Original Research Article

An Accurate System for Face Detection and Recognition

Abstract

During the last few years, Local Binary Patterns (LBP) has aroused increasing interest in image processing and computer vision. LBP was originally proposed for texture analysis, and has proved a simple yet powerful approach to describe local structures. It has been extensively exploited in many applications, for instance, face image analysis, image and video retrieval, environment modeling, visual inspection, motion analysis, biomedical and aerial image analysis, remote sensing. Face recognition is an interesting and challenging problem, and impacts important applications in many areas such as identification for law enforcement, authentication for banking and security system access, and personal identification among others. In this paper we are concerned with face recognition in a video stream using Local Binary Pattern histogram with processed data. First we will detect faces by using a combination of Haar cascade files that uses skin detection, eye detection and nose detection as input of LBP to increase the accuracy of the proposed recognition system. Also, our system can be used to build a dataset of faces and names to be used in a recognition step. The experimental results have shown that the proposed system can achieve accuracy of recognition up to 96.5% which was better than the relevant methods.

Keywords*:*

Face detection; Face Recognition; Local Binary Pattern LBP; Histogram; OpenCV.

1. INTRODUCTION

Authentication is an important issue in system control in computer based communication. Human face recognition is a significant branch of biometric verification and has been used widely in many applications, such as video monitor system, human-computer interaction, and door control system and network security (1). In this paper we are concerned with objects motion so we try to modify face recognition technique to recognize faces in a video stream using Local Binary Pattern histogram with processed data. First, we will detect faces by using a combination of Haar cascade files that uses skin detection, eye detection and nose detection as input of LBP to increase the accuracy of the proposed recognition system.

Motion detection is the process to detect a change in the position of an object comparative to its surroundings also or a variation in the environment which surround to an object. By mechanical or electronic methods motion detection can be achieved. Motion detection is done by natural organisms, we named it motion perception (2).

One of algorithm which used to detect the motion in image processing is motion segmentation algorithm for the detection of objects which are moving in image sequences needed from a moving platform which contains two steps. First, estimate the image plane transformation induced by the moving platform using a sub pixel accuracy image registration algorithm (3)**.** As known that the registration algorithm is fully automatic and performs well under significant camera rotation and translation. The input images are then transformed into a common coordinate system (for example, the coordinate system of the first image). Then, segments the changed regions from the camera motion-compensated frame difference. Finally, the moving object is detected from the closure of changing segments, and the object motion parameters are estimated (4)**.**

In this paper, we are concerned with face recognition in a video stream. The structure of the paper is as follows: Section 2 gives a brief overview of tow face recognition methods which are Eigenfaces and LBPH. In Section 3, the Proposed Methodology is introduced. Section 4 presents the experimental results. We presented the conclusion and future work in section 5 and 6.

2. FACE RECOGNITION

Face Detection has the target of finding the faces (location and size) in an image and may extract them to be used by the face recognition algorithm. But **Face Recognition** with the facial images already extracted, cropped, resized and usually converted to grayscale, the face recognition algorithm is responsible for finding characteristics which best describe the image (5)**.**

A facial recognition system is a technology able to identify or verify a person from a digital image or a video frame from a video source. There are multiple methods in which facial recognition systems work, but in general, all of them work by comparing selected facial features from given image with faces within a dataset or database. It is also described as a Biometric Artificial

Intelligence based application which can uniquely identify a person by analyzing patterns based on the person's facial textures and shape (6)**.** While initially a form of computer application, in recent times it has seen wider uses on mobile platforms and in other forms of technology, such as robotics. It is typically used as access control in security systems and can be compared to other biometrics such as fingerprint or eye iris recognition systems. Although the accuracy of facial recognition system as a biometric technology is lower than iris recognition and fingerprint recognition, it is widely adopted due to its contactless and non-invasive process. Recently, it has also become popular as a commercial identification and marketing tool. Other applications include advanced human-computer interaction, video surveillance, automatic indexing of images, and video database, among others. And over the last ten years or so, face recognition has become a popular area of research in computer vision and one of the most successful applications of image analysis and understanding (7)**.**

Face Recognition systems use computer algorithms to pick out specific, distinctive details about a person's face. These details, such as distance between the eyes or shape of the chin.

