

**Faculty of Natural and  
Mathematical Sciences**  
Department of Informatics

The Strand  
Strand Campus  
London WC2R 2LS  
Telephone 020 7848 2145  
Fax 020 7848 2851



**7CCSMPRJ**

**Individual Project Submission 2016/17**

**Name:** Chaoyue Niu  
**Student Number:** 1642834  
**Degree Programme:** MSc Robotics  
**Project Title:** Face Recognition with the PC/BC – DIM algorithm  
**Supervisor:** Dr.M.W. Spratling  
**Word Count:** 11577

**RELEASE OF PRODUCT**

Following the submission of your project, the Department would like to make it publicly available via the library electronic resources. You will retain copyright of the project.

- ☐ I **agree** to the release of my project  
☒ I **do not** agree to the release of my project

**Signature:**

*Chaoyue Niu*

**Date:** August 25, 2017

Department of Informatics  
King's College London  
WC2R 2LS London  
United Kingdom

---

**Face Recognition with the PC/BC – DIM algorithm**

---

Chaoyue Niu  
Student Number: 1642834  
Course: MSc Individual Project

**Supervisor:** Dr. M.W.Spratling



Thesis submitted as part of the requirements for the award of the MSc in Robotics

7CCSMPRJ - MSc Individual Project - 2017

## Abstract

It is essential to have a deep understanding of how to develop an automatic method to promote the rate of face alignment become fast and accurately. Until now, with the rise of algorithms of machine learning, creating a system for automatic face alignment system is easier than ever. The current challenges of face alignment are divided into three parts, i.e. dealing with posture problem, expression problem and occlusion problem. In this project, I mainly focus on solving the problem of posture when human faces are rotated and at different distances from the camera.

This project gives the detailed background and original designing ideas to develop an automatic method for aligning and resizing faces taken from the camera. The framework elaborates most of the face detection methods involving AdaBoost algorithm and computer vision system toolbox based on existing methods and papers, as well as an implemental process of effective face alignment system by modifying current manual alignment system.

**Keywords:** Face alignment; AdaBoost; Computer vision system toolbox;

## Acknowledgements

I would like to express my special thanks of gratitude to my supervisor, Dr. M.W.Spratling who gave me the golden opportunity to do this wonderful project on the topic (Face Recognition with the PC/BC – DIM algorithm), which also helped me in doing a lot of research and I came to know about so many new things I am really thankful to him.

Secondly, I would also like to thank my parents and my friend Honghu Pan who helped me a lot in finalizing this project within the limited time frame

## Contents

1 Chapter 1.....	6
1. Introduction .....	6
1.1 Motivation.....	6
1.2. Project Objectives .....	6
1.3. Background .....	7
1.3.1. Face Detection.....	7
1.3.2. Face Alignment.....	7
1.3.3. Face Recognition .....	8
1.4 Research Aims .....	8
1.5 Organization .....	9
2 Chapter 2.....	10
2. Background.....	10
2.1 Overview .....	10
2.2 Face detection .....	10
2.2.1 Viola-Jones face detector and AdBoost algorithm [1].....	11
2.2.2 Skin color likelihood method [14] .....	15
2.2.3 Face detection with graphical user interface (GUI).....	17
2.3 Face alignment .....	18
2.3.1 Face alignment across large poses: 3D Dense Face Alignment (3DDFA) [33] .....	18
2.3.2 Large-Pose Face Alignment via CNN-Based Dense 3D Model Fitting [34].....	19
2.3.3 Unconstrained Face Alignment via Cascaded Compositional Learning [35] .....	19
2.4 Face recognition using PC/BC-DIM algorithm [4].....	20
2.4.1 Introduction .....	20
2.4.2 PC/BC – DIM algorithm .....	21
2.5 Summary .....	22
3 Chapter 3.....	23
3. System Design and Implement.....	23
3.1 Overview .....	23
3.2 Face detection .....	23
3.2.1 Face detection using Viola-Jones Algorithm .....	23
3.2.2 Face detection using skin color likelihood.....	24
3.2.3 Image edge detection and segmentation .....	25
3.2.4 Face detection via eye template matching .....	28
3.2.5 Robust Real Time Face Detection.....	30
3.2.6 Face detection based on computer vision system toolbox .....	31
3.2.7 Face Detection GUI based on MATLAB R2016a.....	33
3.2.8 Face++ API website [6] .....	34
3.3 Face alignment .....	34
3.3.1 Original design.....	34
3.3.2 Improved design.....	37
3.4 Summary .....	40
4 Chapter 4.....	42

4. Evaluation .....	42
4.1 Overview .....	42
4.2 Face recognition results by using PC/BC – DIM algorithm[4].....	42
4.3 Comparison .....	44
4.3.1 Normalize images .....	44
4.3.2 Do not normalize images .....	46
4.4 Summary .....	47
5 Chapter 5.....	48
5. Conclusion .....	48
5.1 Introduction .....	48
5.2 Project Overview .....	48
5.3 Contributions .....	49
5.4 Further Work.....	49
5.5 Concluding Remarks.....	50

# 1 Chapter 1

## 1. Introduction

### 1.1 Motivation

Face recognition as a biometric identification compared with other mature identification methods, such as fingerprint and DNA detection, has the following advantages:

1. The acquisition of face images does not need to perform physical contact with the detected individual, i.e. non-invasive detection.
2. It is convenient to install face recognition system by an ordinary camera, digital camera or mobile phone embedded cameras. There are no special installation requirements for users.
3. The computer automatically performs recognition program via preset without human participation.

However, there are so many difficulties in the automatic face recognition systems. Its main difficulties are tripped by following these aspects:

1. Due to the intrinsic changes in the human face:
  - (1) Facial features have quite complicated details, such as different skin color, different expressions and even lack of organs.
  - (2) The occlusion of faces, such as eyeglasses, hair, decorations, and other external objects;
2. Due to the change of external conditions:
  - (1) The multi-gesture of the face because of the different imaging angles, such as the rotation in the plane, the depth rotation and the rotation of the top and bottom.
  - (2) Effects of illumination, such as changes in brightness and contrast
  - (3) Imaging conditions of the image, such as the focal length of the camera, the image distance, and the way of image acquisition.

### 1.2. Project Objectives

The project's objectives have been divided into three sub-objectives. The first sub-objective involved in face detection, considered for the start of the project, are:

1. To find some methods and papers of face detection.
2. To evaluate the most suitable face detection program for this project from above methods.
3. To design and implement a reliable face detection system that can locate position facial features, such as eyes, mouth, and nose.
4. To test whether or not this face detection system can properly integrate with the next sub-objective

The second sub-objective aims to perform face alignment considered absolutely pivotal part for the success of the project. They are:

1. To crop image after transformation
2. To align and resize these cropped images
3. To find the coordinate of two eyes so as to perform face alignment, where the marked point of left eye and right eye can be presented using central coordinate of two rectangles by means of the face detection program of the first sub-objective
4. To ensure these images have fixed position and orientation and then generate new face datasets

The last sub-objective is face recognition by PC/BC-DIM algorithm. They are:

1. To show exemplars from the dictionary learnt from face aligned and resized an image.
2. To present all of the misclassified images from the test set and show responses of the prediction neurons to selected images from the test set
3. To calculate the misclassification error rate

## **1.3. Background**

This section presents the latest research background on face detection and recognition and also defines research direction of the whole industry, its place in the newest progression of computer vision systems and its potential in method and technology.

### **1.3.1. Face Detection**

In the aspect of face detection, the current popular method is Fast Region-based Convolutional Network method (Fast R-CNN) [1] extracted from object detection. Cascade Convolutional Neural Network(Cascade CNN) is a method specially designed for face detection, which combines traditional sliding window methods with deep learning, has also achieved the same effect as much as Fast R-CNN.

Face detection has been considered to be a solved problem, this is not so. When the face is in low resolution, backlight, polarized light, extremely low light illumination and other harsh lighting conditions, it will still have lower detection rate.

In view of this, a new face detection Benchmark [2] was presented last year (2016), which can accurately perform face detection in many harsh conditions including occlusions, poses, event categories, and face bounding boxes and should play an important role in the field of face detection.

### **1.3.2. Face Alignment**

For face alignment, since last year, a trend demonstrating that the method of face alignment

would be transformed from the past popular Cascaded Shape Regression [3] based on Deep feature learning to Recurrent Neural Network (RNN), which tries to resolve face alignment problem under the condition of big gesture has been presented.

### 1.3.3. Face Recognition

For face recognition, there are no many new technical progress. The most widely extensive method still applies a variety of Deep Convolutional Neural Networks (DCNN), especially the ResNet. The PC/BC-DIM algorithm used in this project is also based on neural network. The improvement of performance in actual combat is mainly from increasing face datasets, and the face recognition technology in different scenarios has made great progress and fatly enters the market. Furthermore, this area requires new benchmarks to evaluate the essential progress of technology.

## 1.4 Research Aims

Developing a method to align and resize face images automatically is the aim of this project, and then face recognition will be performed by using PC/BC-DIM algorithm on various face databases.

Face identification can be accurately performed by the PC/BC-DIM algorithm, face images tested should have the same fixed position and orientation and are the same size. [4].

Viola-Jones face detector [5] and the Face++ API [6] as one of the most effective method, can locate landmarks of face images and then transform them into new images with fixed location and orientation in given image. Nonetheless, I also should investigate other methods as evaluation part of this project.

Finally, this automatic and reliable method would be applied to execute face recognition and then the results are compared with other algorithms published.

This aim can be broken down into the following objectives,

1. To implement face detection systems based on existing methods and papers
2. Select a face detection system that is suitable for face alignment from above systems and evaluate why others cannot be used.
3. Implement a face alignment system where it can automatically resize and align landmarks of face to a fixed position and orientation within each image
4. This face alignment program will be used to perform face recognition experiments by PC/BC-DIM algorithm.



## 1.5 Organization

Chapter 2 (Background) provides an overview and previous work of face detection, alignment and recognition. It then analyses how integral image and cascade classifier have been used to solve a problem of locating landmarks of a face in face detection. It finally gave an introduction to PC/BC-DIM algorithm and how it works in improving face recognition experiments and how this hierarchical perceptual inference process performed in the cortex.

Chapter 3 (System Design and Implement) describes the various methods of face detection detailing what the project seeks to achieve. It then selected one method of face detection, i.e. face detection based on computer vision system toolbox and inbuilt function in MATLAB. Moreover, according to combining previous face detection selected with face alignment manually method, this chapter described how to develop an automatic method to locate landmarks of a face and then transform them into new images with fixed location and orientation in given image.

Chapter 4 (Evaluation) makes an evaluation of how well the system addressed the issue of improving face recognition experiments by comparing with published results of other algorithms.

Chapter 5 (Conclusion) concludes the challenge of how to develop a new automatic method to achieve face alignment. It also identifies a research advantage in using the AdaBoost algorithm to perform face detection. A face alignment system using computer vision system toolbox based on AdaBoost algorithm, and manual face alignment system, was therefore designed and implemented to improve the recognition rate and running time.

# 2 Chapter 2

## 2. Background

### 2.1 Overview

The previous chapter mainly introduced the motivation, project objectives, background and research aims of face recognition and what is the significant challenge for face recognition. It also introduced the advanced and potential technology in the field of face detection, alignment and recognition. This chapter provides the necessary background to the subject of face detection, alignment, and recognition. It then delves into the various theoretical methods and foundation of the mathematics of face detection and determines the most suitable face detection method for face alignment. Last, working principle and mathematical analysis of PC/BC-DIM algorithm applied to face recognition will be introduced.

### 2.2 Face detection

Face detection is a target detection problem, mainly reflected in two aspects.

On the one hand, it is caused by face target internal change, including two aspects

- (1) The face has quite complicated and detailed changes and different expressions, such as open and close of eye and mouth open. Different faces have the different appearance, such as the face and skin color.
- (2) Facial occlusion, such as eyeglasses, hair and head decorations.

On the other hand, it is caused by the face target external change, also including two aspects.

- (1) Face has more attitude caused by different imaging angle, such as plane rotation, depth rotation and up and down rotation, which has a greater influence on the depth rotation;
- (2) The influence of illumination, such as the brightness of the image, the change of contrast and the shadow.
- (3) The imaging conditions of the image, such as the focal length of the camera and the imaging distance.

The first stage of this project is to perform face detection, which aims to find face landmarks in a given image. In this area, there are so many methods proposed.

## 2.2.1 Viola-Jones face detector and AdBoost algorithm [1]

### 2.2.1.1 Introduction

The Viola-Jones face detector is one of the most effective methods to perform face location. The rectangle is used as the eigenvector of face detection, which is called the rectangular feature. The algorithm selects the simplest five rectangular feature templates for training, in order to get the most appropriate rectangle features of face detection. Results demonstrating that although training speed of this feature selection is not fast, the efficiency of it is very high have not previously been presented. Viola proposed that integral image might be applied to the calculation of eigenvalues [7]. The usage of the integral image can only perform a traversal calculation of the image so that the calculation of each eigenvalue can be completed with constant time, which greatly improves the speed of training and detecting.

