trans. Pattern Anal. Machine Intell. volume 15(9), pages 850–863 (1993)

[75] Design & Implementation of 3D-DWT for Video Processing Applications, International Journal of Advanced Electrical & Electronics Engineering(IJAEEE), vol 1, issue 3, 2012. ISSN: 2278-8948

[76] R.E. Haralick, K.Shanmugam, I.Dinstein, Textural Features for Image Classification, IEEE
Transactionson Systems, Man and Cybernetics, Vol. SMC-3, No.6, Nov 1973

[77] J.L. Hennesy& D.A. Patterson, Computer Architecture, A Quantitative Approach, Morgan Kaufmann,May 2002

[78] D.E. Maroulis, D.K. Iakovidis, S.A. Karkanis, D.A. Karras, Cold: a versatile detection system for colorectallesions in endoscopy video frames, Computer Methods and Programs in Biomedicine, vol 70, no. 2, pp. 99-186, Elsevier Science, 2003

[79] W. Luk, P. Andreou, A. Derbyshire, F. Dupont-De- Dinechin, J. Rice, N. Shirazi and D. Siganos, Field-Programmable Logic: From FPGAs to Computing Paradigm, SpringerVerilog, Berlin, 1998

[80] M. Nibouche, A. Bouridane, D. Crookes, O. Nibouche, An FPGA-based wavelet transforms coprocessor, in Proc. IEEE Int. Conf. Image Processing, pp. 194-197, vol.3, 2003.

[81] H. Hikawa, Implementation of Simplified Multilayer Neural Network with On-chip Learning, Proc. of theIEEE International Conference on Neural Networks(Part 4), Vol. 4, 1999, pp 16331637.

[82] T. Nakano, T. Morie, and A. Iwata, A Face/Object Recognition System Using FPGA Implementation of Coarse Region Segmentation, SICE Annual Conference 2003, pp. 14181423, Fukui, Aug. 4-6, 2003.

[83] K. Heikkinen and P. Vuorimaa, Computation of Two Texture Features in Hardware, Proceedings of the 10thInternational Conference on Image Analysis and Processing, Venice, Italy, pages 125-129, September 27-29, 1999.

[84] K.C. Chang, Digital Design and Modeling with VHDL and Synthesis, IEEE Computer Society Press - Wiley, 1997

[85] ISO/IEC JTC 1/SC 29/WG 1 N1646R, JPEG2000 part 1 final committee draft version 1.0, 2000.

[86] C,-T.Huang, P-C, Tseng, and L,-G Chen, “Generic RAM-based architectures for twodimensional discrete wavelet transform with line based method,” IEEE Trans. on Circuits and Systems, vol. 1, pp. 363-366, 2002.

[87] L. Hong, A. K. Jain, and S. Pankanti, \Can multibiometrics improve performance," in Proceedings AutoID'99, (Summit(NJ), USA), pp. 59{64, Oct 1999.

[88] A. K. Jain, S. Prabhakar, and S. Chen, \Combining multiple matchers for a high security fingerprint verication system," Pattern Recognition Letters, vol. 20, pp. 1371{1379, 1999.

[89] Y. Zuev and S. Ivanon, \The voting as a way to increase the decision reliability," in Foundations of Information/Decision Fusion with Applications to Engineering Problems, (Washington D.C., USA), pp. 206{210, Aug 1996.

[90] R. Cappelli, D. Maio, and D. Maltoni, Combining fingerprint classifiers," in First International Workshop on Multiple Classier Systems, pp. 351{361, Jun 2000.

[91] J. Kittler, M. Hatef, R. P. Duin, and J. G. Matas, On combining classi_ers," IEEE Transactions on PAMI, pp. 226{239, Mar 1998.

[92] E. Bigun, J. Bigun, B. Duc, and S. Fischer, \Expert conciliation for multimodal person authentication systems using Bayesian statistics," in First International Conference on AVBPA, (Crans Montana, Switzerland), pp. 291{300, Mar 1997.

[93] S. Ben-Yacoub, Y. Abdeljaoued, and E. Mayoraz, \Fusion of face and speech data for person identity veri_cation," Research Paper IDIAP-RR 99-03, IDIAP, CP 592, 1920 Martigny, Switzerland, Jan 1999.