IJERT-A Secured Face Recognition System using Principle Component Analysis and Hybrid Cryptography

IJERT Journal

International Journal of Engineering Research and Technology (IJERT)

Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 02, February-2021

A Secured Face Recognition System using Principle Component Analysis and Hybrid Cryptography

J. Vikram [1]
[1] Researcher,
Department of Computer Science,
Chikkana Government Arts College

Dr. M. Gobi [2]
[2] Assistant Professor,
Department of Computer Science,
Chikkana Government Arts College

***Abstract*:- Face Recognition is used for real time application. So reliability is the more important matter for security. Facial Recognition is rapidly becoming area of interest. Face biometrics is useful for authentication that recognizes face. This paper represents secured face recognition methods and discusses their advantages. The purpose of this paper is to provide secured face recognition methods. Principal component analysis is used for face recognition and cryptography is used to secure the face image.**

INTRODUCTION

A system requires reliability to recognition the identity of an individual entity. The purpose is to ensure that the services are accessed only by a authorized user and not else others. In Biometric, face recognition is based on their physiological and/or behavioral characteristics of individuals. A biometric recognition system consists of four main modules: (i) Image captures of a biometric trait (ii) Feature extraction module that extracts certain features from the biometric data (iii) System database that stores the features extracted from biometric data. (iv) Matcher module that matches the features extracted from the biometric imputed data with the features stored in the system database.

FACE RECOGNITION METHODS

Face recognition methods divided into categories (i) Knowledge-based methods (i) Feature-invariant methods (ii) Template matching methods (i) Appearance-based methods.

Knowledge-based methods are encoding our knowledge of human faces. These are rule-based methods. They try to capture our knowledge of faces, and translate them into a set of rules. It's easy to guess some simple rules. For example, a face usually has two symmetric eyes, and the eye area is darker than the cheeks. Facial features could be the distance between eyes or the color intensity difference between the eye area and the lower zone. The big problem with these methods is the difficulty in building an appropriate set of rules. There could be many false positives if the rules were too general. On the other hand, there could be many false negatives if the rules were too detailed. A solution is to build hierarchical knowledge-based methods to overcome these problems. These methods show themselves efficient with simple inputs. But, what happens if a man is wearing glasses? There are other features that can deal with that problem. For example, there are algorithms that detect face-like textures or the color of human skin.

Feature-invariant methods that try to find invariant features of a face despite its angle or position. Facial recognition utilizes distinctive features of the face – including: distinct micro elements like: Mouth, Nose, Eye, Cheekbones, Chin, Lips, Forehead, Ears, Upper outlines of the eye sockets, the areas surrounding the cheekbones, the sides of the mouth, and the location of the nose and eyes. The distance between the eyes, the length of the nose and the angle of the jaw.

These algorithms compare input images with stored patterns of faces or features. Template matching methods try to define a face as a function. One can try to find a standard template of all the faces. Different features can be defined independently. For example, a face can be divided into eyes, face contour, nose and mouth. Also a face model can be built by edges. But these methods are limited to faces that are frontal. A face can also be represented as a shape. Other templates use the relation between face regions in terms of brightness and darkness. These standard patterns are compared to the input images to detect faces. This approach is simple to implement, but it‟s insufficient for face detection. It cannot achieve good results with variations in pose, scale and shape. However, deformable templates have been proposed to deal with these problems.

A template matching method whose pattern database is learnt from a set of training images. In general, appearance-based methods rely on techniques from statistical analysis and machine learning to find the relevant characteristics of face images.

PCA Algorithm Principal Component Analysis (PCA) is well-organized method for face recognition[1]. It is one of the most usable methods for a face image. It is used to reduce the dimensionality of the image and also holds some of the variations in the image data[2]. It is projecting face image data into a feature space that covers the significant variations among known facial images. Those significant features are known as "Eigen faces", because they are the eigenvectors or Principal Component of the set of faces. That is not necessary to correspond to the features such as eyes, ears, and noses. The projection operation characterizes an individual face by a weighted sum of the Eigen faces features. So to recognize a particular face, it is necessary only to compare these weights to those individuals. The Eigen Object Recognizer class applies PCA on each image, the results of which will be an array of Eigen values. To perform PCA several steps are undertaken:

PRINCIPAL COMPONENT ANALYSIS (PCA)

Principal Component Analysis (PCA) is a simple yet powerful technique used for dimensionality reduction. Through it, we can directly decrease the number of feature variables, thereby narrowing down the important features and saving on computations. From a high-level view PCA has three main steps:

(1) Compute the covariance matrix of the data.
(2) Compute the eigen values and vectors of this covariance matrix.
(3) Use the eigen values and vectors to select only the most important feature vectors and then transform your data onto those vectors for reduced dimensionality!.

