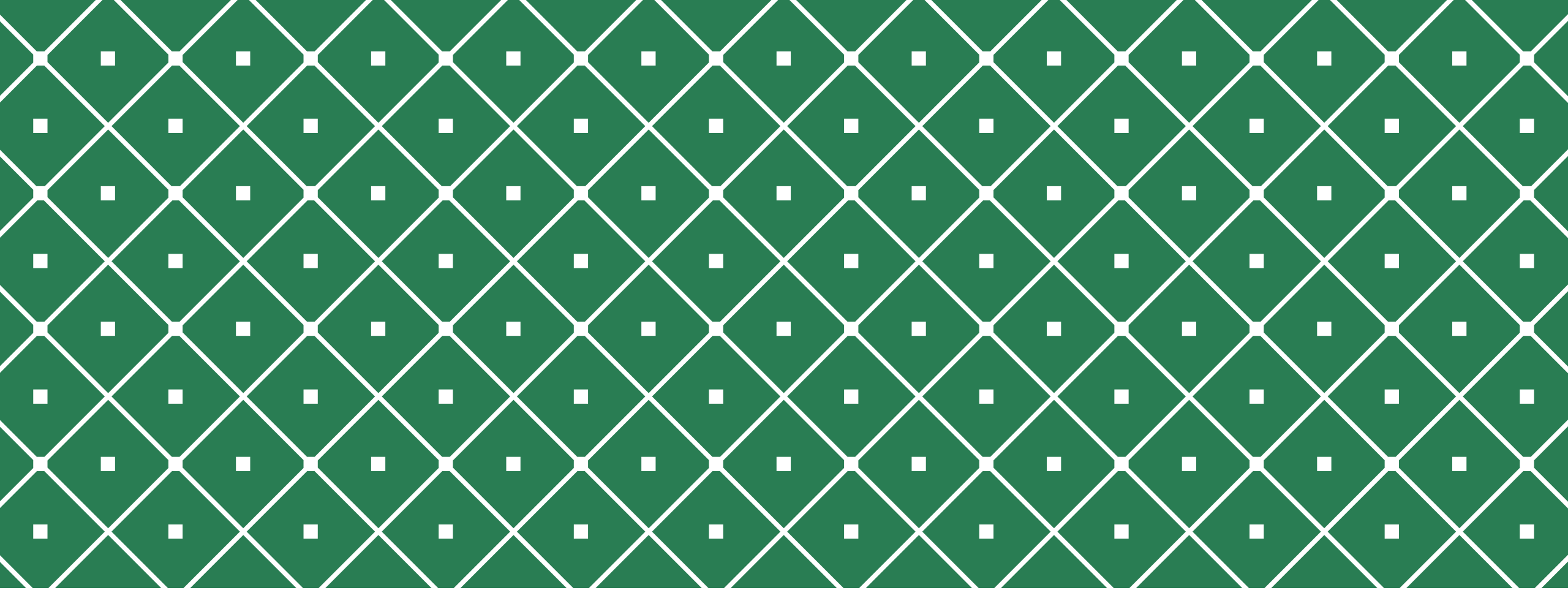
EIGENFACES FOR RECOGNITION

- DEBANGSHU BHATTACHARYA
- SWARAJ BOSE
- AVIRUP CHAKRABORTY
- IPSITA GHOSH

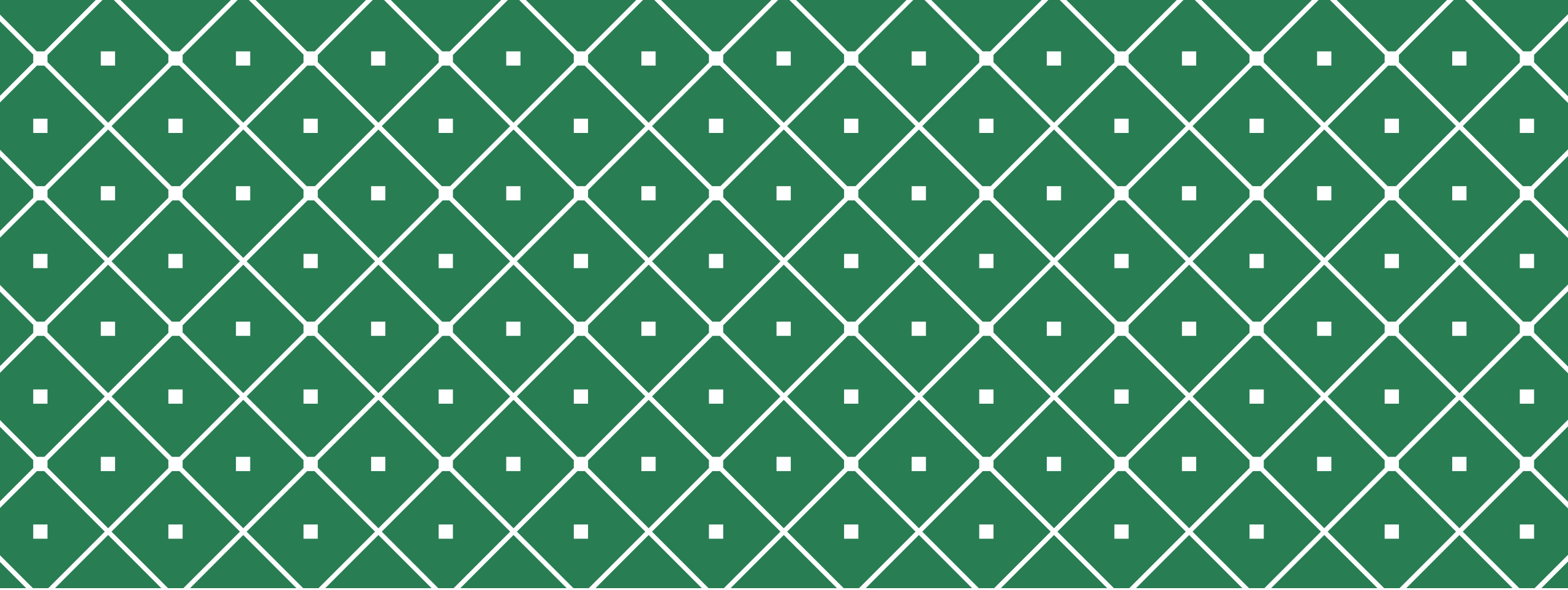


MOTIVATION

- using eigenfaces for recognition
- face image is a weighted sum of eigenface features
- comparing weights of new images with existing classes
- being able to tell whether an image contains a human face
- being able to classify face images as known or unknown based on initial set of images

METHOD OUTLINE

- acquire an initial set of face images (training set).
- calculate eigenfaces of training set
- keep top M′ eigenfaces which makes the face space
- project test image on to face space and calculate the M′ weights
- compute distance from space face to check whether it is a face
- if face, compare patterns of weights with existing classes and classify as known or unknown

