FACE RECOGNITION USING OPEN MP

CSE4001 – PARALLEL AND DISTRIBUTED COMPUTING

FALL SEMESTER PROJECT 2019-20

PROJECT BASED COMPONENT REPORT

by

17BCE0744- PARICHAY DIDWANIA

17BCE0900- RAJAT RATHI

17BCE2033- AMREETA KONER

17BCE0319- PUSPA LAMICHHANE

School of Computer Science and Engineering

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

October 2019

DECLARATION

We hereby declare that the report entitled “Face Recognition using Open MP” submitted by me, for the CSE4001 Parallel and Distributed Computing (EPJ) to VIT is a record of bona fide work carried out by me under the supervision of Dr.Narayanan Prasanth. We further declare that the work reported in this report has not been submitted and will not be submitted, either in part or in full, for any other courses in this institute or any other institute or university.

Place : Vellore

Date :

Signature of the Candidate

ABSTRACT

Facial analysis has been an important research field due its wide range of applications like: law enforcement, surveillance, entertainment like video games and virtual reality, information security, banking, human computer interface, etc. The original interest in facial analysis relied on face recognition, but later on the interest in the field was extended and research efforts where focused in the appearance of model-based image, video coding, face tracking, pose estimation, facial expression, emotion analysis and video indexing

ACKNOWLEDGEMENT

A successful and satisfactory completion of any significant task is the outcome of invaluable aggregate combination of different people in radial direction explicitly and implicitly. We would therefore take the opportunity to thank and express our gratitude to our faculty Dr. Narayanan Prasanth without whom the completion of our project would not be possible.

PARICHAY DIDWANIA
RAJAT RATHI
AMREETA KONER
PUSPA LAMICHHANE

CONTENTS Page No.

1) INTRODUCTION

1.1. OBJECTIVE

A **facial recognition system** is a technology capable of identifying or verifying 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, they work by comparing selected facial features from given image with faces within a database. It is also described as a Biometric Artificial Intelligence based application that can uniquely identify a person by analysing patterns based on the person's facial textures and shape.

Facial recognition is a category of biometric software that maps an individual's facial features mathematically and stores the data as a faceprint. The software uses deep learning algorithms to compare a live capture or digital image to the stored faceprint in order to verify an individual's identity

Facial recognition is a biometric software application capable of uniquely identifying or verifying a person by comparing and analyzing patterns based on the person's facial contours. Facial recognition is mostly used for security purposes, though there is increasing interest in other areas of use. In fact, facial recognition technology has received significant attention as it has potential for a wide range of application related to law enforcement as well as other enterprises.

There are different facial recognition techniques in use, such as the generalized matching face detection method and the adaptive regional blend matching method. Most facial recognition systems function based on the different nodal points on a human face. The values measured against the variable associated with points of a person’s face help in uniquely identifying or verifying the person. With this technique, applications can use data captured from faces and can accurately and quickly identify target individuals. Facial recognition techniques are quickly evolving with new approaches such as 3-D modeling, helping to overcome issues with existing techniques.

1.2 MOTIVATION

There are many advantages associated with facial recognition. Compared to other biometric techniques, facial recognition is of a non-contact nature. Face images can be captured from a distance and can be analyzed without ever requiring any interaction with the user/person. As a result, no user can successfully imitate another person. Facial recognition can serve as an excellent security measure for time tracking and attendance. Facial recognition is also cheap technology as there is less processing involved, like in other biometric techniques

We want to compare the parallel and serial implementation of the face detection algorithm using OpenMP and compare the time taken for face detection

.

2) LITERATURE SURVEY

In [1] paper, a parallel face recognition algorithm based on eigenfaces, which are known as the significant features because they are the eigenvectors (principal components) of a set of face images. Our work is composed of two phases: eigenface construction and face recognition. For the construction of eigenfaces, a solution eigenvalue/eigenvector problem using parallel Jacobi transformation is suggested. Experimental results reveal that time taken for both eigenface construction and recognition is sharply decreased as the number of PE's are increased

In [2] paper, a robust approach for real time face recognition where the images come from live video is proposed. To improve the algorithmic efficiency of face detection, we combine the eigenface method using Haar-like features to detect both of eyes and face, and Robert cross edge detector to locate the human face position. Robert Cross uses the integral image representation and simple rectangular features to eliminate the need of expensive calculation of multi-scale image pyramid.