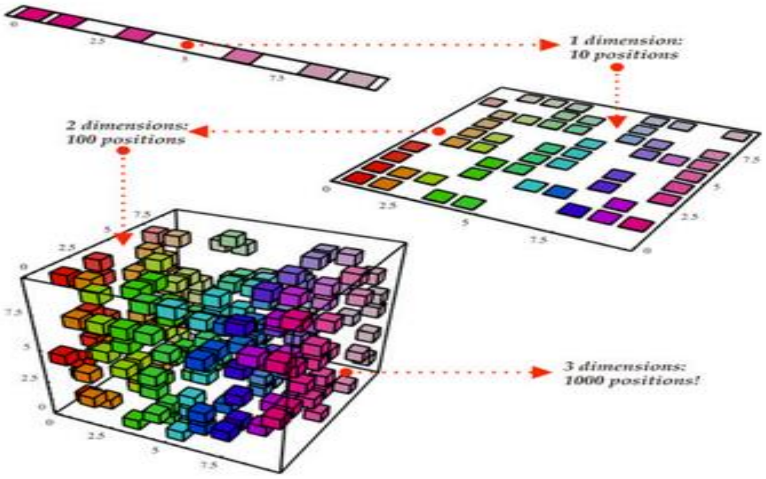


FIG 1. PRINCIPAL COMPONENT ANALYSIS

The entire process is illustrated in the figure above, where the data has been converted from a 3-dimensional space of 1000 points to a 2-dimensional space of 100 points. That's a 10X saving on computation!.

A. Computing the covariance matrix

PCA yields a feature subspace that maximizes the variance along the feature vectors. Therefore, in order to properly measure the variance of those feature vectors, they must be properly balanced. To accomplish this, we first normalise our data to have zero-mean and unit-variance such that each feature will be weighted equally in our calculations. Assuming that our dataset is called *X*: The covariance of two variables measures how "correlated" they are. If the two variables have a positive covariance, then one when variable increases so does the other; with a negative covariance the values of the feature variables will change in opposite directions. The covariance matrix is then just an array where each value specifies the covariance between two feature variables based on the x-y position in the matrix. Where the *x* with the line on top is a vector of mean values for each feature of X. Notice that when we multiply a transposed matrix by the original one we end up multiplying each of the features for each data point together!.

B. Computing Eigen Values and Vectors

The eigen vectors (principal components) of our covariance matrix represent the vector directions of the new feature space and the eigen values represent the magnitudes of those vectors[6]. Since we are looking at our *covariance matrix* the eigen values *quantify* the contributing variance of each vector. If an eigen vector has a corresponding eigen value of high magnitude it means that our data has high variance along that vector in feature space. Thus, this vector holds a lot information about our data, since

any movement along that vector causes large "variance". On the other hand vectors with small eigen values have low variance and thus our data does not vary greatly when moving along that vector. Since nothing changes when moving along that particular feature vector i.e changing the value of that feature vector does not greatly affect our data, then we can say that this feature isn't very important and we can afford to remove it. That's the whole essence of eigen values and vectors within PCA. Find the vectors that are the most important in representing our data and discard the rest. Computing the eigen vectors and values of our covariance matrix is an easy one-liner in numpy. After that, we'll sort the eigen vectors in descending order based on their eigen values.

C. Projection onto new vectors

At this point we have a list of eigen vectors sorted in order of “importance” to our dataset based on their eigen values. Now what we want to do is select the most important feature vectors that we need and discard the rest. We can do this in a clever way by looking at the *explained variance percentage* of the vectors. This percentage quantifies how much information (variance) can be attributed to each of the principal components out of the total 100%. Let’s take an example to illustrate. Say we have a dataset which originally has 10 feature vectors. After computing the covariance matrix, we discover that the eigen values are:
[12, 10, 8, 7, 5, 1, 0.1, 0.03, 0.005, 0.0009]
The total sum of this array is = 43.1359. But the first 6 values represent:

42 / 43.1359 = 99.68% of the total! That means that our first 5 eigen vectors effectively hold 99.68% of the *variance or information* about our dataset. We can thus discard the last 4 feature vectors as they only contain 0.32% of the information, a worthy sacrifice for saving on 40% of the computations!. Therefore, we can simply define a threshold upon which we can decide whether to keep or discard each feature vector. In the code below, we simply count the number of feature vectors we would like to keep based on a selected threshold of 97%. The final step is to actually project our data onto the vectors we decided to keep. We do this by building a *projection matrix*: that’s just a fancy word for a matrix we will multiply by to project our data onto the new vectors. To create it, we simply concatenate all of the eigen vectors we decided to keep. Our final step is to simply take the dot product between our original data and our projection matrix.

SECURED FACE RECOGNITION SYSTEM

The Principal Component Analysis (PCA) is the most successful techniques that have been used in face recognition. PCA is a statistical method under the broad title of factor analysis. The purpose of PCA is to reduce the large dimensionality of the data space observed variables to the smaller intrinsic dimensionality of feature space (independent variables), which are needed to describe the data economically[3]. This is the case when there is a strong correlation between observed variables. When there are different scales used for the measurement of the values of the features, then it is advisable to do the standardization to bring all the feature spaces with mean = 0 and variance = 1.The reason why standardization is very much needed before performing PCA is that PCA is very sensitive to variances. Meaning, if there are large differences between the scales (ranges) of the features, then those with larger scales will dominate over those with the small scales[4]. For example, a feature that ranges between 0 to 100 will dominate over a feature that ranges between 0 to 1 and it will lead to biased results. So, transforming the data to the same scales will prevent this problem[5]. That is where we use standardization to bring the features with mean value 0 and variance 1.

So here is the formula to calculate the standardized value of features:

$$\boldsymbol{Standardized\ value\ of\ x_i = \frac{x_i\ - mean\ of\ x}{std\ Deviation\ of\ x}}$$

To calculate a matrix that summarizes how our variables all relate to one another then break this matrix down into two separate components: direction and magnitude then can understand the “directions” of our data and its “magnitude” (or how “important” each direction is). The “red direction” and the “green direction.” In this case, the “red direction” is the more important one. It’s given how the dots are arranged[6][7].