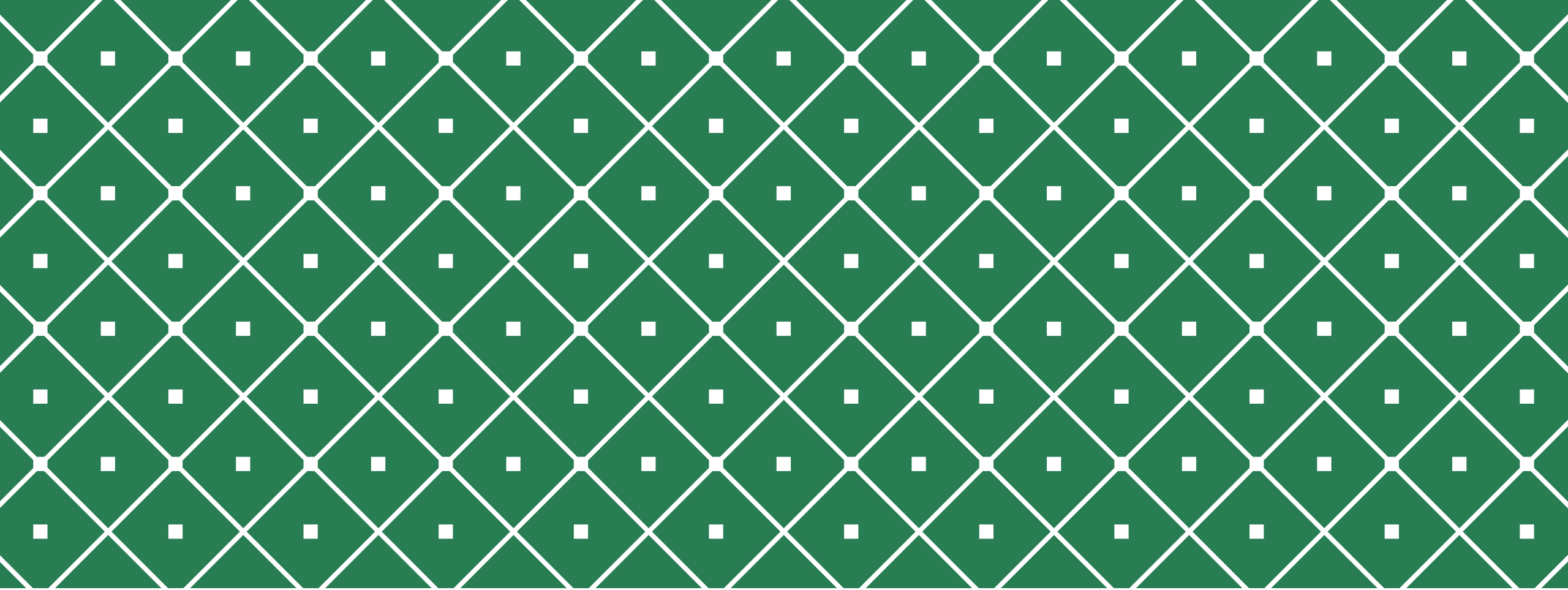


DATA SET USED

YALE DATASET

Key Features:

- 15 persons
- 11 images per person with varying emotions and light
- grayscale images
- GIF format
- size 320x243 (pixels)



CALCULATING EIGENFACES

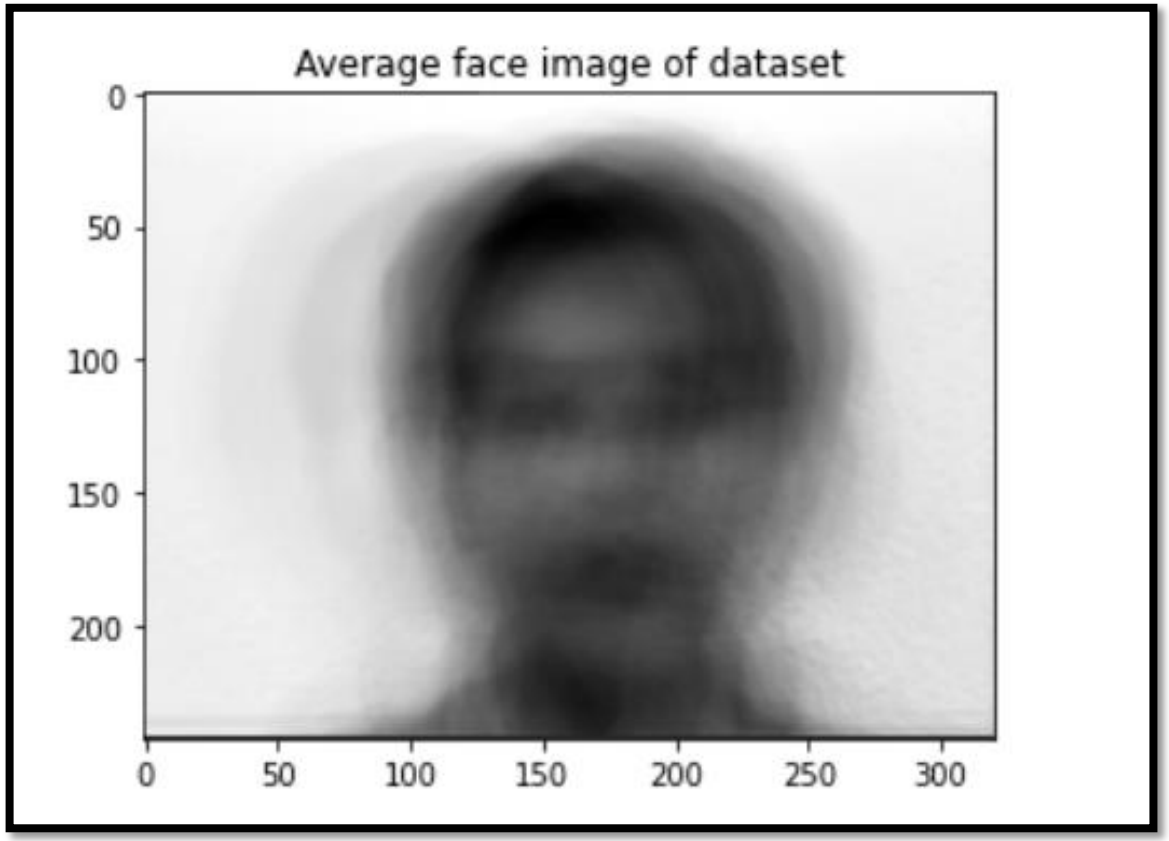
- image I(x,y) can be represented as a 2-D $N_1 \times N_2$ array of 8-bit intensity values
- Flattened to make a 1-D array of dimension $N_1 \times N_2$
- training set of face images: $\tau_1, \tau_2, \ldots\ldots, \tau_M$