In this paper we will use our method that detects faces based on a group of identifiers which were carefully examined and were explained in more details in [20].

There are many algorithms for face recognition, we concentrated on two algorithms –OpenCV and LBPH- and we will modify the second one to get more accurate results.

2.1 Face recognition using Eigenfaces:

OpenCV(Open Source Computer Vision Library), which is an image and video processing library with bindings in C++, C, Python, and Java. OpenCV is used for all sorts of image and video analysis, like facial recognition and detection, license plate reading, photo editing, advanced robotic vision, optical character recognition, and a whole lot more (9)**.**

OpenCV has three built-in face recognizers and thanks to its clean coding, we can use any of them just by changing a single line of code. Here are the names of those face recognizers and their OpenCV calls:

- EigenFaces—cv2.face.createEigenFaceRecognizer()
- FisherFaces—cv2.face.createFisherFaceRecognizer()
- Local Binary Patterns Histograms (LBPH) cv2.face.createLBPHFaceRecognizer() (9)

Eigenfaces is the name given to a group of eigenvectors when they are used in the computer vision problem of human face recognition.Sirovich and Kirby developed the approach of using eigenfaces for recognition in (1987) and this system used by Matthew Turk and Alex Pentland in face classification. The eigenvectors are derived from the covariance matrix of the probability distribution over the high-dimensional vector space of face images. The eigenfaces themselves form a basis set of all images used to construct the covariance matrix. This produces dimension reduction by allowing the smaller set of basis images to represent the original training images. We could achieve a good Classification by comparing how faces are represented by the basis set. (10)

The problem with the image representation we are given is its high dimensionality. For Example Two-dimensional $p \times q$ gray scale images span a $m = pq$-dimensional vector space, so an image with 100×100 pixels lies in a $10,000$-dimensional image space already. The question is: Are all dimensions equally useful for us? We can only make a decision if there's any variance in data, so what we are looking for are the components that account for most of the information. The Principal Component Analysis (PCA) was independently proposed by Karl Pearson (1901) and Harold Hotelling (1933) to turn a set of possibly correlated variables into a smaller set of uncorrelated variables. The idea is, that a high-dimensional dataset is often described by correlated variables and therefore only a few meaningful dimensions account for most of the information. The PCA method finds the directions with the greatest variance in the data, called principal components. (11)

Algorithm Description

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a random vector with observations $x_i \in \mathbb{R}^d$.

- Compute the mean μ

$$\mu = \frac{1}{n} \sum_{i=1}^{n} x_i$$

- Compute the Covariance Matrix *S*

$$S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T,$$

- Compute the eigenvalues λ_i and eigenvectors v_i of S

$$Sv_i = \lambda_i v_i, i = 1, 2, \ldots, n$$

- Order the eigenvectors descending by their eigenvalue. The k principal components are the eigenvectors corresponding to the k largest eigenvalues.

The k principal components of the observed vector x are then given by:

$$y = W^T(x - \mu)$$

where $W = (v_1, v_2, \ldots, v_k)$.

The reconstruction from the PCA basis is given by:

$$x = Wy + \mu$$

where $W = (v_1, v_2, \ldots, v_k)$.

The Eigenfaces method then performs face recognition by:

- Projecting all training samples into the PCA subspace.
- Projecting the query image into the PCA subspace.
- Finding the nearest neighbor between the projected training images and the projected query image.

But still there's one problem left to solve. Imagine we are given 400 images sized 100×100 pixel. The Principal Component Analysis solves the covariance matrix $S = XX^T$, where $size(X) = 10000 \times 400$ in our example. You would end up with a 10000×10000 matrix, roughly $0.8GB$. Solving this problem isn't feasible, so we'll need to apply a trick. From your linear algebra lessons you know that a $M \times N$ matrix with $M > N$ can only have $N - 1$ non-zero eigenvalues. So it's possible to take the eigenvalue decomposition $S = X^TX$ of size $N \times N$ instead:

$$X^TXv_i = \lambda_i v_i$$

and get the original eigenvectors of $S = XX^T$ with a left multiplication of the data matrix:

$$XX^T(Xv_i) = \lambda_i(Xv_i)$$

The resulting eigenvectors are orthogonal; to get orthonormal eigenvectors they need to be normalized to unit length (12).