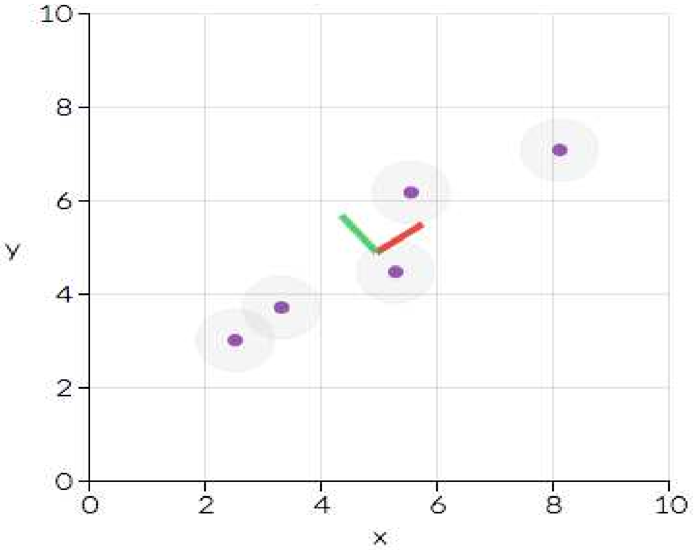


FIG 2.Data in the XY plane

We will transform our original data to align with these important directions, it is transformed so that the *x*- and *y*-axes are now the “red direction” and “green direction.”

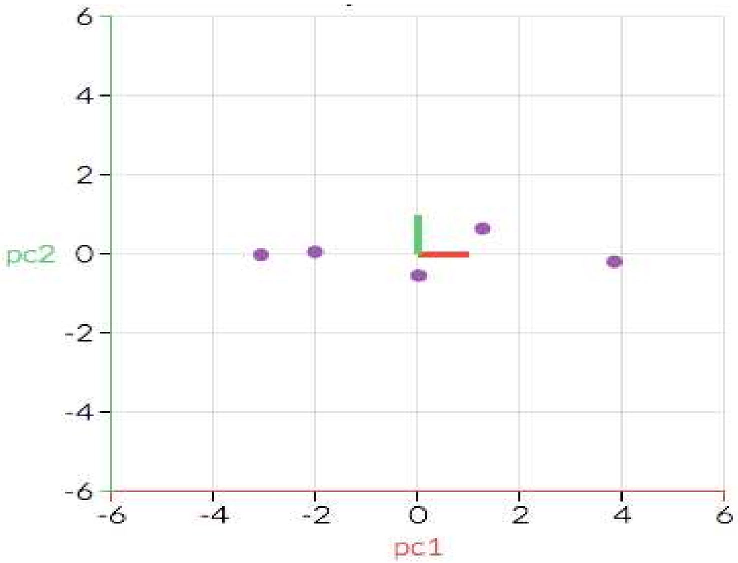


FIG 3.Data transformed by PCA

By identifying which “directions” are most “important,” compress or project the data into a smaller space by dropping the “directions” that are the “least important.” By projecting data into a smaller space, we’re reducing the dimensionality of feature space but because we’ve transformed data in these different “directions,” to made sure to keep all original variables in it.

Every cryptosystem, which has a finite number of keys, can usually be analyzed by *key trial*, deciphering the cipher text with all possible keys until some recognizable text appears[8]. In many “classical” cryptographic systems, the testing of keys could be performed by hand. The stimulus for the development of computers was the need to be able to test large sets of possible keys to decipher coded traffic. Modern cryptosystems are such that the number of possible keys is generally so large as to make exhaustive key trial infeasible[9]. Even computers are limited, and some analysis must precede key testing for the process to be successful. A mixture cryptosystem is a show which uses different figures of different sorts together, each to facilitate it's best potential advantage. One general approach is to make a discretionary puzzle key for a symmetric figure, and a short time later scramble this key by methods for an uneven figure using the recipient's open key. The information itself is then encoded using the symmetric figure and the puzzle key. Both the mixed riddle key and the encoded information are then sent to the recipient. The recipient translates the riddle key first, using his/her own private key, and a while later uses that key to unscramble the data[10].