- Average face computed : $\psi = \frac{1}{M}\sum_{i=1}^{M} \tau_i$

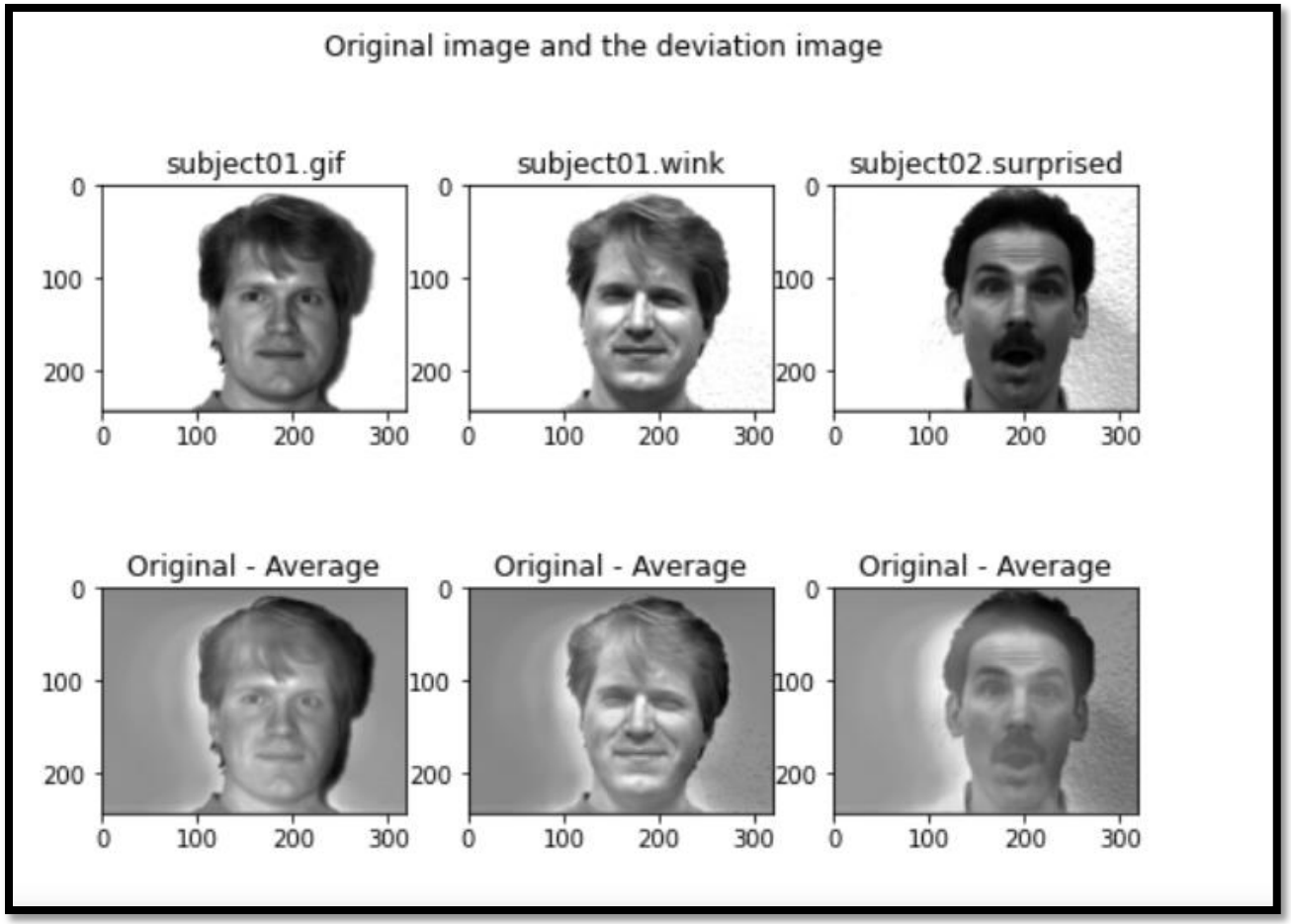
- deviation images : $\Phi_i = \tau_i - \psi$

- So our training data is now represented as : $A = [\Phi_1, \Phi_2, \ldots, \Phi_M]$

AVERAGE FACE IMAGE



ORIGINAL IMAGE AND DEVIATION IMAGE



- Next we compute the covariance matrix :

$$C = \frac{1}{M}\sum_{i=1}^{M} \Phi_i {\Phi_i}^T$$

$$= AA^T$$

- eigenfaces are eigenvectors of C
- dimension of the covariance matrix is $N_1 N_2 \times N_1 N_2$
- computing eigenvalues and vectors computationally expensive
- find eigenvectors of $A^T A$ $(M \times M)$ which reduces computations
- call above eigenvectors v_i

- Now consider eigen vectors v_i of $A^T A$ such that :

$$A^T A v_i = \mu_i v_i$$

- Pre-multiplying both sides with A:

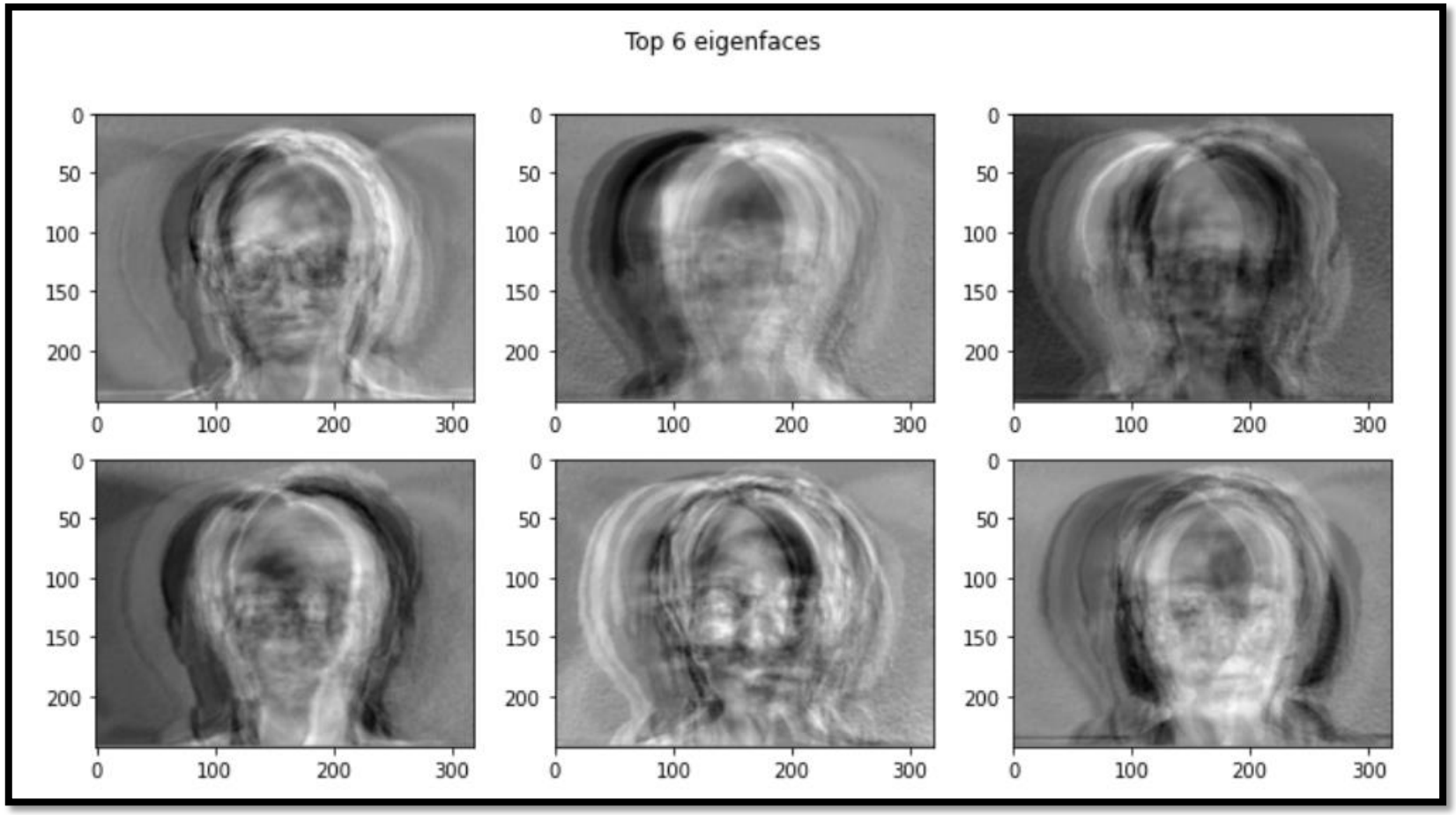
$$AA^T A v_i = \mu_i A v_i$$

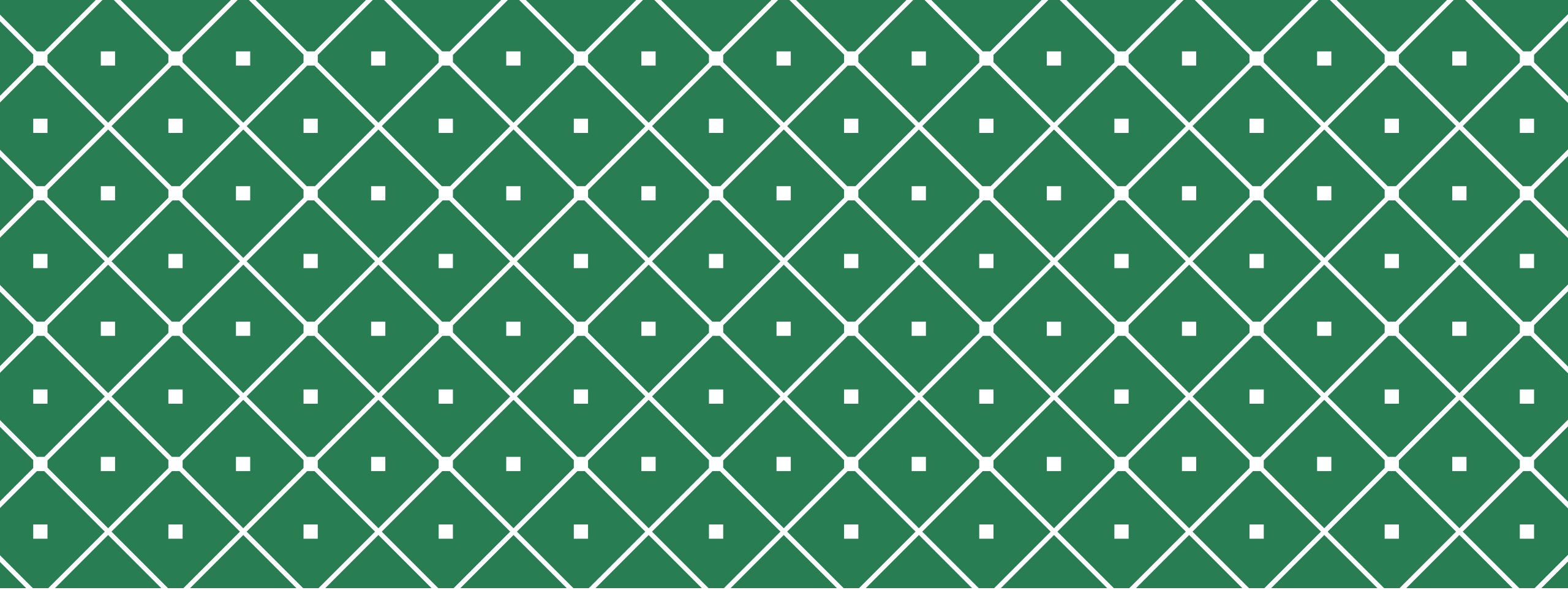
- Thus $Av_i's$ are the eigenvectors of AA^T
- Thus each eigenface u_i can be written as:

$$u_i = \sum_{k=1}^{M} v_{ik} \Phi_k, \qquad i = 1,2, \ldots, M$$

where v_{ik} is the k^{th} component of the i^{th} eigenvector of $A^T A$

- eigenfaces normalized : $||u_i|| = 1, i = 1,2, \ldots, M$
- top $M'(50)$ eigenfaces considered for face space
- The top 6 eigenfaces are shown below:





CLASSIFYING A FACE IMAGE

- For any face image $\Phi = \Gamma\text{-}\Psi$ where Ψ is the average face image of the dataset.

- We project Φ on each of the eigenfaces where the projected weights are computed as :

$$\omega_k = u_k^T \Phi \quad \text{for k} = 1,2\ldots.. M'$$

- The weights form a vector $\Omega^T = [\omega_1, \omega_2 \ldots\ldots \omega_{M'}]$.

- This describes the contribution of each eigenface in representing the face image treating the eigenfaces as basis set vectors.

- We classify a face image as the face class k which minimizes the Euclidean distance :

$$\varepsilon_i = || \Omega - \Omega_i ||^2$$

where Ω_i is the weights of the projection of the average face image of i^{th} class on the face space .

- The distance of an image from face space can be calculated as :

$$\varepsilon^2 = ||\Phi - \Phi_f||^2$$

where Φ_f is the projection of the image on the face space and can be represented as:

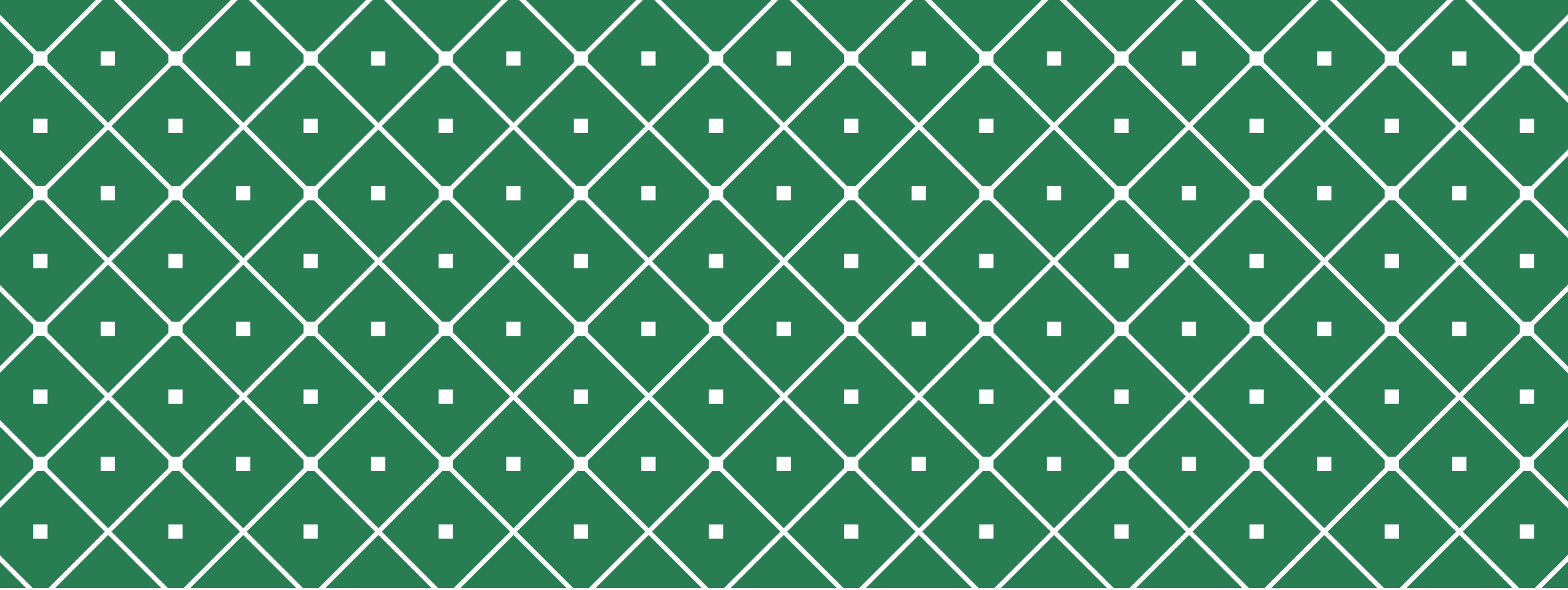
$$\Phi_f = \sum_{i=1}^{M'} \omega_i u_i$$

- If the image has a higher distance from the face space than the threshold θ_e , we confirm that the image is not that of a face.