Figure 1: EigenFace in OpenCV algorithm (12).

2.2 Face Recognition by LBPH Algorithm:

Local binary patterns (LBP) are a kind of visual descriptor which used for classification in computer vision. The first described for (LBP) in 1994.It has since been found to be a powerful feature for texture classification; it has been determined that when LBP is combined with the Histogram of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets. (13)

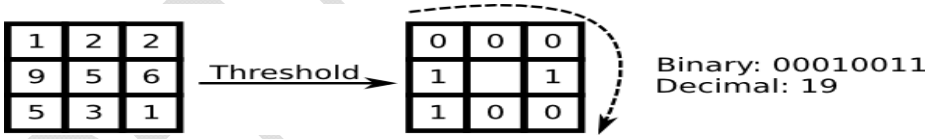
Although the concept of LBPs were introduced as early as 1994, Local Binary Patterns, or LBPs for short, are a texture descriptor made popular by the work of Ojala et al. in their 2002 paper, Multi-resolution Grayscale and Rotation Invariant Texture Classification with Local Binary Patterns. LBPH gives less false positive results even after capturing an image from a height of 4 feet to 7 feet for database of training sets. (14)**.**

LBPs calculate a local representation of texture (LR). This LR is constructed by comparing each pixel with its surrounding neighborhood of pixels. Unlike Haralick texture features which compute a global representation of texture based on the Gray Level Co-occurrence Matrix (15)**.**

To construct the LBP texture descriptor we should first to convert the image to gray scale. Then, we select a neighborhood of size r surrounding the center pixel for each pixel in the gray scale image. Then calculate a LBP value is for this center pixel and stored in the output 2D array with the same width and height as the input image (15)**.**

The idea isn't to look at the whole image as a high-dimensional vector, but describe only local features of an object. The features were extracted by this way have a low-dimensionality implicitly. But we observed that the image representation we are given doesn't only suffer from illumination variations (11)**.**

Take in consideration things like scale, translation or rotation in images -at least our local description has to be a little strong against these things. The Local Binary Patterns methodology has its roots in 2D texture analysis. The main idea of LBP is to summarize the local structure in an image, it do this by comparing each pixel with its neighborhood. Take a pixel as center and threshold its neighbors against. If the intensity of the center pixel is greater-equal its neighbor, write it down with 1 and the value 0 if not.so we'll end up with a binary number for each pixel, just like 11001111. So with 8 surrounding pixels you'll end up with 2^8 possible combinations, called Local Binary Patterns or sometimes referred to as LBP codes. The first LBP operator described in literature actually used a fixed 3 x 3 neighborhood just like this (16)**:**



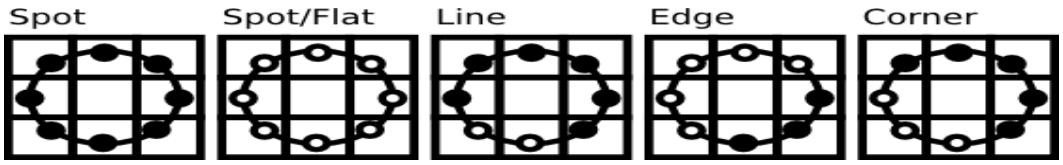
A more formal description of the LBP operator can be given as:

$$LBP(x_c, y_c) = \sum_{p=0}^{P-1} 2^p s(i_p - i_c)$$

, with (x_c, y_c) as central pixel with intensity i_c; and i_n being the intensity of the the neighbor pixel. s is the sign function defined as:

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{else} \end{cases}$$

This description enables you to capture very fine grained details in images. In fact the authors were able to compete with state of the art results for texture classification. Soon after the operator was published it was noted, that a fixed neighborhood fails to encode details differing in scale. So the operator was extended to use a variable neighborhood. The idea is to align an arbitrary number of neighbors on a circle with a variable radius, which enables to capture the following neighborhoods: (17)



For a given Point (x_c, y_c) the position of the neighbor $(x_p, y_p), p \in P$ can be calculated by:

$$x_p = x_c + R\cos\left(\frac{2\pi p}{P}\right)$$

$$y_p = y_c - R\sin\left(\frac{2\pi p}{P}\right)$$

Where R is the radius of the circle and P is the number of sample points (18).