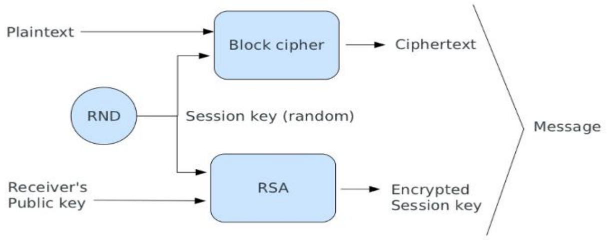


FIG 4. HYBRID – KEY

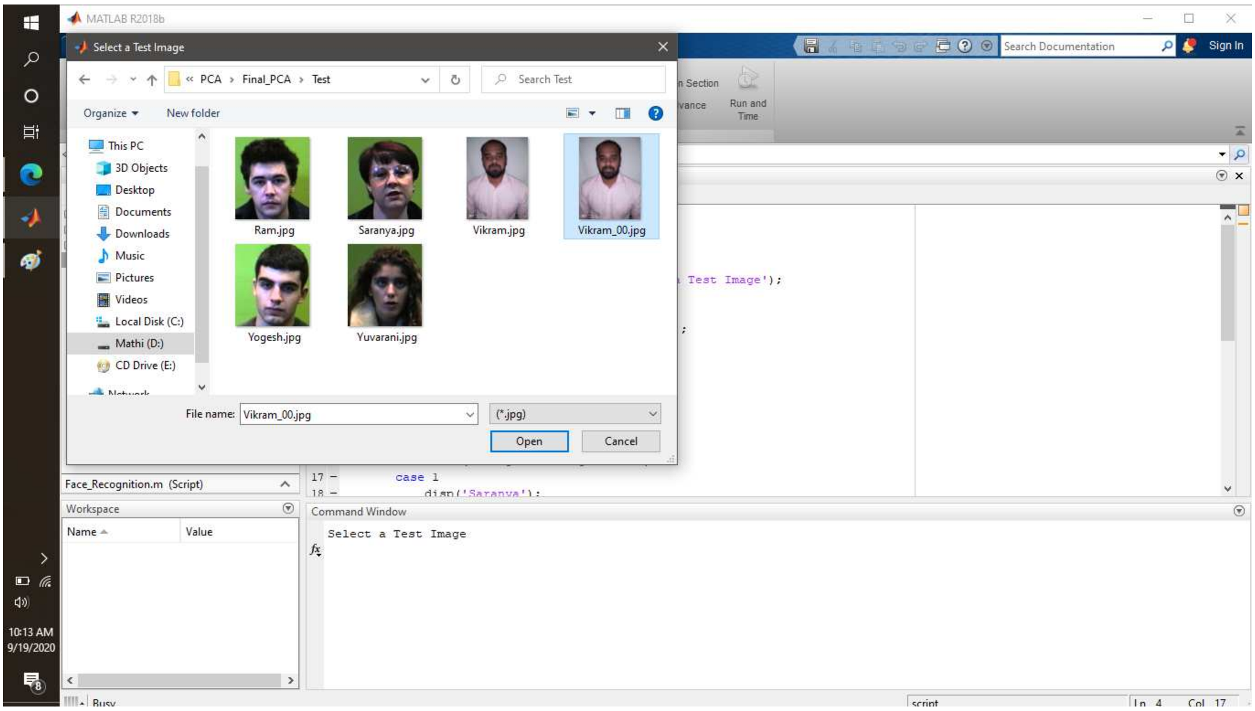
RESULT

FACE RECOGNITION

```
clear all;close all
clc

disp('Select a Test Image');
[file,path] = uigetfile({'*.jpg'},'Select a Test Image');

TestImage = imread(strcat(path,file));
disp('Test Image Selected Successfull ...');
disp('');
index = PCA_analyse(TestImage);
index = floor(index/10);
```

```
switch(index)
    case 0
        disp('Ram');
        imshow(TestImage);
        title('Recognized Image : Ram');
    case 1
        disp('Saranya');
        imshow(TestImage);
        title('Recognized Image : Saranya')

    case 2
        disp('Yogesh');
        imshow(TestImage);
        title('Recognized Image : Yogesh')
    case 3
        disp('Yuvarani');
        imshow(TestImage);
        title('Recognized Image : Yuvarani')
    case 4
        disp('Vikram');
        imshow(TestImage);
        title('Recognized Image : Vikram')
    otherwise
        disp('Unknown');
        imshow(TestImage);
        title('Recognized Image : Unknown')

end

function [rec_Index] = PCA_analyse(testimg)

disp('Select a Trainner Folder');
DataSet_path = uigetdir('Select path of Training images');
disp('Trainner Folder Selected Successfull ...');
disp('');
D = dir(DataSet_path);
imgcount = 0;
for i=1 : size(D,1)
    if not(strcmp(D(i).name,'.')|strcmp(D(i).name,'..')|strcmp(D(i).name,'Thumbs.db'))
        imgcount = imgcount + 1;
    end
end

X = [];
for i = 1 : imgcount


    str = strcat(DataSet_path,'\',int2str(i),'.jpg');
    img = imread(str);
    img = rgb2gray(img);
    img = imresize(img,[200,180],'bilinear');
    [Row,Column] = size(img);
    temp = reshape(img',Row*Column,1);

    X = [X temp];
end

m = mean(X,2);
imgcount = size(X,2);
```

