

Face Recognition under Significant Pose Variation

Feng Yang

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements  
for the Degree of Master of Computer Science at  
Concordia University  
Montreal, Quebec, Canada

April 2007

©Feng Yang, 2007



Library and  
Archives Canada

Bibliothèque et  
Archives Canada

Published Heritage  
Branch

Direction du  
Patrimoine de l'édition

395 Wellington Street  
Ottawa ON K1A 0N4  
Canada

395, rue Wellington  
Ottawa ON K1A 0N4  
Canada

*Your file    Votre référence*

*ISBN: 978-0-494-28958-7*

*Our file    Notre référence*

*ISBN: 978-0-494-28958-7*

#### NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

#### AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

---

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

  
**Canada**

## **ABSTRACT**

### **Face Recognition under Significant Pose Variation**

Feng Yang

Unlike the frontal face detection, multi-pose face detection and recognition techniques, still face the following challenges: large variability of environments such as pose, illumination and backgrounds, and unconstrained capturing of facial images. We introduced a new system to deal with this problem. First, a two-step color-based approach is used to find a candidate area of face from original picture. Then a rough estimator of five poses is created using AdaBoost technique. In order to accurately locate the candidate face, multiple statistical shape models-ASM (Active Shape Models) are proposed to estimate an accurate pose of model of the input image and to extract facial features as well. In the recognition step, we use a geometrical mapping technique to deal with the pose variation and face identification.

## **Acknowledgements**

I would like to take this opportunity to express my sincere appreciation and thanks to my adviser, Dr. Adam Krzyzak for his great help and guidance, and also the inspiration he provided me in all my works. Without his support, criticism, and invaluable help, this work would never have been possible.

Special thanks to my parents and my wife Ji Li for their encouragement and support at those difficult times.

## Table of Contents

List of Figures .....	VIII
List of Tables .....	X
1 Introduction.....	1
1.1 Overview .....	1
1.2 Problem Statement .....	4
1.3 Research Contributions .....	6
1.4 Organization of Thesis .....	11
2 Background and Related Work.....	12
2.1 Face Detection Approaches.....	12
2.1.1 Skin Color .....	12
2.1.2 Active Shape Model .....	13
2.1.3 Edge-Orientation Matching.....	14
2.1.4 Boosting .....	16
2.2 Face Recognition Approaches.....	17
2.2.1 Principal Component Analysis.....	17
2.2.2 3-D Morphable Model.....	19
2.2.3 3-D Face Recognition.....	20
2.2.4 SVM .....	20
3 Databases .....	22
3.1 Database for Rough Face Detection.....	22
3.1.1 Data collection.....	22
3.1.2 Image Processing.....	27
3.2 Database for Accurate Face Detection .....	27
3.3 Database in Face Recognition .....	29
4 Face detection .....	31

4.1	Color-based face detector and rough pose estimator.....	31
4.2	AdaBoost-based face detection .....	33
4.2.1	Features .....	33
4.2.2	Integral Image .....	34
4.2.3	Learning Classification Functions.....	35
4.2.4	Adaboost Training Results .....	39
4.2.5	Multi-pose classifiers .....	41
4.3	Active Shape Models and Gray Level Variations .....	48
4.3.1	Point Distribution Model.....	49
4.3.2	Labeling the Training Set.....	49
4.3.3	Aligning Shape.....	52
4.3.4	Statistics .....	53
4.4	Image Search Using Active Shape Modeling .....	57
4.4.1	The Initial Shape Estimate .....	57
4.4.2	Updating the Shape .....	58
4.4.3	Finding the Desired Movements .....	60
4.5	Multi-Resolution Image Search .....	61
4.6	Facial Feature Extraction and Accurate Pose Estimator .....	64
4.6.1	Face Edge Detection.....	64
4.6.2	Accurate Pose Estimator and Face Detection .....	66
4.6.3	Facial Feature Extraction .....	74
5	Multi-pose face recognition .....	79
5.1	Reference Image Sets Collection .....	79
5.2	Geometrical mapping .....	82
5.3	Geometry assisted face recognition.....	88
5.4	Experimental Results.....	92
6	Conclusion and Future Work .....	94

6.1 Conclusion.....	94
6.2 Future Work .....	94
References.....	95

## List of Figures

Figure 1: Unaligned Facial images after face detection.....	5
Figure 2: Rough face detector & pose estimator .....	7
Figure 3: Accurate face detector .....	9
Figure 4: Algorithm architecture. ....	10
Figure 5: Face recognition process flow.....	11
Figure 6: Face model with labeled points.....	14
Figure 7: Face image after processed by Sobel edge filter .....	15
Figure 8: Training image with feature .....	17
Figure 9: The front training face examples.....	24
Figure 10: The right 45 degree training face examples .....	25
Figure 11: The left 45 degree training face examples.....	25
Figure 12: The left 90 degree training face examples.....	26
Figure 13: The right 90 degree training face examples .....	26
Figure 14: An example of annotation .....	28
Figure 15: The image examples from PIE database .....	30
Figure 16: Original input color image .....	32
Figure 17: Output binary image.....	32
Figure 18: Binary image after opening operator.....	32
Figure 19: Rectangle features .....	34
Figure 20: Integral image.....	34
Figure 21: Computing integral image.....	35
Figure 22: The distribution of sum_det in 5 poses .....	43
Figure 23: The differences between sum_dets and alpha thresholds.....	45
Figure 24: Detected face examples.....	48
Figure 25: Manually annotated face examples for poses of left 67.5, left 45, left 22.5, front, right 22.5, right 45 and right 67.5 .....	51
Figure 26: Manually annotated face examples for poses of left 90 and right 90.....	52
Figure 27: The progress of multi resolution image search .....	62



Figure 28: The process of pyramidal image .....	63
Figure 29: The process of face edge detection.....	65
Figure 30: Facial features localization.....	74
Figure 31: The measurement of extraction and the key points of cropping .....	76
Figure 32: Facial features localization in profile .....	77
Figure 33: Manually select the three key points in profile image.....	80
Figure 34: The measurement of extraction and the key points of cropping for profile image.....	81
Figure 35: Cropped results of profile images .....	82
Figure 36: Geometric mapping: one point on the surface of the ellipsoid maps to a pixel on the image plane. ....	86
Figure 37: Geometrical mapping results.....	87
Figure 38: Geometry assisted face recognition in the poses nearby front .....	90
Figure 39: Geometry assisted face recognition in the poses nearby profile .....	91
Figure 40: Recognition performances of two algorithms on the PIE database.....	93

## List of Tables

Table 1: The number of training and testing face image sets .....	6
Table 2: Adaboost algorithm .....	37
Table 3: Error rates of first 100 features in front faces training procedure .....	39
Table 4: Training results of first six layers of five strong classifiers.....	40
Table 5: The first 500 features' results of <i>sum_det</i> for 5 poses .....	42
Table 6: The first 500 features' results of alpha thresholds with matching poses .....	44
Table 7: The 500 features' differences between <i>sum_dets</i> and alpha thresholds .....	44
Table 8 Accuracy rate of rough face estimator.....	47
Table 9: The algorithm in face edge detection.....	64
Table 10: part of pose accurate detection results with respect to left 67.5, left 45 and left 22.5 poses .....	70
Table 11: Algorithm of pose detection .....	74
Table 12: Extracting and Cropping Algorithm in Testing Stage .....	76
Table 13: Extracting and Cropping Algorithm in Training Stage .....	80
Table 14: The results of face recognition in varying poses .....	92

# 1 Introduction

## 1.1 Overview

Human face recognition has been an active research topic in the field of pattern recognition for decades. Unlike other existing identification technologies such as fingerprint and iris recognition, face recognition has some outstanding characteristics such as non-intrusive and user-friendly interfaces, which mean the face recognition can be carried out without any human intervention or even awareness of the user. Based on these advantages, there are many potential practical applications so far, for example, identification in banking, mug shot searching, security monitoring and surveillance.

Much progress has been made in face detection and recognition in recent years, especially in case of face processing under controlled conditions. The most popular approach studied over two decades was frontal face detection and recognition. Sung and Poggio [1] found a classifier based on the difference feature vector which is computed between the local image pattern and the distribution-based model. Papageorgiou [2] proposed a detection technique based on an overcomplete wavelet representation of an object class [45]. Both are carried out by performing a dimensionality reduction to select the most important basis function in the first place; and then a Support Vector Machine (SVM) is trained [2] to develop final prediction. The SNoW learning architecture was adopted by Roth [3] for learning in the presence of a very large number of features.

Unlike the frontal face detection, most non-frontal face detection techniques in the literature are view-based [39], in which several face models are built; each describes faces in a given range of view. Therefore, explicit 3D modeling is avoided. Pentland et al [39] partitioned the views of face into five channels, and developed a multi-view detector by training a separate detector network for each view. Viola and Jones [9] built a fast multi-pose face detection system. In their work, a cascade of boosting classifiers is built on a set of Haar-like features [2] that integrates the feature selection and classifier design in the same framework. Schneiderman and Kanade [10] used multi-resolution information in different levels of wavelet transform. The system consists of an array of two face detectors in a view-based framework. Each detector is constructed using statistics of products of histograms computed from examples of the respective view. It has achieved the best detection accuracy in the literature, while it is very slow due to the computation complexity.

Another technique directly related to the face detection is face feature extraction (actually it is a part of face detection—accurate face detection). One earlier approach for feature extraction was proposed by Yuille et al.[14]. In this approach, parameterized deformable templates are used to minimize a predefined fitting function. Recently, the Active Shape Model (ASM) and Active Appearance Model (AAM) [15] were proposed as two effective techniques for feature extraction. In those techniques, a constrained statistical shape model is adopted, which offers fast and stable performance.

As successor of face detection, face recognition has always received significant attention. Turk and Pentland [6] introduced a first successful face detection and identification

system by using eigenfaces. Lin et al.[11] proposed an automatic face detection and recognition system by using a Probabilistic Decision Based Neural Network (PDBNN) which includes three components: a face detector, an eye localizer and a face recognizer. Blanz and Vetter [12] suggested an approach combining 3D shape and texture which is estimated using a single image. The estimate is achieved by fitting a statistical, morphable model of faces to the image. The model is learnt from textured 3D data collected with a laser-stripe scanner. A probabilistic PCA-based model is used to represent the statistical variation of shape and texture of human heads. Phillips [13] applied SVM methods in face recognition and made an impressive achievement with high recognition rate by using a support vector machine as the classifier given a set of points belonging to two classes. A SVM can find a hyperplane that separates the largest possible fraction of points of the same class on the same side, while maximizing the distance from either class to the hyperplane.

All those works have achieved satisfactory recognition rate on frontal faces with limited variation of expression, lighting and controlled pose. One of the key remaining problems in face recognition is the need to handle the large variability in appearance due to change in pose and illumination (e.g. in an unconstrained environment) since the relatively unconstrained environment imposes more challenges compared to conventional frontal face recognition. These extra challenges include the following:

1. Large variability of operating environments such as pose, illumination and backgrounds).
2. Nearly unconstrained capturing of facial images.

In this thesis, we will introduce a mug shot identification system which can precisely detect the multi-pose faces and recognize them using geometrical mapping approach. When a police department analyzes suspects' photos, since a large amount of pictures taken from unconstrained and constrained environment need to be processed, it is very inefficient to handle all this work manually. The overall goal of our research is to develop an automatic system which detects the face within a complex background and varying pose, and recognizes the corresponding object from a frontal and profile mug shot database.

## **1.2 Problem Statement**

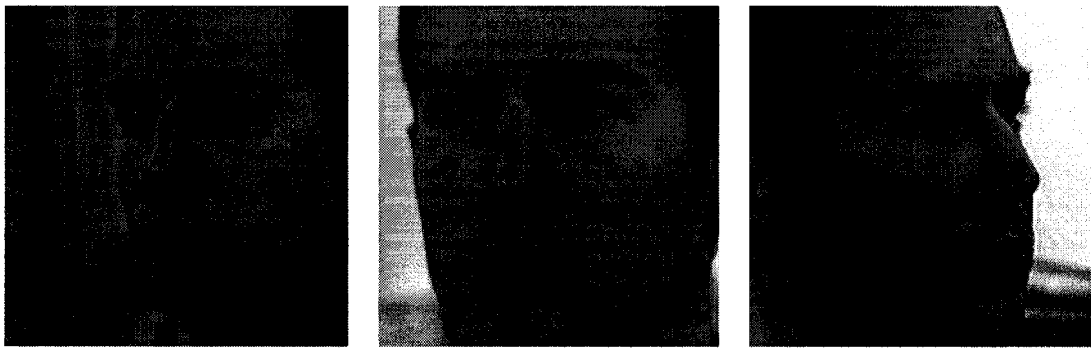
According to the literature and experiment results, the problem of automatic face recognition involves three key steps/subtasks:

- 1) Detection and localization of faces
- 2) Feature extraction and accurate normalization of faces
- 3) Identification and/or verification.

The first step in any automatic face recognition systems is to detect faces within images. Face detection, however, has always been regarded as a challenging problem in the field of computer vision, due to the large intra-class variations caused by the changes in facial appearance, lighting, pose and expression. Such variations result in the face distribution to be highly nonlinear and complex in space. Moreover, in the applications of real life, the pose variations make the distribution of human faces in feature space more dispersed and

complicated than that of frontal faces. It further complicates the problem of robust face detection.

After roughly detecting the candidates of face area, the output from the face detector contain faces with varying scale, position and pose. Those unaligned facial images (Figure 1) can not be directly used in identification since it may potentially cause extremely high rate of failures. Moreover, in order to archive satisfactory result in recognition step, we need an accurate extraction of salient facial features for normalization purposes and pose estimation must be made in the mean time. However, there are two problems with achieving this goal. First, the 2-D data provided by conventional cameras lacks the sensitivity and features required for accurate 3-D pose estimation of arbitrarily shaped objects. Second, approach of pose estimation requires operation which is difficult to perform feature extraction.



**Figure 1: Unaligned Facial images after face detection**

The difficulty in face recognition is magnified by necessity to deal with variations, such as pose, illumination and expression. Among all kinds of variations, pose variation is the hardest one to model and therefore contributes most of the recognition error to a recognition system. Due to time constraints, we did not consider the complex issues of

illumination and expression in this thesis. The difficulty with pose variation is that, the intra-subject variations can be as large as, or even larger than the inter-subject variations when pose variation is present.

### 1.3 Research Contributions

Since multi-pose faces have different facial features, in order to train the face detector with varying pose, image sets with five poses for training and testing were collected manually from internet. The number of training and testing face image sets are shown in the following table:

Pose dataset	testing	training
Left 90	200	1000
Left 45	200	1000
Front	200	4918
Right 45	200	1000
Right 90	200	1000
Non face	200*	15000**

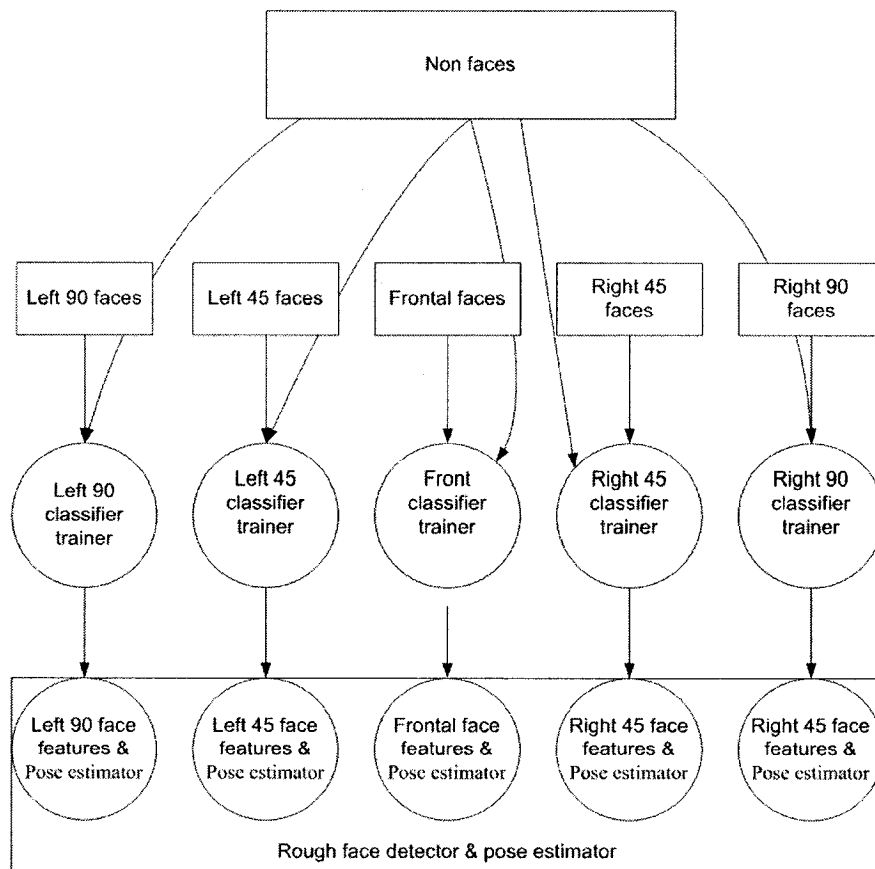
**Table 1: The number of training and testing face image sets**

\* This number is fixed, but the actual nonface images vary in each training task.



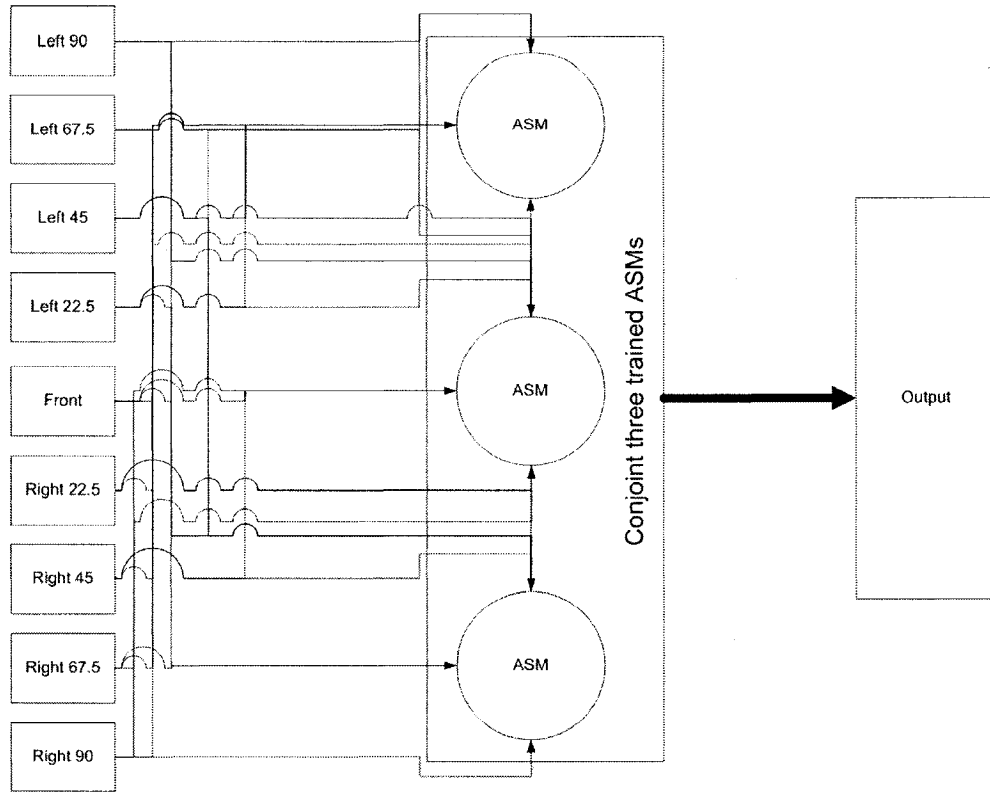
\*\*This is total number, but the actual number of non faces used during training is same as the number of faces

To deal with the face detection in pose variation, first, we adopted the color-based approach to speed up the whole processing, then we introduced a machine learning approach – based on AdaBoost- which selects a small number of critical visual features from a larger set and produces efficient classifiers and combines more complex classifiers. Then we utilized the quantization output of the previous detector, and a rough estimator of five poses, namely left profile, left 45 degree, front, right 45 degree and right profile was created. The following Figure 2 shows this process:



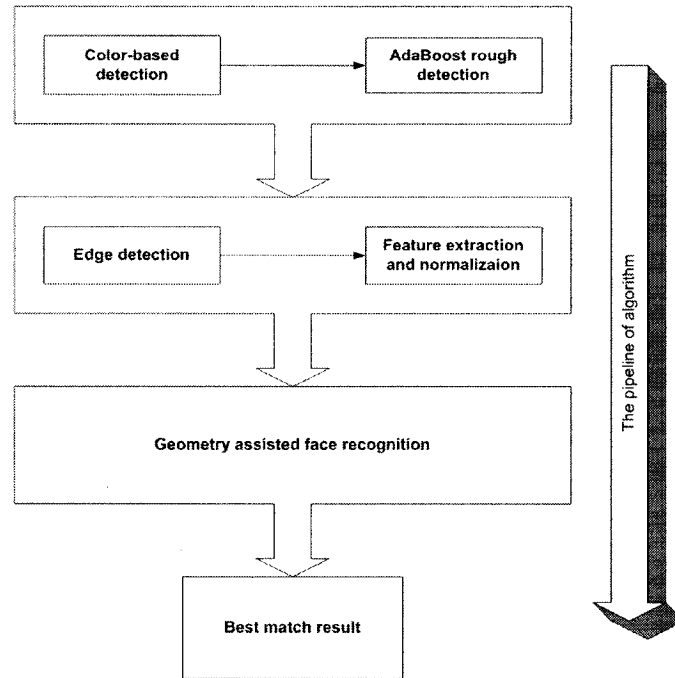
**Figure 2: Rough face detector & pose estimator**

Since the previous output still contains faces with varying scale and position, and the estimated pose is not accurate enough to continue the recognition task, the direct use of unaligned facial images will potentially lead to identification failure. We need to accurately locate the candidate face. To this end, nine statistical shape models- ASM (Active Shape Models) are adopted to estimate the accurate position of model of the input image and to extract facial features as well. Based on results of rough pose detectors, we utilize ASMs of adjacent poses on detected image, for example, carrying out ASMs of degree of Left 45, Left 22.5 and Left 67.5 on previously roughly detected left 45 degree face, since the best ASM has the smallest deviation (the detailed explanation described in section 4.6.2 below), not only the accurate facial features can be found but also the more realistic pose is estimated in the same time. Figure 3 shows the process of accurate facial feature detection. Please note that the three ASMs in the middle of the figure are not the same when dealing with different candidates, but they are always adjacent to each other.