The operator is an extension to the original LBP codes, so it's sometimes called *Extended LBP* (also referred to as *Circular LBP*) . If a point coordinate on the circle doesn't correspond to image coordinates, the point get's interpolated. Computer science has a bunch of clever interpolation schemes; the OpenCV implementation does a bilinear interpolation:

$$f(x,y) \approx \begin{bmatrix} 1-x & x \end{bmatrix} \begin{bmatrix} f(0,0) & f(0,1) \\ f(1,0) & f(1,1) \end{bmatrix} \begin{bmatrix} 1-y \\ y \end{bmatrix}.$$

By definition the LBP operator is robust against monotonic gray scale transformations and can be easily verify this by looking at the LBP image of an artificially modified image (11).

3. PROPOSED METHODOLOGY

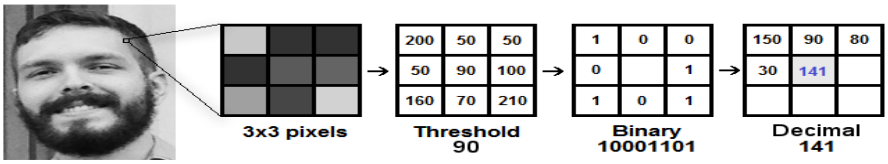
1. **LBPH uses 4 parameters:**
   - **Radius**: the circular local binary pattern was built by the radius and that represents the radius around the central pixel. It is often set to 1.
   - **Neighbors**: they are considered as the number of sample points which used to build the circular local binary pattern. We should keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8
   - **Grid X**: they are considered as the number of cells in the horizontal direction. Note that the more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is often has a value 8.
   - **Grid Y**: the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. Also it is usually set to 8.
2. **Training the Algorithm**.
3. **Applying the LBP operation**: The creation of an intermediate image which describes the original image in a better way is the first computational step of the LBPH, it was done by highlighting the facial characteristics. The algorithm uses a concept of a sliding window, which based on the parameters radius and neighbors. The following image shows this procedure:
4. 

Figure [2]: LBPH Example (12)

Based on the previous image, let's slice it into several small steps to can understand it easily:

- Assume we have a facial image in gray scale.
- We can get one part of this image as a window of 3x3 pixels.
- It can also be represented as a 3x3 matrix which containing the intensity of each pixel (0~255).
- Then, we need to take the central value of the matrix to be used as the threshold.
- We will use this value to define the new values from the 8 neighbors.
- We set a new binary value for each neighbor of the central value (threshold). We set 1 for values equal or higher than the threshold and set 0 for values lower than the threshold.

- So now, the matrix will only contain binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10101001).
- Then, this binary value will be converted to a decimal value and set it to the central value of the matrix, which actually is a pixel from the original image.
- Finally, we have a new image which represents better the characteristics of the original image.

We must note: The LBP procedure was expanded to use a different number of radius and neighbors, it is called Circular LBP (19).

Our modification is that, when we apply LBPH on data processed which is given by our face detection method in [20], this makes the recognition process is much easier and it isolates the background of the images and focus on the face extracted only which helps the recognizer to work better than before.

The Proposed method Algorithm Steps:

- Using webcam/video stream to get image frames.
- Detect Faces of the image frames then draw a square around it.
- Ask user to input names for each face and create a dataset of faces and its names.
- Apply LBP Algorithm.
- Divide the examined window into cells (e.g. 16x16 pixels for each cell).
- For each pixel in a cell, compare the pixel to each of its 8 neighbors.
- If the center pixel's value is greater than the neighbor's value, write "0" Else, write "1".
- Compute the histogram, over the cell, of the frequency of each "number" occurring (i.e., each combination of which pixels are smaller and which are greater than the center). This histogram can be seen as a 256-dimensional feature vector.
- Optionally normalize the histogram.
- Concatenate (normalized) histograms of all cells. This gives a feature vector for the entire window.

Figure 3 shows a flowchart of Creating Dataset Steps, Figure 4 shows a Flow Chart for face Detection System which is represented in our paper (20), *and* Figure 5 shows a Face recognition System Flow Chart.