### 2.2.1.2 Rectangle Feature [1]

Under the condition of given limited data, feature-based testing can encode the state of a particular region, and the speed of the feature-based system is much faster than that of the pixel-based system.

Rectangular features are sensitive to simple graphic structures such as edges and segments. But for a structure that can only describe a particular direction, such as horizontal, vertical, and diagonal, it is not sensitive.

As shown in Figure 1, some features of the face are simply presented by the rectangular features. For example, the color of eyes area is darker than that of the cheeks area, the color of ala nasi is darker than that of apex nasi, and the color of mouth is darker than that of mouth surrounding.



Figure 1: The features selected by Adaboost [7]

### 2.2.1.3 Feature Template [7]

The simple rectangle combination would be used as our feature template. This kind of feature template is made up of more adjacent congruent rectangles, feature template have is two kinds of rectangular, which are white rectangular and black one respectively. Eigenvalues of feature template would be defined the sum of white rectangular pixels minus that of black rectangle pixels. The feature templates are shown in Figure 2.




Template	Feature Template
Side Feature	
Line Feature	
Particular Direction Feature	

Table 1: Harr-like features

#### 2.2.1.4 The total number of features in the detector

For  $m$  by  $m$  pixel resolution detector, the total number of rectangles that satisfy a given condition is calculated in following way.

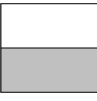

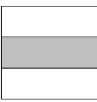
For the  $m$  by  $m$  sub-window, we only need to determine the left upper vertex and the right lower vertex of the rectangle, that is, to determine a rectangle. The rectangle also meet the following two conditions which are namely called  $(s, t)$  condition, the rectangle satisfying  $(s, t)$  condition is called the conditional rectangle:

1. The length of  $x$  direction must be divisible by the natural number  $s$ , i.e. it is divided into  $s$  segments equally.
2. The length of  $y$  direction is divisible by the natural number  $t$ , i.e. it is divided into  $t$  segments equally.

The minimum area and maximum area of the rectangle are  $s \times t$  or  $t \times s$  and  $\left\lceil \frac{m}{s} \right\rceil \cdot s \times \left\lceil \frac{m}{t} \right\rceil \cdot t$  or  $\left\lceil \frac{m}{t} \right\rceil \cdot t \times \left\lceil \frac{m}{s} \right\rceil \cdot s$  respectively, where  $\lceil \cdot \rceil$  is integral operator.

#### 2.2.1.5 Number of feature rectangles in sub-windows

For example, for 24 by 24 sub-window, we calculate the total number of features as shown in table.

Feature template	The number of feature
 1,	$2 \cdot \Omega_{(1,2)}^{24} = 2 \cdot \left( \left\lceil \frac{24}{1} \right\rceil + \left\lceil \frac{23}{1} \right\rceil + \dots + \left\lceil \frac{2}{1} \right\rceil + 1 \right) \cdot \left( \left\lceil \frac{24}{2} \right\rceil + \left\lceil \frac{23}{2} \right\rceil + \dots + \left\lceil \frac{4}{2} \right\rceil + 1 \right)$ $= 2 \times (24 + 23 + \dots + 2 + 1) \times (12 + 11 + 11 + \dots + 2 + 1 + 1)$ $= 2 \times 300 \times 144$ $= 86400$
 2,	
 3,	$2 \cdot \Omega_{(1,3)}^{24} = 2 \cdot \left( \left\lceil \frac{24}{1} \right\rceil + \left\lceil \frac{23}{1} \right\rceil + \dots + \left\lceil \frac{2}{1} \right\rceil + 1 \right) \cdot \left( \left\lceil \frac{24}{3} \right\rceil + \left\lceil \frac{23}{3} \right\rceil + \dots + \left\lceil \frac{4}{3} \right\rceil + 1 \right)$ $= 2 \times (24 + 23 + \dots + 2 + 1) \times (8 + 7 + 7 + \dots + 1 + 1 + 1)$ $\equiv 2 \times 300 \times 92$ $= 55200$


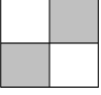
4, 	
5, 	$\Omega_{(2,2)}^{24} = \left( \left\lfloor \frac{24}{2} \right\rfloor + \left\lfloor \frac{23}{2} \right\rfloor + \dots + \left\lfloor \frac{2}{2} \right\rfloor + 1 \right) \cdot \left( \left\lfloor \frac{24}{2} \right\rfloor + \left\lfloor \frac{23}{2} \right\rfloor + \dots + \left\lfloor \frac{4}{2} \right\rfloor + 1 \right)$ $= (12 + 11 + \dots + 1 + 1) \times (12 + 11 + \dots + 1 + 1)$ $\equiv 144 \times 144$ $= 20736$
Total number of features	$\Omega^{24} = 86400 + 55200 + 20736 = 162336$

Table 2: Total number of features

### 2.2.1.6 Integral Image [1, 8]

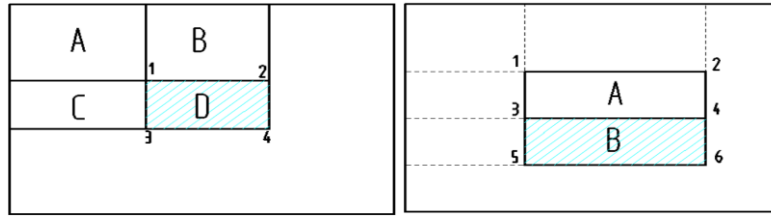


Figure 2: (a) Sum of the pixels of rectangle D (b) Eigenvalues of feature template

The calculation of integral image in image area use following formula and shown in Figure 2 (a).

$$\text{The pixels in rectangle D} = ii_1 + ii_4 - (ii_2 + ii_3)$$

Where

$ii_1$  = the sum of the pixels in rectangle A

$ii_2$  = the sum of the pixels in rectangle A and rectangle B

$ii_3$  = the sum of the pixels in rectangle A and rectangle C

$ii_4$  = the sum of the pixels in rectangle A, rectangle B, rectangle C, and rectangle D

Eigenvalue of rectangular feature can be calculated by means of above method in Figure 2 (b).

$$\text{The pixels in rectangle A} = ii_1 + ii_4 - (ii_2 + ii_3)$$

$$\text{The pixels in rectangle B} = ii_3 + ii_6 - (ii_4 + ii_5)$$

According to the definition that eigenvalues of feature template is the sum of white rectangular pixels minus that of black rectangle pixels. Eigenvalues of feature template is the pixels in rectangle A minus the pixels in rectangle B. Thus, Eigenvalues of feature template =  $(ii_4 - ii_3) - (ii_2 - ii_1) + (ii_4 - ii_3) - (ii_6 - ii_5)$ .

Therefore, eigenvalue calculation of the rectangular feature is related to the integral image of

this feature endpoint, and has nothing to do with the image coordinates. Moreover, the calculation of the eigenvalues is time constant, and it is only simple add and subtraction operation. Because of this, the introduction of integral image greatly improves the speed of detection.

### 2.2.1.7 AdaBoost training algorithm [7]

AdaBoost algorithm is an iterative algorithm for a set of training set. By changing the distribution probability of each sample, it obtains a different training set. For each training set, a weak classifier would be obtained by performing procedure of training. Finally, a strong classifier can also be obtained by these weak classifiers through combination of different weights.

The algorithm of AdaBoost training classifier is described as follows.

1. A series of training examples  $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ , where  $y_i = 0, 1$  for negative and positive examples respectively.  $n$  is the total number of training examples.

2. Initialize weights,  $w_{1,i} = D(i)$

$D(t) = \frac{1}{2m}$  (negative examples) or  $\frac{1}{2l}$  (positive examples), where  $m$  and  $l$  are the number

of negative and positive examples,  $m + l = n$

3. For  $t=1, \dots, T$ :

(1) Normalize the weights,

$$q_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^n w_{t,j}} \quad (1)$$

(2) For each feature  $f$ , train a weak classifier  $h(x, f, p, q)$ ; The weighted ( $q_t$ ) error  $e_f$  is calculated:

$$e_f = \sum_i q_i |h(x_i, f, p, q) - y_i| \quad (2)$$

(3) Choose the weak classifier  $h_t(x)$ , with the lowest error  $e_t$

$$e_t = \min_{f,p,q} \sum_i q_i |h(x_i, f, p, q) - y_i| = \sum_i q_i |h(x_i, f_t, p_t, q_t) - y_i| \quad (3)$$

$$h_t(x) = h(x, f_t, p_t, q_t) \quad (4)$$

4 Update the weights based on the weak classifier

$$w_{t+1,i} = w_{t,i} b_t^{1-e_i} \quad (5)$$

Where  $e_i = 0$  represents  $x_i$  is classified correctly, otherwise,  $e_i = 1, b_t = \frac{e_t}{1-e_t}$ .

4. The final strong classifier is:

$$C(x) = \begin{cases} 1 & \sum_{t=1}^T \alpha_t h_t(x) \geq 0.5 \sum_{t=1}^T \alpha_t \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

Where  $\alpha_t = \log \frac{1}{b_t}$

### 2.2.1.8 Cascade classifier

Cascade classifier would be proposed by Paul Viola and Michael Jones [10]. This method applies integral image to calculate Haar-like [11] feature and then some weak classifiers generated by Harr-like feature using AdaBoost algorithm will be merged to the strong classifier.

For example, there are more than 160,000 rectangular features are associated with each image of 24x24 base window. Even though we can calculate each feature effectively, calculating the complete set is pretty costly. The pivotal key is to find a small set of such features, which can be merged forming an efficient classifier. Degenerate decision tree, as the final form of the detection process, is called as a cascade. The Cascade of classifiers is as follows: The final form of the detection process is a degenerate decision tree, which can be called as a cascade [12] (Figure 3). Determining that whether or not a given sub-window is a face or definitely not a face is the task of each stage in the cascade. It performs execution of next classifier if the previous classifier gives the positive outcome. However, it leads to instant rejection of the corresponding sub-window if the outcome is negative, i.e. a non-face.

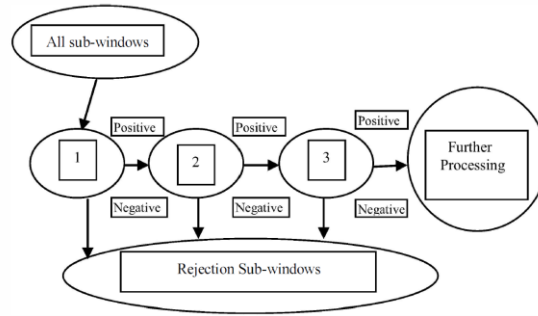


Figure 3: the process of cascade working [13]

Training algorithm of cascade detector are shown as follow:

1. Setting maximum misclassification error rate  $f$  and minimum pass rate  $d$  of each stage and target misclassification error rate  $F_{target}$  of the whole detector. Positive sample set POS and negative sample set NEG are given.
2. Initialize  $F_1 = 1, i = 1$
3. While  $F_i > F_{target}$ 
  - a. The  $i$  th stage is trained by positive sample set and negative sample set. Misclassification error rate is less than  $f$  and pass rate is greater than  $d$
  - b.  $F_{i+1} \leftarrow F_i, i \leftarrow i + 1, NEG \leftarrow \emptyset$
  - c. If  $F_{i+1} \leftarrow F_{target}$ , non-face image is detected by cascade detector. NEG is collected.

### 2.2.2 Skin color likelihood method [14]

By using skin color information, there is an approach proposed, which referred to boosting-based face detection. It is not only for skin color segmentation but also for skin color emphasis, also not containing any parametric fitting or morphological operation.

### 2.2.2.1 Skin likelihood

The YCbCr space [15] can be obtained by simple matrix operation from the RGB space. The conversion (RGB  $\rightarrow$  YCbCr) is shown in following equation.

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.168 & -0.331 & 0.5 \\ 0.5 & -0.418 & -0.081 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \quad (7)$$

Luminance and chrominance are separated by the YCbCr space so that the skin distribution cluster is compactness. A relatively tight cluster in color space can be formed by human skin, so probability of skin color learnt by using the chromaticity information of YCbCr space is pivotal. The color of skin can be emphasized via the probability of skin color, while the color that does not can be ignored. By applying Bayes rule to each pixel of an image, we can obtain probability of skin color by means of two histograms [16]. The probability of skin given (Cb, Cr) color vector is described as:

$$p(\text{skin}|Cb, Cr) = \frac{p(Cb, Cr|\text{skin}) \times p(\text{skin})}{p(Cb, Cr)} \quad (8)$$

where:

$h_{\text{skin}}(Cb, Cr)$ : Histogram of skin colors in an image

$h_{\text{total}}(Cb, Cr)$ : Histogram of entire colors in an image

$N_{\text{skin}}$ : Sum over Cb and Cr of  $h_{\text{skin}}(Cb, Cr)$

$N_{\text{total}}$ : Sum over Cb and Cr of  $h_{\text{total}}(Cb, Cr)$

Ratio between normalized histograms of skin color and non-skin color which are based on manually labeled ground truths of skin pixels can describe the probability of skin color given (Cb, Cr) color vector.