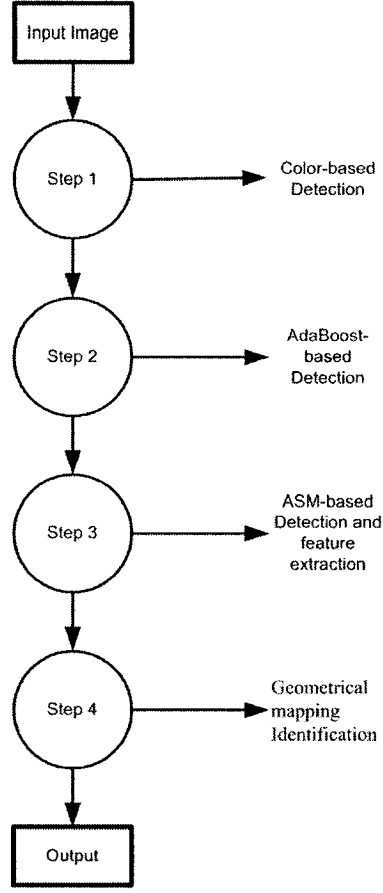
**Figure 3: Accurate face detector**

In the recognition step, we adopted geometrical mapping technique to deal with the pose variation, which essentially maps each face image onto the surface of a 3D ellipsoid. All recognition is then performed on the surface of the ellipsoid. Geometrical mapping could be considered as one way of registering the faces, compensating the pose variation and as well as reducing the intra-subject variations. The following Figure 4 shows the overall architecture of our algorithms.



**Figure 4: Algorithm architecture.**

The matching process for candidate images is shown in Figure 5. The results of our research were published in 20th Canadian Conference on Electrical and Computer Engineering (CCECE 2007) [44].



**Figure 5: Face recognition process flow.**

## 1.4 Organization of Thesis

The remainder of this thesis is organized as follows. In chapter 2, we give an overview of related works on face detection and recognition. In chapter 3, we introduce the major face database used in this thesis and discuss data standard, data collection and preprocessing. Chapter 4 illustrates each processing model of detectors in detail. In the chapter 5, geometry assisted face recognition approach is presented and applied to pose-robust face recognition.

## **2 Background and Related Work**

Since there are too many publications related to face detection and face recognition to thoroughly review them all. We discuss following ones which represent the state of the art and are the most frequently used in the literature and most relevant to our work.

### **2.1 Face Detection Approaches**

There are many ways to detect a face in a scene. Here is a list of the most common approaches in human face detection.

#### **2.1.1 Skin Color**

The color of human skin is distinctive from the color of many other natural objects, hence color is a very important feature that can be used for face detection. Analyzing the skin-tone color statistics, researchers observed that skin colors are distributed over a small area in the chrominance plane and the major difference between skin tones is intensity.

In the beginning of the color-based face detection algorithm, color segmentation is used to classify skin and nonskin color pixels in the input image. There are many ways for this classification [41][42], and we chose the simplest method. It is based on the fact that skin color is compactly clustered in color spaces such as TSL, YCbCr, and HSV with separated luminance and chrominance components. Thus the image is filtered so that only regions likely to contain human skin are marked. This filter was designed using basic mathematical and image processing functions. The binary skin map and the original image

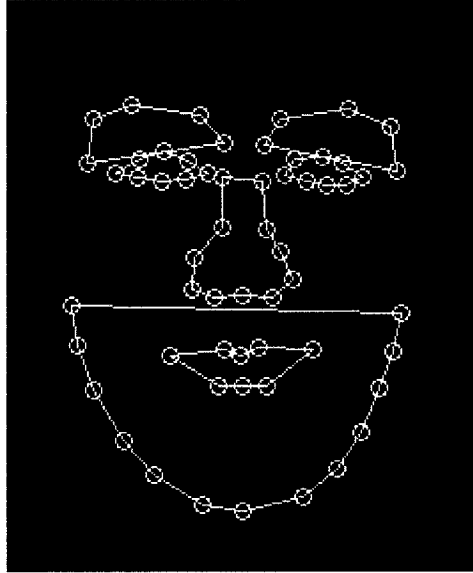
together are used to detect faces in the image. Notice that the criterion for color pruning used in this stage is quite relaxed, therefore, a higher false acceptance rate than a more intricate color-based detection approach is likely to happen. However, for an initial detector in the whole detector cascade, we would rather focus on the speed rather than accuracy and the overall detection performance is reinforced by the succeeding process.

### **2.1.2 Active Shape Model**

Recently, a lot of research have been done on face alignment. Active shape model (ASM), first proposed by Cootes etc[17], is considered to be an efficient method for face alignment and feature localization. ASM is generally divided into three parts: a shape model, an appearance model and the search procedure.

#### **1) Shape Model**

In ASM, a face is represented by a set of labeled points (Figure 6). Each point represents the position of a particular part of the structure. The model is trained by marking landmark points on each of a set of training images. Statistical analysis of the relative positions of the landmarks in different examples allows both the average shape and shape variation to be modeled.



**Figure 6: Face model with labeled points**

## 2) Appearance Model

In ASM, local normalized derivative is used to describe the profile around each landmark [18].

## 3) Search Procedure

Face alignment and feature localization is carried out by finding the desired movement for each landmark with an initial position. Desired movement is achieved by finding the most matched sub-profile, which passes by the original landmark point.

### 2.1.3 Edge-Orientation Matching

Since prominent facial features like face contour, eyes and mouth are often characterized by relatively pronounced edges, edge information (with both strength and orientation) proved to be effective in face processing [16]. The edge map of a given image region is obtained by convolving the image with a horizontal and vertical 3x3 Sobel edge filter. The Sobel method finds edges using the Sobel approximation to the derivative. It returns edges



at those points where the gradient of the image is maximum. The extraction of edge information (strength and orientation) from a 2-D array of pixels  $I(x, y)$  (a grey-scale image) is the basic feature calculation in detection framework. In this work, the Sobel method was used for processing of edge detection. It is a gradient-based method which needs to convolve the image  $I(x, y)$  with two  $3 \times 3$  filter masks. The convolution of the image with the two filter masks gives two edge strength images  $S_x(x, y)$  and  $S_y(x, y)$ . The Edge-Strength image ( $ES$ ), also known as Sobel gradient, and the Edge-Orientation image ( $EO$ ) are generated, given by

$$ES(x, y) = \sqrt{S_x(x, y)^2 + S_y(x, y)^2} \quad (1)$$

$$EO(x, y) = \arctan \frac{S_y(x, y)}{S_x(x, y)} \quad (2)$$

The following figure is an example face processed by Sobel edge filter.



**Figure 7: Face image after processed by Sobel edge filter**

The edge direction as stated in equation (2) takes on values from 0 to  $2\pi$ . The direction of an edge depends on whether the grey value changes from dark to bright or vice versa.

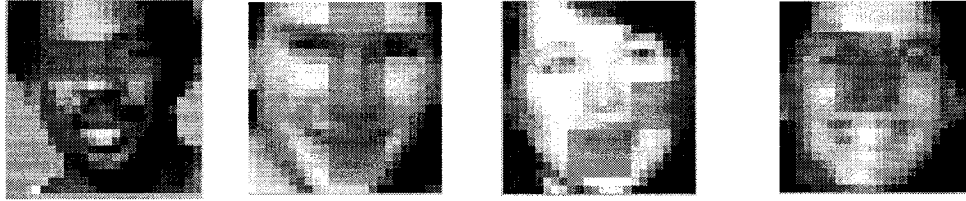
#### **2.1.4 Boosting**

A machine learning approach for face detection which is capable of processing images extremely rapidly and achieving high detection rates by using a new image representation called an integral image that allows for very fast feature evaluation. The integral image can be computed from an image using a few operations per pixel. Once computed, any one of these Harr-like features [2] can be computed at any scale or location in constant time.

Within any image sub-window, the total number of Harr-like features is very large, far larger than the number of pixels. In order to ensure fast classification, the learning process must exclude a large majority of the available features, and focus on a small set of critical features, each weak classifier only depend on a single feature [9]. Figure 8 shows some sample training faces including the features. As a result, each stage of the boosting process, which selects a new weak classifier, can be viewed as a feature selection process. AdaBoost provides an effective learning algorithm and strong bounds on generalization performance combining successively more complex classifiers in a cascade structure which dramatically increases the speed of the detector by focusing attention on promising regions of the image

The idea behind boosting is to sequentially employ a weak learner on a weighted version of a given training sample set to generalize a set of classifiers of its kind. Although any individual classifier may perform slightly better than random guessing, the formed ensemble can provide a very accurate (strong) classifier. Viola and Jones [9] build the first

real-time face detection system by using AdaBoost, which is considered a dramatic breakthrough in the face detection research



**Figure 8: Training image with feature**

## **2.2 Face Recognition Approaches**

A general statement of the face recognition problem can be formulated as follows: Given still or video images of a scene, identify or verify one or more persons in the scene using a stored database of faces. This thesis considers only still images. The basic approaches to recognition of still faces are reviewed next.

### **2.2.1 Principal Component Analysis**

It is well known that there exist significant statistical redundancies in natural images [19]. For a limited class of objects such as face images that are normalized with respect to scale, translation, and rotation, the redundancy is even greater [20]. One of the best global compact representations is KL/PCA, which decorrelates the outputs. Turk and Pentland [6] suggested a classic operation on this approach:

- 1) Acquire an initial set of face images (the training set), and each image has size of  $N^2$ .

- 2) Calculate the eigenfaces from the training set, keeping only the  $M$  eigenfaces ( $M < N^2$ ) that correspond to the highest eigenvalues. These  $M$  images define the face space. As a new face is captured, the eigenfaces can be updated or recalculated.
- 3) Calculate the corresponding distribution in  $M$ -dimensional weight space for each known individual, by projecting their face images onto the "face space."

Having initialized the system, the following steps are then used to recognize new face images:

- 1) Calculate a set of weights based on the input image and the  $M$  eigenfaces by projecting the input image onto each of the eigenfaces.

Input face image  $\Gamma$

$$\Phi = \Gamma - \Psi$$

$$\omega_k = \mu_k^T * (\Gamma - \Psi) \quad (3)$$

where  $\Psi$  is the average face image  $\Psi = \frac{1}{M} \sum_{n=1}^M \Gamma_n$  and  $\mu_k$  is the eigenvector of the covariance matrix  $C$

$$C = \frac{1}{M} \sum_{n=1}^M \Phi_n \Phi_n^T$$

and the weights form a vector

$$\Omega^T = [\omega_1 \dots \omega_M]$$

- 2) Determine if the image is a face at all (whether known or unknown) by checking to see if the image is sufficiently close to "face space."

$$\varepsilon^2 = \|\Omega - \Omega_k\|^2 \quad (4)$$

where  $\Omega_k$  is the vector describing the  $k$ th face class

If  $\varepsilon < \theta$  ( $\theta$  is a threshold)

Then  $\Gamma$  is a face

Else  $\Gamma$  is “unknown”

### 2.2.2 3-D Morphable Model

Human face is a surface lying in the 3-D space. Therefore the 3-D model should be able to represent faces, especially to handle facial variations, such as pose, illumination etc. Blantz et al [12] [22] proposed a method based on a 3-D morphable face model that encodes shape and texture in terms of model parameters, and algorithm that recovers these parameters from a single image of a face.

Starting from an example set of 3D face models, a morphable face model is derived by transforming the shape and texture of the examples into a vector space representation. New faces and expressions can be modeled by forming linear combinations of the prototypes. Shape and texture constraints derived from the statistics of example faces are used to guide manual modeling or automated matching algorithms. Secondly, an approach for face recognition with variations in pose, ranging from frontal to profile views, and across a wide range of illuminations was introduced. In order to account for these variations, the algorithm [24] simulates the process of image formation in 3D space by using computer graphics techniques, and it estimates 3D shape and texture of faces from

single images. The estimate is achieved by fitting a statistical, morphable model of 3D faces to images. The model is learned from a set of textured 3D scans of heads.

### **2.2.3 3-D Face Recognition**

3-D face recognition [25],[26] is an approach having ability to extract the intrinsic geometric features of facial surfaces using geometric invariants and to compare surfaces independent of natural deformations resulting from facial variation such as those resulting from different expressions and poses of the face. The obtained geometric invariants allow mapping 2D facial texture images into special images that incorporate the 3D geometry of the face. These signature images are then decomposed into their principal components. In the first place, the range image and the texture of the face are acquired. Next, the range image is preprocessed by removing certain parts such as hair, which can complicate the recognition process. Finally, a canonical form of the facial surface is computed. Such a representation is insensitive to head orientations and facial expressions, thus significantly simplifying the recognition procedure. The recognition itself is performed on the canonical surfaces. The result is an efficient and accurate face recognition algorithm that is robust to facial variation.

### **2.2.4 SVM**

Support Vector Machines (SVMs) have been recently proposed by Vapnik and his co-workers [23] as a very effective method for general purpose pattern recognition. Intuitively, given a set of points belonging to two classes, a SVM finds the hyperplane that separates the largest possible percentage of points of the same class on the same side, while maximizing the distance from either class to the hyperplane. According to Vapnik,

this hyperplane is called Optimal Separating Hyperplane (OSH) which minimizes the risk of misclassifying not only the examples in the training set but also the unseen examples of the test set. In the field of face recognition [27],[28][29], discrimination functions learned by SVMs can give much higher recognition accuracy than the popular conventional eigenface approach [7] where eigenfaces are used to represent face images. After the features are extracted, the discrimination functions between each pair are learned by SVMs. Then, a different test set enters the system for recognition and a binary tree structure can be built to reinforce recognition on the testing samples.

## 3 Databases

There are many databases used in face detection and face recognition, the choice of an appropriate database to be used should be made based on the given task. It is recommendable to use a standard test data set for researchers to be able to directly compare the results. According to the tasks of this thesis, three image databases are needed in training and testing tasks: database in rough face detection, database in accurate face detection and database in face recognition.

### 3.1 Database for Rough Face Detection

#### 3.1.1 Data collection

To train the rough detector, 6 sets of training images, which are face images with poses of left profile, left 45 degree, frontal, right 45 degree, right profile face and nonface, were used. The frontal face training set consisted of 4916 hand labeled faces scaled and aligned to a base resolution of 24 by 24 pixels (these images are originally from Viola and Jones' work [9]). The other four face sets, namely poses of left 45 degree, right 45 degree, left 90 degree and right 90 degree, each with 1000 images, were extracted from images downloaded during a random crawl of the World Wide Web. Some typical face examples are shown in Figures 9-13. The non-face images used to train the detector were manually inspected and found to not contain any faces from the World Wide Web. Each frontal face classifier was trained with the 4916 training faces and 4916 non-face images (also of size



24 by 24 pixels) using the Adaboost training procedure. Other face classifiers were trained with the 1000 training faces and 1000 non-face sub-windows (size 24 by 24 pixels) respectively using the same training procedure as frontal face. For the initial one feature classifier, the non face training examples were collected by selecting random sub-windows from a set of 4916/1000 images (the number depends on the pose) which did not contain faces. A maximum of 10000 such non-face sub-windows were collected for each layer.

As for testing images, all face testing sets include 200 images which are different from the training sets and each non face testing sets include 200 images which are randomly selected from 10000 testing non face set during the AdaBoost testing procedure.

It is well known that the most important features playing role in recognizing human face from human appearance are eyebrows, eyes, mouth and noses. Based on this rule, all training and testing face pictures are selected. For front faces, the upper boundary is generally cropped at forehead and the low boundary is set at chin. The only limitation on left and right boundaries (from viewer' view) is to make face as close to horizontal center as possible. For faces with out-of-plane rotation, the criterion on upper and low boundary is the same as for frontal faces, left boundary of left faces should be kept some margin space from left edge of face since profile or half profile faces' features are located in one side of face image, vice versa on right face set. However, since there are not many qualified profile images on internet, some half-qualified images are selected as supplement. It may explain why the classifiers of profile face not work well as 45 degrees

classifiers do. The following five figures show the examples of training image sets with five poses.



**Figure 9: The front training face examples**



**Figure 10: The right 45 degree training face examples**



**Figure 11: The left 45 degree training face examples**



**Figure 12: The left 90 degree training face examples**



**Figure 13: The right 90 degree training face examples**

### 3.1.2 Image Processing

All example of faces and nonfaces sub-windows used for training were normalized to minimize the effect of different lighting conditions. Normalization is therefore necessary during detection as well.

The following equations show the image processing procedure before training and testing.

$$x=(x-\text{mean}(x))/\text{std}(x-\text{mean}(x)) \quad (5)$$

$$\text{mean}(x) = \bar{x} = \frac{1}{N} \sum_{i=1}^N x_i = \frac{x_1 + x_2 + \dots + x_N}{N}$$

$$\text{std}(x) = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2}$$

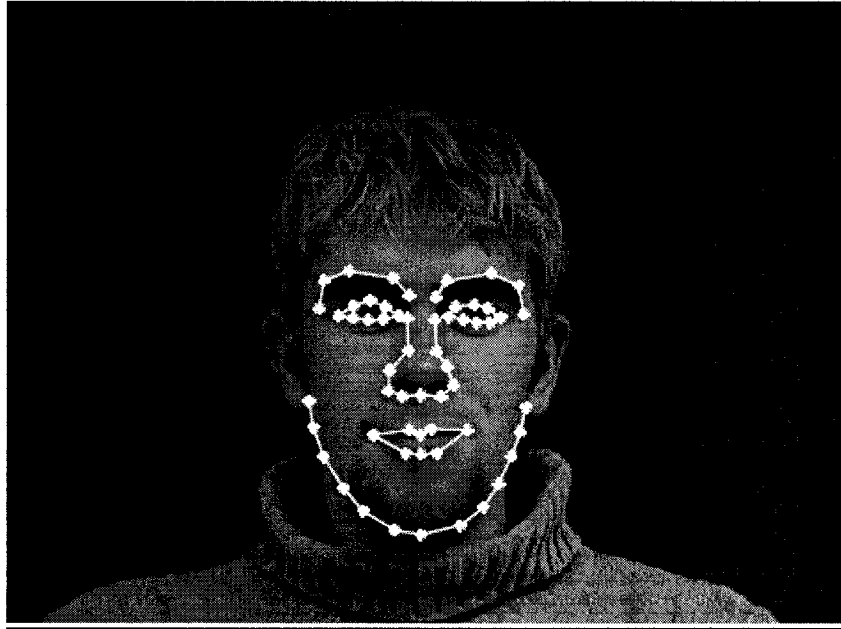
where  $\text{mean}()$  is the general mean operator,  $\text{std}()$  is standard deviation operator,  $x$  is the pixel values within the sub-window in a vector form and  $N$  is the number of pixels. The mean of a sub-window can be computed using the integral image.