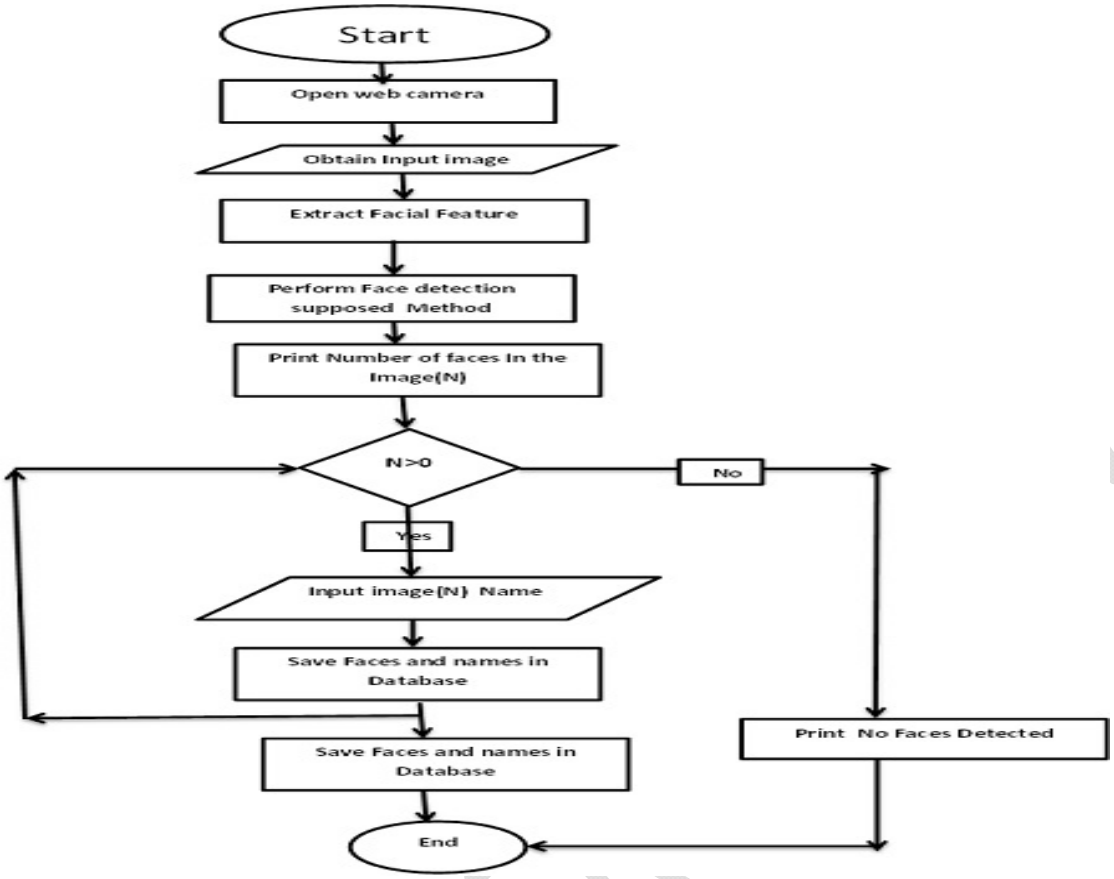


Figure 3: Creating Dataset Steps Flowchart.