### 2.2.2.2 Skin boosted classifier

The skin boosted classifier aims to select complementary weak classifiers and then simultaneously to determine the weights of skin color information emphasized while non-skin color information [10, 17] is reduced. We can describe the process of training scheme of skin boosted classifier.

1. Apply Bayesian lookup table  $x_n'^{[p]} = f_2(x_n^{[p]}, T) \forall n = 1, 2, \dots, N \forall p = 1, 2, \dots, P$
2. Start with initial weight  $w_i = \frac{1}{N}, i = 1, 2, \dots, N, H_{\text{skin}}(x') = 0$
3. Repeat for  $m = 1, 2, \dots, M$ 
  - (a) Fit the regression function by weighted least squares fitting of Y to  $X'$
  - (b) Update  $H_{\text{skin}}(x') \leftarrow H_{\text{skin}}(x') + a_m h_m(x')$
  - (c) Update  $w_i \leftarrow w_i e^{-y_i a_m h_m(x')}$  and normalization
4. Output the skin strong classifier



$$H_{skin}(x') \begin{cases} 1 & \text{when } \sum_{m=1}^M a_m h_m(x') \geq \frac{1}{2} \sum_{m=1}^M a_m \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

Training examples are  $(x_1, y_1), \dots, (x_N, y_N)$ , class label of  $x_i \in R^n$  is  $y_i \in \{+1, -1\}$ ,  $H_{skin}(x')$  is skin strong classifier,  $h_m$  and  $a_m$  are weak classifier and weight respectively.

The total number of combinational features can be described as  $M$

### 2.2.2.3 Performance measure

Face detection result can be executed by using use overlap measure  $A$  [18] in the aspect of performance measure. It is shown in following equation.

$$A = \frac{|A_g \cap A_c|}{\sqrt{|A_g| \times |A_c|}} \quad (10)$$

$A_g$  and  $A_c$  are ground truth bounding box and computed bounding box respectively.  $|A_g|$  and  $|A_c|$  are area of the bounding box in number of pixels.

Thus, boosting-based face detection combined with iterative boosting algorithm is designed and implemented by skin color likelihood, which emphasizes skin color information instead of non-skin color information. Besides, total error rate has been reduced to half of original error rate and a small part of training cascade stages and processing rate will be maintained.

### 2.2.3 Face detection with graphical user interface (GUI)

A user interface involved in graphical objects containing buttons, text fields, sliders and menus, is referred to graphical user interface (GUI) [19]. There are some steps for building mechanics of GUIs with MATLAB: (1) Lay out the components; (2) Program these components to do specific things in response to user actions; (3) Save and launch the GUI

Two basic tasks in the process of implementing a GUI: (1) Lay out the GUI components; (2) Program the GUI components

GUIDE as a set of layout tools, can generate an M-file that includes codes which are to handle the initialization of the GUI. Moreover, implementation of the function (callbacks) that control and launch GUI relies on a framework supplied by M-file.

The detailed implemented procedure will follow following steps:

- (1) Write M-file that includes all the commands to lay out a GUI
- (2) Lay out the components by using GUI
- (3) Generate two files that save and launch the GUI
- (4) Set FIG-file as saving file
- (5) Let FIG-file include:
  - a. The whole description of the GUI
  - b. All of its children (uicontrols and axes)
  - c. The values of all object properties

(6) Set M-file as launching file

- (7) Let M-file include: a. sub functions that launch and execute the GUI  
b. callbacks

M-file is referred to application M-file that does not contain the code of laying out the uicontrols. In such case, the code is saved in the FIG-file. Given a face image, face recognition aims to know whether or not there is any face in image and its detailed position. Towards this goal, face detection will be implemented using Viola-Jones Algorithm and Cascade Architecture (2.1.1).

## 2.3 Face alignment

### 2.3.1 Face alignment across large poses: 3D Dense Face Alignment (3DDFA) [33]

3D Dense Face Alignment (3DDFA) which refer to a novel method, solves the problem of face alignment across large poses. The algorithm framework is shown in Figure 4.

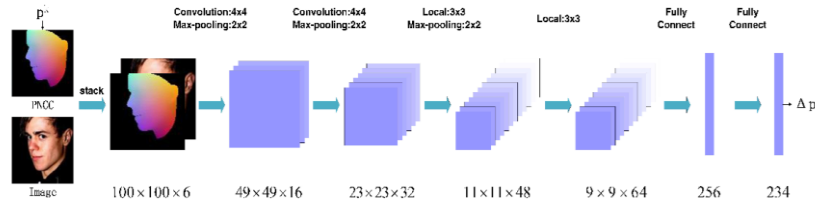


Figure 4: Algorithm framework of 3DDFA [33]

Training the 3DDFA model requires a large number of multi-gesture face samples. To this end, based on existing datasets, we utilize 3D virtual information to generate face image with a different attitude. The core idea is to predict the depth information of the face image, and then to generate face image under the condition of different attitude through the 3D rotation, the results are shown in Figure 5 below:

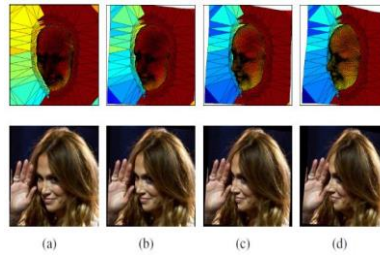


Figure 5: 2D and 3D view of the image rotation [33]

- (a) The original yaw angle  $\text{yaw}_0$  (b)  $\text{yaw}_0 + 20^\circ$  (c)  $\text{yaw}_0 + 30^\circ$  (d)  $\text{yaw}_0 + 40^\circ$

### 2.3.2 Large-Pose Face Alignment via CNN-Based Dense 3D Model Fitting [34]

The author tries to solve the problem of facial feature location by using 3D face modeling. The 2D face shape ‘U’ can be seen as the 3D face shape ‘A’ through the projection to change m, as shown in Figure 6 below:

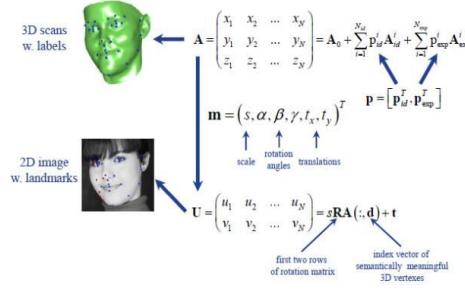


Figure 6: Transformation process [34]

The overall frame diagram of the algorithm is as follows:

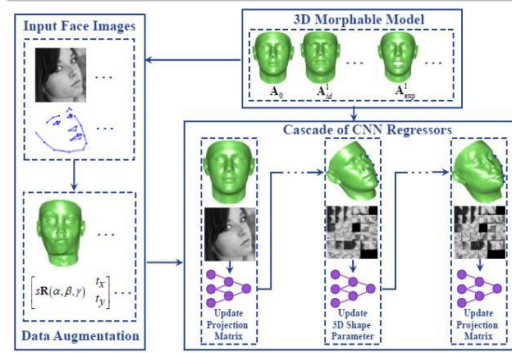


Figure 7: Frame diagram of the algorithm [34]

### 2.3.3 Unconstrained Face Alignment via Cascaded Compositional Learning [35]

The method of Cascaded Compositional Learning (CCL) without starting from the 3D face modeling to solve the problem of face alignment, all face samples are divided into multiple domains to deal with these samples respectively. The starting point of the method is similar to GSDM [36], GSDM relies on the face alignment result of the previous frame in a video to select the domain, so it cannot handle the face alignment problem of static images. The schematic diagram of CCL algorithm is shown in Figure 8:

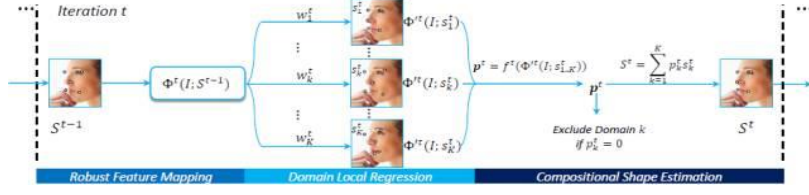


Figure 8: Schematic diagram of CCL algorithm [35]

The overall framework of the algorithm is a cascade shape regression with three blocks per level, which are feature extraction module, shape regression module and combinatorial coefficient prediction module. Feature module extraction on the basis of LBF [37] introduces feature point is visible information, provide important clues for subsequent prediction combination coefficient.

## 2.4 Face recognition using PC/BC-DIM algorithm [4]

### 2.4.1 Introduction

Predictive coding [4] (PC) can be seen a description of the hierarchical perceptual inference process performed in the cortex. It strongly has an impact on the theory of cortical information processing to perform perceptual inference and is achieved as a hierarchical neural network. Nevertheless, the capacity of accomplishing the complex inference desired to recognize face in natural images of PC has not been proposed previously. At the level of functional and neurophysiological, it is suitable to simulate object recognition by using PC. However, up to now, there is no any demonstration to prove PC explicitly. PC can perform object recognition in natural images. Specifically, a particular version of PC which referred to the PC/BC-DIM algorithm, can perform identification and recognition.

As another version of PC, PC/BC-DIM is compatible with Biased Competition (BC) theories created by cortical function. And BC is accomplished by Divisive Input Modulation (DIM), which referred to a method used to update error and prediction neuron activations. By using DIM, reconstruction errors will be calculated using division instead of subtraction compared with other executions of PC. The divisive method is mainly focused on non-negative firing-rates and becomes therefore more biologically-plausible.

By using iteration, an error will be minimized between the expected sensory inputs predicted by the causes and the sensory data. The inference process performs “explaining away” [38, 39, 40, 41, 42]. The process of using a two-stage hierarchical neural network model to perform experiments are shown in Figure 9 (a). PC/BC-DIM algorithm would calculate the activations of the neurons in both stages.

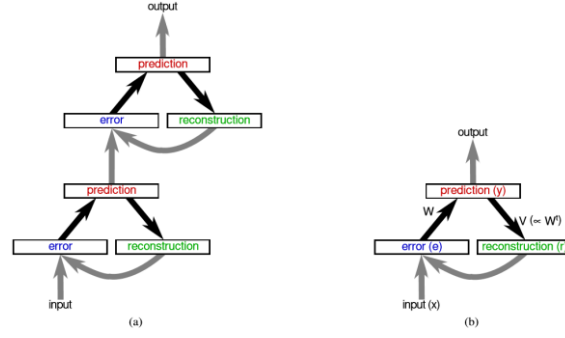


Figure 9 Frame of hierarchical PC/BC-DIM network [4]

(a) The two-stage hierarchical PC/BC-DIM network used in the simulations. (b) In each processing-stage, the population of prediction neurons constitute a model of the input environment of that processing-stage.

In the first processing-stage, the training procedure contains image patches extracted from the grayscale training images and then these patches were clustered to form a dictionary. In the second processing-stage, the procedure contains prediction neuron responses calculated for all the images in the training set of first-stage. According to the responses of the first-stage prediction neurons, weights of the second-stage would be defined. The hierarchical PC/BC-DIM model is applied to recognize face images. The test images pre-processed are as input to the first processing-stage, the PC/BC-DIM algorithm is performed for first stage. And then the first-stage prediction neuron responses are as inputs to the second processing stage, and the PC/BC-DIM algorithm is also performed for second stage. Finally prediction neuron responses of the second-stage are applied to recognize the to-be-recognized face images. Previously on face dataset applied to face recognition by using PC/BC-DIM algorithm, half of face images were used for training set and the other half for testing set.

## 2.4.2 PC/BC – DIM algorithm

The PC/BC-DIM algorithm can be implemented by using convolution or matrix multiplication. The mathematical and theatrical foundation of matrix-multiplication can be performed using the following equations and the corresponding implementations are shown in Figure 9 (b):

$$r = Vy \quad (11)$$

$$e = x \oslash [r]_{\epsilon_2} \quad (12)$$

$$y \leftarrow [y]_{\epsilon_1} \odot We \quad (13)$$

Symbols	Explanations
$x$	$m \times 1$ vector of input activations
$e$	$m \times 1$ vector of error neuron activations
$r$	$m \times 1$ vector of reconstruction neuron activations
$y$	$n \times 1$ vector of prediction neuron activations
$W$	$n \times m$ matrix of feedforward synaptic weight values
$V$	$m \times n$ matrix of feedback synaptic weight values
$[v]_{\epsilon}$	$\max(\epsilon, v)$
$\epsilon_1$ & $\epsilon_2$	parameters

$\oslash$	element-wise division
$\odot$	multiplication
$\leftarrow$	left-hand side of the equation is assigned the value of the right-hand side

Table 3: Symbols and the corresponding explanations of formula of matrix-multiplication  
Also, the convolution can be performed using the following equations:

$$R_i = \sum_{j=1}^p (V_{ji} \star Y_j) \quad (14)$$

$$E_i = X_i \oslash [R_i]_{\epsilon_2} \quad (15)$$

$$Y_j \leftarrow [Y_j]_{\epsilon_1} \odot \sum_{i=1}^k (w_{ji} \star E_i) \quad (16)$$

Symbols	Explanations
$X_i$	Two-dimensional array representing channel $i$ of the input
$R_i$	Two -dimensional array representing the network's reconstruction of $X_i$
$E_i$	Two -dimensional array representing the error between $X_i$ and $R_i$
$Y_j$	Two -dimensional array that represent the prediction neuron responses for a particular class $j$ of prediction neuron
$w_{ji}$	Two -dimensional kernel representing the feedforward synaptic weights from a particular channel, $i$ , of the input to a particular class, $j$ , of prediction neuron
$v_{ji}$	Two -dimensional kernel representing the feedback synaptic weights from a particular class, $j$ , of prediction neuron to a particular channel, $i$ of the input
$\star$	cross-correlation

Table 4: Symbols and the corresponding explanations of formula of convolution

## 2.5 Summary

This chapter provided the necessary background to the implemental processes and methods of face detection, feature extraction, and face recognition. Especially it also showed the detailed definition of integral image and cascade classifier of AdaBoost algorithm and how they work and achieve objectives of locating landmarks of face, the advanced technology for face alignment in the context of this project. A review of works related to this project was also presented which revealed that how hierarchical perceptual inference process performed in the cortex and how PC/BC-DIM algorithm works in improving face recognition experiments.

