Chapter 1 Introduction

1.1 Motivation

Automatic human recognition systems are increasingly popular in the recent years. There are recognition systems called biometric identification systems which use physiological characteristics such as a fingerprint, iris, or behavior patterns such as handwriting, voice and key-stroke patterns. However, people must contact some particular devices in order to be verified in these recognition systems, which is regarded as a risk action sometimes; People are reluctant to use iris identification systems because of human inherent protectiveness of his eyes. Comparatively, face recognition has the benefit of being a passive, nonintrusive mechanism to verify a personal identity in a noncontact way.

Face recognition technology has been applied in many fields such as law enforcement applications and video surveillance. Many factors will affect the recognition accuracy. For instance, the influence of the illumination may cause a recognition system poor performance. Anyway, in an indoor environment, the difficulties can be lessened. In this paper, we aim to develop a face recognition system under an indoor environment such as offices and houses. To make the system more efficient, we aim to reduce the storage and the processing time by reducing the feature dimension of human faces.

1.2 Problem Definition

Here we propose two problem definitions:

1. Which features can be used to differentiate human faces more

efficiently and effectively?

A human can recognize different persons easily from different features exposed. In a computer vision, we have to determine the features can be used to differentiate human faces efficiently and effectively.

Fig. 1.2.1 Examples for four different faces.

2. How to recognize the faces under different orientations?

A human face is a three-dimension (3-D) object. A human can rotate his head voluntarily. Because the images can only be displayed in a two-dimensioned space, the faces under different orientations would not be the same. The method how to recognize the faces under different orientations is another important issue to tackle in this paper.

Fig. 1.2.2 Face images under limited side movements.

1.3 Related Works

1. Principal Component Analysis

Sirovich and Kirby [1] proposed principal components to represent face images. A face image could be approximately reconstructed by a small collection of weights and a standard face picture (eigenpicture). In mathematical terms, eigenfaces are the principal components of the distribution of faces, or the eigenvectors of the covariance matrix of the set of face images. The eigenvectors represent the different amounts of the variation among the faces, i.e. faces can be distinguished according to these eigenvectors. Each face can be represented exactly by a linear combination of the eigenfaces. In order to reduce processing times, only the best eigenvectors with the largest eigenvalues are used practically. Then the face image can be described from the weights obtained by projecting the face image onto the eigenpicture.

Turk and Pentland [2] also used eigenfaces.. The authors reported 96%, 85%, and 64% correct classifications averaged over lighting, orientation, and size variations, respectively. Their database contained 2,500 images of 16 individuals. When the images include a large quantity of background area, the results are influenced significantly. Besides, the results are also influenced by lighting conditions and orientations.

To overcome the influence of the background and the variations in orientation, Pentland et al. [16] extended their early work on eigenface to eigenfeatures corresponding to face components, such as eyes, nose, and mouth. They used a modular eigenspace which was composed of the above eigenfeatures (i.e., eigeneyes, eigennose, and eigenmouth). They also propose a view-based approach, in which each view has it own face space. The test images were compared in different view face spaces. They claimed their method would be less sensitive to appearance changes than the standard eigenface method. The system achieved a recognition rate of 95% on the FERET database of 7,562 images of approximately 3,000 individuals.

In summary, an eigenface based method appears as a simple and practical method. However, it does not provide invariance over lighting conditions. It also need much computation to obtain the eigenvectors.

2. Fisherface

Belhumeur et al. [4] proposed a face recognition method based on Fisher's linear discriminant function to overcome illumination effects under the condition that if the object is Lambertian. As eigenface, they also need to calculate the eigenvectors. Based on Fisher's linear disciminant function, their method linearly projects the image into a subspace in a manner which discounts those regions of the face with large deviation. They claimed their method has a lower error rate than those of the eigenface techniques for tests on the Harvard and Yale face databases. But the method also needs much computation to calculate the eigenvectors and the eigenvalues. Besides, they did not concern the faces under variations in orientation.

3. Hidden Markov Model

Stochastic modeling of nonstationary time series based on hidden Markov models (HMM) has been very successful for speech applications. [5] Samaria and Fallside [8] tried to apply this method to human face recognition. Faces were intuitively divided into regions such as the eyes, nose, mouth, etc., which can be associated with the states of a hidden Markov model. Since HMMs require a one-dimensional observation sequence and images are two-dimensional, the images should be converted into either 1D temporal sequences or 1D spatial sequences. They extracted a spatial observation sequence from a face image by using a band sampling technique. Each face image was represented by a 1D vector series of pixel observations. Each observation vector is a block of L lines and there is an M lines overlap between successive observations. An unknown test image is first sampled to an observation sequence. Then, it is matched against every HMMs in the model face database (each HMM represents a different subject). The match with the highest likelihood is considered the best match and the relevant model reveals the identity of the test face. The recognition rate was 87 % using ORL database consisting of 400 images of 40 individuals. A pseudo 2D HMM [7] was reported to achieve a 95% recognition rate in their preliminary experiments. Its classification time and training time were not given (believed to be very expensive). Besides, the choice of parameters was based on subjective intuitions.