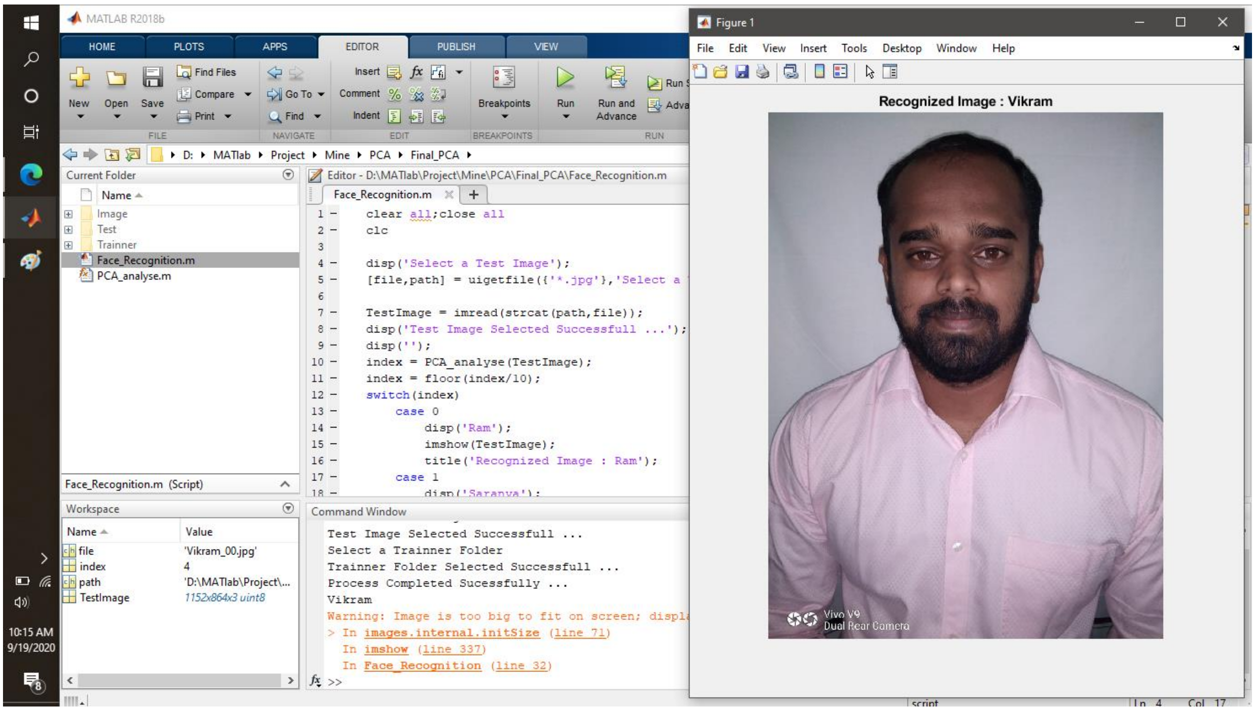
```
A = [];
for i=1 : imgcount
    temp = double(X(:,i)) - m;
    A = [A temp];
end

L= A' * A;
[V,D]=eig(L);

L_eig_vec = [];
for i = 1 : size(V,2)
    if( D(i,i) > 1 )
        L_eig_vec = [L_eig_vec V(:,i)];
    end
end
eigenfaces = A * L_eig_vec;
projectimg = [ ];
for i = 1 : size(eigenfaces,2)
    temp = eigenfaces' * A(:,i);
    projectimg = [projectimg temp];
end
test_image = testimg;
test_image = test_image(:,:,1);
test_image = imresize(test_image,[200,180],'bilinear');
[Row,Column] = size(test_image);
temp = reshape(test_image',Row*Column,1);
temp = double(temp)-m;
projtestimg = eigenfaces'*temp;
euclide_dist = [ ];
for i=1 : size(eigenfaces,2)
    temp = (norm(projtestimg-projectimg(:,i)))^2;
    euclide_dist = [euclide_dist temp];
end
[euclide_dist_min index] = min(euclide_dist);
rec_Index = index;
%disp(index);
disp('Process Completed Sucessfully ...');
disp('');
```



```
17 -        case 1
18 -            disp('Saranya');
```

```
Select a Test Image
```

FIG 5. SELECTING TRAINED FACE IMAGE



Editor - D:\MATlab\Project\Mine\PCA\Final_PCA\Face_Recognition.m

```
    clear all;close all
    clc

    disp('Select a Test Image');
    [file,path] = uigetfile({'*.jpg'},'Select a

    TestImage = imread(strcat(path,file));
    disp('Test Image Selected Successfull ...');
    disp('');
    index = PCA_analyse(TestImage);
    index = floor(index/10);
    switch(index)
        case 0
            disp('Ram');
            imshow(TestImage);
            title('Recognized Image : Ram');
        case 1
            disp('Saranya');
```

Name	Value
file	'Vikram_00.jpg'
index	4
path	'D:\MATlab\Project\...
TestImage	1152x864x3 uint8

Command Window

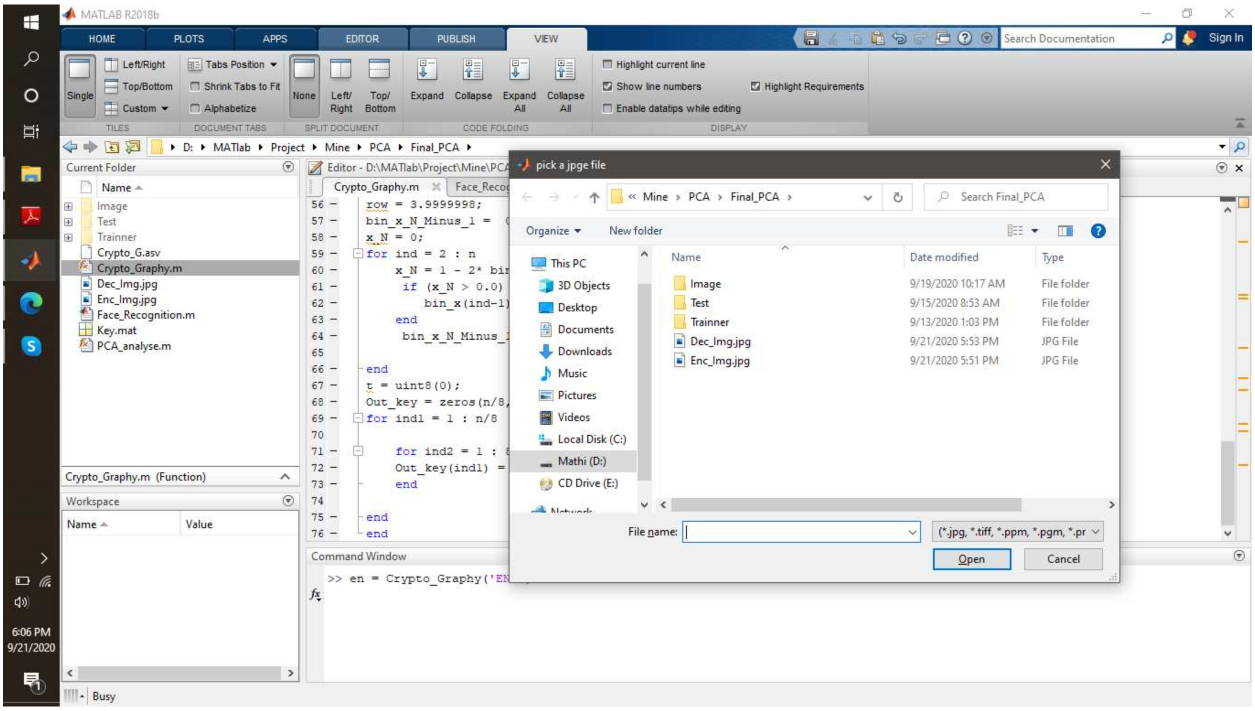
```
Test Image Selected Successfull ...
Select a Trainner Folder
Trainner Folder Selected Successfull ...
Process Completed Sucessfully ...
Vikram
Warning: Image is too big to fit on screen; displa
> In images.internal.initSize (line 71)
  In imshow (line 337)
  In Face_Recognition (line 32)
>>
```