# 3 Chapter 3

## 3. System Design and Implement

### 3.1 Overview

The previous chapter gave a detailed background overview on the subject of face detection, face alignment and face recognition. It mainly delved into the subject of face detection and what makes them become one of the most important parts in this project. It then presented related works in feature extraction and face alignment and how feature extraction has been used in face alignment. Noticeable absent within the review is the use of image processing toolbox and inbuilt function in MATLAB in improving face alignment, which referred to an issue this project seeks to address. After evaluating various methods of face detection, this chapter describes the design of an automatic method detailing what the project seeks to achieve, and presents how the design components were implemented

### 3.2 Face detection

#### 3.2.1 Face detection using Viola-Jones Algorithm

AdaBoost is the abbreviation of the Adaptive Boosting. This algorithm is different from Boosting algorithm. Boosting algorithm needs to know the lower limit of error rate in advance, however, AdaBoost have capacity of adjusting the error rate by means of weak learning feedback. That is to say, the AdaBoost algorithm does not need any prior knowledge about the weak learning performance, so it can be easily applied to practical problems.

We can implement face detection by using the inbuilt function of MATLAB, i.e. `vision.CascadeObjectDetector`. Based on theory of chapter 2, this approach is used to improve running time of program and detection rate.

In such case, the computation speed of current detection system is about 15 times faster than that of any previous detection system. I have applied my photo and image of ORL as training images to test this detection system, the results are as shown in Figure 10.



Figure 10: Results of Face detection using Viola-Jones Algorithm

AdaBoost, thus as a traditional and effective identification method based on the iteration of classifier and cascade detector, can accurately detect the face landmarks, which can be used as a method in face alignment.

### 3.2.2 Face detection using skin color likelihood

This method mainly applies image color space conversion, threshold segmentation and morphological processing to image detected.

The purpose of threshold segmentation is to separate the skin from the image background. The goal of image binary processing is to separate the skin area from the rest of the region. The skin area is represented by "1" (white), and "0" (black) in other areas.

By locating the face by means of connected domain, i.e. 'white area'. By searching maximal connected domain of the face and fixing length ratio condition ( $< 1.8$ ), this algorithm will find the largest external rectangle as detection results.

Implement process of detection system is shown as follows:

The detection system is first to read and display original face image and then to create color transformation structure. And it applies device-independent color space to execute transformation from RGB space to LAB space and displays the image of lab space. The a component of image is extracted from image of LAB space and is displayed. A global threshold that can be used to convert an intensity image to a binary image will be computed.

Based on above threshold calculated by previous step and display result, images are converted to the binary image. Similarly, the b component of the image is also processed following above steps. The two binary images created by a and b component are performed by 'and' operation.

Connected components within the two-dimensional binary image are labeled. Parameters related to the minimum external rectangular within each connected domain will be found, including the top left vertex coordinates (x, y) of the rectangle, and the width and height of the rectangle.

At the same time, the external rectangle structure is converted to cell structure in such case, the



number of connected domains is equal to that of cells. Then the cell type data is converted to mat type data that holds all of the external rectangular parameters within the connected domains in which each of four consecutive data is corresponding to an external rectangular parameter of connected domain (x, y, W, H).

Finally, the number of parameters of external rectangle and the area of the  $k_{th}$  external rectangle will be calculated to determine whether or not the area of current connected domain is larger than the maximum area of previous connected domain and length ratio condition ( $< 1.8$ ). The largest area of external rectangle that meets the requirement of ratio will be recorded as the final detection result.

Detection of face area on the basis of skin color and establishing bounding box are the main procedure. My photo is tested as training image using above method and its results are shown in Figure 11.

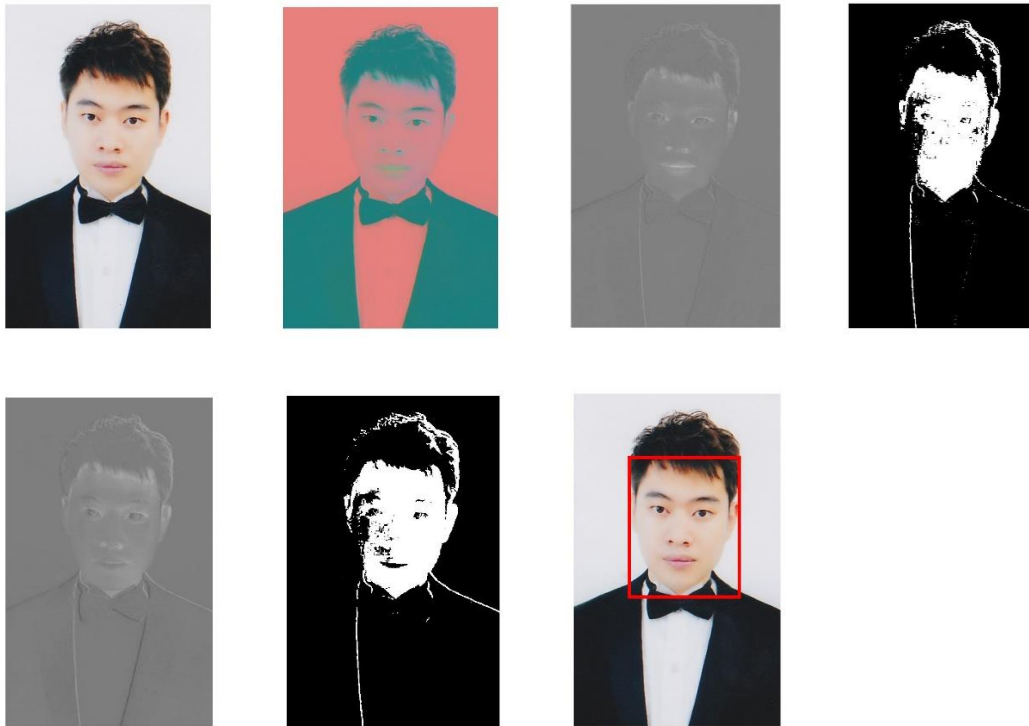


Figure 11: Result of skin color method

In my opinion, although face detection method based on skin color likelihood has a high recognition rate, it cannot be able to detect feature points of human face. By searching the biggest morphological area as the main object, it achieves face detection. This is a major reason why facial alignment cannot be performed.

### 3.2.3 Image edge detection and segmentation

The first input image is to find the pixels where the value of R, G, B, H, S, V, Y, Cb, and Cr satisfy the value of the pixels of the trained image. Output sets to 1 if it satisfies, otherwise sets

to 0.

After then we have to apply various necessary morphological operations to output image. Moreover, filling any holes [21] or gaps in the individual blobs will use hole filling operation.

The image is subtracted from the main binary image so as to open the narrowly connected blobs and then the narrowly connected regions are separated. As loss in size of the individual blobs leaded by subtraction, a particular size of the skin detected individual blobs are maintained using a dilate function [22] applied by the final morphological operation.

Additional region properties are introduced to determine whether likelihood of a skin region is a face region [23, 24, 25]. Box ratio is used as the width to height ratio of the region bounding box and to examine and classify the shape of each skin region.

The main procedure of performing face detection are:

- (1) Input image
- (2) Skin region segmentation
- (3) Morphological operations
- (4) Region labeling
- (5) Provable face region
- (6) Cropping face region
- (7) Marking face detection

This method is similar to previous method to complete edge detection [4] and traversal process. We divide face images into 100 image patches to execute image edge detection, assuming that the size of each one patch is  $r \times c$ .

The ratio related to points of black background of the image corresponding to the binary image will be calculated in the most peripheral image blocks. If, the image patch is set to 0 when the ratio is less than the given threshold in which the threshold is set to 100, its color is thus black, otherwise, is set to 1.

Therefore, the color of a peripheral image of the binary image processed is black. By searching maximal connected domain of the face and fixing length ratio condition ( $< 1.8$ ), this algorithm will find the largest external rectangle as detection results.

Implement process of detection system is shown as follows:

Firstly, after calculating the size of image, this detection system judges whether or not the current image patch is the peripheral image block and then locates the black point coordinates in the current image block.

Secondly, the number of black points in the current image patch and the proportion of black regions are calculated in order. This detection system can also determine if the ratio of black regions is less than the given threshold and then assign the current image block, as well as

display the binary image processed.

Furthermore, connected components within the two-dimensional binary image are labeled. Parameters related to the minimum external rectangular within each connected domain will be found, including the top left vertex coordinates  $(x, y)$  of the rectangle, and the width and height of the rectangle.

At the same time, the external rectangle structure is converted to cell structure in such case, the number of connected domains is equal to that of cells. Then the cell type data is converted to mat type data that holds all of the external rectangular parameters within the connected domains in which each of four consecutive data is corresponding to an external rectangular parameter of connected domain  $(x, y, W, H)$ .

Finally, the number of parameters of external rectangle and the area of the  $k$  th external rectangle will be calculated to determine whether or not the area of current connected domain is larger than the maximum area of previous connected domain and length ratio condition ( $< 1.8$ ). The largest area of external rectangle that meets the requirement of ratio will be recorded as the final detection result. I have applied some face images as training images to test this detection system and results are shown in Figure 12.



(a) Correctly detect face area



(b) Inaccurately detect face area

Figure 12: Results of image edge detection and segmentation

From my perspective, face detection method based on image edge detection and segmentation has a lower recognition rate than that of AdaBoost algorithm and skin color likelihood. By searching the biggest morphological area as the main object, it achieves face detection.

However, in the process of edge detection and image segmentation, due to traversing all the feature points in the image, in some degree, the actual color of the face region is same or similar with that of background, so these will cause led to an inevitable error of the final result.

The detection conditions of the face area is the maximum long - width ratio in morphological region, the area of final result detected is larger than that of actual face result. This method cannot accurately complete face detection, so this is a major reason why facial alignment cannot be performed.

### **3.2.4 Face detection via eye template matching**

As we known, the skin color likelihood method may not perform face detection for grayscale image accurately, so other methods should be applied for face detection on grayscale image rather than RGB image in this project. In this section, face detection based on detecting eye position will be evaluated and makes comparison with others.

This method is mainly performed by locating eye position to find face landmarks in the following steps:

1. Define a rectangular area with the top left vertex  $(i, j)$ , this area is the same size as eye template.
2. Read image of defined rectangular area in original image.
3. Compute the correlation coefficient between image in rectangular area and eye template.
4. Record correlation coefficient of rectangular area corresponding to point  $(i, j)$ .
5. Extend the area of testing to get final face image.
6. Compute the boundary of the face area.
7. Mark the face region.

First of all, in face image, we need to find the area that is similar to eye template image. Point  $(i, j)$  is designed to upper left corner of vertices. Width and height of rectangle (i.e. eye template image) are described respectively.

And then, rectangle that is the same size of eye template will be cropped in face image and correlation coefficient will be computed between them. Finally, we will change the value of the vertex coordinates  $(i, j)$ , and implement the traversal on the entire image to solve correlation coefficient between rectangular image and eye template in each position.

In terms of the correlation coefficient above, in next step, we need to find upper left vertex coordinates  $(i, j)$  in rectangle which has the most similar area to eye template. We need to initialize the optimal solution and then Perform all the points to solve coordinate  $(x, y)$  that has the highest correlation coefficient and maximum value of the correlation coefficient. According

to above tests, we can solve the coordinate of eye region (x, y, py1-1, px1) and then locate final face result based on locating eye region.

The way to get final face image is to extend the area of the test, extending the width of the rectangle frame of eye region to three times. Then the vertices of the polygon of the face area are defined. The program is defined through a rectangular face region, so there are four points (bx(1),by(1)),((bx(2),by(2)),((bx(3),by(3)),((bx(4),by(4)) .