In [3] paper comparison of GPU performance to multithreaded CPU performance by implementing a neural network face recognition scenario. From the results we derived the following findings: for simulation (training and evaluation) of BPNN algorithm GPU based parallelization should be preferred generally to CPU based multithreaded program. However, for BPNN simulations with small input and few hidden neurons CPU based execution is better. But, if the input size and thus the number of input neurons increases GPU-based parallelization is suitable to reduce training and evaluation

In [4] paper parallel face detection system for shared-memory multiprocessors using the OpenMP programming model. OpenMP provides a fast and portable means of parallelizing the sequential code and allows for efficient exploitation of the computational resources. The experimental results demonstrate significant improvement in the processing time of a single image. Our efficient runtime support, based on the OMP OpenMP compiler with lightweight multithreading, provides efficient exploitation of the nested loop-level parallelism and better load balancing in the face detection algorithm. The experiments on a quad-core system show that the speedup of the second step of the algorithm is improved by up to 20%, when multiple levels of parallelism are exploited.

In [5] With the aid of a regular web camera, a machine is able to detect and recognize a person's face; a custom login screen with the ability to filter user access based on the users' facial features will be developed. The objectives of this thesis are to provide a set of detection algorithms that can be later packaged in an easily portable framework amongst the different processor architectures we see in machines (computers) today. These algorithms must provide at least a 95% successful recognition rate, out of which less than 3% of the detected faces are false positives

In [9] Ratnawati Ibrahim et al (2011) have proposed an automated face recognition system is done in typical face recognition's two stages which are training stage and evaluation stage. In the first stage, the specific number of training image of face candidate is captured. The features are extracted from the intensity image of human frontal faces using principle component analysis. The system will then learn on the extracted features and store them in its database. In the second stage, the system will recognize new faces in an unsupervised manner and that is easy to implement using artificial neural network. The scope of improvement from the paper is found to be the lighting settings, the background in front of the camera.

In [8] Mozammel Chowdhury et alA robust stereo algorithm based on neural network is used for matching correspondences between the detected face regions. The human can be localized with the matching outcomes. The low computation cost required for the detection and localization process makes it suitable to be employed in real time applications. They propose a robust approach for detecting humans and their locations from the video scenes in a secure access control system. The first step of the approach is the face detection to identify that the object is a human, while the next step is the localization process that analyses the intention of the human towards the entrance. The general architecture of our proposed approach. The proposed method consists of the following components: (i) Stereo scene capturing, (ii) Human detection by face detection, (iii) Face extraction and registration, (iv) Correspondence matching between the detected face regions, (v) Estimation of depth information, and (v) Human localization.

In [8] Mrutyunjaya Sahani et al proposed system is a wireless access control system designed and developed for smart home environment. The paper proposes a Raspberry pi-based door access control and home security system through webpage with ZigBee based technology. The system identifies the visitor's presence, capture and transfers the image through email and/or an alert SMS via GSM network automatically to home owner to recognize the visitors. The system capability to provide access through internet, where subject of received email is read by the developed algorithm fed into Raspberry pi and system responds to the corresponding instruction with high security. The user can directly login and interact with the embedded device in real time without the need to maintain an additional server. It has a variety of features such as energy efficient, intelligence, low cost, portability and high performance

In [6] M. Carikci et al. proposed a work on A Face Recognition System based on Eigen face method in which they used Eigen method for face recognition and Euclidean distance method to compare the image of the person concerned with the images in the database.
It was very efficient and fast method and also gave high accuracy.

In [7] S. Jogdand.et.al proposed a work on Implementation of Automated Door Accessing System with Face Design and Recognition in which they used Viola Jones method for face detection and PCA (Principal Component Analysis) for the comparison of images. The limitation of this work was that it is not robust and the efficiency is less.