Notice that the images may not be visible after normalization since the pixels range from **-5.8486** to **8.7431** in general.

## 3.2 Database for Accurate Face Detection

The IMM face database [30], an annotated dataset of 40 different human face images, is adopted to train the mean face contour model. Points of correspondence are placed on each image so the dataset can be readily used for building statistical models of shape. The IMM face database comprises 240 still images of 40 different human faces, all without glasses. The gender distribution is 7 females and 33 males. Images were acquired in January 2001 using a 640\*480 JPEG format with a Sony DV video camera, DCR-TRV900E PAL. The

following facial structures were manually annotated using 58 landmarks: eyebrows, eyes, nose, mouth and jaw. A total of seven point paths were used; three closed and four open. The following Figure 14 shows an annotated frontal face of IMM.



**Figure 14: An example of annotation**

Since there are only frontal and left and right 30 degree faces in IMM database, we randomly select part of PIE faces [31] with poses of left profile, left 67.5, left 45, left 22.5, right 22.5, right 45, right 67.5 and right profile as training faces on the purpose of making annotated datasets for our Point Distribution Models (PDM) [34]. Except left profile and right profile models, which have less annotated points with three point paths and 32 points, other seven face datasets have total of 58 points and seven point paths -- three closed and four open.

### **3.3 Database in Face Recognition**

PIE Database of CMU [31], the database of CMU Pose, Illumination, and Expression (PIE) database includes 41,368 images of 68 people, each person under 13 different poses, 43 different illumination conditions, and with 4 different expressions.

As mentioned before, the practice only focuses on the pose sensitive environment, only those natural and representative pictures are selected for this purpose. The Figure 15 shows some examples of selected images. The labels below the figure are provided by the author for distinguishing between different poses.



C34



C14



C11



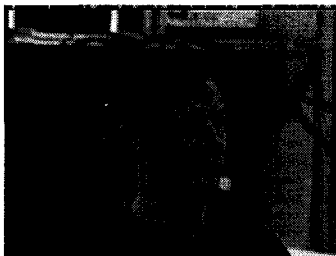
C29



C27



C05



C37



C02



C22

**Figure 15: The image examples from PIE database**



## 4 Face detection

### 4.1 Color-based face detector and rough pose estimator

Color information has been proven to be an effective image feature for coarsely locating potential facial regions. We analyzed a sample set of facial pixels taken from pictures of people of different race, gender, age and under various lighting conditions. We adopted the YCbCr space since the chrominance components CbCr in the YCbCr color space form a condensed cluster and it is perceptually uniform, and widely used. A pixel is classified to be a skin pixel if its color values satisfy:  $Cb_1 < Cb < Cb_2$ ,  $Cr_1 < Cr < Cr_2$ , for example,  $77 < Cb < 127$   $133 < Cr < 173$ . The thresholds are based on skin patches collected from different skin races from the Internet. As a result, skin-regions in an input image can be quickly determined by checking a look-up table and a binary image demonstrates this segmentation. Then, an opening operator is used to remove tiny noise in the segmented image.

Given an input color image as Figure 16, color-based face detection algorithm is as follows:



**Figure 16: Original input color image**

- 1) Find skin pixels and nonskin pixels, then get a skin image (binary image).



**Figure 17: Output binary image**

- 2) Remove noise from the skin image by an opening operator.



**Figure 18: Binary image after opening operator**

Notice that the color pruning criterion used in this stage is quite relaxed, resulting in a higher false acceptance rate. However, for an initial detector in the whole detector cascade, we intend to put more focus on speed than accuracy and the overall detection performance is reinforced by the next processing.

The input color image should be in RGB or BMP format with color intensity values ranging from 0 to 255. Due to restrictions on speed and performance, this project used images with  $484 \times 640$  in size.

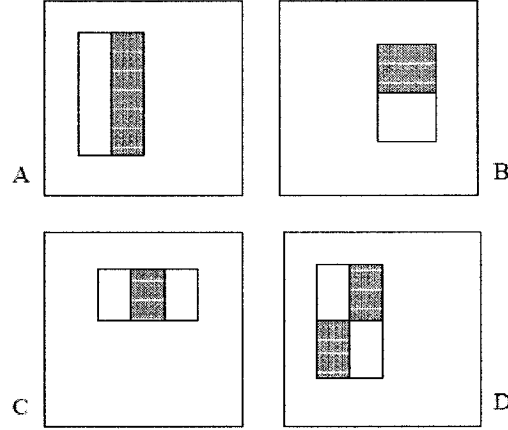
## **4.2 AdaBoost-based face detection**

### **4.2.1 Features**

AdaBoost-based face detection procedure classifies images based on the value of simple features. There are many reasons for using features rather than directly using the pixels. The most common reason is that features can discover ad-hoc domain knowledge that is difficult to learn using training data. The second critical reason is that the feature-based system operates much faster than the pixel-based system.

There are three kinds of features used in the detection procedure (see Figure 19): two-rectangle feature, three-rectangle feature and four-rectangle feature. The value of a two-rectangle feature is the difference between the sum of the pixels within two rectangular regions. The regions have the same size and shape and are horizontally or vertically adjacent. A three-rectangle feature computes the sum within two outside rectangles subtracted from the sum in a center rectangle. Finally a four-rectangle feature computes the difference between diagonal pairs of rectangles. Given that the base

resolution of gray value image is  $24 \times 24$ , the exhaustive set of rectangle features is more than 180,000.

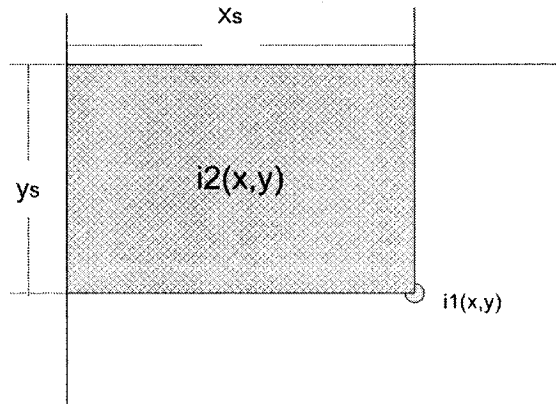


**Figure 19: Rectangle features**

#### 4.2.2 Integral Image

Rectangle features can be computed very fast using integral image which is an intermediate representation for the image. The integral image at location  $(x, y)$  is the sum of the pixels above and to the left of  $(x, y)$  inclusive:

$$i2(x, y) = \sum_{x_s < x, y_s < y} i1(x_s, y_s) \quad (6)$$



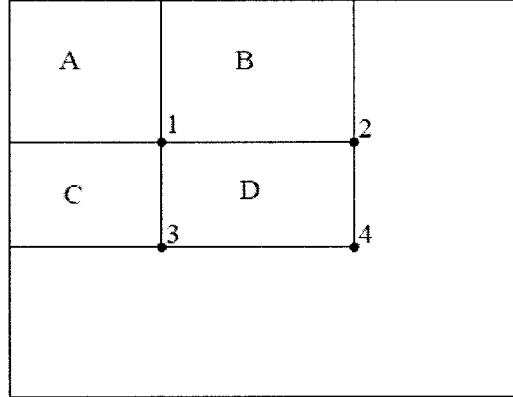
**Figure 20: Integral image**

where  $i2$  is the integral image and  $i1$  is the original image. Using the following equations:

$$s(x, y) = s(x, y-1) + i1(x, y) \quad (7)$$

$$i2(x, y) = i2(x, -1) + s(x, y) \quad (8)$$

where  $s(x, y)$  is the cumulative row sum,  $s(x, -1) = 0$ , and  $i2(-1, y) = 0$



**Figure 21: Computing integral image**

The integral image can be computed very quickly from the original image. For example, the sum of the pixels within rectangle D can be computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value at location 2 is  $A+B$ , The value at location 3 is  $A+C$ , and location 4 is  $A+B+C+D$ . The integral value within D can be calculated as  $4+1-2-3$

#### 4.2.3 Learning Classification Functions

The AdaBoost learning algorithm is used to boost the classification performance of a weak learning algorithm. Freund and Schapire [32] proved that the training error of the strong classifier approaches zero exponentially in the number of rounds. More important is the margin of the examples which represents the distance between positive and negative ones, and AdaBoost achieves large margins rapidly. Recall that there are over 180,000 rectangle

features associated with each image sub-window, a number far larger than the number of pixels. Even though each feature can be computed very efficiently, computing the complete set is prohibitively expensive. According to theory of AdaBoost, there must be a very small number of these features which can be combined to form an effective classifier. The problem is how to find these features. To this end, the weak learning algorithm is designed to find the single feature which separates the positive and negative examples in best way. For each feature, the weak learner determines the optimal threshold classification function which only misclassifies the minimum number of examples. Generally, a weak classifier  $h_i(x)$  uses feature  $f_i$ , a threshold  $\theta_i$  and a parity  $p_i$  indicating the direction of the inequality sign and it is determined by following equation:

$$h_i(x) = \begin{cases} 1 & \text{if } p_i f_i(x) < p_i \theta_i \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

Here  $x$  is a  $24 \times 24$  pixel sub-window of an image. For convenience of explanation, the algorithm of boosting is cited in Table 2.

- Given example images  $(x_1, y_1), \dots, (x_n, y_n)$  where  $y_i=0,1$ , for negative and positive examples respectively.
- Initialize weights  $w_{1,i} = \frac{1}{2m}, \frac{1}{2n}$  for  $y_i=0,1$  respectively, where  $m$  and  $n$  are the number of negatives and positives respectively.
- For  $t = 1:T$  ( $T$  is the number of features)
  1. Normalize the weights,

$$w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^n w_{t,j}}$$

so that  $w_t$  is a probability distribution.

2. For each feature,  $j$ , train a classifier  $h_j$  which is restricted to using a single feature. The error is evaluated with respect to  $w_t$ ,  $\varepsilon_j = \sum_i w_i |h_j(x_i) - y_i|$ .

3. Choose the classifier,  $h_t$ , with the lowest error  $\varepsilon_t$ .

4. Update the weights:

$$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$

where  $e_i=0$  if example  $x_i$  is classified correctly,  $e_i=1$  otherwise, and

$$\beta_t = \frac{\varepsilon_t}{1 - \varepsilon_t}.$$

■ The final strong classifier is:

$$h(x) = \begin{cases} 1 & \sum_{t=1}^T \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^T \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

$$\text{where } \alpha_t = \log \frac{1}{\beta_t}$$

**Table 2: Adaboost algorithm**

In practice, however, a single feature cannot perform the classification task very well. Features selected in early rounds of the boosting process yielded smaller error rates between 0.1 and 0.3. Features selected in later, as the task becomes more difficult, yield

higher error rates between 0.4 and 0.5. The following Table 3 shows the error rates for the first 100 features in front faces training procedure.

0.217416882074048	0.249372145854198	0.337982605897554	0.335136159451548
0.311099728692453	0.352419805241122	0.314315317309299	0.360794914558039
0.377706534309039	0.337470647225941	0.372157748066781	0.338082946005033
0.390601597294953	0.395667459059407	0.403338207508573	0.364886861309005
0.398205601759972	0.402026623539409	0.371678112518419	0.387935884339697
0.394488827399747	0.389931587514685	0.403066501757382	0.402756041230427
0.373232278127317	0.40731343817633	0.399077277693885	0.394693578182786
0.388592864555788	0.413679554806191	0.39740433770493	0.380917122373186
0.415476737449021	0.384875615516151	0.424350995927251	0.416905478447953
0.405676265382858	0.391638592041118	0.407617272770534	0.416905425336689
0.419408303210348	0.423997843332274	0.419190313894789	0.391904239676132
0.412715585599672	0.423867001337388	0.407991417257902	0.424721898058991
0.421255645284539	0.423016316166857	0.429913471098981	0.426723507617801



0.407853436159744	0.416504078841078	0.413962781625027	0.420465043242339
0.429549521273117	0.429308051916065	0.428884023989761	0.428232348119165
0.431974072098753	0.410244389647722	0.421501663534312	0.425365767094102
0.419758012166191	0.431132181739371	0.427432957098281	0.411360319911128
0.417240418347826	0.432116714967818	0.426514218915175	0.437897083766761
0.43343345894173	0.412369034096297	0.422747620674204	0.432383666369697
0.421006708423026	0.424085233118108	0.42557829627753	0.436105478163636
0.410672495142484	0.426373784338713	0.431783344065312	0.430847477685185
0.425049740773109	0.434238101035977	0.434841429073589	0.434162868403974
0.436945304440289	0.432259081381678	0.428838984415802	0.431084268766217
0.424265989747516	0.414042959418853	0.421489385022201	0.435620756977353
0.428234464759664	0.436747802771526	0.433409253903986	0.431439381902035

**Table 3: Error rates of first 100 features in front faces training procedure**

#### **4.2.4 Adaboost Training Results**

To train all 5 poses' strong classifiers is very time consuming job for a general home computer and even for TACITUS server in CS department. The trained first six layers of

each strong classifier and the number of features for each layer are as Table 4 . Please note that the subsequent layers (if the process is continued or the number of layers is bigger than 6) for each classifier may have increasingly more features. The average training time for each frontal face feature is 502 seconds, while the average training time for a face feature of other poses is 93 seconds (speed tested at different time on server TACITUS).

	Number of training face images	Number of non face images	Number of trained features in each layer					
			1	2	3	4	5	6
Left profile	1000	1000	6	13	46	336	563	2463
Left 45	1000	1000	3	13	105	96	130	485
Frontal	4916	4916	2	33	49	76	302	122
Right 45	1000	1000	3	11	47	132	93	807
Right profile	1000	1000	6	11	45	56	104	975

**Table 4: Training results of first six layers of five strong classifiers**

To train a detector, a set of face and nonface training images were used with same quantity (same number of face and nonface got better result). The training set of frontal face consists of 4916 hand labeled faces scaled and aligned to a resolution of 24 by 24 pixels. Other training sets consist of 1000 labeled faces with same resolution as the frontal data

set, the faces were extracted from images downloaded from the world wide web such as Google image, Yahoo image.

#### 4.2.5 Multi-pose classifiers

Based on the previous training procedure and result, it is unrealistic to train all 38 layer cascaded classifiers for all five poses as Viola and Jones did in their work because more than hundred thousands features would be needed to feed multiple face detector and the whole procedure would take several months! It is necessary to adopt another approach on the purpose of efficient training process. A new approach based on AdaBoost is introduced as follows:

First, a value called *summation of detector* is introduced for the purpose of multi-pose classification, and it can be calculated by following equation:

$$\text{sum\_det} = \sum_{t=1}^T \alpha_t h_t(x) \quad (10)$$

where  $\alpha_t$ ,  $\theta_t$ ,  $p_t$  and  $h_t(x)$  are achieved from pose related training procedures and  $h_t(x)$  can be computed from equation (9).

The sum\_det abbreviation for *summation of detectors* refers to first 500 features within 200 test faces of different poses were calculated as Table 5.

Notice that each row of table represents the sum\_det values calculated under different poses. For example, in the row number 2, the first cell shows that the real face pose is left 90 and the values from column 2 to column 6 were calculated using features under poses varying from left 90, left 45, front, right 45, right 90. Only column 2(shaded cell),

however, is in correspondence with the correct pose, while other values are computed by the features of wrong poses.

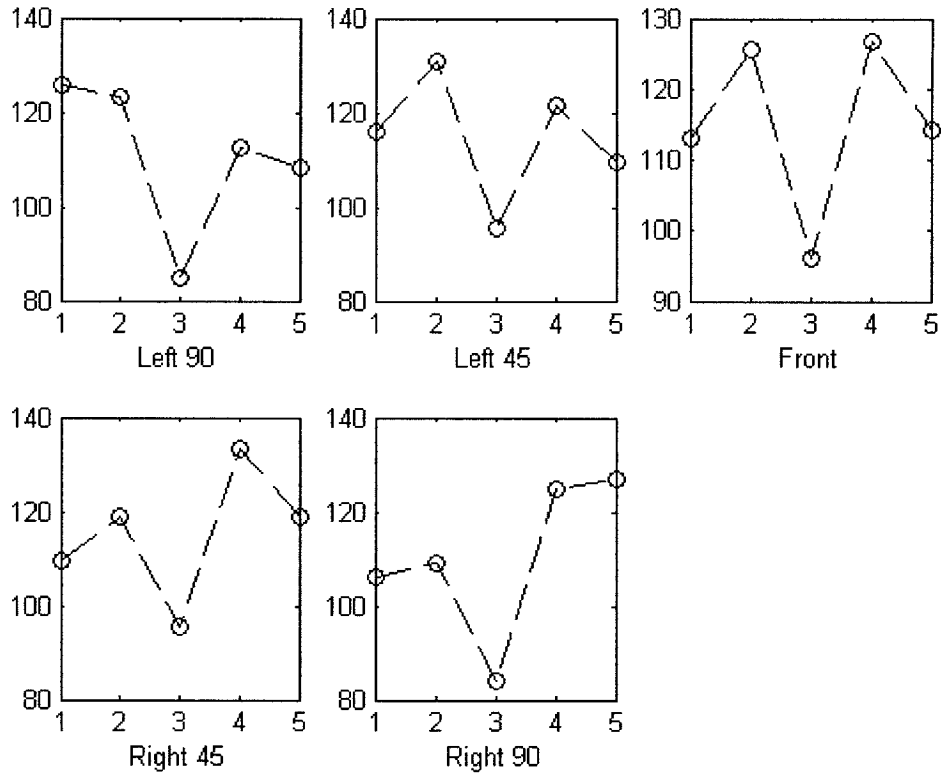
According to the data of following table, an outstanding characteristic is that the biggest value (with shading in table) in each column is the one with matching pose and other values (without shading) in same column are smaller than it.

	Left 90	Left 45	Front	Right 45	Right 90
Left 90	125.7	123.29	84.701	112.45	108.5
Left 45	115.79	130.87	95.692	121.69	109.64
Front	113.02	125.71	96.093	126.85	114.13
Right 45	109.74	118.86	95.439	133.27	119.07
Right 90	106.26	109.03	84.228	124.9	126.93

**Table 5: The first 500 features' results of  $sum\_det$  for 5 poses**

In order to clearly show the distribution of data in comprehensible way, the values in each row are shown as following Figure 22.

The values in X axis are 1, 2, 3, 4 and 5 corresponding to the poses of left 90, left 45, front, right 45 and right 90. The values in Y axis ranging between 80 and 140 correspond to the values of  $sum\_det$ .



**Figure 22: The distribution of  $\text{sum\_det}$  in 5 poses**

However, only these data are not enough to differentiate a pose from 5 possibilities. Thus another  $\text{sum\_det}$  related value is taken into account for pose classification.

Recall AdaBoost algorithm, the strong classifier is defined as follows in Table 2:

$$h(x) = \begin{cases} 1 & \sum_{t=1}^T \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^T \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

where  $\frac{1}{2} \sum_{t=1}^T \alpha_t$  is a threshold which we call it *alpha threshold*.

The following *alpha threshold* values were calculated from first 500 features within 200 test faces of five poses. Notice that  $\alpha_t$ s were achieved from different pose-related training procedures, and all values were calculated with its own matching pose in this time.