Computing the boundary of the face area (i.e. the border of the polygon), we firstly mark the face area as BW=roipoly(q,bx,by).Face regions are marked as 1, the rest are marked as 0 in BW. The face area will be magnified in a pixel to get the polygon vertex bx1 and by1, then we mark up the enlarged face area to get BW1=roipoly(q,bx1,by1) where q is input grayscale face image. Then according to BW2 = BW1 - BW, the original face area is marked as 0 and the outermost circle pixel in enlarged area is marked as 1.

K(x, y) denotes the obtained edge of the image where, K(x, y) =1 if (x, y) is an edge pixel and otherwise, K(x, y) =0. Each column x and each row y can be calculated as V(x) and H(y) respectively [43,44].

For vertical direction:

$$V(x) = \sum_{y=0}^{N-1} K(x, y)$$

For horizontal direction

$$H(y) = \sum_{x=0}^{M-1} K(x, y)$$

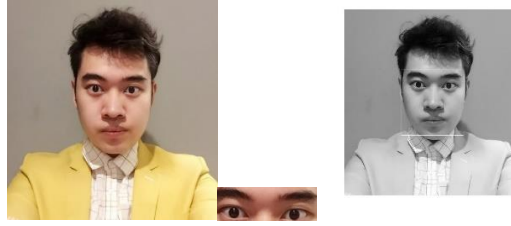
The marking on the face bounding box of the input image is determined. The width of this image (in pixels) is defined by [45]:

$$bb_{xw} = \frac{zy\_zy}{2 * pupil\_se} * d_{GT}$$

Symbol	Description
$d_{GT}$	distance (in pixels) between both center of the eyes
$zy\_zy$	mean width of a human face
$pupil\_se$	half of the inter-pupil distance

Table 5: Explanations of formula components

If the value of  $y\_up$  is equal to  $pupil\_se$ , the position of the bounding box can be computed. Finally, we convert images to floating-point data and add the border to the image to display test result. And then I use my photos to test this method. The results are shown in Figure 1. However, this accurate rate of detection depends on the shape of eye template, it will not work well if we cannot choose an accurate template.



(a) My photo



(b) FDDB database

Figure 13: Results of eye template matching

I conclude that this cannot be used to develop an automatic method for face alignment. The reason is original image and eye template must be supplied to face detection system at the same time in the running process.

Moreover, the eye template image must be extracted from the original image to complete the subsequent traversal process. For a given image, we assume that two images have been extracted from that given image with different sizes, i.e. image 1 and image 2. If eye template of image 1 matched with image 2, there is no way to complete the traversal and detection process.

The limitations of this approach are not likely to be useful as a follow-up to face alignment, which can be seen the biggest drawback for creating an automatic system. It is also complicated if the requirement is only to locate landmarks of face. Thus, it only can be an investigated and evaluated part in project.

### 3.2.5 Robust Real Time Face Detection

This program contains many sub-functions, which are written by D.Kroon University of Twente (November 2010). [5]

This program mainly focus on Haar-cascade classifier to perform face detection by using AdaBoost algorithm and cascade classifier to gradually narrow the range of facial features so as to find the most accurate facial features. Testing image and results are shown in Figure 14, 15 and 16.



Figure 14: Results of real time face detection by ORL dataset



Figure 15: Results of real time face detection by FDDB dataset



Figure 16: Results of real time face detection by my photo

In summary, this method by narrowing area detected to locate face landmarks, has a higher detection rate for performing face detection. Central idea is to use AdaBoost algorithm, and to find the number of objects detected through some constraints in order to locate the face. It can be seen a very effective method and be used to execute face alignment.

### 3.2.6 Face detection based on computer vision system toolbox

This program: Copyright (c) 2014, Masayuki Tanaka, all rights reserved.

This code is implemented based on CascadeObjectDetector of FrontalFaceCART, LeftEye, RightEye, Mouth, and Nose. I have used FDDB dataset to test this face detection system, result are shown in Figure 17 and 18.

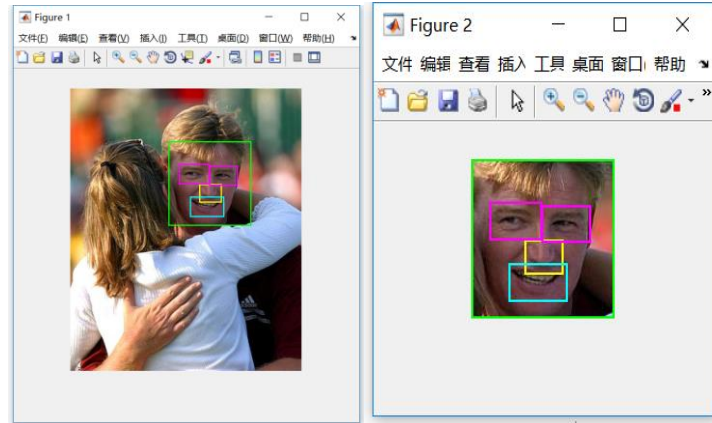


Figure 17: Face and its landmarks



Figure 18: Results of face detection using toolbox

This code uses a rectangle to mark the position of detected face, eyes, nose, and mouth. So I modify some of codes so as to delete these rectangles and extract directly landmarks of face from original image.

One of the sub-functions is `detectFaceParts`. Input parameters include the detection object built by other sub function `buildDetector`, image data that is `uint8` and found faces with boxes stored as cell array. Output parameters contain bounding box for face, left eye, right eye, mouth and nose, image with found faces which are shown as boxes, found faces with boxes stored as cell array and found faces stored as cell array. This sub-function is to detect faces with parts and is usually used with another sub-function.

Another sub-function is `buildDetector` that builds face parts detector object. Its input parameters consist of `MergeThreshold` for face detector, `MergeThreshold` for face parts detector and size of normalized face. Output parameter is built detector object.

Example:

```
detector = buildDetector();
img = imread('img.jpg');
[bbbox, bbimg] = detectFaceParts(detector, img);
```

The purpose of sub-function `detectRotFaceParts` is to detect faces with parts rotating input image and output four corners points of face, left eye, right eye, mouth, and nose, image with found faces which are shown as boxes, found faces stored as cell array and found faces with boxes stored as cell array. However, fourpoints of `detectRotFaceParts` are different from `bbox`



of detectFaceParts.

Determining top left corner of face, top right corner of face, bottom right corner of face, bottom left corner of face and face detection is achieved through set of four corner of left eye, set of four corners of right eye, set of four corners of mouth and set of four corners of nose.

The detection object built by sub-function buildDetector, image data which should be uint8, thickness of bounding box and degree step of rotation will be read as input parameters.

Example:

```
detector = buildDetector(2, 2);  
img = imread('img.jpg');  
[fp, bbimg] = detectRotFaceParts(detector, img);
```

The sub-function drawFourPoints is to output image drawn bounding boxes by drawing bounding boxes on input image, where input parameters include input, fourpoints data to be drawn and thickness of bounding boxes.

Example:

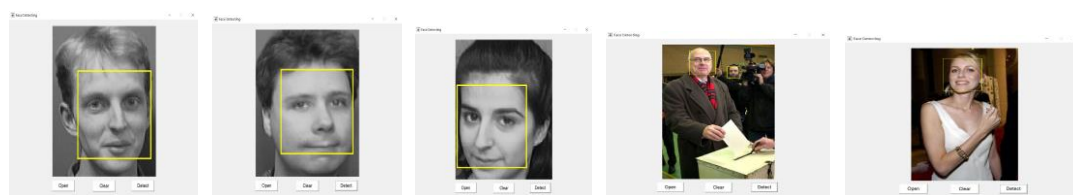
```
detector = buildDetector(2, 2);  
img = imread('img.jpg');  
fp = detectRotFaceParts(detector, img);  
bbimg = drawFourPoints(img, fp);
```

The sub-function mergeFourPoints will output merged fourpoints data by means of merges multiple detection of same face and input parameter is fourpoints data to be merged. These above two sub-functions (i.e. drawFourPoints and mergeFourPoints) are internally called in detectRotFaceParts.

In summary, this method can quickly locate the facial features of the human eye, nose and mouth, and then mark them with some rectangles, which provides a potential basis for improving the manual face alignment method later

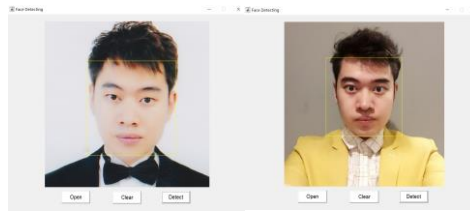
### 3.2.7 Face Detection GUI based on MATLAB R2016a

This original idea is from Viola-Jones Algorithm. The main function is based on inbuilt function in MATLAB, i.e. vision.CascadeObjectDetector. Then the design of GUI is shown in section 2.1.4. Testing image and results are shown in Figure 19.



(a) ORL database

(b) FDDB database



(c) My photo

Figure 19: Results using different face datasets

This method also uses the AdaBoost algorithm, which is not easy to extract the detected images from the GUI system.

### 3.2.8 Face++ API website [6]

Face ++ API get back face bounding box and token for each detected face. The face token can be passed to other APIs for further performing. Detect API can get back 83-point landmarks and attributes for the top 5 largest detected faces. 83-point landmarks and attributes can be obtained by passing its face token to Face Analyze API. And face token is achieved by means of Detect API. Moreover, Face Analyze API can process 5 face token at a time. Face++ Face Landmark SDK enables application to implement face recognition on mobile devices locally. All of face images can be detected and tracked in videos streams in real time. We can get back high-precision 106-point landmarks for each face. I also have applied images of ORL to test this face detection system and get results as shown in Figure 20.

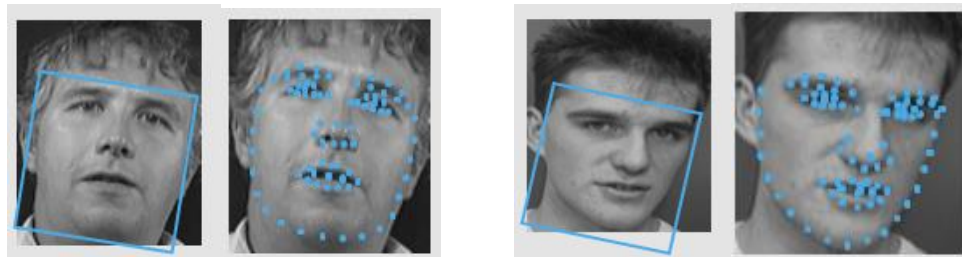


Figure 20: Result for Face++ API

## 3.3 Face alignment

### 3.3.1 Original design

The original idea is to crop image after transformation, and to set size of output image. Besides, the distance between two eyes is fixed. The center of two eyes is located in  $1/3$  of width between the upper and lower edges of image, and  $1/2$  of length between the right and left edges of image. We introduce an example to explain this idea, the size of image is  $130 \times 150$ . Some detailed distribution of position and orientation of two eyes are shown in Figure 21, where 30, 70, 30 and 45 are fixed distance. Some testing results are shown in Figure 22.

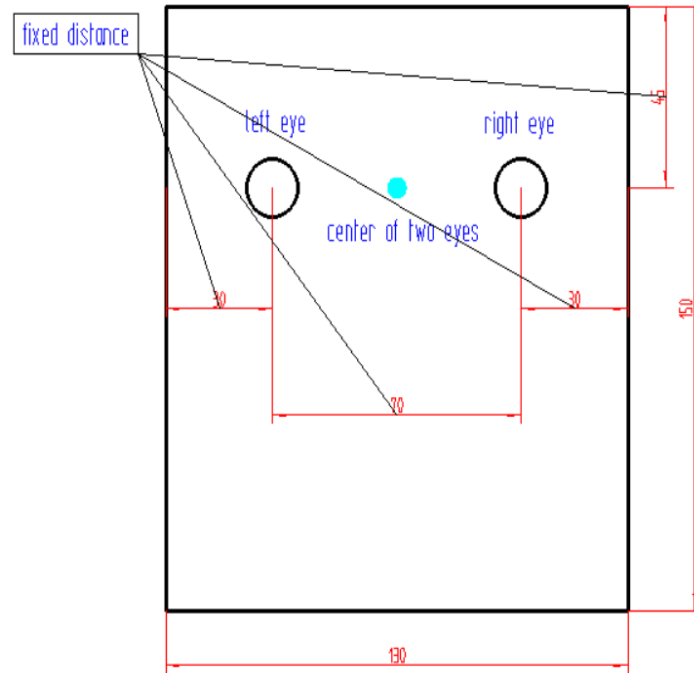


Figure 21: Distribution of face landmarks

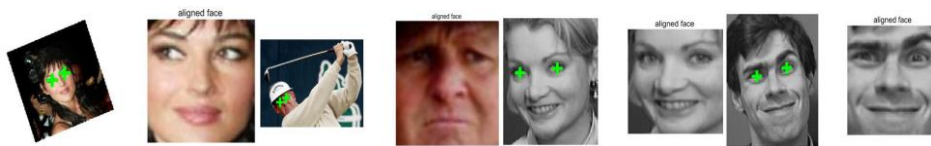
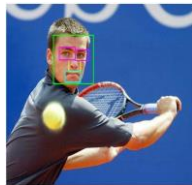
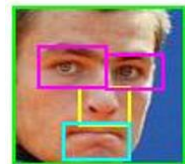


Figure 22: Results of original idea

By labeling the coordinate of left eye and right eye, this method is to perform face alignment. Previously, face detection based on computer vision system toolbox can label the position of two eyes using two rectangles, so from my perspective, the central coordinate of those rectangles are as marked coordinate to label and then face alignment will be executed by means of central coordinates. The automatic alignment system can be created. In such case, the position and orientation of two eyes are fixed. The procedure of performing face alignment is shown in Table 6.