TECHNICAL SPECIFICATIONS

Tools used:

OpenMP

OpenMP (Open Multi-Processing) is an application programming interface (API) that supports multi-platform shared memory multiprocessing programming in C, C++, and
Fortran, on most platforms, processor architectures and operating systems, including Solaris, Linux, OS X, and Windows. It consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior We use ***pragma omp's*** as following:
Firstly, an optimization was needed for reading the files as that function is called 360 times, thus a parallelizing statement was put before the two nested loops inside which image read and enhanced image matrix was being created.

Another important issue within those loops is creating the histograms which is as well called 360 times. Being that together in the serial implementation in the parallelised version an obvious improvement will be obtained. Furthermore, another #pragma omp for is put before the 2 nested for loops inside which the **find_closest** function is called which as well calls distance function

Furthermore, a pragma is put for the four loop which initializes the 3D array where the histograms will be stored speeds up the initialization of the array elements with 0. Another important part is reading from files and creating the histograms; thus, a pragma is inserted there to speed up the process of creating the histograms. Although allocation and deallocation of the
3D matrix do not require a significant amount of time, we have also parallelized that task

DESIGN/METHODOLOGY

1) Training Step:

For each image in the dataset, the following steps are applied:

i) For each pixel in an image, pixel is compared with its 8 neighbors (on its lefttop, left middle, left-bottom, right-top, etc.).

ii) Where the center pixel's value is less than the neighbor's value, "1". Otherwise, “0".

iii) Value is converted into a decimal value.

iv) Histogram is computed, over the pixels, of the frequency of each "number" occurring (i.e. each combination of which pixels are smaller and which are greater than the center).

2) Test Step:

i) Histogram for the test image are generated as described previously.

ii) Distance values between test image's histogram and training histograms are computed.

iii) Closest training histogram to determine the person is selected.

iv) Results are displayed

PROPOSED SYSTEM

Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number.

It was first described in 1994 (LBP) and has since been found to be a powerful feature for texture classification. It has further been determined that when LBP is combined with histograms of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets. Using the LBP combined with histograms we can represent the face images with a simple data vector.

As LBP is a visual descriptor it can also be used for face recognition tasks, as can be seen in the following step-by-step explanation.

Step-by-Step process:

Now that we know a little more about face recognition and the LBPH, let's go further and see the steps of the algorithm:

1. ***Parameters***: the LBPH uses 4 parameters:

Radius: the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.

- Neighbours**:** the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.
- Grid X: the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 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. It is usually set to 8.

2. Training the Algorithm: First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let’s see the LBPH computational steps.

2. Applying the LBP operation: The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameter’s radius and neighbours**.**

let’s break it into several small steps so we can understand it easily:

- Suppose we have a facial image in grayscale.
- We can get part of this image as a window of 3x3 pixels.
- It can also be represented as a 3x3 matrix 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.
- This value will be used to define the new values from the 8 neighbours.
- For each neighbour of the central value (threshold), we set a new binary value. We set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.
- Now, the matrix will contain only 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. 10001101). Note: some authors use other approaches to concatenate the binary values (e.g. clockwise direction), but the final result will be the same.
- Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original image.
- At the end of this procedure (LBP procedure), we have a new image which represents better the characteristics of the original image.

4. ***Performing the face recognition***: In this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, we perform the steps again for this new image and creates a histogram which represents the image.

- So, to find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram.

- We can use various approaches to compare the histograms (calculate the distance between two histograms), for example**:** Euclidean distance**,** chi-square**,** absolute value, etc.

