Face Detection and Recognition

Muhammad Haziq Bin Razali
U1322810E

Contents

1. Introduction

Finding faces in an arbitrary scene and successfully recognizing them have been an active topic in Computer Vision for decades. In this project, we present a fully automatic facial recognition system capable of handling in-plane rotation. Experiments verify the efficacy of the methods. Illumination variations are explicitly taken care of using a histogram equalization technique.

2. Face Detection

Face recognition is challenging at multiple levels. The first challenge is to detect faces in each frame in the presence of considerable in-plane head pose variations. Misalignments in face localization significantly deteriorate the usability of the system. In the first section, we present a brief description of the detection paradigm adopted for the detection of faces in an image. Following that, we will present a simple solution to cope in plane rotation up to a certain extent.

2.1 Haar-Like Features

Haar-like features are digital image features used in recognition. The difference between adjacent rectangular regions at specific locations in an image indicates the existence or absence of certain characteristics. For example as shown in the figure below, common features of faces include rectangles that lie above the eye and the cheek region.

Figure 2.1: Input frame (left) and histogram of green channel (right)

During run-time, the presence of faces at different scales will be detected by a sliding window approach.

2.2 Rotation Invariant Detector

The method described in the previous section suffers from an inherent inability to detect faces that are rotated by more than a certain angle. In our system, we have considerably simplified the solution using 3 threads to process the incoming video stream while maintaining real time capability. Each thread except the first rotates the frame by 30 (we have found that the Haar face detector is unable to detect faces tilted by more than 30) before invoking the face detector. Faces detected (bounding boxes) in every thread are then de-rotated before results from multiple threads are fused to be displayed in the 1st thread. Note that the figure below illustrates this process with 3 threads. Computers with higher processing power would be able to support more threads and hence detect faces tilted at greater angles while maintaining real time capability.

3. Face Recognition

In the previous section, we showed how given an input video, we can detect faces. We now seek to label these faces as someone belonging to a database or as an unknown individual or an object. Facial recognition algorithms come in 2 stages; face training i.e. the generation of a database and recognition. We will begin with an overview of the training pipeline.

3.1 Training

Recall that the aim of this project is to develop a real-time facial recognition system. The speed of the recognition process is vastly affected by a multitude of factors, including the size of the database and the incoming face. As images in the real world generally come in high dimensions (resolution), performing recognition at a high dimension will impede the real-time capability of the system. The problem of generating concise, representations for describing faces is therefore a vital component in such system.

Over the next two sections, we will describe the methods used to generate the database of faces. We will first pre-process the data to account for any lighting variation before presenting a technique widely used for multivariate analysis, known as Principal Components Analysis.

3.1.1 Global Illumination Normalization

Varying face illumination represents a major problem for many recognition systems. The differences due to varying illumination can be more significant than the differences between individuals face features [1]. As such, techniques aimed to alleviate the problem of illumination variation have become an important precursor to many facial recognition algorithms.

While it is a highly challenging problem to address, the authors in [2] concluded that simple global illumination normalization significantly helps to address inaccuracy elevated from such an issue. In our case, we pre-process all faces in the training database via histogram equalization. This technique adjusts image intensities such that the intensities across all images (faces) become more standardized as shown in the figure below.

Algorithm 1: Histogram Equalization

1. Plot a histogram of the input image
2. Re-distribute the data such that the resultant histogram becomes approximately flat