Step	Results
1 Locate landmarks of face and mark position of facial features, such as left eye, right eye, mouth and nose.	
2 Extract these features from original image	

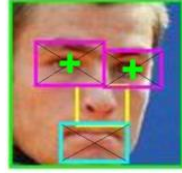
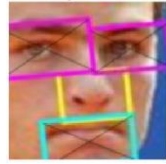
3 Mark the central coordinates of two eyes as points detected.	
4 Let image have fixed position and orientation to complete face alignment	

Table 6: Procedure of performing face alignment

Main program modified after combining face detection based on computer vision system toolbox with manual face alignment method are shown in Listing 1

```
reqToolboxes = {'Computer Vision System Toolbox', 'Image Processing Toolbox'};
if( ~checkToolboxes(reqToolboxes) )
    error('detectFaceParts requires: Computer Vision System Toolbox and Image
Processing Toolbox. Please install these toolboxes.');
```

```
end

for i=1:40
    datadir=['att_faces/orl_faces/s',int2str(i),'/'];
    filenames=[datadir,'*.pgm'];
    files=dir(filenames);
    for j=1:length(files)
        img=imread([datadir,files(j).name]);
        name = strsplit(files(j).name, '.');
        if length(size(img)) == 2
            tmp = ones(112, 92, 3, 'uint8');
            tmp(:, :, 1) = img;
            tmp(:, :, 2) = img;
            tmp(:, :, 3) = img;
            img = tmp;
        end
        detector = buildDetector();
        [bbox, ~, ~, bbfaces] = detectFaceParts(detector,img,2);
        dir_ = ['s', int2str(i), '/'];
        if ~exist(dir_)
            mkdir(dir_)
        end
        path = ['s', int2str(i), '/', name{1}, '.jpg'];
        if size(bbox, 1) == 1
            lx = bbox(5) + bbox(7) / 2;
```

```

        ly = bbox(6) + bbox(8) / 2;
        rx = bbox(9) + bbox(11) / 2;
        ry = bbox(10) + bbox(12) / 2;
        pts = round([lx, ly rx, ry]);
        Face = my_alignment_2points_demo(img, pts);
        size(Face)
        imwrite(Face, path);
    end
end
end

```

Listing 1: Modified codes of combination method

### 3.3.2 Improved design

Generally, the distance between two eyes, the vertical distance between the center of two eyes and upper edge, the distance between left eye and left edge and the distance between right eye and right edge would be fixed.

The source code for area of size and position distribution are shown:

```

inter_ocular = norm([pts(1:2) - pts(3:4)]);
inSize = [130*(inter_ocular/70), 150*(inter_ocular/70)];
inSize = round(inSize);
outLE = [30*(inter_ocular/70), 45*(inter_ocular/70)];
outLE = round(outLE);
outRE = [100*(inter_ocular/70), 45*(inter_ocular/70)];
outRE = round(outRE);

```

Listing 2: Set each of fixed distance within image

However, the image appears black area after the detected image has been transformed, resized and aligned. The reason is that fixed distance between left eye and left edge or between right eye and right edge beyond the boundary of the original image. To ensure the size of new images are the same, those parts have to be filled with black area.



Figure 23: Drawbacks of original method

In such case, the new transformed image could not properly meet alignment requirements. So fixing the distance between two eyes and the vertical distance between the center of two eyes and upper edge is enough to make image has fixed position and orientation.

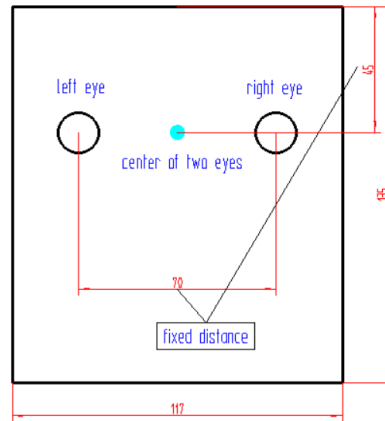


Figure 24: Improved distribution of landmarks

Modified area of codes are shown:

```
inSize = [117, 135];
outLE = [30, 45];
outRE = [100, 45];
```

Listing 3: Improved distribution of position within image

In summary, there are 400 images in ORL dataset. 364 images can automatically be resized and aligned, so alignment rate is 91%. 36 images can manually be resized and aligned if necessary. Some details are shown in Table 7:

The folder where images do not meet the alignment requirements	The number of images in this folder that do not meet the alignment requirements
s1	2, 5, 10
s2	1, 3
s3	5
s4	9, 10
s8	4
s9	2
s11	4
s15	5
s25	2
s26	5
s27	6
s28	1, 3
s31	3, 4, 5, 10

s32	7
s34	1, 2, 3, 4, 5, 6, 7, 8, 10
s35	2
s36	7, 10
s39	6
s40	7

Table 7: Some details of 40 folders

The reason for misaligned is that this program may recognize multiple faces or eyes, may also recognize only one eye in the process of running. This will have a negative effect on the marked points that will be aligned.

To find the size of image with the minimum classification error, I have designed new location of two eyes and the whole face image to test these new images with different size as below:

The distance of two eyes is  $\frac{m}{2}$ , and that of between the center of two eyes and the upper edge is  $\frac{n}{3}$ , where m and n are width and height of the whole face image respectively.

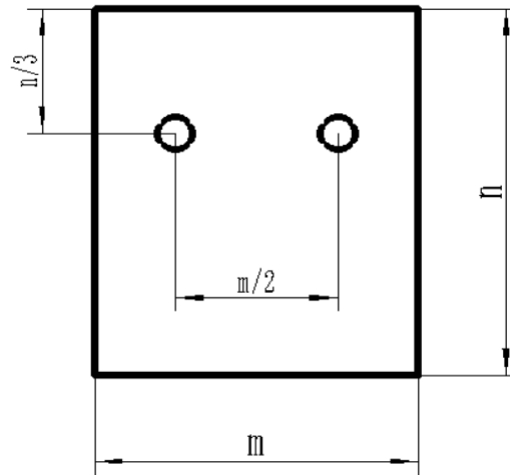


Figure 25: Design of template

Also, I have tested different misclassification error of different size based on above location distribution under the condition of the same sigma value as shown in Table 8.

Sigma=1.25	Image Size	Classification Error
	11.5 × 14	24
	46 × 56	21
	117 × 135	19
	184 × 224	24

Table 8: result of classification error under the condition of different size

The design idea of the image size is from the manual alignment program source code, in which

the program set the image size to 130 by 150. The size of image is reduced by 0.9 times and thus is resized to 117 by 135. In the evaluation part, according to the requirements of image size used in the other published algorithm, the size of image transformed will be adjusted again. Finally, the size of  $117 \times 135$  is chosen for searching the most suitable sigma value. Results are shown in Table 9.

Image Size = $117 \times 135$	Sigma	Classification Error
	1	22.1106
	5	16.0804
	10	14.5729
	15	13.5678
	20	13.5678
	25	13.5678
	30	12.5628
	31	11.5578
	32	11.5578
	35	12.0603
	40	12.5628

Table 9: result of classification error under the condition of different sigma

Therefore, the minimum classification error is 11.5578 under the condition of  $\sigma = 32$ . I have selected some of exemplars from ORL dataset and new aligned and resized dataset. Top dataset is from original ORL dataset and bottom one is from new dataset. As shown in Figure 34. The distribution of position for new image with fixed position and orientation is shown in Table 10.

Size	117 x 135
The distance between two eyes	70
The vertical distance between the center of two eyes and upper edge	45

Table 10: Distribution of parameters of new face image

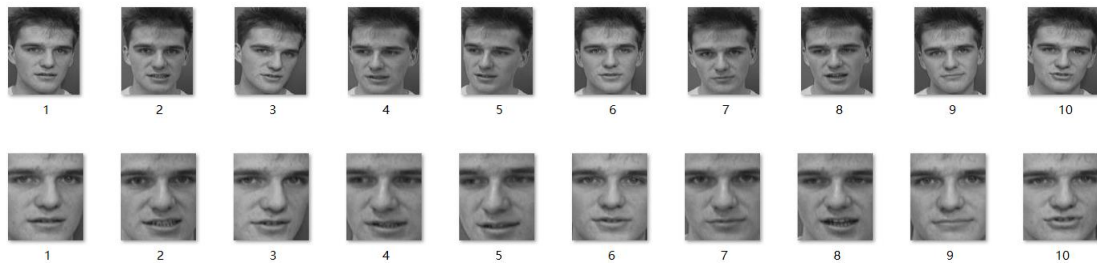


Figure 26: Result of face alignment

### 3.4 Summary

This chapter looked at an automatic method which was used to implement the face alignment



system. It also presented how to distribute position and orientation of two eyes and the size of new images, and then chose the suitable size of new image resized and aligned by changing the sigma value. Finally, it showed a satisfied result compared to original images.

Face alignment system I designed can train 155 images per minute, which has a quick alignment speed. Currently, available data in the literature shows that existing explicit shape regression alignment system performs 100 images per minute for training. This face alignment system has more potential in terms of speed aspect

# 4 Chapter 4

## 4. Evaluation

### 4.1 Overview

The last chapter looked at an automatic method which was used to implement face alignment system. It also presented how all the design parts of the system were implemented. This chapter begins by taking a look at the method and results used to evaluate the system, i.e. PC/BC-DIM algorithm. It finally makes an evaluation of how well the system addressed the issue of improving face recognition experiments by comparing with published results of other algorithms.

### 4.2 Face recognition results by using PC/BC – DIM algorithm[4]

ORL face dataset consists of 400 images of size 112 x 92. There are 40 individuals, and 10 different images per each person. All of the images were taken at different times, lighting and facial expressions and were taken against a dark homogeneous background with the individuals in an upright, frontal position with tolerance for some side movement. The face images are in an upright position in frontal view, with a slight left-right rotation.

In previous work, face images are used to down-sampled to 21-by-24 pixels in Extended Yale B Dataset to perform classification, and execute some pre-processing steps including the calculation of Eigen faces and Laplacian-faces. They are too memory to be performed intensive on larger images. Compared with previous methods, these face images will be resized by a scale factor  $\delta \approx \frac{5}{4}$  to 117-by-135 are proposed.

For this task, we have to set value of some parameters, such as similarity threshold ( $\kappa = 0.9$ ), threshold on the number of patches ( $\lambda = 0$ ), standard deviation( $\sigma = 2.5\sqrt{\delta}$ ), the range of each individual input image([0,1]).

For the 117-by-135 pixel images, a dictionary has been generated including 184 elements in training procedure of the first processing stage. Exemplars from the dictionary learnt from image patches are shown in Figure 30 (a). The weights of 184 prediction neurons are defined by this dictionary in the first processing stage. The predicted class and true class are shown in Figure 30 (b).

The second processing stage included 40 prediction neurons due to 40 individuals. The responses of the above two stages prediction neurons to two test images are shown in Figure

30 (c) & (d).