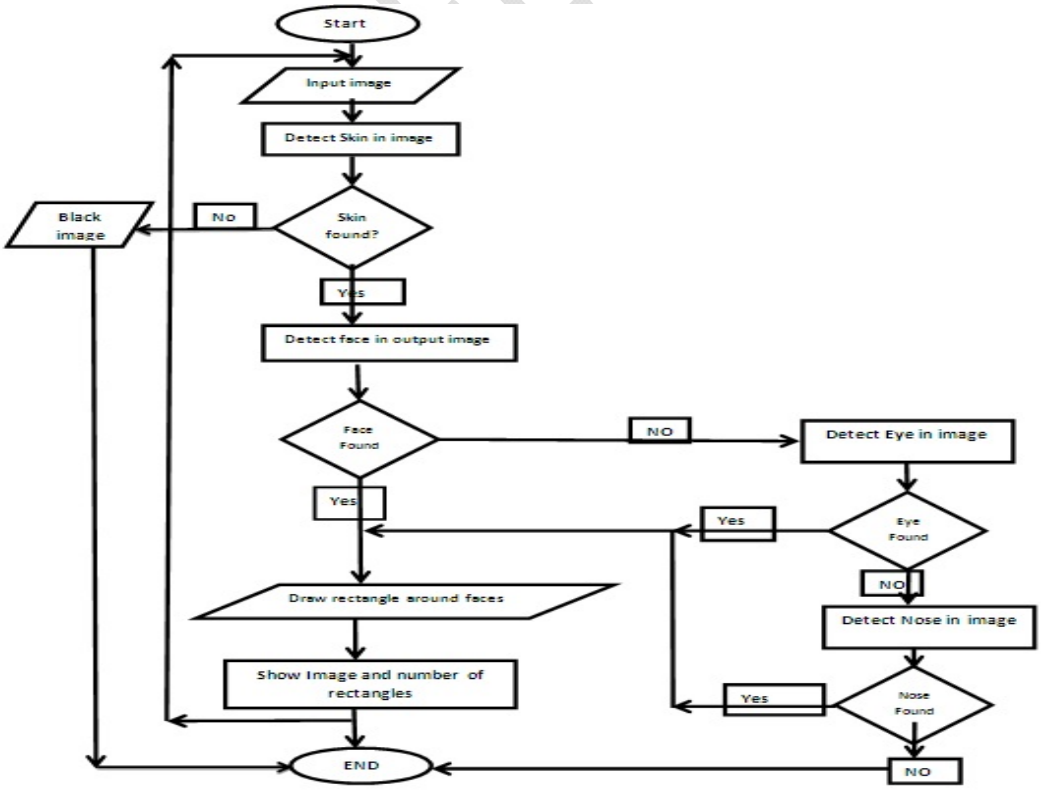


Figure 4: Flow Chart for face Detection System (20)

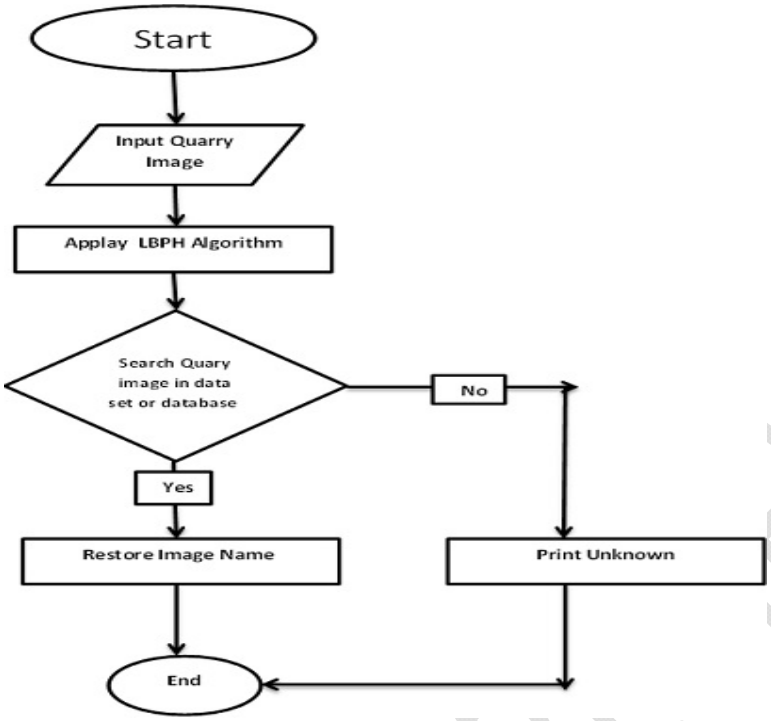


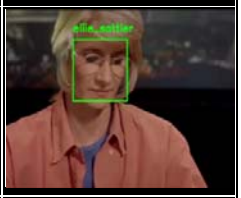
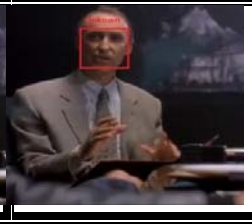
Figure 5: Face recognition System Flow Chart.