	Left 90	Left 45	Front	Right 45	Right 90
Alpha threshold	110.29	114.72	74.98	116.2	111.66

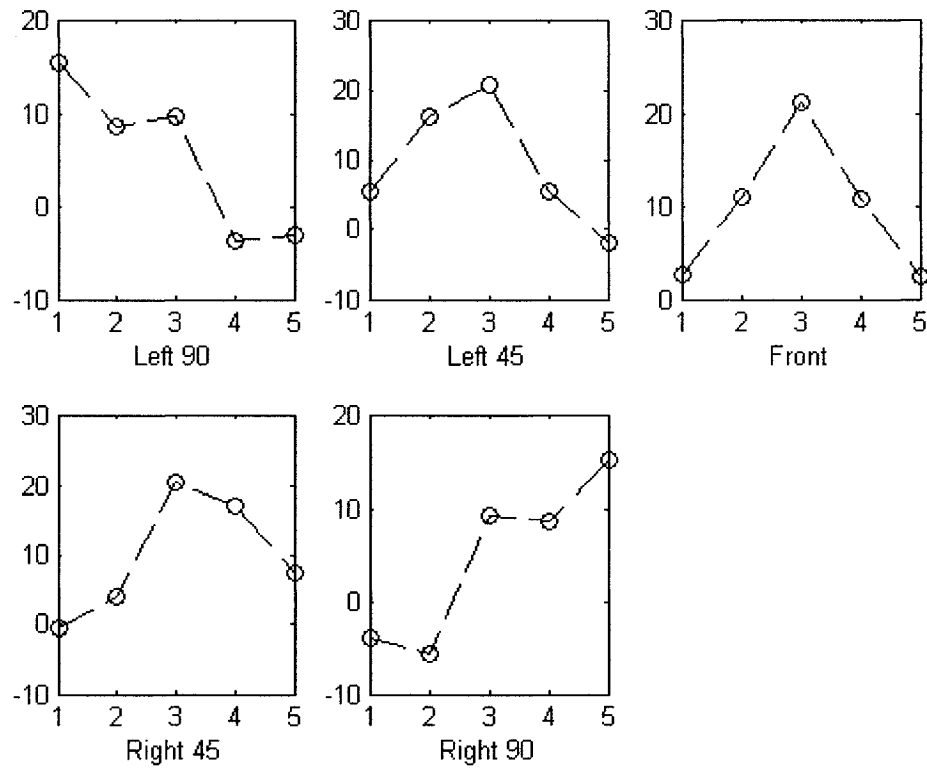
**Table 6: The first 500 features' results of alpha thresholds with matching poses**

Having sum\_dets and alpha thresholds, the *differences* between them calculated with the same manner are as following:

	Left 90	Left 45	Front	Right 45	Right 90
Left 90	15.415	8.5667	9.7197	-3.7483	-3.1617
Left 45	5.5051	16.147	20.71	5.4925	-2.023
Front	2.7315	10.992	21.111	10.654	2.4699
Right 45	-0.54513	4.1361	20.457	17.074	7.4074
Right 90	-4.0333	-5.6974	9.2461	8.7017	15.267

**Table 7: The 500 features' differences between sum\_dets and alpha thresholds**

We still show the distribution of these data by another way, the values of *differences* in each row of previous table are shown as Figure 23.



**Figure 23: The differences between sum\_dets and alpha thresholds**

There are three observations from above figures:

- 1) The figures are roughly symmetrical around front pose, such as left 45 and right 45, left 90 and right 90.
- 2) The shapes for left 90 and right 90 are prominently different from other, the shapes of right 45 and left 45 are not very different from frontal one.
- 3) The *differences* between *summation of detectors* and *alpha thresholds* are more prominent than *summation of detectors*.

Thus the algorithm of rough pose detector can be described as follow:

- For each candidate face sub-window, calculate the summation of detectors

and alpha thresholds according to the AdaBoost algorithm, and then get the differences between them

$$\text{difference}(i) = \text{sum\_det}(i) - \text{threshold}(i)$$

where  $i=1,2,3,4$  and  $5$ ,  $\text{difference}(i)$  represents the result of poses in left 90, left 45, frontal, right 45 and right 90 respectively.

■ Comparing the differences:

$$\text{index} = \max(\text{difference})$$

Switch index

case 1

the pose is left 90

case 5

the pose is right 90

others

\* The pose may be front, left 45 or right 45. And the accurate estimation of pose will be done in next round processing—accurate face detector and pose estimator.

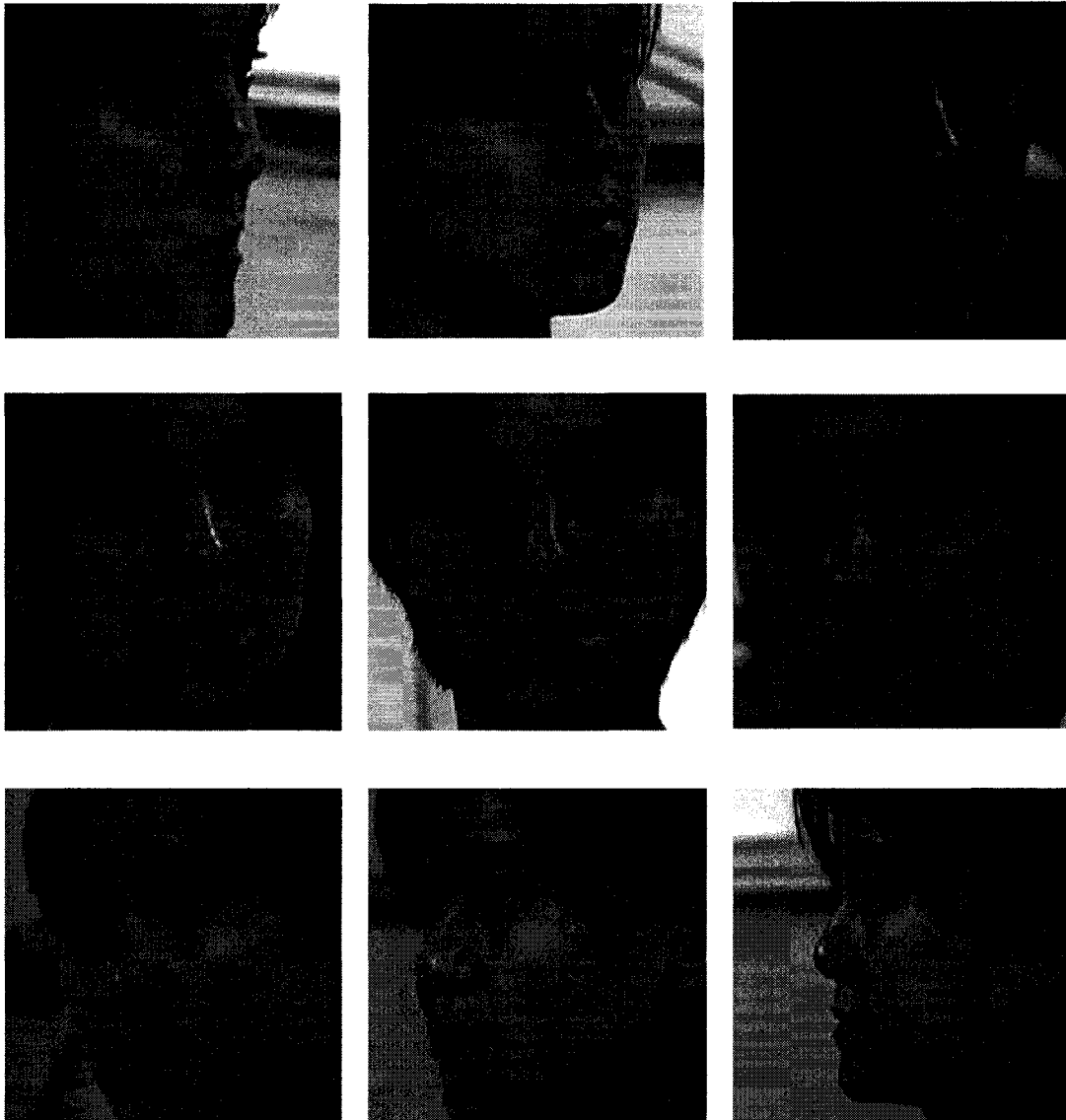
The experiment result with respect to this algorithm is as follow:



	Left 90	Left 45   Front   Right 45	Right 90
Accuracy rate	78.4%	92.1%	84%
Accuracy rate with $\pm 45$ degree tolerance	88.3%		94%

**Table 8 Accuracy rate of rough face estimator**

Some faces detected by previous algorithm are shown as following Figure 24.



**Figure 24: Detected face examples**

### **4.3 Active Shape Models and Gray Level Variations**

After roughly locating the face image and determining the pose of the face, it's time to find the accurate location of face features. The ASM (Active Shape Models)[17] is adopted for this purpose with modifications.

In order to build a model that is flexible enough to cover the most typical variations of faces, nine sets of face images were involved as mentioned in chapter 3.

#### **4.3.1 Point Distribution Model**

An object shape can be represented by a set of labeled points or landmarks. The model used to describe a shape and its typical appearances is based on the variations of the spatial position of each landmark point within the training set. Each point will thus have a certain distribution in the image space and therefore the shape model is being referred to as a Point Distribution Model (PDM). In order to obtain the PDM, we first need to label the landmarks, to align the shapes, and finally, to summarize the landmark variations in a compact form. The number of landmarks should be large enough to show the overall shape of face object. In what follows, detailed steps will be described.

#### **4.3.2 Labeling the Training Set**

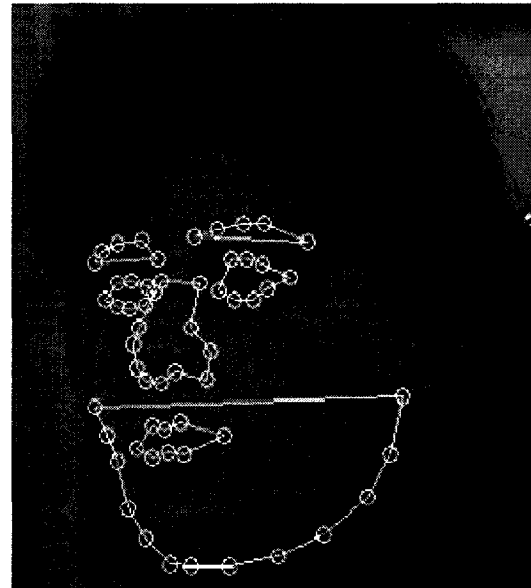
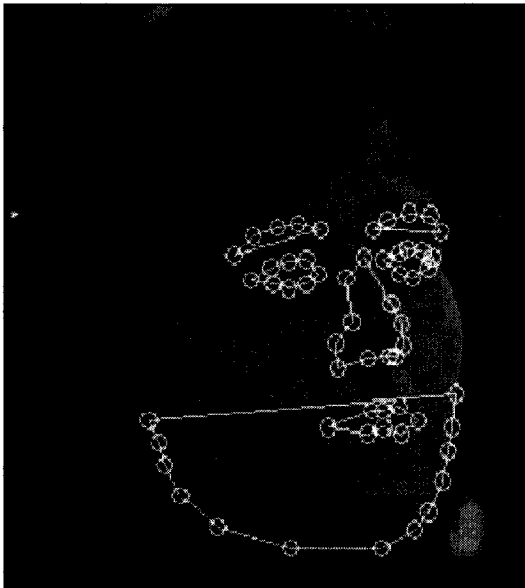
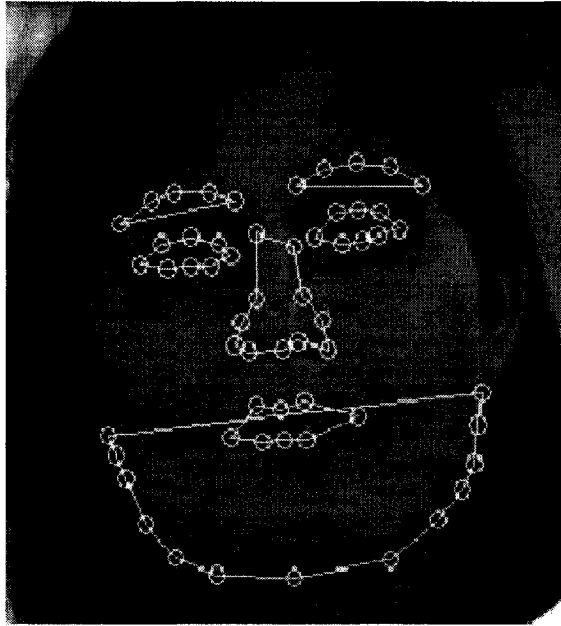
Before labeling the shapes of the training face image set, we need to determine the number of landmark points that can adequately represent the shape. In this study we apply 32 on profile faces and 58 on others since part of face features can not be seen for profile faces. For each image of the training set, we manually locate the shape, and then identify significant landmarks on that shape. It is important that the landmarks are accurately located and that there is an exact correspondence between labels in different instances of training shapes. There are at least three basic types of landmarks that can be used [33] :

- 1) Application-dependent landmarks.
- 2) Application-independent landmarks.

3) Landmarks interpolated from the two above.

Example of application-dependent landmarks in face images are the centers of the eyes. Application independent landmarks may be the highest or lowest point of an object with a certain orientation. Interpolated landmarks can be points which are separated by equal distances or criterion with evident characteristic and located along a certain path between two landmarks of type 1 or 2. Typically, type 3 will dominate and describe most of the boundary of the shape.

Assume that the labeled training set is denoted by  $S$ . It contains  $N$  shapes, each of which has  $n$  landmarks (58 for frontal faces and 45 degree faces, and 32 for profile faces). In other words, there are  $N$  coordinate points for each landmark of the certain shape. The  $j$ th landmark coordinate point of the  $i$ th shape of the training is set by  $(x_{ij}, y_{ij})$ , and the vector describing the  $i$ th shape in the training set by:  $x_i = [(x_{i1}, y_{i1}), (x_{i2}, y_{i2}) \dots (x_{in}, y_{in})]$ , where  $1 \leq i \leq N$  and  $n=58$  or  $32$ . For poses of left 67.5, left 45, left 22.5, front, right 22.5, right 45 and right 67.5, the following facial structures were manually annotated using 58 landmarks: eyebrows, eyes, nose, mouth and jaw. A total of seven point paths were used. The following **Figure 25** shows three manually annotated face examples.



**Figure 25: Manually annotated face examples for poses of left 67.5, left 45, left 22.5, front, right 22.5, right 45 and right 67.5**

For poses of left 90 and right 90, the following facial structures were manually annotated using 32 landmarks: left eyebrow, left eye, nose, mouth and jaw (for pose of right 90, it would be right eyebrow and right eye). A total of three point paths were used.



**Figure 26: Manually annotated face examples for poses of left 90 and right 90**

### 4.3.3 Aligning Shape

#### 1. Aligning Two Shapes

In order to get the variations of each landmark throughout the specific training set, all shapes, each of which is represented by its corresponding landmark vector  $\mathbf{x}$ , must be aligned to each other. This is done by changing the pose (scale, rotation, and translation) of consecutive shapes until the complete set is properly aligned. We take two shapes as an example first, given two vectors  $\mathbf{x}_i$  and  $\mathbf{x}_j$ , the scaling value  $s$ , the rotation angle  $q$  and the value of translation  $(t_x, t_y)$  can be found by aligning the vector  $\mathbf{x}_j$  to  $\mathbf{x}_i$  in a weighted least-squares sense[34]. Then we can achieve significance of the more stable landmark points by weighting. The stability of a point is measured by the amount of variation in the distance between that point and the other points.

#### 2. Aligning All Shapes

Then the set of  $N$  shapes can be aligned with each other. The pose of a shape is described by its scaling, rotation and translation, with respect to a known reference. If aligning all shapes, normalization must be made first. Normalization of the pose includes:

- 1) *Scaling*      the distance between two points becomes a certain constant
- 2) *Rotation*      the line joining two pre-specified landmarks is directed in a certain direction
- 3) *Translation*      the shape becomes centered at a certain coordinate.

The purpose of normalization is to force the process to converge, otherwise the mean shape may rotate, translate or expand (or shrink) indefinitely. Convergence is achieved if the shape changing is not more than a pre-defined threshold.

#### 4.3.4 Statistics

Assume that the  $i$ th aligned shape of the training set of face images is represented by vector  $\mathbf{x}_i$ , where  $\mathbf{x}_i$  contains the new coordinates resulting from alignment. This vector is of dimension  $2n$  (including  $n$  Xs and  $n$  Ys coordinates), so it can be represented by a point in a  $2n$ -dimensional space. The  $N$  vectors representing the  $N$  aligned shapes will then map to a ‘cloud’ of  $N$  points in the same  $2n$ -D space [17]. Assume that these  $N$  points are contained within a region of this  $2n$ -D space referred to as the ‘Allowable Shape Domain’ (ASD). We know that the shorter the Euclidean distance between two points (that represent two shapes in same pose) the more similar to the two shapes. The weighted Euclidean distance  $d$  between the two points representing the two shapes  $\mathbf{x}_i$  and  $\mathbf{x}_k$  is given by:

$$d_{ik} = \sqrt{(x_i - x_k)^T W (x_i - x_k)} \quad (11)$$

where

$$x_i = [(x_{i1}, y_{i1}), (x_{i2}, y_{i2}) \dots (x_{in}, y_{in})]$$

$$W = \text{diag}(w_1, w_1, w_2, w_2, \dots, w_n, w_n)$$

The weighting matrix  $W$  is used to give more importance to those landmark points that vary less in the training set. Now, we want to find those dominating the behavior of the variations of the  $N$  points in the  $2n$ -D space defined by the  $2n$  variables of  $x$ . We can generate a new set of variables called the principal components after applying Principal Component Analysis (PCA). Each principal component is a linear combination of the original variables [36]. All the principal components are orthogonal to each other so there is no redundant information. All principal components form an orthogonal basis for the space of data. Generally speaking, it can be assumed that the first few principal components describe a high percentage of the total variance of the original data. Hence, the dimension of the model can be reduced and the variations can be described by a less number of variables (much less than  $2n$ ). Now, each landmark vector can be represented as a linear combination of the principal components. Moreover, we can express the difference between each vector and the mean of all vectors as a linear combination of the principal components, because this difference vector will also lie in the  $2n$ -D space spanned by the principal components. Denoting the mean vector by  $\bar{x}$ , and the difference vector between the vector  $x_i$  and  $\bar{x}$  by  $d_{xi}$ , we have

$$d_{xi} = x_i - \bar{x} \quad (12)$$



where  $\bar{\mathbf{x}} = \frac{1}{N} \sum_{i=1}^n \mathbf{x}_i$

The covariance matrix for the landmarks is given by

$$\mathbf{C}_x = \frac{1}{N} \sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \quad (13)$$

As mentioned in [34], we can represent the difference  $d\mathbf{x}_i$  as a linear combination of the principal components,

$$d\mathbf{x}_i = b_{i0}\mathbf{p}_0 + b_{i1}\mathbf{p}_1 + \dots + b_{i2n-1}\mathbf{p}_{2n-1} \quad (14)$$

where  $\mathbf{p}_l$  is the  $l$ th principle component axis or vector and  $b_{il}$  is a scalar that weighs  $\mathbf{p}_l$ . And also the principal components are mutually orthogonal, so they are orthonormal. Thus we have

$$\mathbf{x}_i = \bar{\mathbf{x}} + d\mathbf{x}_i$$

where  $d\mathbf{x}_i = \mathbf{P}\mathbf{b}_i$

This yields  $\mathbf{x}_i = \bar{\mathbf{x}} + \mathbf{P}\mathbf{b}_i$  and  $\mathbf{b}_i = \mathbf{P}^{-1}(\mathbf{x}_i - \bar{\mathbf{x}})$

Since  $\mathbf{P}$  is an orthogonal matrix, we have  $\mathbf{P}^{-1} = \mathbf{P}^T$  and

$$\mathbf{b}_i = \mathbf{P}^T(\mathbf{x}_i - \bar{\mathbf{x}}) \quad (15)$$

where  $1 \leq i \leq N$

In order to reduce the dimension of the data and describe the variations with a fewer number of variables, we now express the  $N$  landmark vectors as the sum of their mean  $\mathbf{x}$  and a weighted sum of some of the principal components. Assume that the first  $d$  (out of  $2n$ ) principal components explain a sufficiently high percentage of the total variance of the

original data such as 90 percent. If only  $d$  principle components, the basic equation becomes

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{P}\mathbf{b} \quad (16)$$

where

$$\mathbf{b} = [b_1, b_2, \dots, b_d] \quad \text{and}$$

$$\mathbf{P} = [p_1, p_2, \dots, p_d]$$

#### 4.1.1 Modeling the Gray Level Appearance

The main idea of modeling the gray level information is to examine the gray levels in a region around each landmark throughout the training set. In general, any region around a landmark can be studied, but we only focus on the gray levels along a line passing through the landmark. For every landmark point  $j$  in the image  $i$  of the training set, we extract a gray level profile  $\mathbf{g}_{ij}$  with length of  $p$  pixels centered at the landmark point. Instead of using the actual gray level profile, its normalized derivative is adopted. This gives invariance to the offsets and uniform scaling of the gray levels [37].