FLOWCHART DIAGRAM

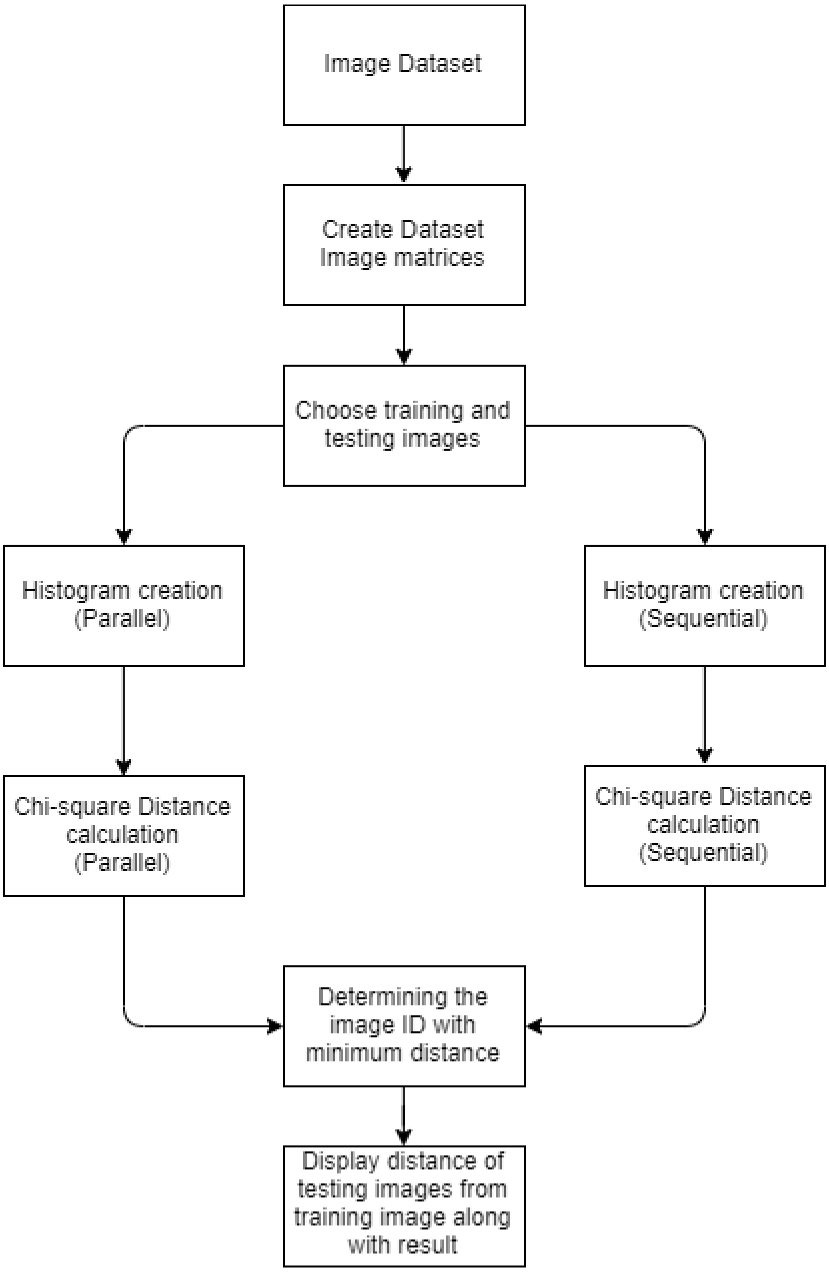
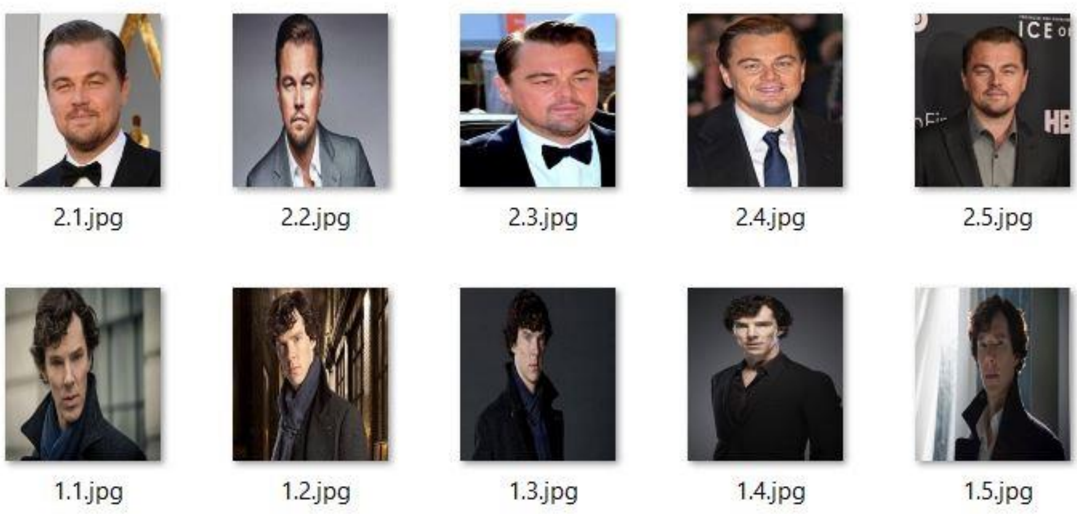