- If the distance of the image is less than threshold θ_e , then the image is that of a face as it is close to the face space.

- Then we compute the distance of the image from each known class label and find the possible classes where $\varepsilon_k < \theta_k$ and we classify the image as the one with minimum ε_k .

- If no possible candidates exist, then we classify it as a face image of an unknown person.



EXPERIMENTS & RESULTS

- In the first part of our experiment, we have considered the entire dataset for computing the eigenfaces, i.e., 11 images of each of the 15 people , totaling165 images.

- Subsequently, for testing the accuracy of classification, we tested on “normal” images of each person (15 of them in total).

Here, we found that the classifying mechanism was able to **correctly classify all the 15 images** belonging to the test set.

- In the second part of the experiment, we had a mutually exclusive training set for computing eigenfaces and a separate unseen test set for predicting the class of the faces.

  As before, we took the “normal” images of each person (15 of them in total) as our test set. However, this time for computing the eigen faces, our train set consisted of the entire dataset images **minus** the test set images. So our training set consisted of 10 images of each class totaling 150 images.

  Here, we found that the classifying mechanism was able to **correctly classify 14 out of the 15** test set images.

CHOICE OF M'

For both these experiments, we tried out different values of the number of top eigenfaces (i.e., M') to consider as the face space.

Now, considering more number of top eigenfaces gives a better representation of a face image but there is the cost of added computational complexity.

So, there is a tradeoff between the computational cost and classification accuracy.

However, we found out that considering the top 50 eigenfaces gave sufficiently good results w.r.t. accuracy. It is also computationally effective as we are only considering about 1/3rd of the total eigenfaces. So for our experiments, we considered the top 50 eigenfaces as our facespace.

CALCULATING THE THRESHOLD VALUE

- For calculating the face threshold value (θ_e) we planned to use the maximum distance of an image from the face space out of all the images in the training set.

  For calculating the class threshold value vector($\theta = [\theta_1, \theta_2, \ldots\ldots \theta_k]$), for each class label k, we planned to take the maximum distance of the image from that class label for all images of that class label in the training set.

- However, after some experimentation, we observed that the threshold values do not vary much if we choose 2-3 images instead of all of them, per class.

  Thus to reduce computations and run time complexity we have calculated the class thresholds by taking only 3 ("no-glasses", "happy" and "sad") images for each class, i.e., a total of 45 images.

So, instead of the whole training set we did our computations for the threshold values on these 45 images.

CALCULATING THE THRESHOLD VALUE

- In the second part of the experiment however, we took the outlier detection approach instead of taking the maximum distance from class label as threshold value.

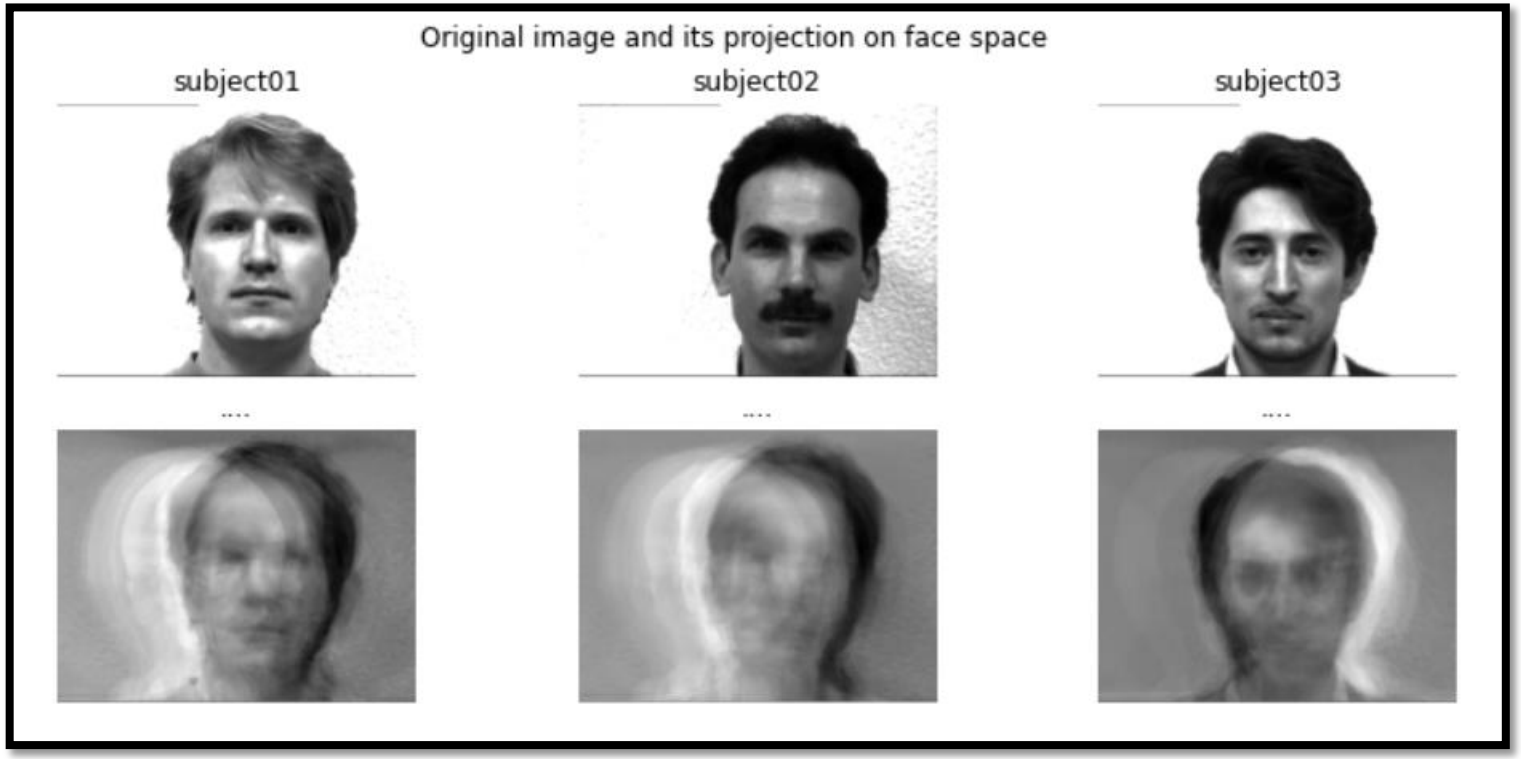
$$\theta_k = Q_3 + 4.5 * IQR$$

where Q_3 is the third quartile value and IQR is the interquartile range value.

- This approach gave better results both in terms of accuracy and also minimizing the number of possible candidates.

- So, even if there is a misclassification, we are not off by much!!