The gray level profile of the landmark  $j$  in the image  $i$  is a vector of  $p$  values,  $\mathbf{g}_{ij} = [g_{ij1} \ g_{ij2} \dots \ g_{ijp}]$  and the derivative profile of length  $p - 1$  becomes  $\mathbf{dg}_{ij} = [g_{ij2} - g_{ij1} \ g_{ij3} - g_{ij2} \ \dots \ g_{ijp} - g_{ijp-1}]$ .

The normalized derivative profile is given by

$$y_{ij} = \frac{dg_{ij}}{\sum_{m=1}^{p-1} |dg_{ijm}|} \quad (17)$$

Now, we calculate the mean of the normalized derivative profiles of each landmark throughout the training set for landmark  $j$

$$\bar{y}_j = \frac{1}{N} \sum_{i=1}^N dg_{ij} \quad (18)$$

The covariance matrix of the normalized derivative is given by

$$C_{yj} = \frac{1}{N} \sum_{i=1}^N (y_{ij} - \bar{y}_j)(y_{ij} - \bar{y}_j)^T \quad (19)$$

Thus we get a model for the gray levels around any landmark  $j$  represented by  $\bar{y}_j$  and  $C_{yj}$ .

#### 4.4 Image Search Using Active Shape Modeling

The combined technique of PDM and iterative model deforming, updating and matching is called Active Shape Modeling (ASM).

##### 4.4.1 The Initial Shape Estimate

According to equation(16), a shape of an object  $x_i$  can be described as the sum of the mean shape obtained from the training set and a weighted sum of the principle components, with the possibility of this sum being translated, rotated, and scaled. That is, we can express the initial estimate  $x_i$  of a shape as a scaled, rotated and translated version of reference shape  $x_1$ :

$$x_i = M(s_i, \theta_i)[x_1] + t_i \quad (20)$$

where

$$M(s, \theta) = s \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

and

$$\mathbf{t}_i = [t_{xi} \ t_{yi} \ t_{xi} \ t_{yi} \ \dots \ t_{xi} \ t_{yi}]^T$$

The matrix  $M(s, \theta)$  scales the shape by  $s$ , rotates it by  $\theta$ , and the vector  $\mathbf{t}_i$  translates the shape in the  $x$  and  $y$  directions respectively.

The shape  $\mathbf{x}_i$  can be expressed as  $\mathbf{x}_i = \bar{\mathbf{x}} + d\mathbf{x}_i$ , where  $d\mathbf{x}_i = \mathbf{P}\mathbf{b}_i$ . Then, the initial estimate can be written as  $\mathbf{x}_i = M(s_i, \theta_i)[\bar{\mathbf{x}} + d\mathbf{x}_i] + \mathbf{t}_i$

#### 4.4.2 Updating the Shape

After examining the image region around each landmark point of  $\mathbf{x}_i$ , a new desired location  $\mathbf{x}_i + d\mathbf{x}_i$  is obtained by following steps. First, an adjustment should be made to the pose (scaling, rotation and translation) parameters, as well as the shape parameters (the weights of the principal components) in order to move current estimate  $\mathbf{x}_i$  as close as possible to  $\mathbf{x}_i + d\mathbf{x}_i$ , while still satisfying the shape constraints imposed to produce an allowable shape[35]. To do that, we first find the additional scaling  $1 + ds$ , rotation  $d\theta$  and translation  $(dt_x, dt_y)$ , required to move  $\mathbf{x}_i$  as close as possible to  $\mathbf{x}_i + d\mathbf{x}_i$ . Secondly, we need to solve the following equation for  $d\mathbf{x}$ :

$$M(s_i(1 + ds), \theta_i + d\theta)[\mathbf{x}_i + d\mathbf{x}] + \mathbf{t}_i + d\mathbf{t} = \mathbf{x}_i + d\mathbf{x}_i$$

Since  $\mathbf{x}_i = M(s_i, \theta_i)[\mathbf{x}_i] + \mathbf{t}_i$

then

$$dx = M((s_i(1 + ds))^{-1}, -(\theta_i + d\theta)) [M(s_i, \theta_i) [x_i] + dx_i - dt] - x_i \quad (21)$$

In general, the resulting vector  $dx$  is in  $2n$ -D space, but since there are only  $d$  (less than  $2n$ ) modes of variation described in our model, we can only move the shape in  $d$  dimensions described by the first  $d$  principal axes. So we seek the vector that is most similar to  $dx$  but lies in the  $d$ -D space. If we adopt the least-squares approach, then the solution  $dx'$  is the projection of  $dx$  onto the  $d$ -D space (the space spanned by the vectors of principal components, or the  $d$  columns of  $P$ ). We have:

$$dx' = A dx$$

where  $A = P(P^T P)^{-1} P^T$ .

Since the columns of  $P$  are orthonormal and  $P$  is no longer square, we have  $P^T P = I$  and thus  $dx' = P P^T dx$ .

So,  $x_i$  will actually move to  $x_i + dx'$ . Since  $dx' = P db'$  and multiplying by  $P^T$  from the left, we get  $db' = P^T dx'$

We are now ready to update the shape and pose parameters of our initial estimate. We obtain a new estimate  $x_i^{(1)}$  where

$$x_i^{(1)} = M(s_i(1 + ds), \theta_i + d\theta) [x_i + P db'] + t_i + dt \quad (22)$$

We then repeat the same step on  $x_i^{(1)}$  as we started with  $x_i$  and produce  $x_i^{(2)}$ , and so on, until no significant change in the shape is noticeable.

Notice that the resulting (updated) shape is within the allowable shape domain which is done by limiting the values of  $\mathbf{b}$ .

#### 4.4.3 Finding the Desired Movements

Here we will describe how the modeling of gray level statistics around each landmark can be used to determine the adjustment of each landmark ( $dx_i$ ). To find such adjustments, we search along a line passing through the landmark and perpendicular to the boundary formed by the landmark and its neighbors. In this way, we obtain a search profile. Within this search profile, we look for a sub-profile with characteristics that match the ones obtained from training. In order to do so, we collect the gray level values along the search profile, compute the derivative and normalize it. We then search within the normalized derivative search profile (having length  $ns$ ) for a sub-profile that matches the mean normalized derivative profile (of length  $p$ ) obtained from the training set.

The search profile along the landmark  $i$  is given by

$$s_i = [s_{i1}, s_{i2} \dots s_{ins}]$$

The derivative search profile of landmark  $i$  will be of length  $ns - 1$  as follows:

$$ds_i = [s_{i2} - s_{i1}, s_{i3} - s_{i2} \dots s_{ins} - s_{i(ns-1)}]$$

The normalized derivative search profile is

$$y_{si} = \frac{ds_i}{\sum_{m=1}^{ns-1} |ds_{im}|} \quad (23)$$

Then we analyze  $y_{si}$  for sub-profiles that match  $y_i$  which is the mean normalized derivative profile obtained from the training process. Denoting the sub-interval of  $y_{si}$  centred at the  $d$ th pixel of  $y_{si}$  by  $h(d)$ , then we try to find the value of  $d$  that makes the sub-interval  $h(d)$

most similar to  $\mathbf{y}_i$  . This can be done by defining the following square error function (which decreases as the fit becomes better) and minimizing it.







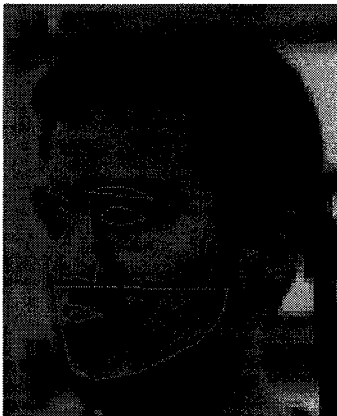
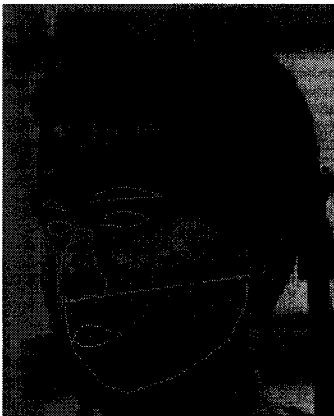
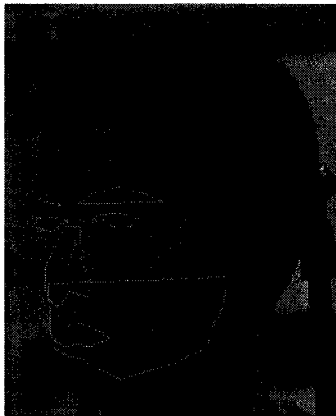
$$f(d) = (\mathbf{h}(d) - \bar{\mathbf{y}}_i)^T \mathbf{C}_{\mathbf{y}_i}^{-1} (\mathbf{h}(d) - \bar{\mathbf{y}}_i) \quad (24)$$

where  $\mathbf{C}_{\mathbf{y}_i}^{-1}$  is the inverse of the covariance matrix of  $\mathbf{y}_i$  .

In this way, we determine the location of the point to which the landmark  $i$  should move. Also, the same procedure is carried out for all the landmark points to obtain the adjustment vector  $\mathbf{dx}_i$ .

#### 4.5 Multi-Resolution Image Search

How to choose the length of the search profile  $ns$  still is a critical problem that will affect the image search considerably since we need to consider two contradictories in the same time. First, the search profile should be long enough to contain the target point to which the original landmark is supposed to move. On the other hand, we want the search process along searching profile to be as short as possible in order to reduce the computation. Also, if the searching profile is long and the target point is close to the current position of the landmark then it may move too far and miss the target. This problem can be solved by a multi-resolution approach. First, the search is carried out in large extent to include far points. Then, when the search progresses closer to a target structure, the search is limited to near points. In order to achieve such multi-resolution search, a pyramid of images with different resolutions is generated. At the base of the pyramid (Level 0) we have the original image and on higher levels (Level 1 to  $L-1$ ) we decrease the resolution by a factor of two (see Figure 27).

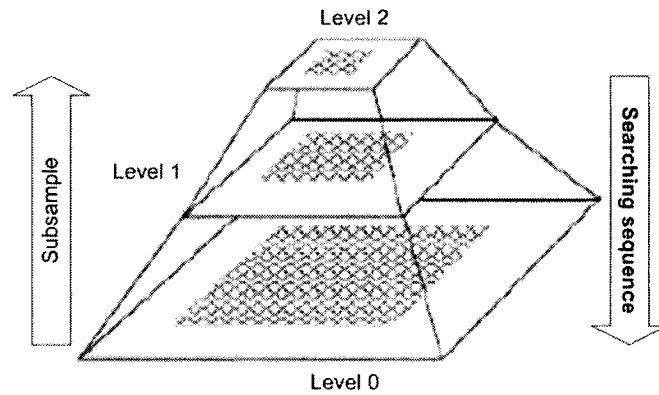
	Level 2	Level 1	Level 0
Non matching pose			
matching pose			
Non matching pose			

**Figure 27: The progress of multi resolution image search**

In order to obtain the pyramid, we subsample the image at the lower level in every second pixel to get the image at the higher level [38]. Level 0 is the original image. Level 1 is an



image with half the number of pixels along axes, and so on. We start searching at the top level of the pyramid and then continue at a lower level using the search output of the previous level (Figure 28). In level 1 and 2, ASM just searches for an approximate position. So the detail losing phenomena seems not to be an important factor for accuracy of the algorithm. The procedure is repeated until the lowest level (the original image) is reached. In order to carry out this multi-resolution search, we must use the information about the gray level profiles at each of these levels. This means that during the training stage, we need to obtain the mean normalized derivative profile for each landmark in *all* the pyramidal levels.



**Figure 28: The process of pyramidal image**

We denote the mean normalized derivative profile for the landmark  $i$  at the pyramidal level  $l$  by  $\bar{y}_{il}$  where  $0 \leq i \leq n-1$  and  $0 \leq l \leq L-1$ . The mean is obtained by calculating the average of the normalized profile for a certain landmark along the  $N$  images of the training set. Before searching process, we build a criterion for determining when to change the level of search within the pyramid. One possibility is to move to a lower level (detailed image) when a certain percentage of the land marks do not change considerably, for example when 95% of the landmarks move only within the central 50% of the search

profile[38]. A maximum number of iterations such as 20,30,40 or 50 can also be used to avoid getting stuck at a higher level.

## 4.6 Facial Feature Extraction and Accurate Pose Estimator

### 4.6.1 Face Edge Detection

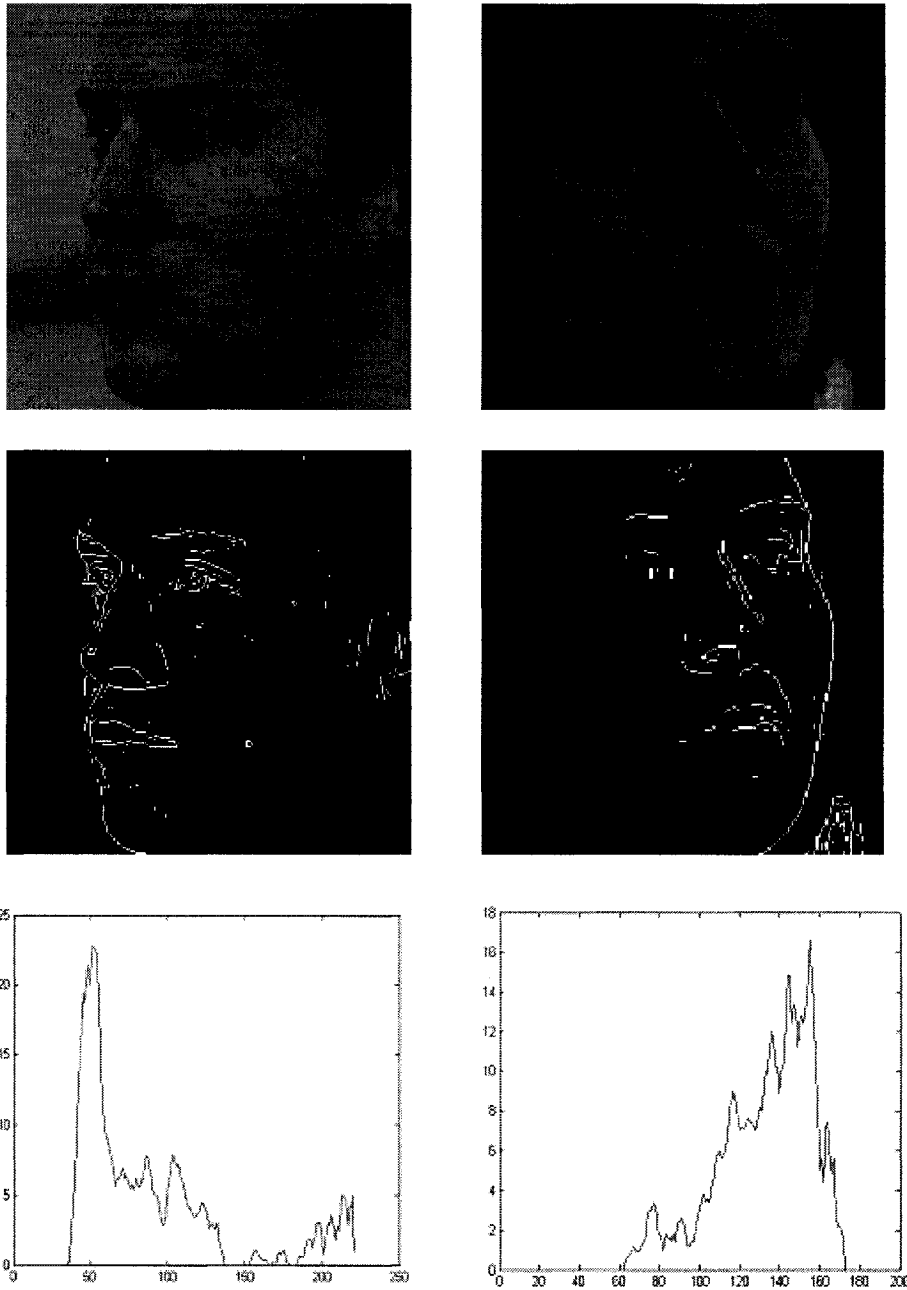
Since most of detected face sub-windows have nonface margin on one side which is inherited from the characteristic of corresponding training faces, normally the left faces (like poses of left 22.5, left 45 ...) have left margin, and the right faces (like poses of right 22.5, right 45 ...) have right margin. In order to speed up the processing and improve the accuracy, it is necessary to carry out face edge detection.

The following is the algorithm in face edge detection.

- Finding the edge contour of detected face sub-window by using sobel operation which we mentioned in section 2.1.3
- Calculate the sum of sobel matrix in vertical direction and result a vector `sum_sobel`. The distribution of `sum_sobel` is as bottom line of following figure
- The coordinate of biggest value in the *first* (toward to the opposition of face direction) half of `sum_sobel` is the edge.

**Table 9: The algorithm in face edge detection**

The following is the figures expressing the process of face edge detection.



**Figure 29: The process of face edge detection.**

#### 4.6.2 Accurate Pose Estimator and Face Detection

As mentioned in section 4.2, the rough pose estimators may not work very well on poses of left 45, front and right 45 faces. Thus an ASM-based technique is proposed to reinforce the pose estimation and features extraction.

Recall the equation in chapter 4,

$$\mathbf{x}_i^{(1)} = M(s_i(1 + ds), \theta_i + d\theta)[\mathbf{x}l + Pdb'] + \mathbf{t}_i + dt$$

and

$$f(d) = (\mathbf{h}(d) - \bar{\mathbf{y}}_i)^T \mathbf{C}_{y_i}^{-1} (\mathbf{h}(d) - \bar{\mathbf{y}}_i)$$

Assume that the original shape of object  $\mathbf{x}_i$  is set by mean shape of PDM. Thus we can get  $\mathbf{x}_i^{(1)} \mathbf{x}_i^{(2)} \dots \mathbf{x}_i^{(K)}$  in the same manner as above approach. Here  $\mathbf{K}$  can be interpreted as the number of search loops varying from 20 to 50 according to need. Please note that the iterations of searching procedure must be carried out in each pyramidal level. After minimizing the square error function  $f(d)$ , the  $dx_i$  and  $Pdb'$  can be estimated in each iterative step.

Assume that the vector  $X_{end}$  describes the shape of  $\mathbf{x}_i$  after pyramidal searching process with  $\mathbf{K}$  times' iteration in each level and  $X_{begin}$  is the original  $\mathbf{x}_i$ , the difference between them is  $diffX$ , so we have

$$diffX = X_{end} - X_{begin} \tag{25}$$

It is obvious that the value of  $diffX$  express the variation of  $\mathbf{x}_i$ . In other words, if the components of  $diffX$  have large absolute value, it means that the original shape changed dramatically during searching procedure. On the other hand, if the components of  $diffX$

have small absolute values, it shows that the original shape just changed a bit during searching procedure and this happened only when the face shape are very similar to the trained mean shape with certain pose. If several trained mean shapes with adjacent poses are carried out on candidate sub-windows for the purpose of detecting the matching pose among these shape models, for instance, to carry out the trained mean shapes Left 67.5, Left 45 and Left 22.5 on the roughly detected Left 45 pose, the one with smallest variation of  $diffX$  would be the most likely one, and experiment data in Table 10 proved this observation in detail. Furthermore, considering the side effect of sign of  $diffX$ , we use standard deviation of  $diffX$  instead of  $diffX$ ,

$$StddiffX = \text{std}(diffX) \quad (26)$$

Where  $\text{std}()$  stand for standard deviation operation.

According to the experiment, there are two observations of  $StddiffXs$ .

- 1) Points of a shape moved as a whole in same direction are intended to lead small  $StddiffX$  value. This can be found in “matching pose” section of Figure 27.
- 2) When the shape model mismatches the face contour, it will result random adjustment of points which explains why bigger  $StddiffX$  values are generated in this situation. This can be illuminated by the “Non matching pose” part of Figure 27.