FIG 6. RECOGNIZED FACE IMAGE

HYBRID-CRYPTOGRAPHY

```
function EncImg = Crypto_Graphy(Input)

if strcmp(Input,'ENC') == 1
    [file,path] = uigetfile('*.jpg;*.tiff;*.ppm;*.pgm;*.png','pick a jpge file');
    Orgimg = imread(strcat(path,file));
    figure;
    imshow(Orgimg);
    [r,c,p] = size(Orgimg);
    key = rand_key(r*c);
    save('Key.mat','key');
    Enc_img = Cryptography(Orgimg,key);
    EncImg = Enc_img;
    figure;
    imshow(Enc_img);
    imwrite(Enc_img,'Enc_Img.jpg','jpg');
else
    %[file,path] = uigetfile('*.jpg;*.tiff;*.ppm;*.pgm;*.png','pick a jpge file');
    %Enc_img = imread(strcat(path,file));
    Enc_img = Input;
    figure;
    imshow(Enc_img);
    load('Key.mat');
    Dec_img = Cryptography(Enc_img,key);
    figure;
    imshow(Dec_img);
    imwrite(Dec_img,'Dec_Img.jpg','jpg');
end

end

function [ImageOut] = Cryptography(ImgIn,key)


[n,m,k] = size(ImgIn);
for ind = 1 : m
    Fkey(:,ind) = key((1+(ind-1)*n) : (((ind-1)*n)+n));
end
len = n;
bre = m;
for ind = 1 : k
    Org_img = ImgIn(:,:,ind);
for ind1 = 1 : len
    for ind2 = 1 : bre
        PI(ind1,ind2) = bitxor(Org_img(ind1,ind2),Fkey(ind1,ind2));
    end
end
ImageOut(:,:,ind) = PI(:,:,1);
end
return;
end


function [Out_key] = rand_key(n)
n = n*8;
bin_x = zeros(n,1,'uint8');
row = 3.9999998;
bin_x_N_Minus_1 =  0.300001;
x_N = 0;
for ind = 2 : n
    x_N = 1 - 2* bin_x_N_Minus_1 * bin_x_N_Minus_1;
     if (x_N > 0.0)
        bin_x(ind-1) = 1;
    end
     bin_x_N_Minus_1 =  x_N;

end
t = uint8(0);
Out_key = zeros(n/8,1,'uint8');
for ind1 = 1 : n/8
```

```
    for ind2 = 1 : 8
    Out_key(ind1) = Out_key(ind1) + bin_x(ind2*ind1)* 2 ^ (ind2-1);
    end

end
end
```