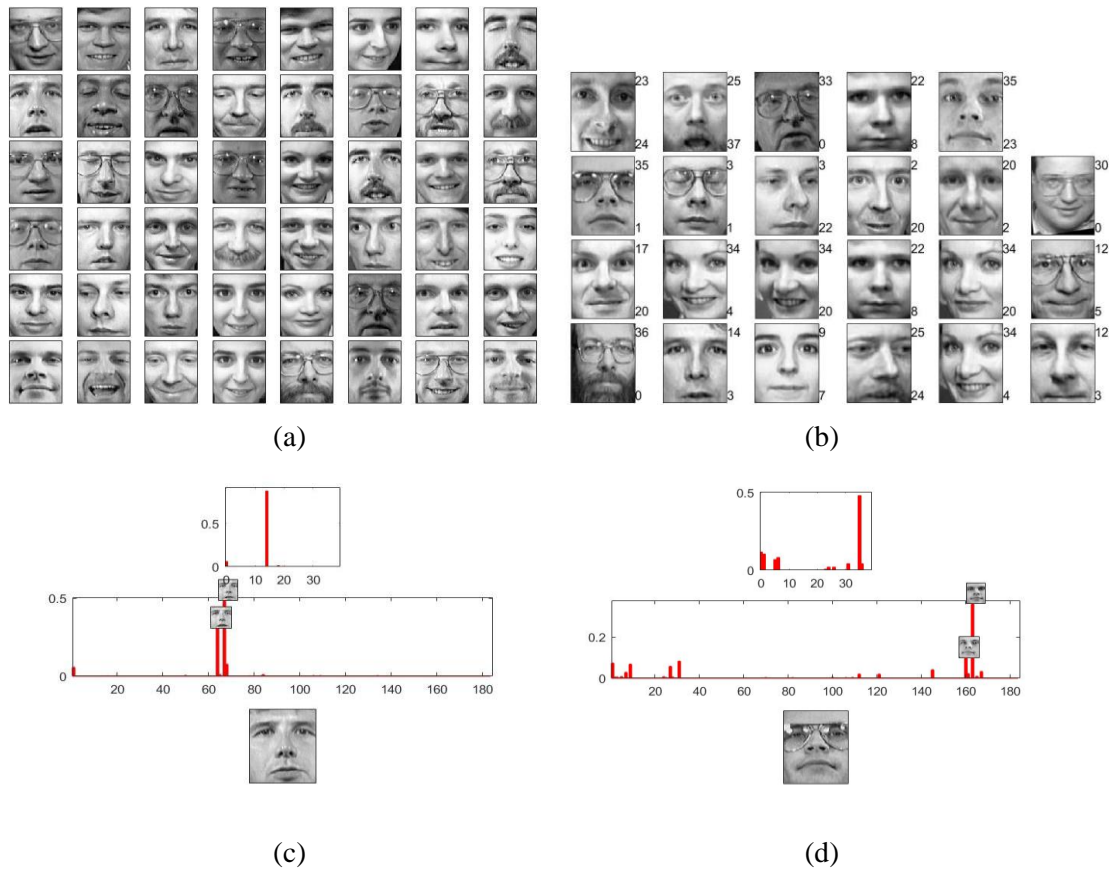


Figure 27: Results of using 117-by-135 pixel images

(a) Exemplars from the dictionary learnt from image patches. (b) All of mis-classified images from the test set.

There are two sets of figures in the right side of the image. The explanations of them are shown in Table 11.

Location	Explanation
Lower number	class predicted by the PC/BC-DIM network
Top number	true class of the image

Table 11: Explanation of Figure 27 (a) and (b)

(c) and (d) indicate the feedback of the prediction neurons of two images from the testing set. Every location of each image can be shown in Table 12. Images superimposed on the histogram show RFs of the most active prediction neurons.

Location	Explanation
Bottom panel	Input to the PC/BC-DIM network
Middle panel	Response of the prediction neurons in the first processing stage
Top panel	Response of the prediction neurons in the second processing stage

Table 12: Explanation of Figure 27 (c) and (d)

Classification can be performed by analyzing the reconstruction errors that are generated through dictionary elements relevance to different classes. The classification error needs to be compared to other algorithms. The current state-of-the-art algorithms are proposed according to sparse coding.

PC/BC-DIM uses sparse code to show the input images (i.e. in Figure 10 (c) & (d), only a very small subset of the first stage prediction neurons are active). However, compared with current existing sparse dictionary-based classifiers, PC/BC-DIM makes the classification using the sparse code rather than the reconstruction error (i.e. the prediction neuron responses instead of the error neuron responses).

## 4.3 Comparison

### 4.3.1 Normalize images

Most of papers resize and align the images with size of  $64 \times 64$  within ORL face datasets. After resizing and aligning the images, I have tested the classification error under the condition of different value of sigma and found the value of minimum classification error and results are shown in Table 13 and Figure 36 respectively.

$64 \times 64$	Sigma	Classification Error
	1	20.1005
	5	16.5829
	10	13.0653
	15	11.5578
	20	12.0603
	25	10.5528
	30	10.5528
	31	10.5528
	35	11.0553
	40	10.5528

Table 13: Sigma and classification error of image with size of  $64 \times 64$

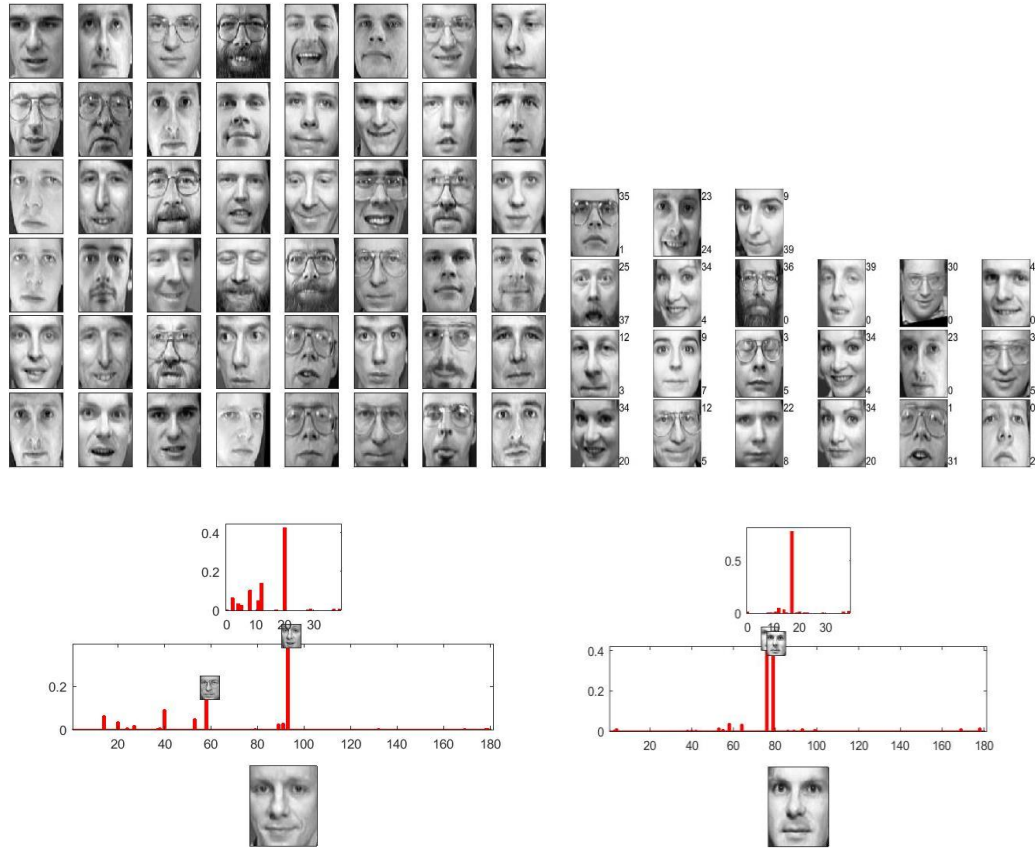


Figure 28: Results of using 64-by-64 pixel images

The classification error of this automatic method by using PC/BC - DIM algorithm is compared to those of a wider range of other algorithms in Table 14.

Method	ORL ( $64 \times 64$ )
hierarchical PC/BC-DIM	10.5
KPCA (Kernel Principal Component Analysis) [46]	9.5
KFD (Kernel Fisher Discriminant) [46]	8.7
CKFD (Complete Kernel Fisher Discriminant) [46]	7
GKFD (Generalized Kernel Fisher Discriminant)[46]	6.5
TS [47]	37.77
LBP [47]	21.12
LTP [47]	16.81
OLTP[47]	7.77
Global expansion ACNN [48]	11.97
Global + local expansion ACNN [48]	9.61

Table 14: Percentage classification error of various methods on the ORL dataset ( $64 \times 64$ )

### 4.3.2 Do not normalize images

Besides, some of papers use original ORL dataset to test classification error of their method, so I also test minimum error under the condition of images with size of  $92 \times 112$ .

$92 \times 112$	Sigma	Classification Error
	1	17.5
	5	12.5
	10	9.5
	15	8.5
	20	7
	25	6
	30	6
	35	6
	40	6.5

Table 15: Sigma and classification error of image with size of  $92 \times 112$

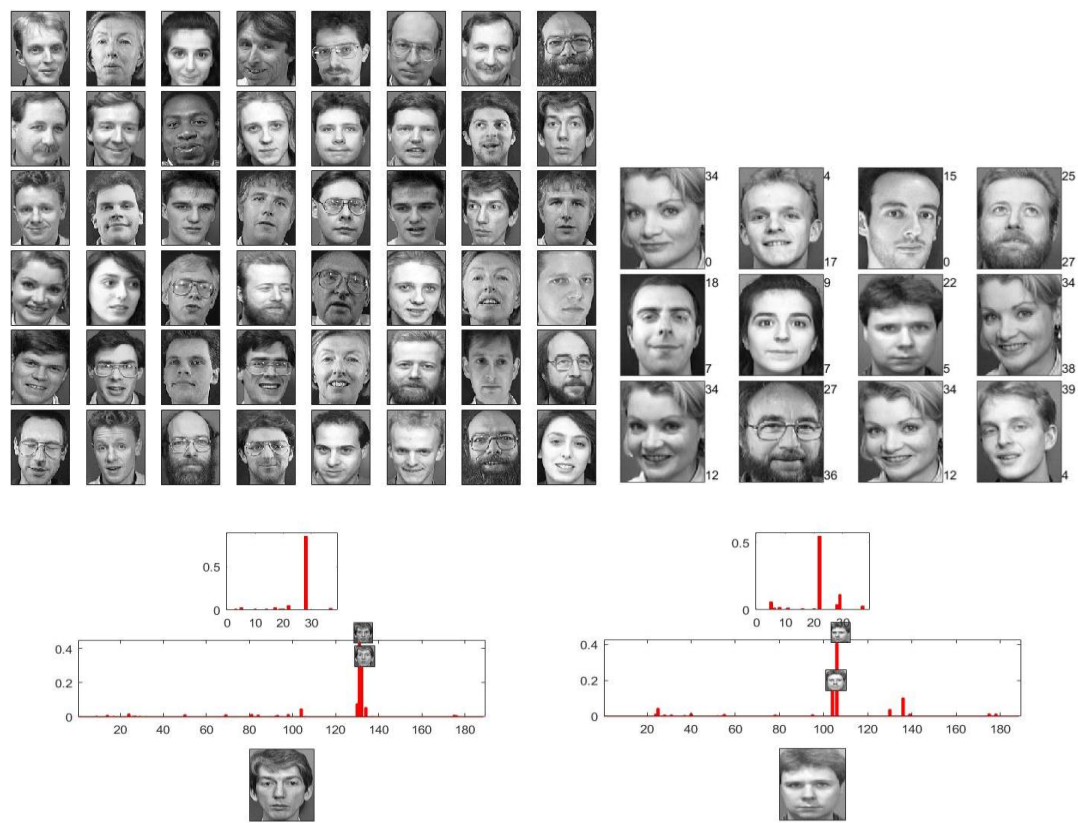


Figure 29: Results of using 92-by-112 pixel images (original ORL face dataset)

The classification error of PC/BC - DIM algorithm is compared to those of a wider range of other algorithms in Table 16.

Method	ORL ( $92 \times 112$ )
--------	-------------------------

hierarchical PC/BC-DIM	6
PCA [49]	13.8
LDA [49]	12.6
Scaled CGA [49]	10
F-CS [49]	8
Powel-Beale CGA [49]	8
CA-SRC [49]	7
EI-CS [49]	7
Powel-Beale-CGA-PCA [49]	6
Fast Fourier Transform (FFT) [49]	4
ODSMMA [50]	9.43
MFA [50]	11.5
SPP [50]	13.29
KNN [51]	8
SVM [51]	5.9
KNN+SVM [51]	3.5
LR [52]	6.5
NS [52]	6.5
Gumus E [52]	4.7
PCA and logistic regression analysis [52]	4
Top-down HMM + gray tone features [53]	13
Eigenface [53]	9.5
Pseudo 2D HMM + gray tone features [53]	5.5
Elastic matching [53]	20.0
PDNN [53]	4.0
Continuous n-tuple classifier [53]	2.7
Top-down HMM + DCT [53]	16
Point-matching and correlation [53]	16
Ergodic HMM + DCT [53]	0.5
Pseudo 2D HMM + DCT [53]	0
SVM + PCA [53]	3
Indipendent Component Analysis [53]	15
Gabor filters + rank correlation [53]	8.5
Wavelet + HMM [53]	0

Table 16: Percentage classification error of various methods on the ORL dataset

## 4.4 Summary

This chapter had evaluated automatic face alignment system and had compared to the results of other algorithms. It not only made a comparison with new face dataset with the size of 64 by 64, but also include results compared with original ORL face dataset with the size of 92 by 112. It concludes that the capability of the PC/BC – DIM algorithm is competitive with the current state-of-the-art for improving identification rate of face recognition.

# 5 Chapter 5

## 5. Conclusion

### 5.1 Introduction

This project has looked at the importance of face detection for research objectives and the significant challenge of how to develop an automatic method to achieve face alignment. It looked at the current ways face detection is carried out and the potential of using AdaBoost algorithm to carry out face detection and locate landmarks of face image.