Since the whole data sheet is too big to be shown here, only part of accurate detection data are represented as following Table 10:

No. of faces	Real pose left 67.5			Real pose left 45			Real pose left 22.5		
	Trained mean shape								
	left 67.5	left 45	left 22.5	left 67.5	left 45	left 22.5	left 67.5	left 45	left 22.5
1	2.623	7.699	7.839	3.405	4.673	10.307	5.422	4.554	3.713
2	2.222	6.805	5.553	10.352	4.191	8.004	9.561	11.704	3.350
3	5.404	5.816	10.444	9.027	1.829	2.807	7.116	6.133	2.977
4	4.739	7.529	11.506	11.254	6.859	6.158	16.629	7.153	7.505
5	2.390	20.867	19.233	3.539	5.734	15.341	5.452	1.967	1.941
6	4.049	7.960	14.787	14.710	7.861	10.161	10.538	5.071	5.426
7	2.670	6.243	17.784	7.571	7.293	6.142	11.445	2.810	6.526
8	3.235	9.316	10.128	14.870	3.833	4.013	10.621	2.608	4.426
9	5.378	6.615	9.497	10.155	9.926	3.065	11.827	5.191	5.942
10	1.910	17.091	10.656	8.733	14.172	13.447	9.806	4.765	11.946
11	4.239	14.898	18.072	7.004	5.547	3.173	13.468	5.919	5.277
12	4.873	5.176	11.052	8.483	5.903	4.801	8.548	7.060	7.601
13	2.883	5.394	6.669	13.031	2.444	6.112	23.900	5.445	6.868
14	2.168	8.616	9.552	8.109	2.626	4.625	8.724	2.338	3.863

15	11.988	13.348	13.889	15.876	6.234	8.817	6.675	16.912	19.436
16	6.151	9.743	9.143	8.257	5.534	10.194	12.899	14.898	10.654
17	3.942	19.219	14.055	7.907	2.260	0.931	13.629	7.886	2.661
18	28.460	34.101	25.871	4.789	5.582	4.782	16.736	6.237	3.083
19	3.785	11.991	13.170	2.893	5.072	9.679	12.481	4.928	4.211
20	7.435	9.791	10.673	3.421	10.152	7.745	15.010	16.977	9.223
21	5.033	4.139	6.182	11.885	4.308	3.927	7.869	16.182	7.113
22	3.928	9.212	21.716	16.761	3.235	4.720	12.598	4.306	1.957
23	7.204	11.893	13.121	18.741	5.653	15.622	12.837	4.952	6.779
24	2.924	8.168	9.955	3.968	5.839	5.671	9.315	4.954	3.155
25	2.011	6.962	15.634	6.663	1.917	4.653	6.796	5.830	2.977
26	3.289	9.130	20.524	15.148	9.718	3.447	11.299	3.888	3.579
27	1.667	18.000	18.788	4.235	7.663	21.073	4.954	4.041	4.612
28	10.907	9.213	7.475	9.503	19.012	6.135	9.694	21.915	5.618
29	2.566	12.389	13.441	7.456	2.611	7.255	12.525	4.762	7.213
30	2.923	9.307	8.282	8.400	5.876	3.590	9.068	11.325	5.699
31	7.980	7.030	24.369	8.103	3.625	3.402	11.745	3.664	5.718

32	6.416	11.660	15.722	9.078	6.660	6.717	11.169	9.118	5.837
----	-------	--------	--------	-------	-------	-------	--------	-------	-------

**Table 10: part of pose accurate detection results with respect to left 67.5, left 45 and left 22.5 poses**

According to above data, 60 out of 96 *StddiffX* values (62.5%) with matching pose are the smallest values among three adjacent candidate models, and 93 values out of 96 (96.9%) within a range of  $\pm 22.5$  are smallest ones among three adjacent candidate models.

There are two reasons to explain why some of the smallest data not appear on the right places:

- 1) Not every face ‘looks like’ corresponding mean face model, some of them may ‘look like’ mean face model with other adjacent poses.
- 2) Some smallest values can not represent the results since the values are far bigger than the allowable extent of value such as 10.00, it results failure of accurate location of facial feature. Taking the No. 18 of column “Real pose left 67.5” for example, all three values are 28.460, 34.101 and 25.871 which are too big to be accepted and may be resulted by the failure of previous rough face detector.

Notice that even the result may not appear on ‘right’ place, most the smallest data within allowable extent still can accurately locate the facial feature, such as eyes, eyebrows, noses and jaw. Actually this is an outstanding merit of the ASM technique, after iterative seeking in three pyramidal levels, the selected mean shape model is the one having smallest *StddiffX* value which also means the smallest variation, and it does n’ t matter which one of candidates is ‘right’ at all.



Then the similar approach is used for all pose estimators. Based on the rough face detectors and rough pose estimator, the location of candidate face can be determined by following algorithm:

- If pose is estimated as left 90 by previous rough pose estimator

Then calculate three stds of corresponding trained mean shapes:

stdL90, stdL67.5 and std45

[minval index]=min(stdL90, stdL67.5, stdL45)

where min() is minimum function which return the minimum and the index of the minimum.

- If pose is estimated as right 90 by previous rough pose estimator

Then calculate three stds of corresponding trained mean shapes:

stdR90, stdR67.5 and stdR45

[minval index]=min(stdR90, stdR67.5, stdR45)

where min() is minimum function which return the minimum and the index of the minimum.

- If pose is estimated among right 45, right 45 or front poses by previous rough pose estimator

Then calculate three stds of corresponding trained mean shapes:

stdL45, stdFront and stdR45

[minval index1]=min(stdL45, stdFront, stdR45)

Switch index1

Case 1

Then calculate two stds of corresponding trained mean  
shapes:stdL67.5 and stdL22.5

[minval index2]=min(stdL67.5, stdL45, stdL22.5)

Switch index2

Case 1

The pose is left 67.5

Case 2

The pose is left 45

Case 3

The pose is left 22.5

end

Case 2

Then calculate two stds of corresponding trained mean  
shapes:stdL22.5 and stdR22.5

[minval index2]=min(stdL22.5, stdFront, stdR22.5)

Switch index2

Case 1

The pose is left 22.5

Case 2

The pose is front

Case 3

The pose is right 22.5

end

Case 3

Then calculate two stds of corresponding trained mean shapes:

stdR22.5 and stdR67.5

[minval index2]=min(stdR22.5, stdR45, stdR67.5)

Switch index2

Case 1

The pose is right 22.5

Case 2

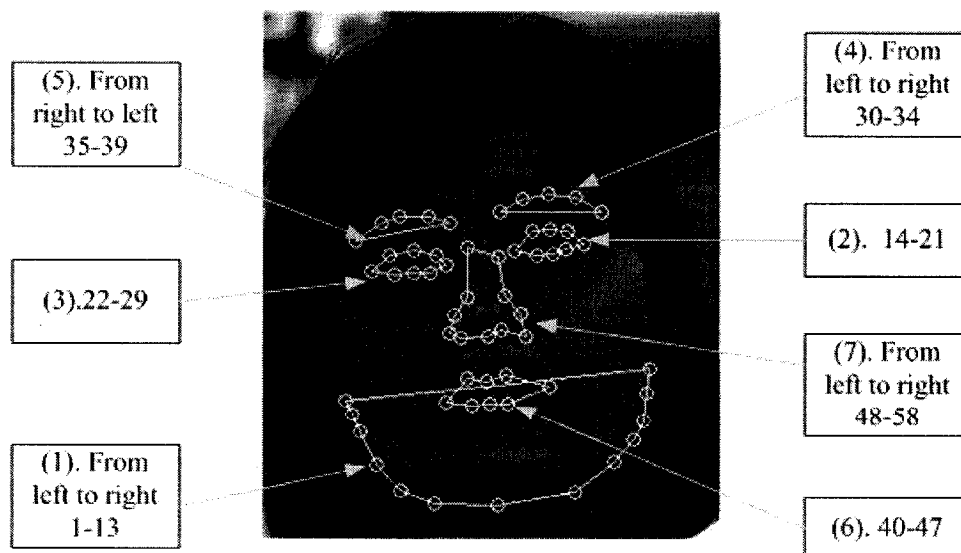
The pose is right 45

Case 3
The pose is right 67.5
end
end

**Table 11: Algorithm of pose detection**

#### 4.6.3 Facial Feature Extraction

After accurate localization and edge detection of candidate faces, it is time to extract the facial features from detected face image. The Figure 30 shows 58 key points in a detected frontal facial shape. The others 6 shape models, such as left 67.5, left 45, left 22.5, right 22.5, right 45 and right 67.5, have the same numbers and annotated sequences as frontal one.



**Figure 30: Facial features localization**

In above figure, each text box has a parenthesized number describing the number of path it belongs to; for example, in textbox No.3, “22-29” represents the left eye path with 8 points aligning from number 22 to number 29.

Experiment results show that the most prominent features on the face are eyes, noses and mouth, this is also accordant with the observation of human vision, that is to say that the paths 2, 3 and 6 are most important on face recognition task. Furthermore, these three paths have bigger gradient variation than other paths of face such as jaw and nose, thus the points of shape model around eyes and mouth are unlikely to have big update during iterative searching procedure. The following Table 12 is the extracting and cropping algorithm.

- Calculate the mean coordinates of two eyes (16 points) in the vertical directions  $Y_e$  and  $X_e$ .
- Calculate the mean coordinates of mouth (8 points) in the vertical and horizontal directions  $Y_m$  and  $X_m$ .
- Distance between  $Y_e$  and  $Y_m$

$$DisY = Y_m - Y_e$$

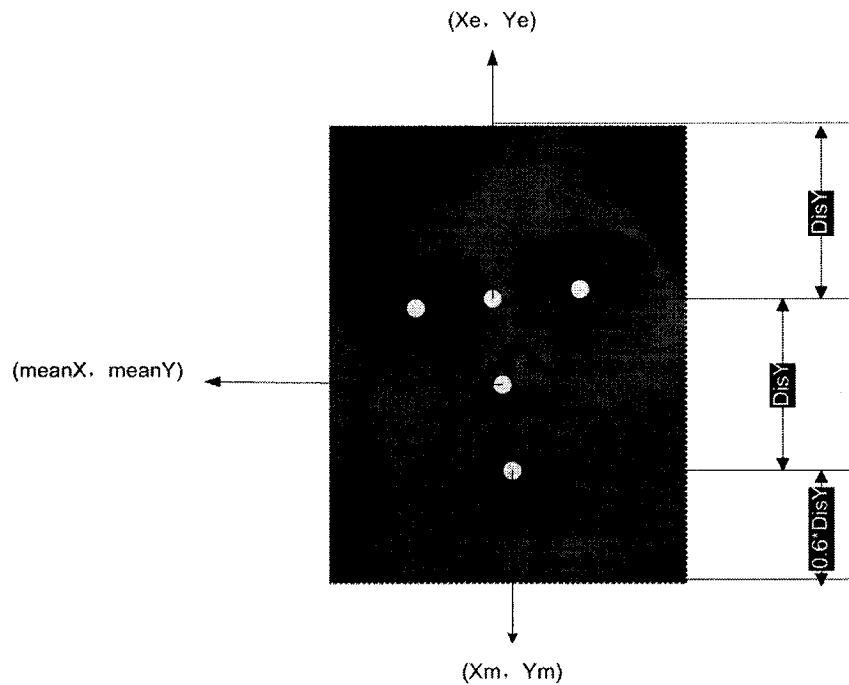
- Mean of  $X_e$  and  $X_m$

$$meanX = (X_e + X_m) / 2$$

- Set boundaries for cropping
  - Left boundary  $B_l = \text{meanX} - \text{DisY}$
  - Right boundary  $B_r = \text{meanX} + \text{DisY}$ ,
  - Up boundary  $B_u = Y_e - \text{DisY}$
  - Bottom boundary  $B_b = Y_m + \text{rate}^* \times \text{DisY}$

\* *rate* is generally set to 0.6-0.8 If rate is too big, some irrelevant part like neck may be cropped into target image.

**Table 12: Extracting and Cropping Algorithm in Testing Stage**

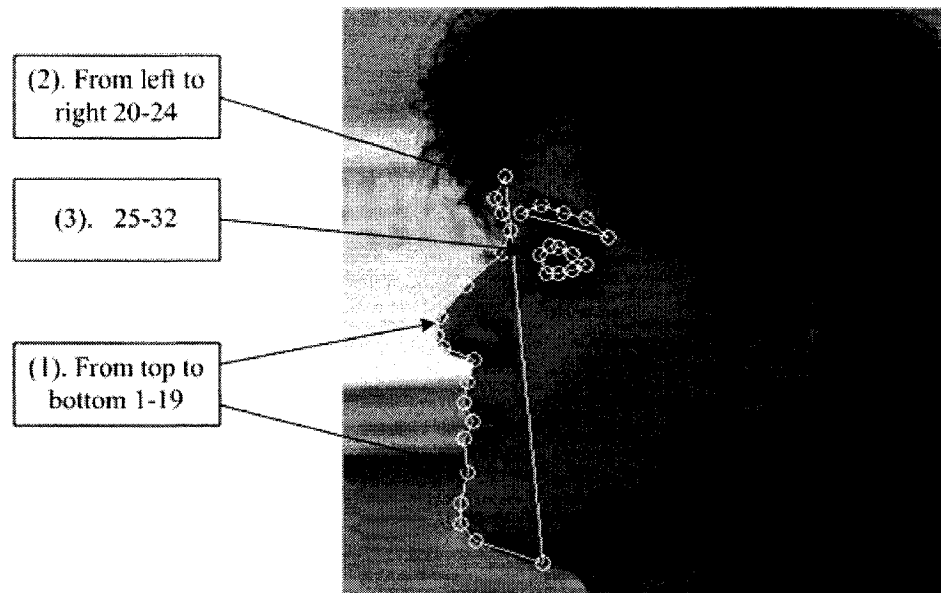


**Figure 31: The measurement of extraction and the key points of cropping**

Please note that the previous algorithm is dedicated to the feature extraction of frontal face during testing stage; since we can utilize the edge detection on faces with out-of plane

rotation, the feature extraction routine of other poses may implement in the similar manner, even the profile ones.

Another concern must be addressed, however, the shape models with poses of left 90 and right 90 have some differences since one side of face features such as eye and eyebrow can not be seen (Figure 32).



**Figure 32: Facial features localization in profile**

In above figure, shape model in pose left 90 just has 3 paths pointed by textboxes which describe the point in each path.

In order to compare the face data set of pose left 90 and right 90 with other poses, a common standard must be built to carry out meaningful comparison among faces with different pose, we found that sub-window faces with the poses of left 90 and right 90 have same characteristics as other poses have.

There are two of them:

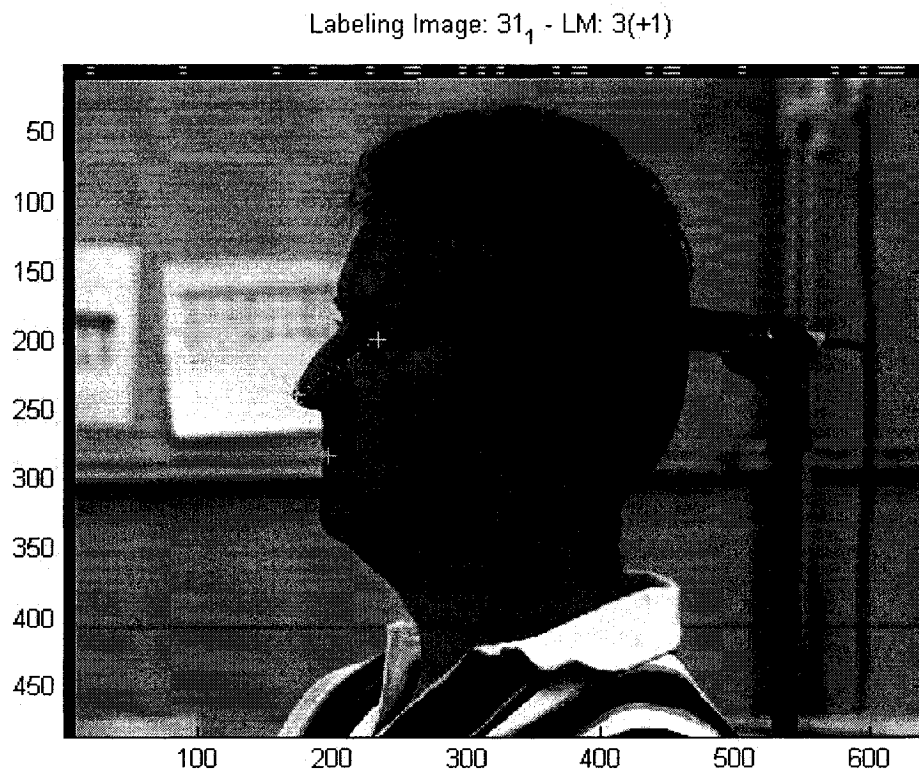
- 1) The paths of eyes and mouth in any pose have greater gradient variation than other part of face. In other words, we can find points of eye and mouth paths by ASMs no matter which pose the face is, thus we can take advantage of this characteristic in face cropping and comparison.
- 2) Most prominent and comparable parts in face are eyes, eyebrows, noses, mouth and the areas around these features. Other parts such as hair, ears and even jaw have very limited contributions to face recognition.



## 5 Multi-pose face recognition

### 5.1 Reference Image Sets Collection

In order to carry out mug shot face recognition in multi-pose environment, the reference image data for comparison, namely the datasets of poses of left 90, right 90 and front, should be created first. The main procedure to produce these three face data sets are similar with the one mentioned previously; however, instead of using automatic approach to locate the face feature, manual approach is introduced to collect these data sets for accurate purpose(see Figure 33).