The projections of the first three images belonging to the test dataset (corresponding to the first three subjects) on to the face space



In the next few slides, we have shown the results that we observed by running our algorithm and some other observations we made.

Our computed face threshold value : $\theta_e = 987.72$

The computed class threshold values($\theta'_k s$) are:

Class	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
θ_k	47.66	16.36	17.71	20.74	11.72	22.76	48.13	85.80	25.36	19.70	65.70	131.93	23.31	37.87	87.48

Normal Face 1

True: Face of Class 1

$$34.85 < \theta_1 = 47.66$$

Prediction: Face of Class 1

Distance from face space: 213.87 (< 987.72)

Class no.	Distance from each class label
1	**34.85 (minimum)**
2	138.35
3	159.61
4	113.09
5	76.11
6	377.66
7	162.30
8	146.61
9	165.49
10	118.14
11	125.73
12	160.20
13	102.45
14	283.91
15	147.99

Normal Face 8

True: Face of Class 8

$$24.64 < \theta_8 = 85.80$$

Prediction: Face of Class 8

Distance from face space: 302.82 (< 987.72)

Class no.	Distance from each class label
1	180.15
2	268.82
3	42.20
4	232.70
5	186.18
6	311.65
7	51.90
8	**24.64 (minimum)**
9	66.43
10	105.21
11	223.71
12	274.47
13	173.55
14	197.55
15	43.04

Normal Face 12

True: Face of Class 12

$$0.00 < \theta_{12} = 131.93$$

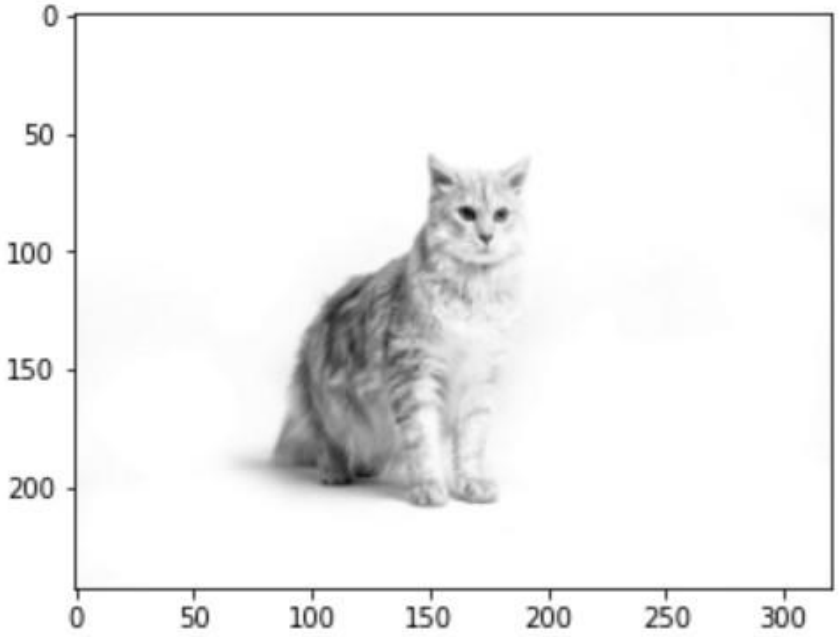
Prediction: Face of Class 12

Distance from face space: 483.23 (< 987.72)

Class no.	Distance from each class label
1	133.27
2	94.33
3	275.82
4	140.03
5	140.43
6	445.02
7	276.10
8	262.85
9	269.88
10	230.37
11	144.15
12	**0.00 (minimum)**
13	183.50
14	374.95
15	265.11

TESTING ON SOME OTHER IMAGES

We also took some different images from Google, preprocessed them to grayscale and adjusted the dimensions to match those of the Yale dataset and observed that the algorithm was able to correctly classify the images.

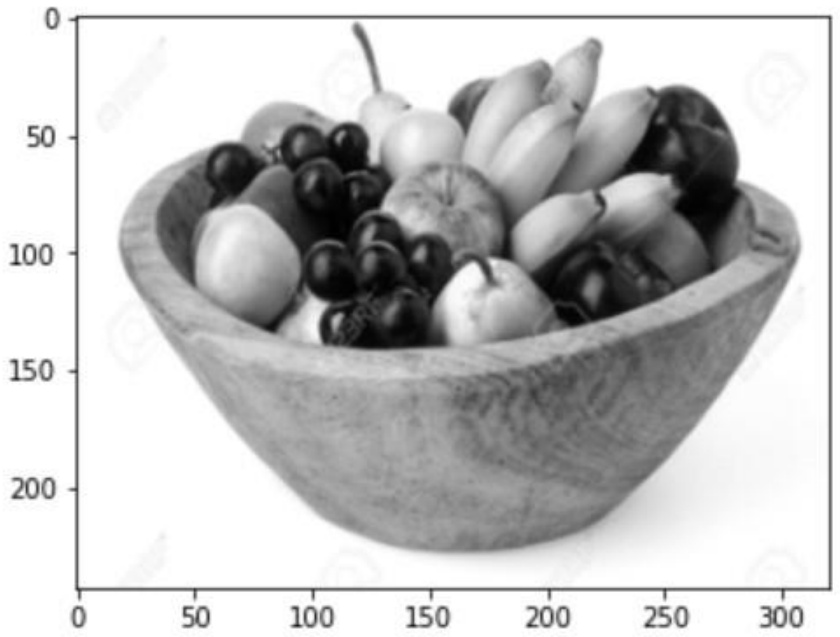


True: Not a face

Computed threshold from the training set = 987.72

Distance from face space = 1368.25 (> 987.72)

Prediction: Not a face

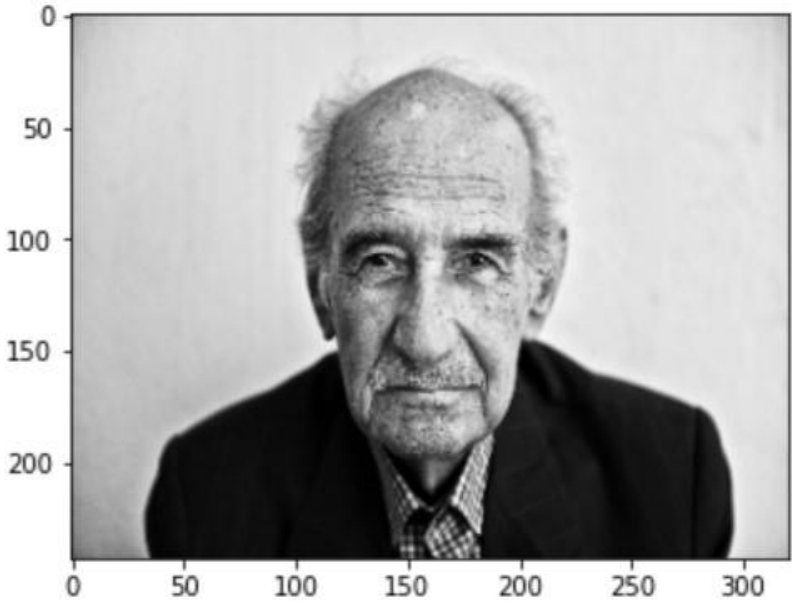


True: Not a face

Computed threshold from the training set = 987.72

Distance from face space = 1106.10 (>987.72)