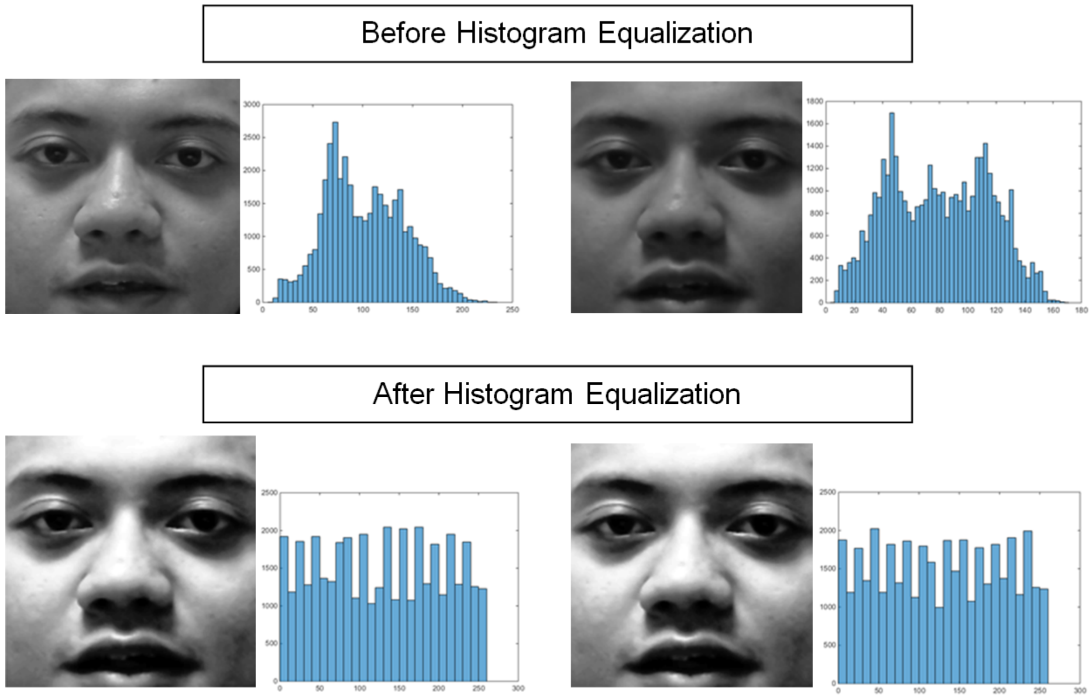

Figure 3.1: Input frame (left) and histogram of green channel (right)

3.1.2 Principal Components Analysis

Given the histogram equalized faces, we now begin the task of representing the set of faces in our training database in a compact form. Principal Components Analysis accomplishes this by projecting the data from a higher dimension to a lower one such that the variance of projection along each new dimension is maximized, retaining as much information as possible.

The benefits include providing a simpler representation of the data, reduction in memory, and faster classification. The algorithm is as follows.

Algorithm 2: Principal Components Analysis

1. Reshape every image into a column vector, simply by concatenating the rows of pixels. Store all images in a single matrix T of size [M rows x N columns], where each column of the matrix is an image and each row the resolution of the image.

2. Compute the average face vector by averaging along the columns of T

3. Subtract the mean from every face in the database and denote it as A i.e. $A = T - \mu$

4. Find the covariance matrix $C = AA^T$ [M rows x M columns]

In our implementation, we have avoided computing the sample covariance explicitly as the sample size (200 faces in our database) is much smaller than the resolution (200^2 pixels per face). The derivation, listed in Appendix A shows that the dimensionality of the covariance matrix can be reduced from M^2 to N^2, resulting in a significant reduction in computation.

3.1.3 Feature Selection

Given the eigenvectors of the covariance matrix, normalized to unit length and rearranged in descending order, the next task would be to project the training faces onto the eigenspace. Although the covariance matrix generates N Eigenfaces, only a fraction of those are required to represent the training database. Additionally, since the structure of the data affects the eigenvalues of each corresponding eigenvector (Figure 3.2), we have chosen to select the number of principal components such that the minimum percentage of variation is met rather than selecting a fixed number of principal components, allowing the training framework to be more adaptive. This concludes the section of training.

3.2 Recognition

At run time, given the faces returned from section 2, we seek to classify these faces into their respective classes or declare them as outliers. As explained before, the faces will be pre-processed via histogram equalization (3.1.1) before being projected onto the eigenspace (3.1.3).