### Figure 33: Manually select the three key points in profile image

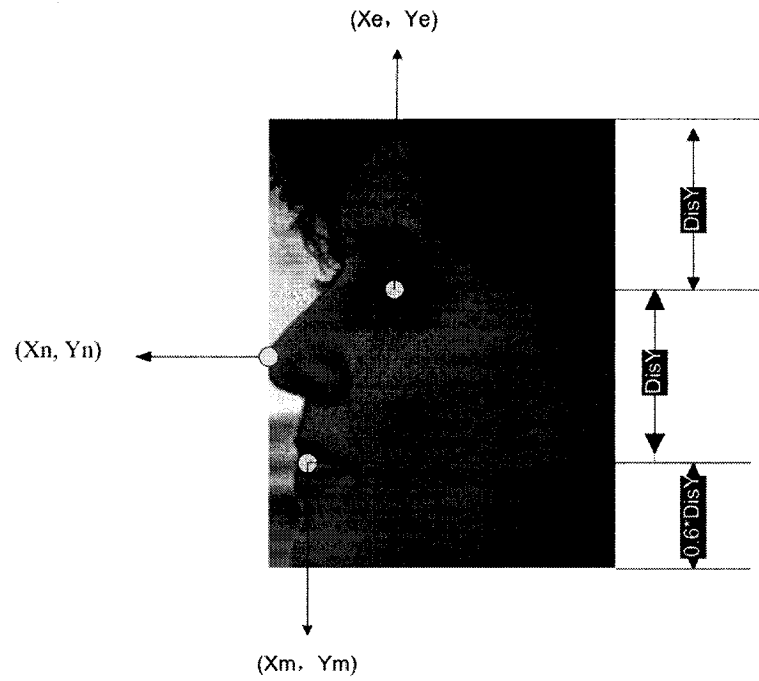
We choose three points in each picture of three image data sets as “+” point in above figure. The algorithm to extract and crop in training step is as follows:

- Select the eye centre( $X_e, Y_e$ ), Tip of nose( $X_n, Y_n$ ) and mouth( $X_m, Y_m$ )
- Distance between  $Y_e$  and  $Y_m$ 
  - $DisY = Y_m - Y_e$
- Set boundaries for cropping
  - Left boundary  $B_l = X_n$
  - Right boundary  $B_r = B_l + 2 * DisY$
  - Up boundary  $B_u = Y_e - DisY$
  - Bottom boundary  $B_b = Y_m + rate * DisY$

\* rate generally is set as 0.6-0.8. If rate is too big, some irrelevant part like neck may be cropped into target image. This rate should be identical with the one in Table 12

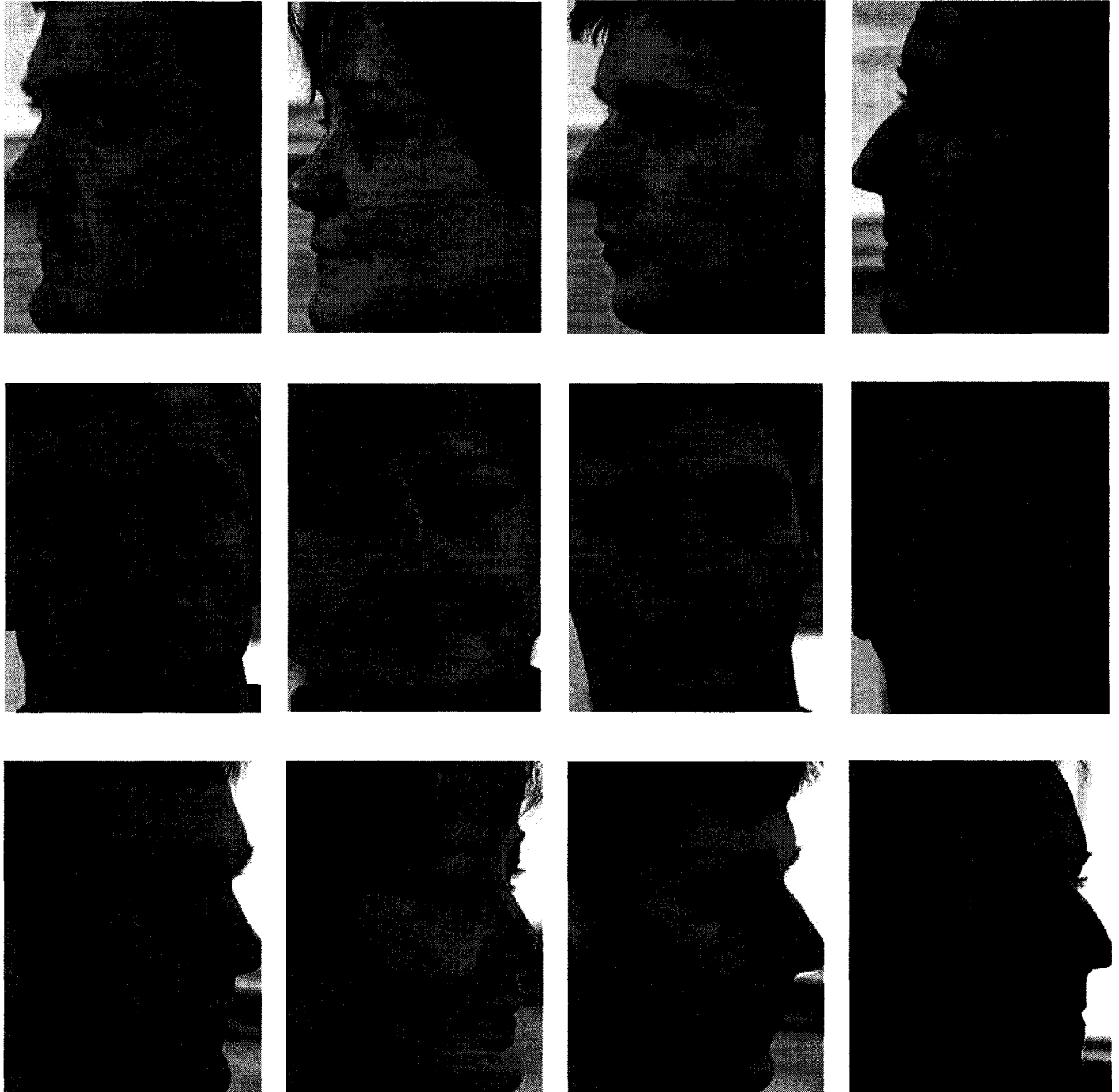
**Table 13: Extracting and Cropping Algorithm in Training Stage**

The following figure shows the selected coordinators and the variables use in above algorithm.



**Figure 34: The measurement of extraction and the key points of cropping for profile image**

Some cropped results are as follow:



**Figure 35: Cropped results of profile images**

## **5.2 Geometrical mapping**

It is always failure to compare two face images of the same subject captured at two different view angles pixel-by-pixel because these two images are not registered/aligned

with respect to each other and the pixel-by-pixel difference is relatively big. Many traditional approaches do not work well under the circumstance of pose variation. To solve this problem, we can take advantage of image registration to fix this problem.

It is well known that a human head has the non-planar geometry, one way to register face images is to project images back to the surface of a 3D ellipsoid based on their specific poses. The procedure of back projection is called geometrical mapping, which is a key component in our face recognition algorithm. In this section, we will introduce how to generate a texture map  $\mathbf{s}$  from a face image  $\mathbf{f}$  with a known mapping parameter  $\mathbf{x}$ . Two assumptions are made. First, assume that a human head is an approximate 3D ellipsoid with radiuses  $r_x$ ,  $r_y$  and  $r_z$ . Second, a face image is captured with the weak-perspective camera model [43] and the camera's focal length equals to one. Thus mapping parameter  $\mathbf{x}$  describing the relation between a face image and its texture map is as follows:

$$\mathbf{x} = [cv \ ch \ d \ R\alpha \ R\beta \ R\chi]^T$$

Where  $cv$  and  $ch$  are the center of the face area in the image,  $d$  indicates the average distance between the face and the camera, and  $R\alpha$ ,  $R\beta$  and  $R\chi$  indicate the rotation of the human head in three dimensions respectively.

In order to generate a texture map  $\mathbf{s}$  from  $\mathbf{f}$ , for each pixel in  $\mathbf{s}(\alpha, \beta)$ , we need to find its corresponding coordinate  $\mathbf{f}(v, u)$  with the mapping parameter  $\mathbf{x}$ . The parameters  $v$  and  $u$  are the axes of the original image;  $\alpha$  and  $\beta$  are the axes of the texture map. There are four steps for the mapping from  $\mathbf{s}(\alpha, \beta)$  to  $\mathbf{f}(v, u)$  as shown in Figure 36.

- 1) A pixel  $s(\alpha, \beta)$  in the texture map corresponds to one coordinate  $(P_x, P_y, P_z)$  on the surface of a sphere, whose radius is one:

$$\begin{cases} P_x = \sin(\alpha)\sin(\beta) \\ P_y = \cos(\alpha) \\ P_z = \sin(\alpha)\cos(\beta) \end{cases}$$

As shown in the left part of Figure 36, the sphere is then converted into an ellipsoid by stretching each radius according to  $rx$ ,  $ry$ , and  $rz$ :

$$\begin{cases} P_x = rxP_x \\ P_y = ryP_y \\ P_z = rzP_z \end{cases}$$

- 2) Rotate the head ellipsoid by  $R\alpha$ ,  $R\beta$  and  $R\chi$  with respect to the XYZ axes. Since all test face image only rotate around Y axis, the rotation is simplified. As shown in Figure 36,  $(P_x, P_y, P_z)$  moves to a new coordinate  $(P'_x, P'_y, P'_z)$  by the following equation ( $R\alpha$  and  $R\chi$  are zero):

$$\begin{bmatrix} P'_x \\ P'_y \\ P'_z \end{bmatrix} = \begin{bmatrix} \cos(R_\beta) & 0 & \sin(R_\beta) \\ 0 & 1 & 0 \\ -\sin(R_\beta) & 0 & \cos(R_\beta) \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \quad (27)$$

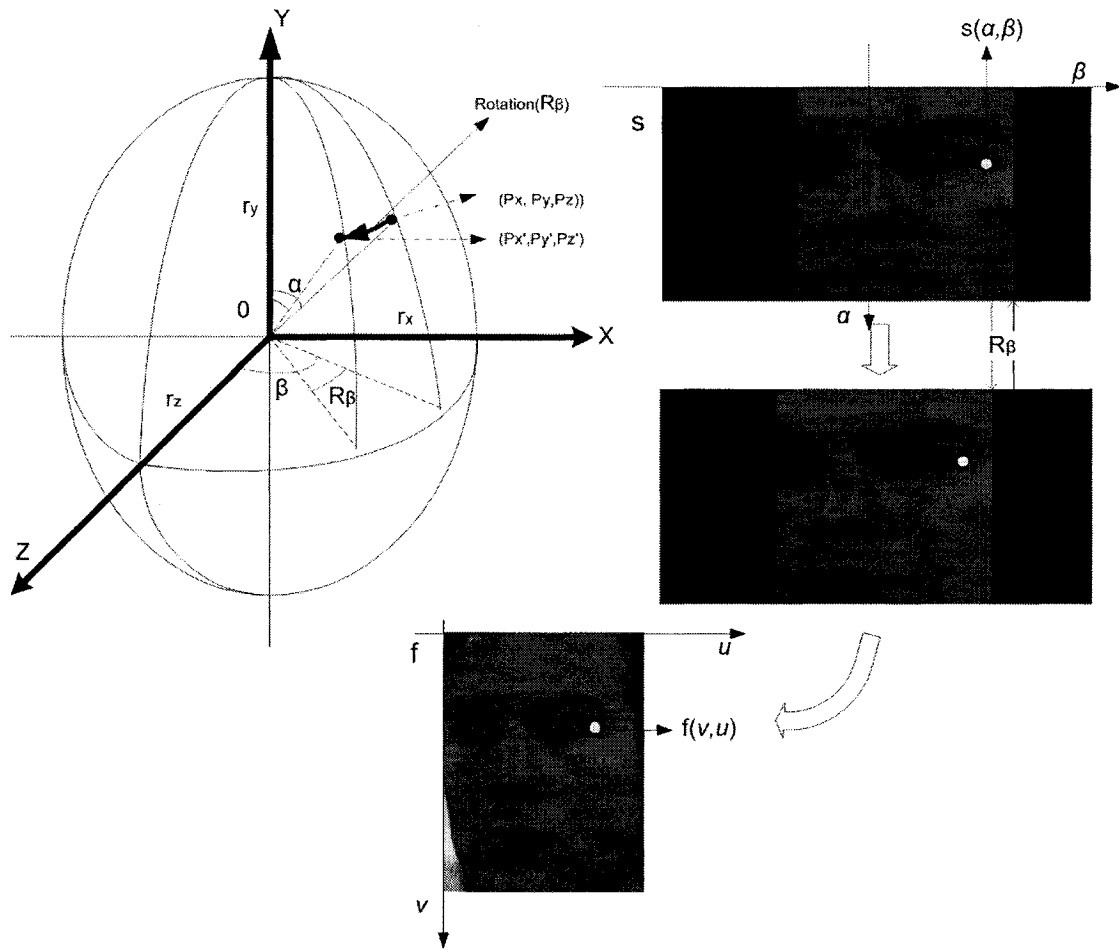
- 3) Project the coordinate  $(P'_x, P'_y, P'_z)$  to the image plane by using the weak-perspective camera model and translating the resulting coordinate by  $cv$  and  $ch$  in both vertical and horizontal directions:

$$\begin{cases} v = P'_y / d + c_v \\ u = P'_x / d + c_h \end{cases} \quad (28)$$

We get the new coordinate  $(v, u)$  in the image plane. Because not all pixels on the texture map can be visible from the camera, we need to determine the visibility of each coordinate  $(P'_x, P'_y, P'_z)$  by the following. That is, we rotate the normal of the point at  $(P_x, P_y, P_z)$  by  $R\alpha$ ,  $R\beta$  and  $R\chi$ , as done in (27)(here  $R\alpha$  and  $R\chi$  are zero). If the angle between the resulting normal and the positive Z axis is smaller than  $90^\circ$ , i.e., the normal points to the positive Z axis,  $(v, u)$  is a valid coordinate.

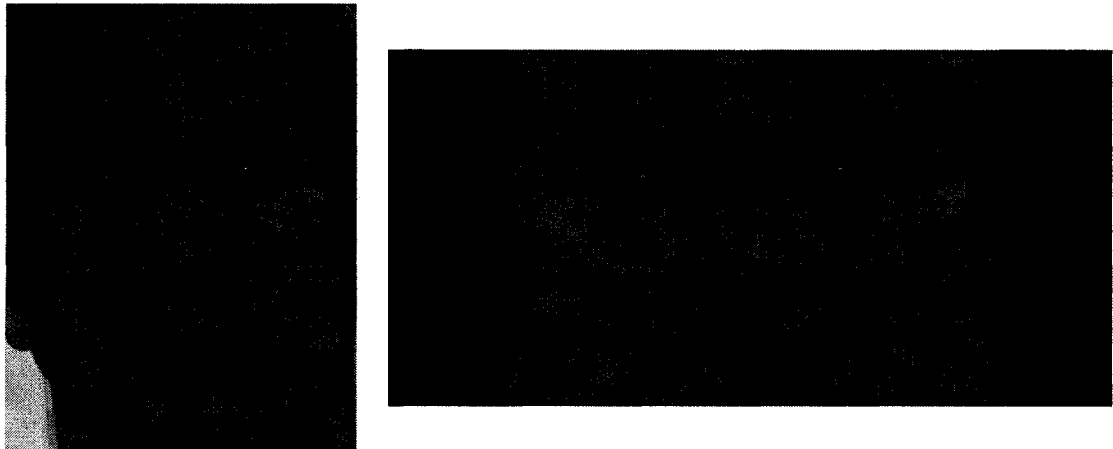
Please note that there are three explanations about mapping process:

- 1) Since the first step does not influence the process result too much, it can be neglected in algorithm.
- 2) The previous four steps describe the procedure from  $s(\alpha, \beta)$  to  $f(v, u)$ , however, we should implement algorithm inversely to get texture map  $s$  from face image  $f$ .
- 3) Reference face image of front pose has more comparable area than other poses have when generating its corresponding texture map.

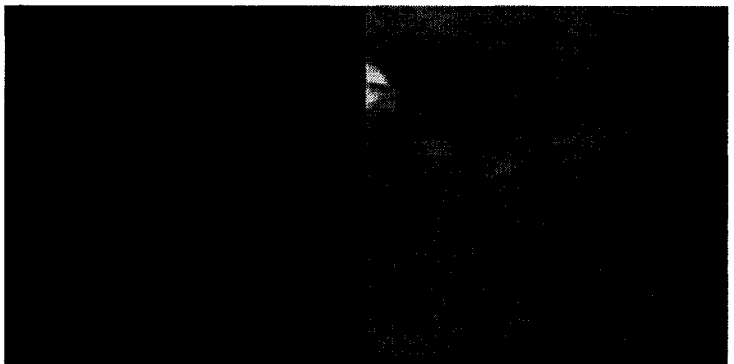
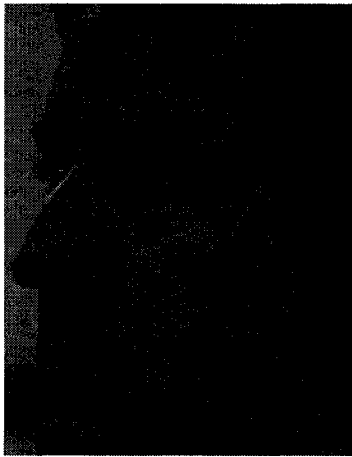
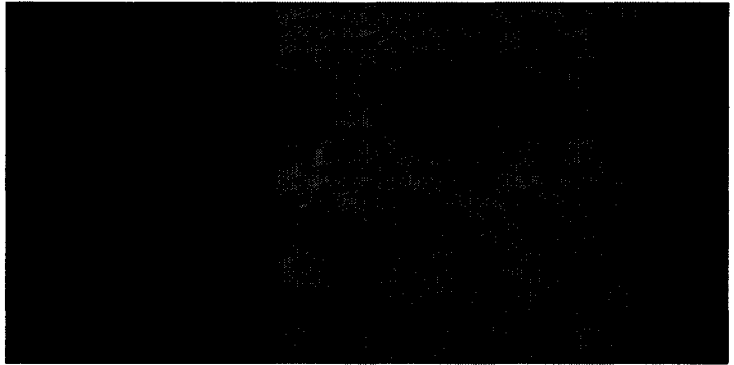
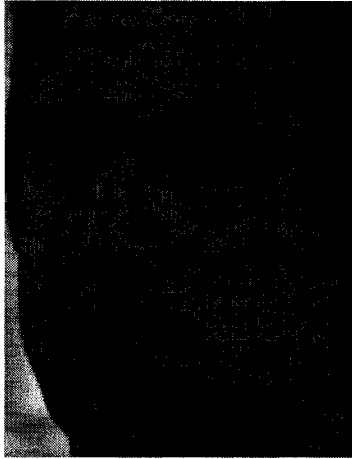


**Figure 36: Geometric mapping: one point on the surface of the ellipsoid maps to a pixel on the image plane.**

The Figure 37 shows some geometrical mapping results:







**Figure 37: Geometrical mapping results**

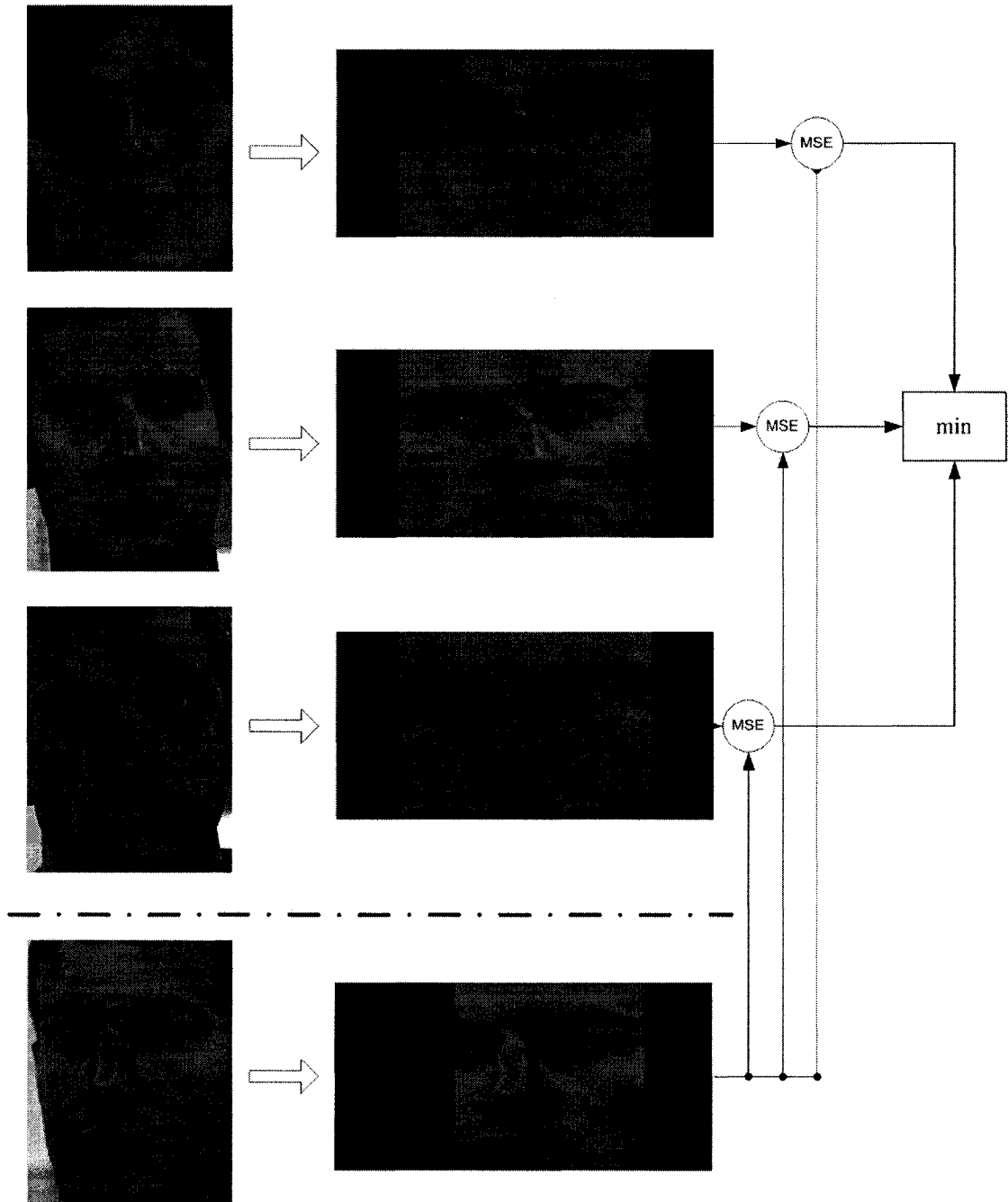
### 5.3 Geometry assisted face recognition