Prediction: Not a face

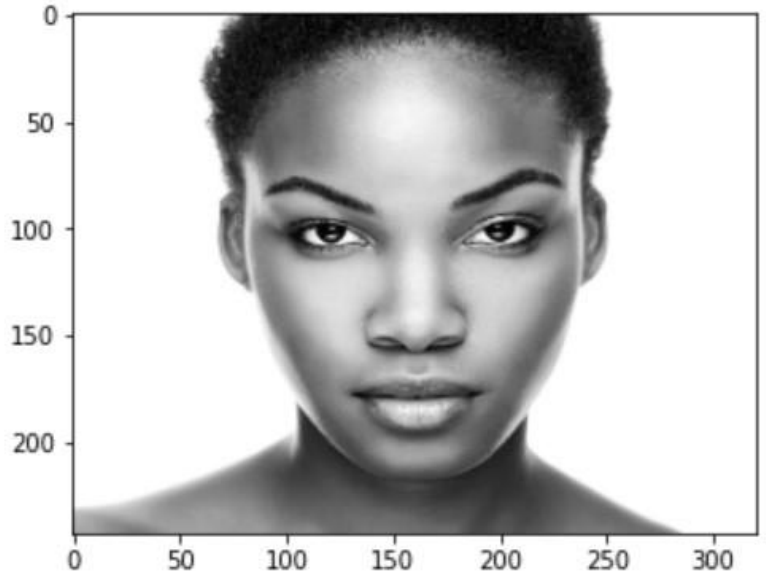


True: Unknown face

Computed threshold from the training set = 987.72
Distance from face space = 346.14 (<987.72)

Prediction: Unknown Face

Predicted Distance to classes	Epsilon k thresholds(θ_k)
209.07	47.66
255.22	16.36
223.20	17.71
235.40	20.74
208.91	11.72
229.82	22.76
212.67	48.13
198.50	85.80
191.56	25.36
197.84	19.70
201.54	65.70
239.04	131.93
228.91	23.31
216.99	37.87
196.25	87.47



True: Unknown face

Computed threshold from the training set = 987.72
Distance from face space = 259.08 (<987.72)

Prediction: Unknown Face

Predicted Distance to classes	Epsilon k thresholds(θ_k)
155.01	47.66
244.69	16.36
130.94	17.71
214.22	20.74
166.05	11.72
248.51	22.76
132.77	48.13
107.74	85.80
96.82	25.36
135.21	19.70
200.50	65.70
247.02	131.93
177.03	23.31
167.89	37.87
100.50	87.47

True: Unknown face

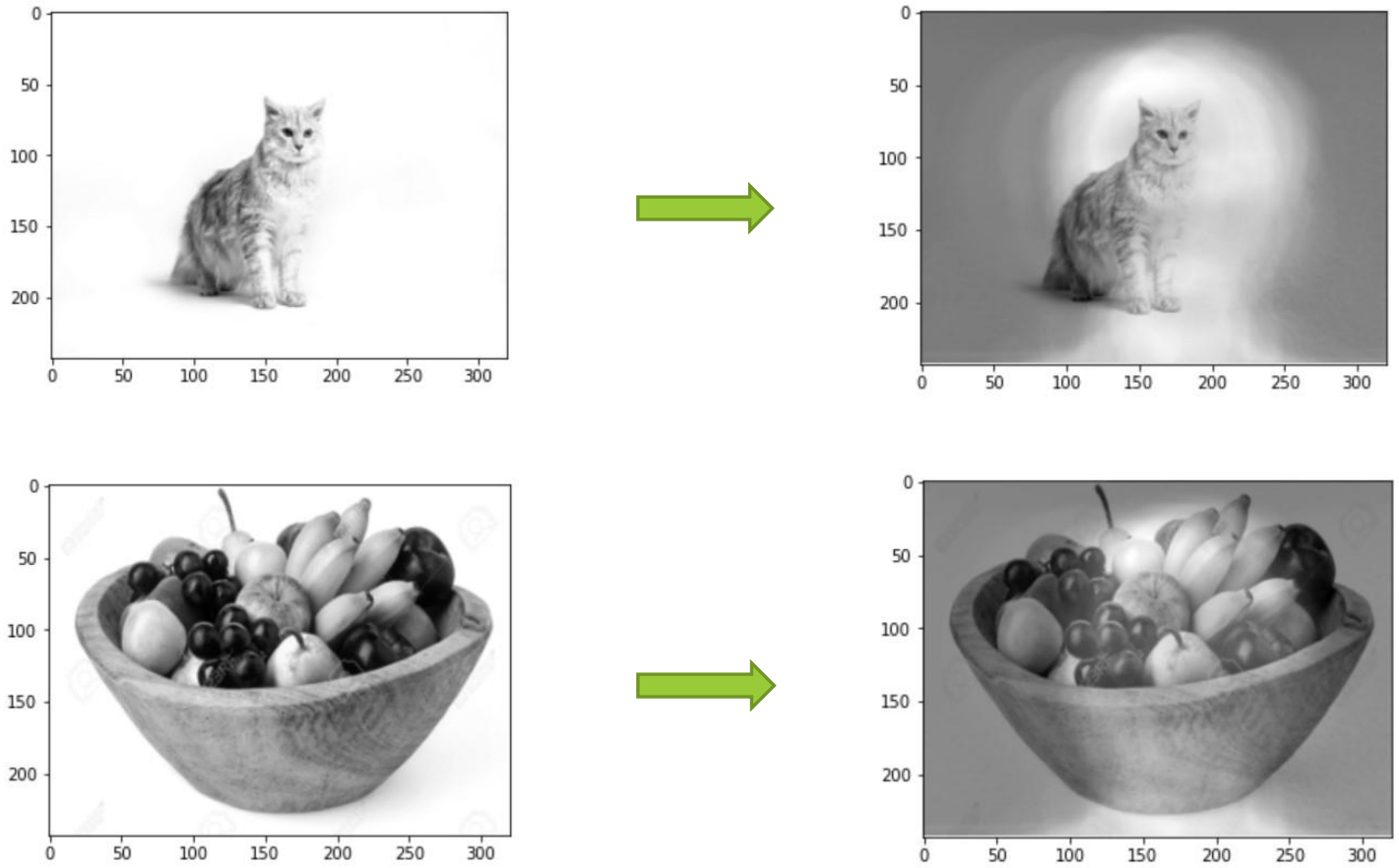
Computed threshold from the training set = 987.72
Distance from face space = 173.55 (<987.72)

Prediction: Unknown Face

Predicted Distance to classes	Epsilon k thresholds(θ_k)
142.81	47.66
215.22	16.36
156.41	17.71
188.48	20.74
165.75	11.72
264.23	22.76
154.54	48.13
130.13	85.80
138.59	25.36
129.98	19.70
162.34	65.70
213.92	131.93
176.50	23.31
199.19	37.87
122.35	87.48

SOME OTHER FINDINGS

- We found that for non face images, subtracting the average face image (computed from the training images), results in a “halo” on the deviation image as shown here:



This resulted in the images to be classified as face images as the halo structure resembles that of a face and hence reduces the distance from the face space.