At the beginning we should open the camera to get an input image, the first step is to get an input image and apply the face detection process to improve the performance, then we should extract facial feature of the face, give it a name and save it to create dataset to make it as a reference Later. Second part of the system is to test an input image to know if it belongs to our dataset or not then apply local binary pattern histogram face recognition algorithm LBPH. Finally, we get the output name if this image was found in the dataset. If the image was not found in the dataset, the system will give the image a default name like 'Unknown'.

4. EXPERIMENTAL RESULTS

Using 25 sample images, captured from video and webcams, have 57 faces appearance in different illuminations and backgrounds. Table 1 shows sample results of the implemented face recognition techniques. The first column shows the image number, the next column shows the original images, and the next 3 columns shows the Face Recognition with EigenFace, Face Recognition with LBPH only, and Face Recognition with LBPH with the proposed face detection method, respectively.

Table 1: Results of the implemented face recognition techniques.

#	Original image	Face Recognition with EigenFace	Face Recognition with LBPH only	Face Recognition with LBPH + proposed face detection method
1				
2				


3
4
5
6
7
8
9
10
john_hammond
ian_malcolm
alan_grant
owen_grady
Unknown
ellie_sattler

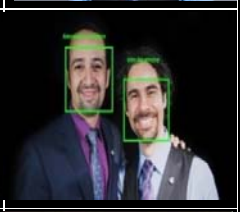
11				
12				
13				
14				
15				
16				
17				
18				
19				

Table 2 shows the results of the true positive and true negative of each face recognition technique.

Table 2 : The results of the true positive and true negative of each face recognition technique.

#	No.of.Faces	True Positive			True Negative		
		EigenFace	LBPH	LBPH+Proposed	EigenFace	LBPH	LBPH+Proposed
1	1	1	1	1	0	0	0
2	1	1	1	1	0	0	0
3	1	1	1	1	0	0	0
4	2	1	1	2	1	1	0
5	4	2	3	4	2	1	0
6	2	2	2	2	0	0	0
7	2	1	1	1	1	1	1
8	2	2	2	2	0	0	0

9	3	1	2	3	2	1	0
10	2	1	2	2	1	0	0
11	2	2	2	2	0	0	0
12	4	3	4	4	1	0	0
13	6	4	5	5	2	1	1
14	2	1	1	2	1	1	0
15	1	1	1	1	0	0	0
16	2	2	2	2	0	0	0
17	1	1	1	1	0	0	0
18	1	0	1	1	1	0	0
19	3	2	2	3	1	1	0
20	2	1	2	2	1	0	0
21	4	2	3	4	2	1	0
22	3	2	2	3	1	1	0
23	3	3	3	3	0	0	0
24	1	1	1	1	0	0	0
25	2	2	2	2	0	0	0
Total	57	40	48	55	17	9	2

experimental results for rate and error rate.

Table 3 summarizes the the recognition success

experimental results for

Table 3 : The summary.

	EigenFace	LBPH only	Proposed
Recognition Rate	70%	84.2%	96.5%
Average Error Rate	29.8 %	%15	3.5%

As shown in Table 2 and 3 the modified LBPH method improved the success rate to 96.5 % with error rate 3.5%, while using EigenFace only has a high error rate 29% and only 70 % recognition rate. Also using LBPH has approximate 84% of recognition rate and 15 % error rate. We can say that, the success rate of the proposed recognition system was better than the EigenFaces and LBPH only techniques.

5. CONCLUSION

Authentication is an important issue in system control in computer based communication. Human face recognition is a significant branch of biometric verification and has been used widely in many applications, such as video monitor system, human-computer interaction, and door control system and network securit. In this paper we are concerned with faces on a video stream so we built a modified face recognition technique based on Local Binary Pattern histogram with processed data (faces detected before by using a combination of Haar cascade files that uses skin detection, eye detection and nose detection) as input of LBP to increase the accuracy of the proposed recognition system. The modified LBPH method improved the success rate to 96.5 % with error rate 3.5%, while using EigenFace only has a high error rate 29% and only 70 % recognition rate. Also using LBPH has approximate 84% of recognition rate and 15 % error rate. The success rate of the proposed recognition system was better than the EigenFaces and LBPH only techniques.