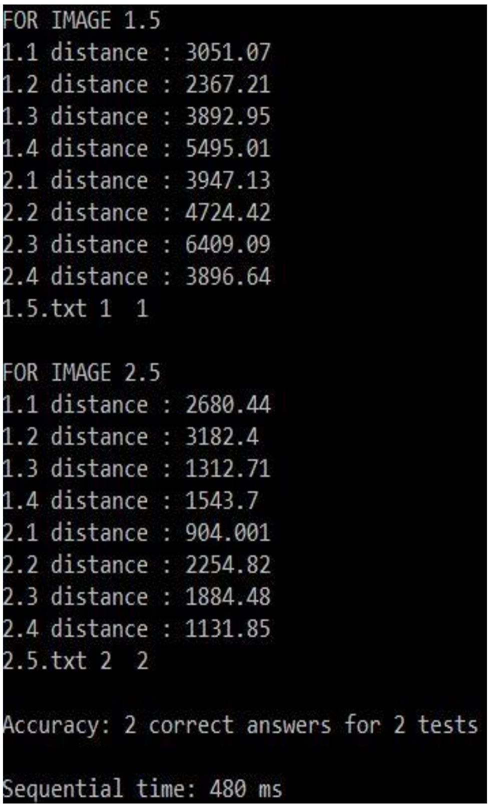
Image Dataset
Create Dataset Image matrices
Choose training and testing images
Histogram creation (Parallel)
Histogram creation (Sequential)
Chi-square Distance calculation (Parallel)
Chi-square Distance calculation (Sequential)
Determining the image ID with minimum distance
Display distance of testing images from training image along with result

RESULT AND CONCLUSION

Image Directory:



Sequential Result:



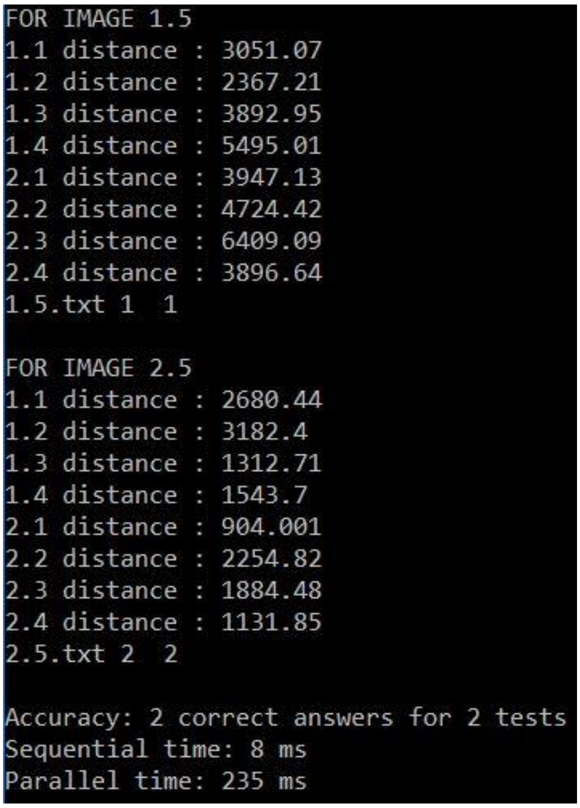
```
FOR IMAGE 1.5
1.1 distance : 3051.07
1.2 distance : 2367.21
1.3 distance : 3892.95
1.4 distance : 5495.01
2.1 distance : 3947.13
2.2 distance : 4724.42
2.3 distance : 6409.09
2.4 distance : 3896.64
1.5.txt 1  1

FOR IMAGE 2.5
1.1 distance : 2680.44
1.2 distance : 3182.4
1.3 distance : 1312.71
1.4 distance : 1543.7
2.1 distance : 904.001
2.2 distance : 2254.82
2.3 distance : 1884.48
2.4 distance : 1131.85
2.5.txt 2  2

Accuracy: 2 correct answers for 2 tests

Sequential time: 480 ms
```