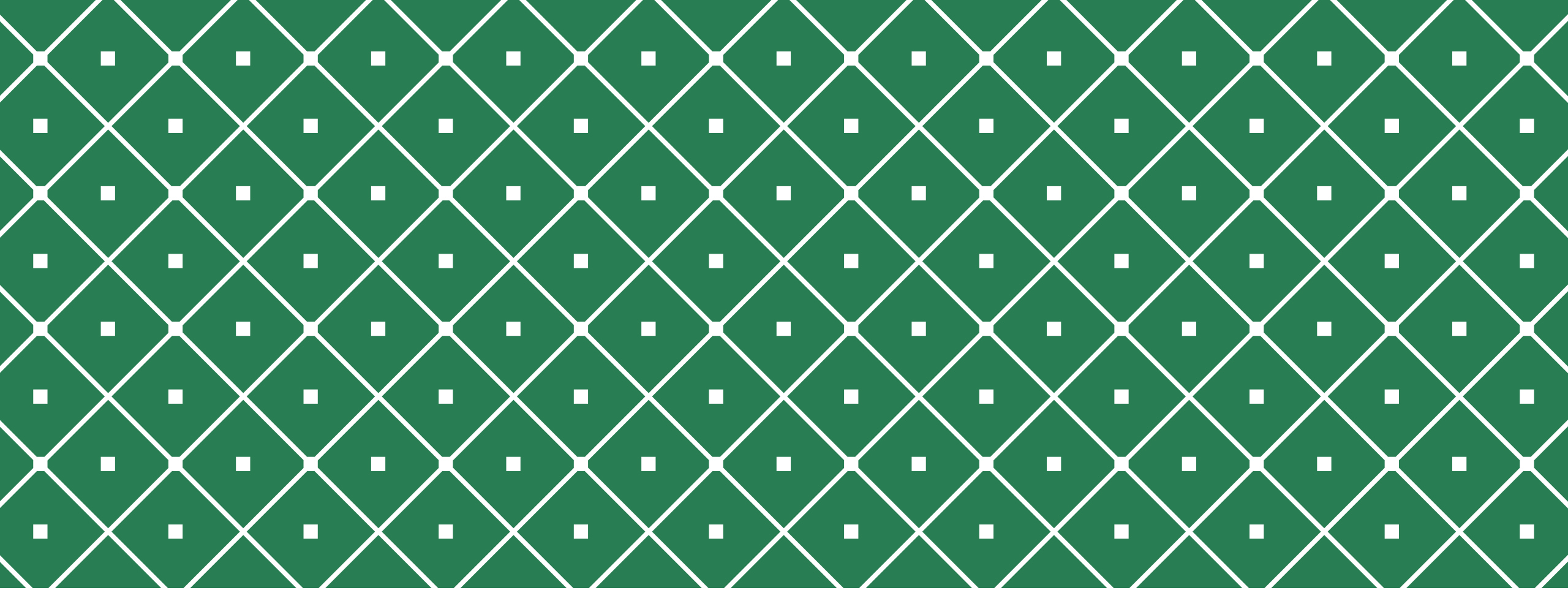
To counter this issue :

- For a non face image we did not subtract the average image before checking its distance from the face space, while
- for a face image, we subtracted the average image and then computed its distance from the face space.

SOME OTHER FINDINGS (CONTD.)

- In the paper, the dataset they used primarily consisted of all images taken in same resolution and the subjects had the same upright posture. However, in our dataset there is a wide variation of emotions for each subject.

While experimenting,we have seen that while considering the projection of average face image of a class, it is better to consider lower number of images with less emotions (like “center light” or “no glasses”) for the average image computation, to have a better classification accuracy.



CONCLUSION

In this project, we

- ✓ could verify that a non-face image was not a face image
- ✓ have been able to correctly classify images with high accuracy
- ✓ could verify whether a face image was that of an existing person or not

Thus, we can say that we have been successful in replicating the works of the authors M. Turk et al. in their paper.

FUTURE SCOPE

UPDATING EIGENFACES

- creating new class when new image of a face occurs
- updating the eigenfaces to incorporate new image
- running the algorithm again for a new face image

FACE DETECTION

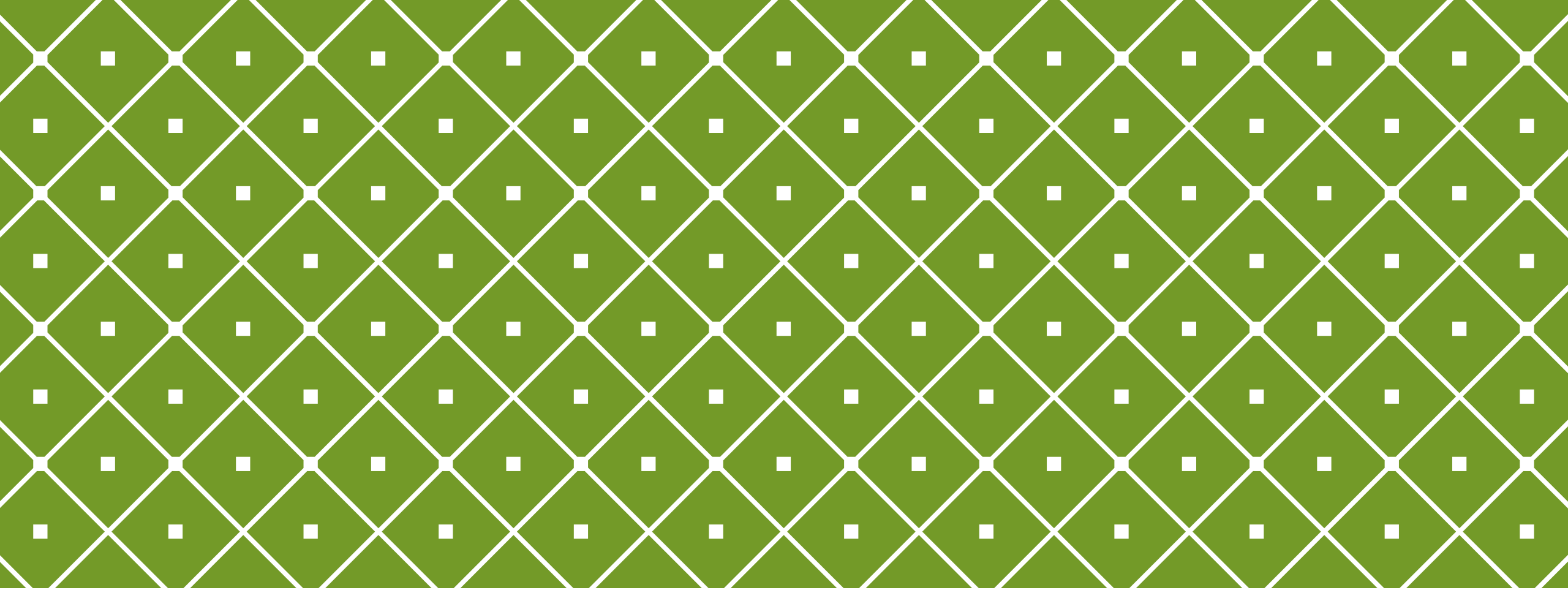
- dividing images into smaller images
- checking for face in each sub-image
- drawing a boundary around the sub-image with minimum distance from face space

REFERENCES

- **M. Turk and A. Pentland, "*Eigenfaces for recognition,*"** Journal of cognitive neuroscience, 1991.

- **M. Turk and A. Pentland, "*Face recognition using eigenfaces*,"** in *Proceedings*. 1991 IEEE computer society conference on computer vision and pattern recognition, 1991, pp. 586–587.

- **http://vision.ucsd.edu/content/yale-face-database**



THANK YOU