6. DISCUSSION AND FUTURE WORK

There is many possible data could be processed to recognize faces like face expression like happiness sadness this enable us to analysis the character of the person and will help to get more information about the person which was recognized .It will be helpful in many fields Like Psychiatry.

REFERENCES

1. *Study of implementing automated attendance system using face recognition technique.* **Kar, N., Debbarma, M.K., Saha, A. and Pal, D.R.** 2012, International Journal of computer and communication engineering, p. p.100.

2. wikipedia.org. [Online] 2015. https://en.wikipedia.org/wiki/Motion_detection.

3. *Motion detection in image sequences acquired from a moving platform.* **Zheng, Q. and Chellappa, R.** 5, 1993, IEEE International Conference on Acoustics, Speech, and Signal Processing , pp. 201-204.

4. *Computation and analysis of image motion: A synopsis of current problems and methods.* **Mitiche, A. and Bouthemy, P.** 1, 1996, International journal of computer vision, Vol. 19, pp. .29-55.

5. *Face recognition: A literature survey.* **Zhao, W., Chellappa, R., Phillips, P.J. and Rosenfeld, A.** 35, 2003, ACM computing surveys (CSUR), Vol. 4, pp. 399-458.

6. *Face detection: A survey. Computer vision and image understanding,.* **Hjelmås, E. and Low, B.K.** 3, 2001, Vol. 83, pp. .236-274.

7. *A literature survey on face recognition techniques.* **Patel, R. and Yagnik, S.B.** 5, 2013, International Journal of Computer Trends and Technology (IJCTT), Vol. 4, pp. 189-194.

8. *Learning OpenCV: Computer vision with the OpenCV library.* **Bradski, G. and Kaehler, A.** s.l. : O'Reilly Media, 2008.

9. *Learning OpenCV 3: computer vision in C++ with the OpenCV library.* **Kaehler, A. and Bradski, G.** 2016, O'Reilly Media, Inc.

10. *June. Face recognition using eigenfaces. In Proceedings.* **Turk, M.A. and Pentland, A.P.** 1991, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 586-591.

11. OpenCV 2.4.13.7 documentation . *opencv.org.* [Online] 2019. https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html.

12. *Learning local binary patterns for gender classification on real-world face images.* **Shan, C.** 4, 2012, Pattern recognition letters, Vol. 33, pp. 431-437.

13. *Automated attendance management system based on face recognition algorithms.* **Chintalapati, S. and Raghunadh, M.V.** 2013. In 2013 IEEE International Conference on Computational Intelligence and Computing Research. pp. 1-5.

14. *Conceptual Model for Proficient Automated Attendance System based on Face Recognition and Gender Classification using Haar-Cascade, LBPH Algorithm along with LDA Model.* **Shrivastava, K., Manda, S., Chavan, P.S., Patil, T.B. and Sawant-Patil, S.T.** 2018, International Journal of Applied Engineering Research, Vol. 10, pp. pp.8075-8080.

15. *Face description with local binary patterns: Application to face recognition.* **Ahonen, T., Hadid, A. and Pietikainen, M.** 2006, IEEE Transactions on Pattern Analysis & Machine Intelligence,, Vol. 12, pp. .2037-2041.

16. *A survey on comparison of face recognition algorithms.* **Özdil, A. and Özbilen, M.M.** 2014. 2014 IEEE 8th International Conference on Application of Information and Communication Technologies (AICT).

17. *Image Training and LBPH Based Algorithm for Face Tracking in Different Background Video Sequence.* **Ranganatha, S. and Gowramma, Y.P.** 6, 2018, International Journal of Computer Sciences and Engineering, Vol. 9, pp. 349-354.

18. *A real-time face recognition system based on the improved LBPH algorithm.* **Zhao, X. and Wei, C.** 2017. 2017 IEEE 2nd International Conference on Signal and Image Processing (ICSIP). pp. 72-76.

19. face recognition LBPH. *towardsdatascience.* [Online] 2017. https://towardsdatascience.com.

20. *A Robust and Efficient System to Detect Human Faces Based on Facial Features.* **Ali, A.A., El-Hafeez, T.A. and Mohany, Y.K.** 4, s.l. : Asian Journal of Research in Computer Science, 2018, Vol. 2, pp. 1-12.