Editor - D:\MATlab\Project\Mine\PCA

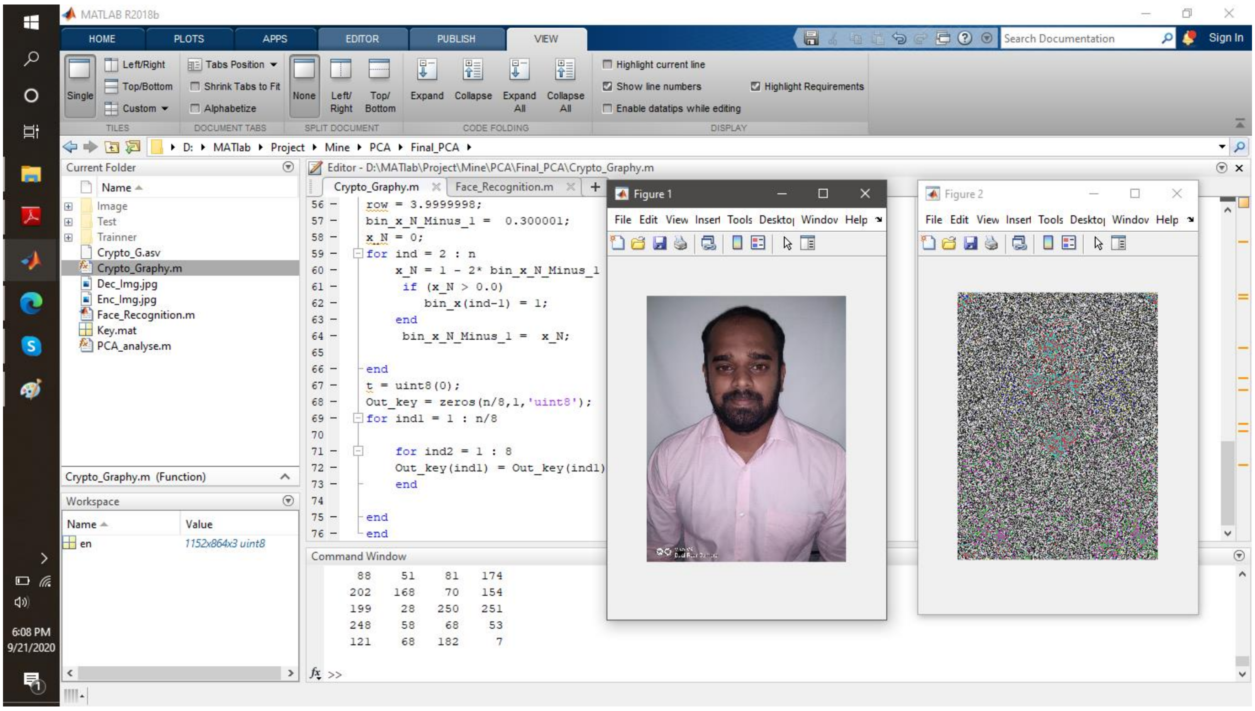
```
row = 3.9999998;
bin_x_N_Minus_1 =
x_N = 0;
for ind = 2 : n
    x_N = 1 - 2* bin
     if (x_N > 0.0)
        bin_x(ind-1)
    end
     bin_x_N_Minus_1

end
t = uint8(0);
Out_key = zeros(n/8,
for ind1 = 1 : n/8

    for ind2 = 1 : 8
    Out_key(ind1) =
    end

end
end
```

Command Window

```
>> en = Crypto_Graphy('EN
```

pick a jpge file
Mine › PCA › Final_PCA
Search Final_PCA
Organize
New folder
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
Mathi (D:)
CD Drive (E:)
Name
Date modified
Type
Image 9/19/2020 10:17 AM File folder
Test 9/15/2020 8:53 AM File folder
Trainner 9/13/2020 1:03 PM File folder
Dec_Img.jpg 9/21/2020 5:53 PM JPG File
Enc_Img.jpg 9/21/2020 5:51 PM JPG File
File name:
(*.jpg, *.tiff, *.ppm, *.pgm, *.pr
Open
Cancel
Busy

FIG 7. SELECTING HYBRID-CRYPTOGRAPHY



Editor - D:\MATlab\Project\Mine\PCA\Final_PCA\Crypto_Graphy.m

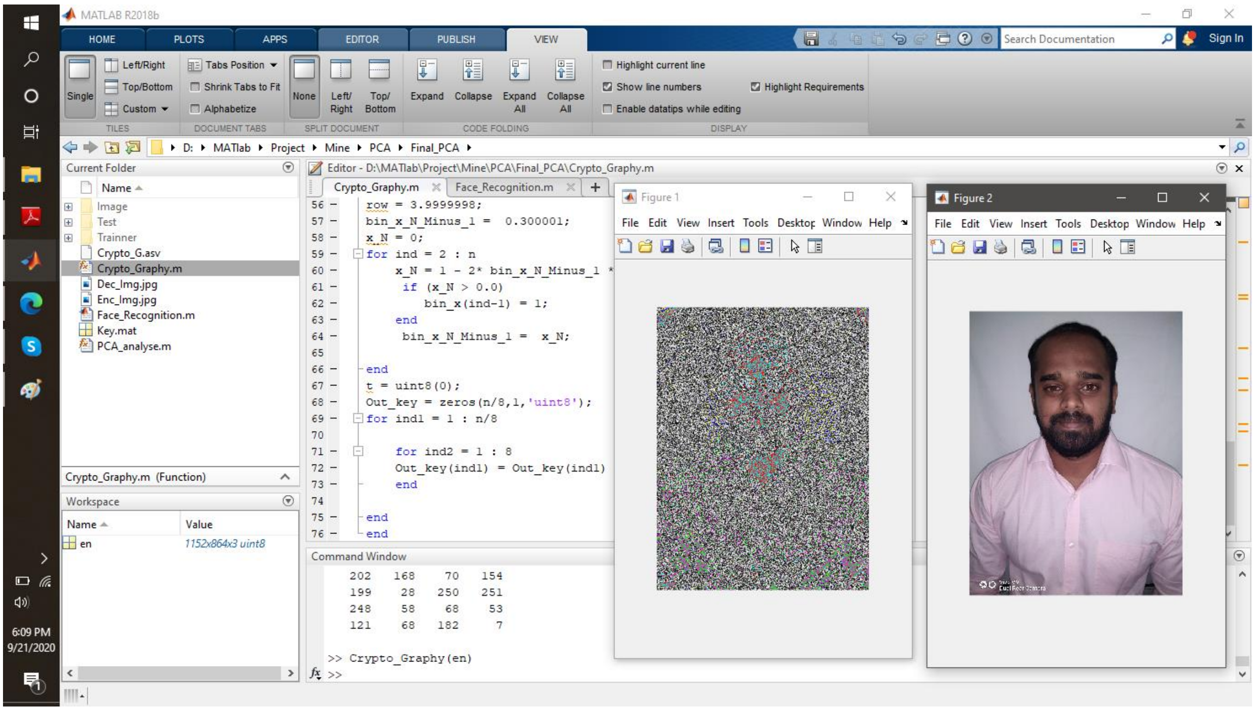
```
row = 3.9999998;
bin_x_N_Minus_1 =  0.300001;
x_N = 0;
for ind = 2 : n
    x_N = 1 - 2* bin_x_N_Minus_1
     if (x_N > 0.0)
        bin_x(ind-1) = 1;
    end
     bin_x_N_Minus_1 =  x_N;

end
t = uint8(0);
Out_key = zeros(n/8,1,'uint8');
for ind1 = 1 : n/8

    for ind2 = 1 : 8
    Out_key(ind1) = Out_key(ind1)
    end

end
end
```