In many face recognition systems, there is only one face image, normally the frontal view face image, during the training stage. However, in the test stage, there might be test images that correspond to different poses of human faces. This is a hard problem because the same face might appear very differently under various poses. In this section, we present our geometry assisted approach to deal with this case. Given a face database with  $L$  subjects, there are frontal and profile face images,  $\mathbf{f}_{lp}$  ( $l = 1, 2, \dots, L$  and  $p$  represent one of three training poses), for each subject that are available for training. During the training stage, the optimal mapping parameter  $\mathbf{x}_{lp}$  is estimated for each training image  $\mathbf{f}_{lp}$  based on a universal mosaic model, which is generated by combining texture maps from multiple subjects. Essentially this estimation process is trying to minimize the difference between the universal mosaic model and the texture map controlled by the mapping parameter, which provides information about the position, the distance, and the pose of the face. Notice that some of the parameters might be known from external sources. For example, if we know all training images have frontal view faces, their pose parameters,  $R\alpha$ ,  $R\beta$  and  $R\chi$ , are all known as zero. Once the estimation is done, the corresponding texture map  $\mathbf{s}_{lp}$  is generated from each training image  $\mathbf{f}_{lp}$ . It is obvious that in the texture map  $\mathbf{s}_{lp}$ , only part of the pixels are valid information of the facial appearance, while the rest are missing pixels since each face image only corresponds to one portion of a 3D head ellipsoid's surface. To describe this missing pixel information, we also generate a mask map,  $\mathbf{a}_{lp}$ , which has the same dimension as the texture map  $\mathbf{s}_{lp}$ . For all missing pixels in  $\mathbf{s}_{lp}$ , the corresponding pixels in  $\mathbf{a}_{lp}$  are zero and the others are one. During the test stage, given one test image  $\mathbf{f}_{lp}$ ,

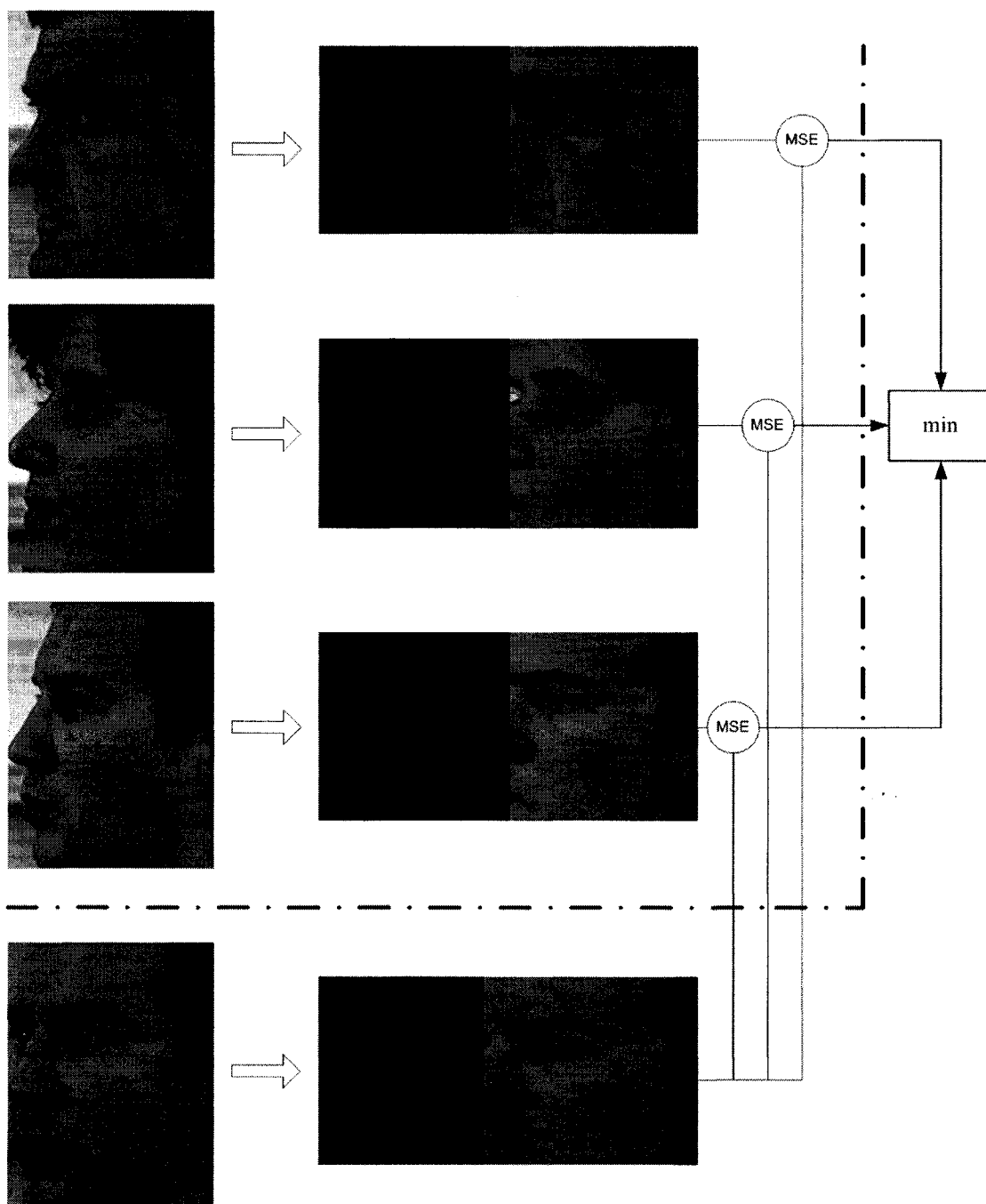
first we estimate the optimal mapping parameter based on the universal mosaic model. Second, the resulting texture map  $\mathbf{s}_{tp}$  and the mask map  $\mathbf{a}_{tp}$  are compared with each of the training texture maps as the following:

$$dl = \frac{1}{\|\mathbf{a}_{tp} \circ \mathbf{a}_{lp}\|} \left\| (\mathbf{s}_{tp} - \mathbf{s}_{lp}) \circ \mathbf{a}_{tp} \circ \mathbf{a}_{lp} \right\| \quad (29)$$

where  $\circ$  refers to the element-wise multiplication. Basically  $dl$  is the normalized mean-square-error (MSE) between the overlap area of the test texture map  $\mathbf{s}_t$  and the training texture map  $\mathbf{s}_l$ , and  $\|\mathbf{a}_t \circ \mathbf{a}_l\|$  indicates the size of the overlap area between two texture maps. There is a degeneration case when the two texture maps have a very small overlap area, which leads to a small  $dl$ . Because in our estimation algorithm, the mapping parameter changes slowly, there is a very low chance that we will fall into this degeneration case. Eventually, the test image is recognized as the subject with the minimal  $dl$ . The following Figure 38 shows the processing of geometry assisted face recognition in the poses nearby front, and Figure 39 shows the poses nearby profile.



**Figure 38: Geometry assisted face recognition in the poses nearby front**



**Figure 39: Geometry assisted face recognition in the poses nearby profile**

## 5.4 Experimental Results

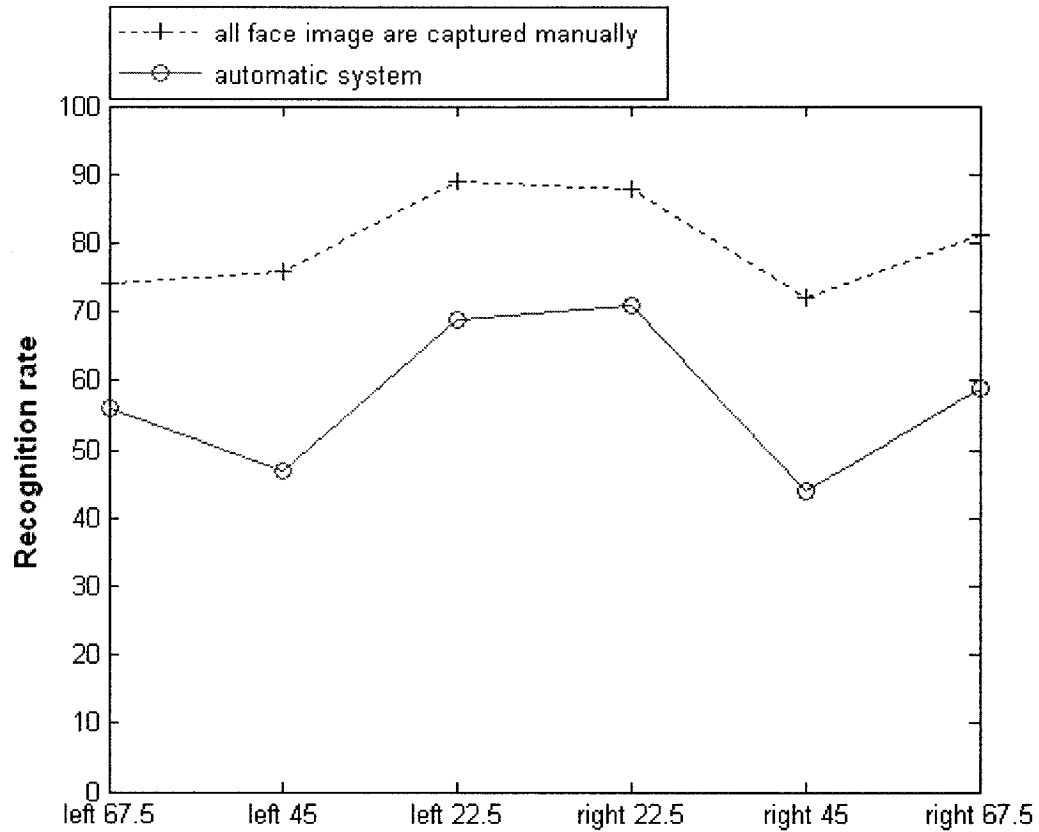
To evaluate the performance of Multi-Pose Mug Shot Recognition system, we have tested PIE database in indoor environments under neutral illumination condition. The above Figure 38 and Figure 39 show some intermediate face processing results under different poses prior to the final identification. The processing gives good performance with respect to complex backgrounds. For the identification, we set up a face database composed of half number of PIE database subject (34 subjects) for the purpose of comparison with [5]. The Table 14 shows the testing results of our system under varying pose environment.

	Left 67.5 (c14)	Left 45 (c11)	Left 22.5 (c29)	Right 22.5 (c05)	Right 45 (c37)	Right 67.5 (c02)
Matching rate	56	47	69	71	44	59

**Table 14: The results of face recognition in varying poses**

In our experiment, the frontal and profile view images were used for reference, the other six poses' images are used for testing. As shown in Figure 40:, the horizontal axis represents the labels of 6 test poses, c14, c11, c29, c05, c37, c02, from the right 67.5 to the left 67.5. The vertical axis shows the recognition rate of two algorithms for each specific pose.

Comparing with our automatic method, the human face for both the training and test images are manually cropped and normalized in Liu's approach [5].



**Figure 40: Recognition performances of two algorithms on the PIE database**

Two conclusions can be drawn from these results.

- 1) When the pose of the test image is changed toward the profile view, the recognition rate decreases. This is consistent with observation of human vision.
- 2) Our algorithm performance is not better than the baseline algorithm. However, the advantages of our algorithms are efficiency and user-friendly interfaces.

# 6 Conclusion and Future Work

## 6.1 Conclusion

In this thesis, we have presented a Multi-Pose Mug Shot Recognition system. In contrast with face recognition under constraint environments, our proposal concentrates on recognition under unconstrained environments. To meet these requirements, we developed a pipeline of face processing algorithms, namely, face detection, facial feature extraction for face normalization, and face identification. During each processing stage, we use step-wise refinements for improved robustness and accuracy. The system has achieved satisfactory recognition rate. The current system is able to handle all view faces with out-of-plane rotation within  $\pm 90$  degrees.

## 6.2 Future Work

Multi-pose face recognition is a challenging problem. To realize a high performance system for commercial use, future work can proceed in the following directions:

- Since the biggest contribution to error rate lies in the failure of previous detection steps, the highest priority of future works should be focused on the improvement of face detection.
- The faces recognition near profile performs not as good as poses nearby frontal do, the partial reason is that the profile training dataset is short of enough



qualified images. Thus to enlarge the training image databases in profile datasets is necessary to achieve better performance.

- Optimization is a way to accelerate the process. Optimization includes the optimization of algorithms and using more efficient coding skill even coding language.
- In face recognition step, computer graphics technique can be adopted in the works of modeling head and estimating pose.

## References

- [1] K. Sung and T. Poggio. Example-based learning for view based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol 20, pp. 39–51, 1998.
- [2] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. *Proceedings of the International Conference on Computer Vision*, pp. 555-562, Bombay, India, 1998.
- [3] D. Roth, M. Yang, and N. Ahuja. A SNoW-based face detector. *Proceedings of Neural Information Processing systems*, vol. 12, pp. 855- 861, 2000.
- [4] F. Zuo and P.H.N. de With. Real-time face recognition for smart home applications. *Consumer Electronics, 2005. ICCE. 2005 Digest of Technical Papers*. pp. 35- 36, Jan. 2005.

- [5] X. Liu and T. Chen. Pose-robust face recognition using geometry assisted probabilistic modeling. *Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol.1 pp. 502 – 509, NY, USA, June 20-25, 2005.
- [6] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal Cognitive Neuroscience*, vol.3, no.1. pp. 71–86. 1991.
- [7] T. Sim, S. Baker, and M. Bsat. The CMU pose, illumination, and expression Database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1615 – 1618, Dec. 2003.
- [8] B. Froba and C. Kublbeck. Real-time face detection using Edge-Orientation Matching. *Proceedings on Audio and Video-Based Biometric Person Authentication*, pp. 78-83, 2001.
- [9] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 511-518, 2001.
- [10] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 511-518, Kauai, Hawaii , Dec 9-14, 2001.

- [11] S. H. Lin, S. Y. Kung, and L. J. Lin. Face recognition/detection by probabilistic decision based neural network. *IEEE Transactions on Neural Networks*. vol 8, pp. 114–132. 1997.
- [12] V. Blanz and T. Vetter. Face recognition based on fitting a 3D morphable Model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. vol. 25 pp. 1063–1074, 2003.
- [13] P. J. Phillips. Support vector machines applied to face recognition. *Proceedings of the Conference on Advances in Neural Information Processing System*. vol. 11, pp. 803–809, Denver, Colorado, Dec1-5, 1998.
- [14] A. Yuille, P.W. Hallinan and D. Cohen. Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, vol. 8, pp. 99-112, 1992.
- [15] T. Cootes. Statistical models of appearance for computer vision. *Technical Report*. Imaging Science and Biomedical Engineering, University of Manchester, 2001.
- [16] B. Froba and C. Kublbeck. Real-time face detection using edge-orientation matching. *Proceedings Audio-and Video-Based Biometric Person Authentication*, pp. 78-83, 2001.
- [17] T. Cootes, C. Taylor, D. Cooper and J. Graham. Active shape models - their training and application. *Computer Vision and Image Understanding*, vol. 61, no. 1, pp. 38-59, 1995.

- [18] S. Ordas, L. Boisrobert, M. Huguet and A.F. Frangi. Active shape models with invariant optimal features (IOF-ASM) application to cardiac MRI segmentation. *Computers in Cardiology*, pp. 633-636, Sept 2003.
- [19] D. L. Ruderman. The statistics of natural images. *Network: Computation in Neural Systems*, vol. 5, pp. 598–605, 1994.
- [20] W. Zhao, R. Chellappa, P. J. Phillips. 1999. Subspace linear discriminant analysis for face recognition. *Technical Reports CAR-TR-914*, Center for Automation Research, University of Maryland, College Park, MD.
- [21] W. Zhao, R. Chellappa, P. J. Phillips and A. Rosenfeld. Face recognition: a literature survey. *ACM Computing Surveys*, vol. 35, No. 4, pp. 399–458, Dec 2003.
- [22] V. Blanz, T. Vetter. A morphable model for the synthesis of 3D faces. *Proceedings of the Special Interest Group for Computer Graphics*, pp. 187-194, Los Angeles, USA, Aug 8-13, 1999.
- [23] V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.
- [24] V. Blanz and T. Vetter. Face recognition based on fitting a 3D morphable model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 9, Sep 2003.
- [25] A. Bronstein, M. Bronstein, R. Kimmel, and A. Spira. 3D face recognition without facial surface reconstruction. *Proceedings of the European Conference on Computer Vision 2004*, Prague, Czech Republic, May 11-14, 2004.

- [26] A. Bronstein, M. Bronstein, and R. Kimmel, Expression-invariant 3D face recognition. *Proceedings Audio & Video-based Biometric Person Authentication (AVBPA), Lecture Notes in Computer Science 2688*, pp. 62-69, 2003.
- [27] G. Guo, S.Z. Li and K. Chan, Face recognition by support vector machines. *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, Grenoble, France, pp. 196-201, March 26-30, 2000.
- [28] B. Heisele, P. Ho and T. Poggio, Face recognition with support vector machines: global versus component-based approach. *Proceedings of the IEEE International Conference on Computer Vision (ICCV 2001)*, vol. 2, Vancouver, Canada, pp. 688-694, July 9-12, 2001.
- [29] K. Jonsson, J. Matas, J. Kittler and Y.P. Li, Learning support vectors for face verification and recognition. *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, Grenoble, France, pp. 208-213, March 26-30, 2000.
- [30] M. B. Stegmann, B. K. Ersboll, and R. Larsen. FAME: A flexible appearance modeling environment. *IEEE Transactions on Medical Imaging*, vol. 22, pp. 1319-1331, 2003.
- [31] T. Sim, S. Baker, and M. Bsat. The CMU Pose, Illumination, and expression database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 12, pp. 1615-1618, December, 2003.

- [32] R. E. Schapire, Y. Freund, P. Bartlett and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *Proceedings of the 14th International Conference on Machine Learning*, pp. 322-330, 1997.
- [33] M. B. Stegmann, B. K. Ersboll, and R. Larsen. FAME - a flexible appearance modeling environment. *IEEE Transactions on Medical Imaging*, pp. 1319- 1331, May, 2003.
- [34] G. Hamarneh, R. A. Gharbieh and T. Gustavsson. Active shape models - part I: modeling shape and gray level variations. *Proceedings of the Swedish Symposium on Image Analysis*, 1998.
- [35] G. Hamarneh, R. A. Gharbieh and T. Gustavsson. Active shape models - part II: image search and classification. *Proceedings of the Swedish Symposium on Image Analysis*, 1998.
- [36] K. Mardia, J. Kent, J. Bibby, *Multivariate Analysis*. Academic Press, 1995.
- [37] T. Cootes, C. Taylor, A. Hill, J. Halsam, The use of active shape models for locating structures in medical images. *Proceedings of the 13th International Conference on Information Processing in Medical Imaging*, (Eds. H.H.Barrett, A.F.Gmitro), Springer-Verlag, pp. 33-47, 1993.
- [38] T. Cootes, C. Taylor, A. Lanitis, Active shape models: evaluation of a multi-resolution Method for Improving Image Search. *Proceedings of the British Machine Vision Conference*, pp. 327-336, 1994.

- [39] A. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces of face recognition. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 84-91, Seattle, June 1994.
- [40] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. *IEEE Conference on Computer Vision and Pattern Recognition*, 2001.
- [41] J. P. Kapur. Face detection in color images. *EE499 Capstone Design Project*, Department of Electrical Engineering in University of Washington. Spring 1997.
- [42] M. Störring, H. J. Andersen, and E. Granum. Skin colour detection under changing lighting conditions. *7th Symposium on Intelligent Robotics Systems*, pp 187-195, Coimbra, Portugal, July 20-23, 1999.
- [43] O. D. Faugeras. *Three-Dimensional Computer Vision*. MIT Press, Cambridge, 1993.
- [44] F. Yang and A. Krzyżak. Face Recognition under Significant Pose Variation. *20th Canadian Conference on Electrical and Computer Engineering (CCECE 2007)*, Vancouver, Canada, April 2007.
- [45] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian detection using wavelet templates. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 193-99, 1997.