This led to the design and implementation of face alignment system which achieve contribution of improving face detection. A significant contribution achieving that an automatic method to align and resize face images would be implemented by face detection using a method of computer vision system toolbox to locate position of left eye and right eye by means of two rectangles, and face alignment based on fixed position and orientation of facial features in each given image, i.e. central coordinate of two rectangles of left eye and right eye, has been presented within this project. And then a wider variety of face images would be automatically resized and aligned according to above method to generate new face datasets so that PC/BC-DIM algorithm could be tested in face recognition.

Compared to other algorithms of face recognition on the same face data set, this automatic method is to improve the recognition rate and running time. This chapter concludes the project by giving an overview of the previous chapters, followed by the contributions and results of the project. Areas of further work are then presented, before closing the project with some concluding remarks.

### 5.2 Project Overview

Chapter 1 presented an introduction to face detection and face recognition. It also defined the subject of face detection and face recognition, and their place and advanced technology in the current recognition system.

Chapter 2 presented an overview and previous work of face detection, face alignment and face recognition. It then analyses how integral image and cascade classifier have been used to solve problem of locating landmarks of face in face detection. It finally gave an introduction to PC/BC-DIM algorithm and how it works in improving face recognition experiments and how this hierarchical perceptual inference process performed in the cortex.

Chapter 3 described the various methods of face detection detailing what the project seeks to



achieve. It then selected one method of face detection, i.e. face detection based on computer vision system toolbox and inbuilt function in MATLAB. Moreover, according to combining previous face detection selected with face alignment manually method, this chapter described how to develop an automatic method to locate landmarks that will allow faces to be aligned and resized to a fixed location within each image.

Chapter 4 made an evaluation of how well the system addressed the issue of improving face recognition experiments by comparing with published results of other algorithms.

## 5.3 Contributions

A review of existing literature revealed a gap in exploring the potentials of using AdaBoost algorithm to achieve face detection process. A key contribution of this project was the design and implementation of combining manual face alignment with face detection so as to create an automatic face alignment system.

Previous existing methods involving the use of explicit shape regression are clearly slow. In explicit shape regression for face alignment, two-level boosted regression, shape indexed features and a correlation-based feature selection method are designed. However, it will take 20 minutes for 2000 training images learning accurate models from large training data, average speed is 100 images per minute. In this project, this face alignment system will spend 2 minutes and 35 seconds for 400 training images, the average speed is 155 images per minute. Experiments show that our approach significantly outperforms the state-of-the-art in terms of both accuracy and efficiency.

## 5.4 Further Work

The algorithm of existing work is designed based on the attitude, expression, occlusion to improve the robustness of the feature point positioning model. All methods are more or less related to the cascade shape regression framework and show the effectiveness of the method of cascade shape regression. However, every regression model under the cascade shape regression framework is independently trained and is not an end-to-end approach.

George Trigeorgis, etc. in Imperial College London put forward a method by using Convolutional Recurrent Neural Network to solve the problem of feature point positioning [54]. It can be trained end-to-end feature point localization model, and have significant performance improvements than the traditional cascade regression methods. In addition, the attitude estimation strongly relies on feature point location task. Analyzing these tasks may be good solution in the application. HyperFace integrates the feature map of different layers of convolution neural network can complete face detection, facial feature point positioning, posture prediction and gender identity. Finally, how to study the positioning of the network with low complexity, and how handset facial point can be located efficiently and accurately is also a problem that worth exploring.

## 5.5 Concluding Remarks

This project has identified the challenge of how to develop a new automatic method to achieve face alignment. It also identified a research advantage in using Adaboost algorithm to perform face detection. A face alignment system using computer vision system toolbox based on AdaBoost algorithm, and manual face alignment system, was therefore designed and implemented to improve the recognition rate and running time. It has also shown that face alignment system can be implemented by using mainly neural network and shape regression in the future.

## References

- [1] Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440-1448).
- [2] Yang, S., Luo, P., Loy, C. C., & Tang, X. (2016). Wider face: A face detection benchmark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 5525-5533).
- [3] Cao, X., Wei, Y., Wen, F., & Sun, J. (2014). Face alignment by explicit shape regression. *International Journal of Computer Vision*, 107(2), 177-190.
- [4] Spratling, M. W. (2017). A hierarchical predictive coding model of object recognition in natural images. *Cognitive Computation*, 9(2), 151-167.
- [5] Viola, P., & Jones, M. J. (2004). Robust real-time face detection. *International journal of computer vision*, 57(2), 137-154.
- [6] <https://www.faceplusplus.com/>
- [7] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on* (Vol. 1, pp. I-I). IEEE.
- [8] <http://www.cis.pku.edu.cn/vision/Visual&Robot/publication/doc/thesis/Face%20Detection%20Based%20on%20AdaBoost%20-%20ZhaoNan.pdf>
- [9] Xiong, S. W., Zong, X. L., & Zhu, G. F. (2007). Improved face detection method based on AdaBoost algorithm. *Jisuanji Yingyong Yanjiu/ Application Research of Computers*, 24(11), 298-300.
- [10] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on* (Vol. 1, pp. I-I). IEEE.
- [11] Lienhart, R., & Maydt, J. (2002). An extended set of haar-like features for rapid object detection. In *Image Processing. 2002. Proceedings. 2002 International Conference on* (Vol. 1, pp. I-I). IEEE.
- [12] Mehrabian, A. (2008). Communication without words. *Communication theory*, 193-200.
- [13] Das, A., Pukhrambam, M., & Saha, A. (2015, October). Real-time robust face detection and tracking using extended haar functions and improved boosting algorithm. In *Green*

*Computing and Internet of Things (ICGCIoT), 2015 International Conference on* (pp. 981-985). IEEE.

[14] Ban, Y., Kim, S. K., Kim, S., Toh, K. A., & Lee, S. (2014). Face detection based on skin color likelihood. *Pattern Recognition*, 47(4), 1573-1585.

[15] Hamilton, E. (2004). JPEG file interchange format.

[16] Schwerdt, K., & Crowley, J. L. (2000). Robust face tracking using color. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on* (pp. 90-95). IEEE.

[17] Freund, Y., & Schapire, R. E. (1995, March). A decision-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory* (pp. 23-37). Springer, Berlin, Heidelberg.

[18] Soriano, M., Martinkauppi, B., Huovinen, S., & Laaksonen, M. (2003). Adaptive skin color modeling using the skin locus for selecting training pixels. *Pattern Recognition*, 36(3), 681-690.

[19] Himani, K., Sankalp, M. C., Kumar, K. P., & Shashidhar, K. P. Face Recognition System with GUI Using Digital Image Processing.

[20] Mondal, A., Dutta, P., Roy, D., Pandey, M., & Saha, A. K. Region Marking of Face and Eye Using Skin Tone Segmentation.

[21] Lakshmi, H. V., & Kulakarni, S. P. (2010). Face Localization and Detection Algorithm for Colour Images Using Wavelet Approximations. *Computer Vision and Information Technology: Advances and Applications*, 305.

[22] Gonzalez, R. C., & Woods, R. E. (2005). Book on "Digital image processing".

[23] Farkas, L. G. (Ed.). (1994). *Anthropometry of the Head and Face*. Raven Pr.

[24] Thakur, S., Paul, S., Mondal, A., Das, S., & Abraham, A. (2011, December). Face detection using skin tone segmentation. In *Information and Communication Technologies (WICT), 2011 World Congress on* (pp. 53-60). IEEE.

[25] Ruangyam, P., & Covavisaruch, N. (2009, November). An efficient region-based skin color model for reliable face localization. In *Image and Vision Computing New Zealand, 2009. IVCNZ'09. 24th International Conference* (pp. 260-265). IEEE.

[26] He, X., Yan, S., Hu, Y., Niyogi, P., & Zhang, H. J. (2005). Face recognition using laplacianfaces. *IEEE transactions on pattern analysis and machine intelligence*, 27(3), 328-340.

- [27] Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. J. (1997). Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7), 711-720.
- [28] Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1), 71-86.
- [29] Sinha, P., Balas, B., Ostrovsky, Y., & Russell, R. (2006). Face recognition by humans: Nineteen results all computer vision researchers should know about. *Proceedings of the IEEE*, 94(11), 1948-1962.
- [30] Savvides, M., Abiantun, R., Heo, J., Park, S., Xie, C., & Vijayakumar, B. V. K. (2006, June). Partial & holistic face recognition on frgc-ii data using support vector machine. In *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on* (pp. 48-48). IEEE.
- [31] Liu, C. (2006). Capitalize on dimensionality increasing techniques for improving face recognition grand challenge performance. *IEEE transactions on pattern analysis and machine intelligence*, 28(5), 725-737.
- [32] Phillips, P. J., Scruggs, W. T., O'Toole, A. J., Flynn, P. J., Bowyer, K. W., Schott, C. L., & Sharpe, M. (2007). FRVT 2006 and ICE 2006 large-scale results. *National Institute of Standards and Technology, NISTIR*, 7408(1).
- [33] Zhu, X., Lei, Z., Liu, X., Shi, H., & Li, S. Z. (2016). Face alignment across large poses: A 3d solution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 146-155).
- [34] Jourabloo, A., & Liu, X. (2016). Large-pose face alignment via CNN-based dense 3D model fitting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4188-4196).
- [35] Zhu, S., Li, C., Loy, C. C., & Tang, X. (2016). Unconstrained face alignment via cascaded compositional learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3409-3417).
- [36] Xiong, X., & De la Torre, F. (2015). Global supervised descent method. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2664-2673).
- [37] Ren, S., Cao, X., Wei, Y., & Sun, J. (2014). Face alignment at 3000 fps via regressing local binary features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1685-1692).

- [38] Kersten, D., Mamassian, P., & Yuille, A. (2004). Object perception as Bayesian inference. *Annu. Rev. Psychol.*, 55, 271-304.
- [39] Lochmann, T., & Deneve, S. (2011). Neural processing as causal inference. *Current opinion in neurobiology*, 21(5), 774-781.
- [40] Lochmann, T., Ernst, U. A., & Deneve, S. (2012). Perceptual inference predicts contextual modulations of sensory responses. *Journal of neuroscience*, 32(12), 4179-4195.
- [41] Spratling, M. W. (2012). Unsupervised learning of generative and discriminative weights encoding elementary image components in a predictive coding model of cortical function. *Neural Computation*, 24(1), 60-103.
- [42] Spratling, M. W., De Meyer, K., & Kompass, R. (2009). Unsupervised learning of overlapping image components using divisive input modulation. *Computational intelligence and neuroscience*, 2009.
- [43] Dutta, P., & Bhattacharjee, D. (2014, January). Face detection using generic eye template matching. In *Business and Information Management (ICBIM), 2014 2nd International Conference on* (pp. 36-40). IEEE.
- [44] Yuille, A. L., Cohen, D. S., & Hallinan, P. W. (1989, June). Feature extraction from faces using deformable templates. In *Computer Vision and Pattern Recognition, 1989. Proceedings CVPR'89., IEEE Computer Society Conference on* (pp. 104-109). IEEE.
- [45] Lin, C. H., & Wu, J. L. (1999). Automatic facial feature extraction by genetic algorithms. *IEEE Transactions on Image processing*, 8(6), 834-845.
- [46] Shi, Y., Ren, X., Yang, S., & Gong, P. (2016, August). A generalized Kernel Fisher Discriminant framework used for feature extraction and face recognition. In *Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), 2016 12th International Conference on* (pp. 1487-1491). IEEE.
- [47] Raja, G. M., & Sadasivam, V. (2017). Face Representation and Face Recognition using Optimized Local Ternary Patterns (OLTP). *Journal of Electrical Engineering & Technology*, 12(1), 402-410.
- [48] Zhang, Y., Zhao, D., Sun, J., Zou, G., & Li, W. (2016). Adaptive convolutional neural network and its application in face recognition. *Neural Processing Letters*, 43(2), 389-399.
- [49] Banitalebi-Dehkordi, M., Banitalebi-Dehkordi, A., Abouei, J., & Plataniotis, K. N. (2017). Face recognition using a new compressive sensing-based feature extraction method. *Multimedia Tools and Applications*, 1-21.

- [50] Lin, Y. E., Xu, J., Zhang, Y., & Liang, X. (2017). Orthogonal Discriminant Sparse Maximum Margin Analysis. *DEStech Transactions on Engineering and Technology Research*, (apetc).
- [51] Yan, H., Wang, P., Chen, W. D., & Liu, J. (2015). Face recognition based on gabor wavelet transform and modular 2dpca. In *International Conference on Power Electronics and Energy Engineering (PEEE 2015)* (pp. 245-248).
- [52] Zhou, C., Wang, L., Zhang, Q., & Wei, X. (2014). Face recognition based on PCA and logistic regression analysis. *Optik-International Journal for Light and Electron Optics*, 125(20), 5916-5919.
- [53] Bicego, M., Castellani, U., & Murino, V. (2003, September). Using hidden Markov models and wavelets for face recognition. In *Image Analysis and Processing, 2003. Proceedings. 12th International Conference on* (pp. 52-56). IEEE.
- [54] Trigeorgis, G., Snape, P., Nicolaou, M. A., Antonakos, E., & Zafeiriou, S. (2016). Mnemonic descent method: A recurrent process applied for end-to-end face alignment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4177-4187).