Workspace: en 1152x864x3 uint8

Command Window

```
   88    51    81   174
  202   168    70   154
  199    28   250   251
  248    58    68    53
  121    68   182     7

>>
```

FIG 8. AFTER ENCRYPTION



Editor - D:\MATlab\Project\Mine\PCA\Final_PCA\Crypto_Graphy.m

```
    row = 3.9999998;
    bin_x_N_Minus_1 =  0.300001;
    x_N = 0;
    for ind = 2 : n
        x_N = 1 - 2* bin_x_N_Minus_1 *
         if (x_N > 0.0)
            bin_x(ind-1) = 1;
        end
         bin_x_N_Minus_1 =  x_N;

    end
    t = uint8(0);
    Out_key = zeros(n/8,1,'uint8');
    for ind1 = 1 : n/8

        for ind2 = 1 : 8
        Out_key(ind1) = Out_key(ind1)
        end

    end
    end
```

Command Window

```
   202   168    70   154
   199    28   250   251
   248    58    68    53
   121    68   182     7

>> Crypto_Graphy(en)
fx >>
```

Workspace

Name	Value
en	1152x864x3 uint8

FIG 9. AFTER DECRYPTION

CONCLUSION

This paper presents a secured face recognition system. Principal component analysis is used for face recognition and in cryptography hybrid is used to secure the face image. Face recognition using PCA method has a very good and fast ability to recognize human face. PCA is used to reduce the large dimensionality of the data space observed variables to the smaller intrinsic dimensionality of feature space independent variables. Hybrid encryption is a mode of encryption that merges two or more encryption systems. It incorporates a combination of asymmetric and symmetric encryption to benefit from the strengths of each form of encryption. These strengths are respectively defined as speed and security. No More Time Fraud – One of the big benefits of using this system in company to protect from time fraud. It will be impossible for unauthorized user punching to occur, since everyone has to have their face scanned to clock in. Not only can you track employees, but any visitors can be added to the system and tracked throughout the area too. Anyone that is not in the system will not be given access. Automated System – Many companies like the fact that biometric imaging systems are automated. You won't have to worry about having someone there to monitor the system. Easy Integration – This face recognition system are secured to integrate to companies, hospitals to make authorizes users (patients) get access in hospitals and know their health status and in banking sector etc.

REFERENCES

[1] A.Singh, and S. Kumar, "Face recognition using PCA and Eigenface approach", Thesis: National Institute of Technology Rourkela, 2012.
[2] G. C. Feng, P. C. Yuen, and D. Q. Dai, "Human face recognition using PCA on wavelet subband", J. Electron. Imaging, vol. 9, pp. 226-233.
[3] D. Kumar, Rajni, "An efficient method of PCA based face recognition using simulink", International journal of advanced in computer science and technology, vol. 3, pp. 364-368, May 2014.
[4] Two-dimensional Weighted PCA algorithm for Face Recognition Vo Dinh Minh Nhat 0-7803-9355-4 / 05 / $20.00 ©2005 IEEE.
[5] Koren, Y.; Carmel, L. "Robust linear dimensionality reduction" Visualization and Computer Graphics, IEEE Transactions on , Volume: 10 , Issue: 4 , July-Aug. 2004 Pages:459 – 470.
[6] Jian Yang; Zhang, D.; Frangi, A.F.; Jing-yu Yang "Two-dimensional PCA: a new approach to appearance-based face representation and recognition" Pattern Analysis and Machine Intelligence, IEEE Transactions on, Volume: 26, Issue: 1, Jan 2004 Pages:131 – 137.
[7] (2D)^2 PCA: 2-Directional 2-Dimensional PCA for Efficient Face Representation and Recognition Daoqiang Zhang[1,2] and Zhi-Hua Zhou[1*].
[8] Saleh Saraireh, "A Secure Data Communication System Using Cryptography And Steganography", International Journal of Computer Networks & Communications (IJCNC) Vol.5, No.3, May (2013).
[9] "A comparative survey of symmetric and asymmetric key cryptography," International Conference on Electronics, Communication and Computational Engineering (ICECCE) (2014).
[10] ECC Based Threshold Cryptography for Secure Data Forwarding and Secure Key Exchange in MANET (I) Levent Ertaul and Weimin Lu Department of Math and Computer Science, California State University, East Bay, 25800 Carlos Bee Blvd, Hayward, CA 94542-3092 USA.