Parallel Result:



```
FOR IMAGE 1.5
1.1 distance : 3051.07
1.2 distance : 2367.21
1.3 distance : 3892.95
1.4 distance : 5495.01
2.1 distance : 3947.13
2.2 distance : 4724.42
2.3 distance : 6409.09
2.4 distance : 3896.64
1.5.txt 1  1

FOR IMAGE 2.5
1.1 distance : 2680.44
1.2 distance : 3182.4
1.3 distance : 1312.71
1.4 distance : 1543.7
2.1 distance : 904.001
2.2 distance : 2254.82
2.3 distance : 1884.48
2.4 distance : 1131.85
2.5.txt 2  2

Accuracy: 2 correct answers for 2 tests
Sequential time: 8 ms
Parallel time: 235 ms
```

From the result we can conclude that the program is successfully able to perform face recognition and the accuracy increases as the number of training images increases. Here we can notice the execution time of serial compared to parallel is almost double. This difference increases as the number of images increases leading to an exponential improvement in execution time.

All in all, it can be said that this project is a good example of task parallelizing with OpenMP Dealing with multiple data to read, compute and compare results in a high amount of time if the tasks are to be run sequentially. The serial version in this case requires about twice the time of the parallel version because in the latter, inserting pragmas for parallelizing the most timeconsuming tasks as reading from files and creating the histogram for each of 360 images. Regarding the time elapsed, the serial implementation takes about 10-11 seconds and the parallel one approximately half of the time. The improvement was obtained by looking at results for the serial part and optimizing it such that different nr of threads take almost equal parts of the tasks and execute them in parallel, thus resulting in a better performance in terms of time.

7) REFERENCES

[1] Yong-Seok Cheon and Chang-Sung Jeong ,Parallel Processing for Automatic Face Recognition using Eigenfaces published by SiteCeer in 2007.

[2] Varun Pande, Khaled Elleithy, Laiali Almazaydeh, Parallel Processing for Multi Face Detection and Recognition Conference: 5th International Conference on Computers and Their Applications in Industry and Engineering (CAINE-2012), At New Orleans, Louisiana, USA.

[3] Altaf Ahmad Huqqania, ∗, Erich Schikutaa , Sicen Yea , Peng Chena, Multicore and GPU Parallelization of Neural Networks for Face Recognition International Conference on Computational Science, ICCS 2013

[4] P. E. Hadjidoukas, V. V. Dimakopoulos, M. Delakis and C. Garcia,A high-performance face detection system using OpenMP in Concurrency and Computation Practice and Experience 21(15):1819-1837 · October 2009

[5] Shervin EMAMI , Valentin Petruț SUCIU, Facial Recognition using OpenCV by Research gate March 2012

[6] Çarıkçı, M., Özen, F. 2012 A Face Recognition System Based on Eigen faces Method by Science Direct in Volume 1, 2012, Pages 118-123

[7] Jogdand, S., Karanjkar, M. 2015Implementation of Automated Door Accessing System with Face Design and Recognition International Journal of Science and Research (IJSR) ISSN (Online): 2319-7064 Index Copernicus Value (2013): 6.14

[8] Chowdhury, M., Gao, J., & Islam, R. (2016, June). Human detection and localization in secure access control by analysing facial features. In 2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA) (pp. 1311-1316). IEEE.

[9] Ibrahim, R., & Zin, Z. M. (2011, May). Study of automated face recognition system for office door access control application. In 2011 IEEE 3rd International Conference on Communication Software and Networks (pp. 132-136). IEEE