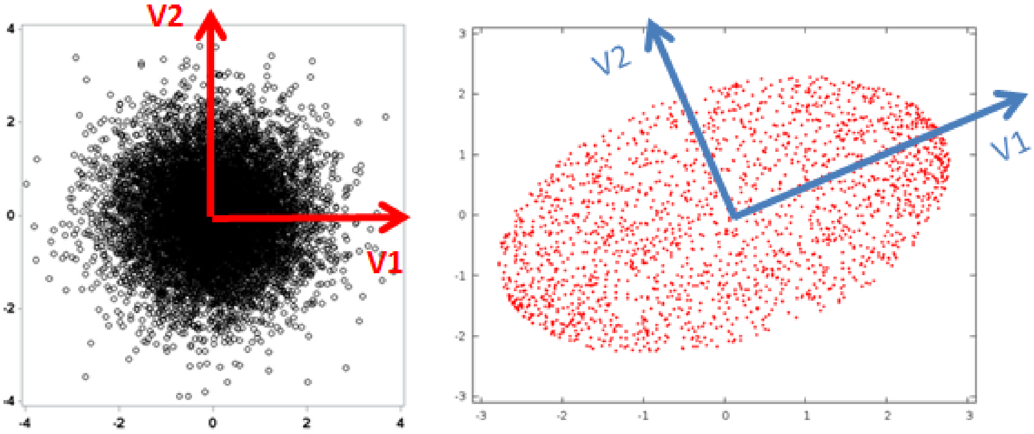

Figure 3.2: The eigenvectors of a circle (left) and an ellipse (right). The eigenvector V1 of the ellipse captures a larger percentage of variance compared to V1 of the circle. The amount of data lost when projecting onto a lower dimensional space is therefore highly depending on the structure of the data.

3.2.1 Outlier Rejection

Test samples may not come from the database. As such, there needs to be a mechanism to identify and subsequently reject outliers. To this end, we have utilized the normalized Euclidean for outlier rejection. It is defined as follows:

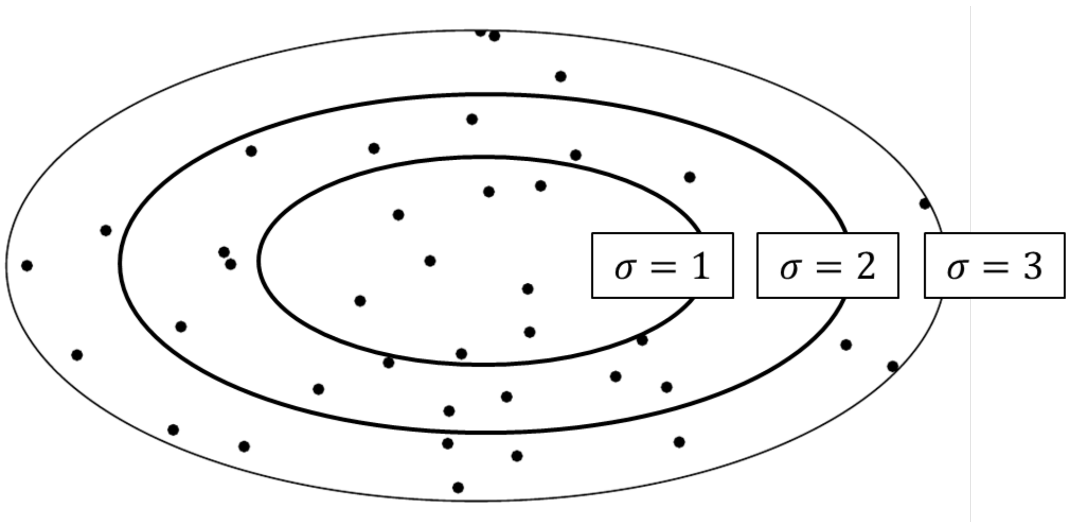

Figure 3.3: An illustration of the normalized Euclidean distance in 2D. The test data X is declared an outlier since its distance is above 3 standard deviations from the mean

3.2.2 Classification

The final task then remains for all test samples not rejected by the previous section, i.e. classification. For it to be more robust against noise, we have utilized the K nearest neighbour (KNN) which classifies data based on the majority voting scheme.

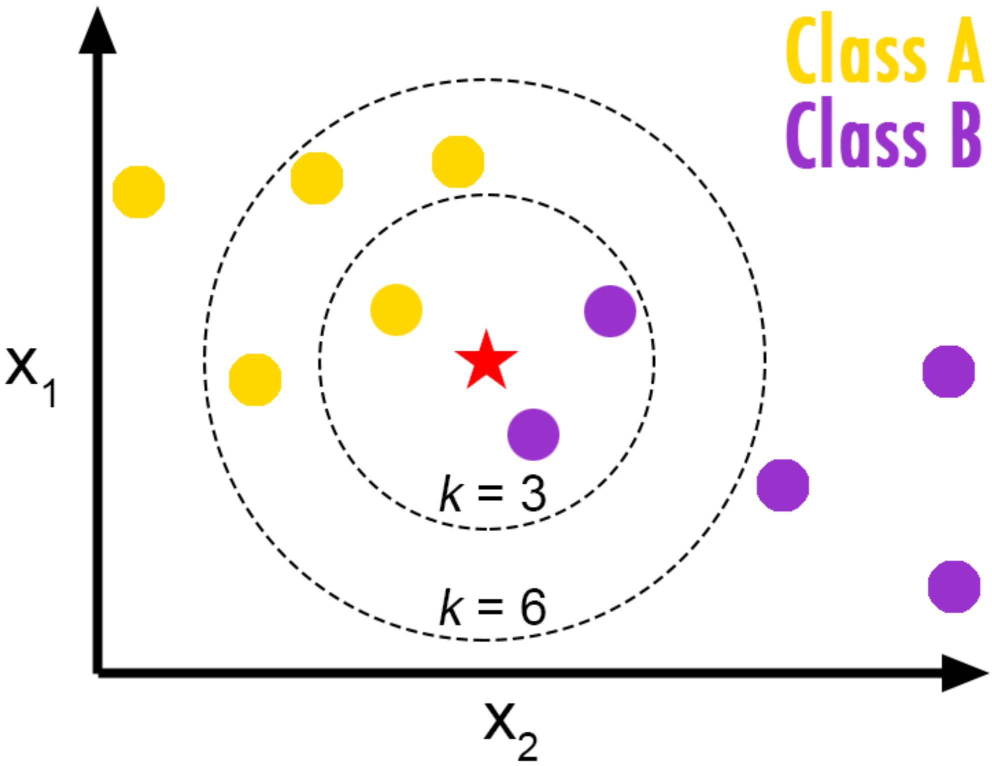

Figure 3.4: The test data X is classified as class B if k = 3 and as class A if k = 6

4. Conclusion

We have presented an automatic facial recognition system capable of handling in-plane rotation for about 50 degrees. All faces are pre-processed via histogram equalization to mitigate the effect of lighting variation. The resultant data are then projected onto a lower dimensional manifold via Principal Components Analysis. Despite its attractive features, PCA models have several shortcomings i.e. it is very sensitive to lighting, scale and translation. As such, PCA should be used in conjunction with other techniques that optimize the discrimination between classes. Additionally, lighting from natural or artificial sources do not often affect the image on a global scale. Thus, methods that partition the face into separate regions for illumination correction could significantly enhance accuracy.

Bibliography

[1] Bo Cao Shiguang Shan, Wen Gao and Debin Zhao. Illumination normalization for robust face recognition against varying lighting conditions. In *Proceedings of the IEEE International Workshop on Analysis and Modeling of Faces and Gestures.*

[2] Josef Kittler Xuan Zou and Kieron Messer. Illumination invariant face recognition: A survey. In *IEEE International Conference on Biometrics: Theory, Applications, and Systems.*

Appendix

The computation of covariance matrix of $\mathrm{AA^T}$ is impractical if the sample size << sample dimension. An alternative method is as follows:

1. Let D represent the set of M faces, each reshaped to a column vector, i.e. $\mathrm{D} = (\mathrm{F}_1, \mathrm{F}_2, ...\mathrm{F}_3)$
2. The 'large' covariance matrix, L is $\frac{1}{\mathrm{N}}\mathrm{DD^T}$ where N denotes the resolution of the images.
3. Let S be the M M 'small' covariance matrix, i.e. $\mathrm{S} = \frac{1}{\mathrm{N}}\mathrm{D^TD}$
4. Let the eigenvector, $\mathrm{e_i}$ be the unit length, orthogonal eigenvectors of T with corresponding eigenvalues $mathrm\gamma$ i.e. $\mathrm{Te_i} = \gamma_\mathrm{i}\mathrm{e_i}$ where (i=1,,N)
5. Hence, $\frac{1}{\mathrm{N}}\mathrm{D^TDe_i} = \gamma_\mathrm{i}\mathrm{e_i}$
6. Multiplying by D gives us $\frac{1}{\mathrm{N}}\mathrm{DD^TnDe_i} = \gamma_\mathrm{i}\mathrm{De_i}$
7. Replacing $\frac{1}{\mathrm{N}}\mathrm{DD^T}$ by L results in the following: $\mathrm{L(De_i)}\gamma_\mathrm{i}\mathrm{(De_i)}$

The eigenvectors of the large covariance matrix L therefore is equal to $\mathrm{De_i}$ where $\mathrm{e_i}$ denote the eigenvector of the small covariance matrix S. Both eigenvectors share the same eigenvalue.