4. Graph matching

Graph matching is another approach to face recognition. The approach tries to find the "best matching" for the face using the deformable mask. Lades et al. [18] presented a dynamic link structure for distortion-invariant object recognition that employed elastic graph matching to find the closest stored graph. Dynamic link architecture is an extension to classical artificial neural networks. Faces are represented by sparse graphs, whose vertices are labeled with a multiresolution description in terms of a local power spectrum and whose edges are labeled with geometrical distance vectors. Face recognition can be formulated as elastic graph matching which is performed by stochastic optimization of a matching cost function. They reported good results on a database of 87 people and a small set of office items comprising different expressions with a rotation of 15 degrees. Note the matching process is computationally expensive.

Wiskott and Malsburg [11] extended the technique and matched human faces against a gallery of 112 neutral frontal view faces. Probe images were distorted due to rotation in depth and changing facial expression. They reported recognition rates of 86.5% and 66.4% for the matching tests of 111 faces of 15 degree rotation and 110 faces of 30 degree rotation to a gallery of 112 neutral frontal views.

In summary, graph matching such as dynamic link architecture is superior to other face recognition techniques in terms of rotation invariant. However, the matching process is computationally expensive and the usefulness of point features should be evaluated on large datasets.

5. 3-D Face Model

Except the methods that find invariant parts of a face under variations in orientation, such as elastic matching [11] and Gabor wavelet filter[9], there are some researches [19-20] which solve the problem using a 3-D face model. Zhang and Cohen [19] reported recognition rate of 97% that 30 out of 31 people were classified correctly. Although 3-D face model could solve the face orientation problem, even nonfrontal face images, it is inefficient to build a 3-D model.

6. Edge Map

Cognitive psychological studies [21], [22] indicated that human beings recognize line drawings as quickly and almost as accurately as gray-level pictures. These results might imply that edge images of objects could be used for object recognition and to achieve similar accuracy as gray-level images. Takacs [15] made use of edge maps, which was motivated by the above finding, to measure the similarity of face images. The faces were encoded into binary edge maps using the Sobel edge detector. The Hausdorff distance was chosen to measure the similarity of the two point sets, i.e., the edge maps of two faces, because the Hausdorff distance can be calculated without an explicit pairing of points in their respective data sets. The recognition rate 92% was reported in their experiments. Takacs argued that the process of face recognition might start at a much earlier stage and edge images can be used for the recognition of faces without the involvement of high-level cognitive functions, which means the processing time might be speeded up using only the edge map information. However, the Hausdorff distance uses only the spatial information of an edge map without considering the inherent local structural characteristics inside such a map. It also requires that all matching pairs fall within a given neighborhood of each other in consistency.

Considering a combination of template and geometrical feature matching, Gao et al. [12] proposed face recognition using a line edge map. They first integrated the structural information with spatial information of a face image by grouping pixels of the face edge map to line segments, and then use Line Segment Hausdorff Distance (LHD) to recognize faces. They claimed their method is relatively insensitive to lighting changes and facial expression. They reported recognition rates of 96.4 percent for Bern face database and 100 percent for Purdue Univ. Face Database. However, it still requires that all matching pairs fall within a given neighborhood of each other in consistency, and they did not concern faces under variations in orientation.

In summary, face recognition using only edge map information might be accurate and quick. If we could resolve the location of face and overcome the face under variation in orientations, face recognition might be performed well.

1.4 Assumptions

Because the environment that we consider is indoor, we may assume the lighting variation is uniform. Many face detection and feature extraction methods have been proposed [13][25, 26]. It is difficult to solve all the case of face orientation. We only consider frontal faces with limited rotations.

1.5 System Flow

Fig. 1.5 shows the system flow in this paper. The system proposed in this paper consists of two modules: face feature description and face recognition.

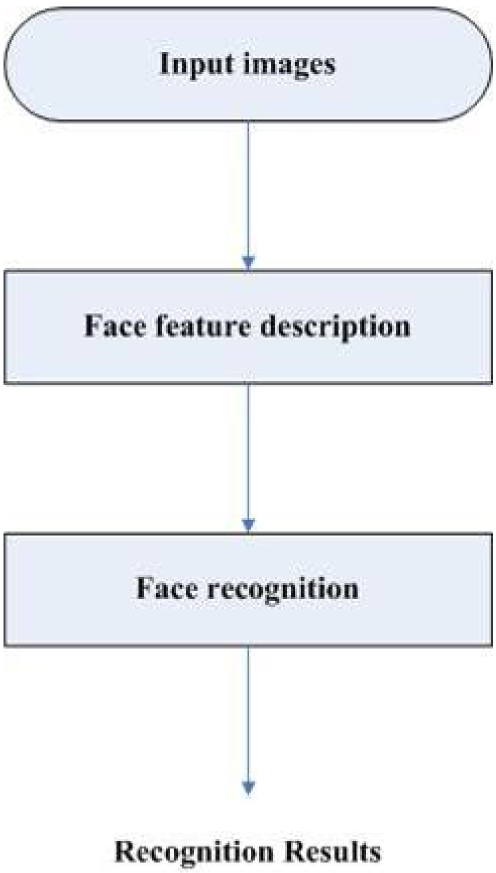


Fig. 1.5.1 System Flow in this paper.

Face representation

In the first module, we transform the face image into a binary edge map and represent the face by recording the distance of edge points in vertical sampling lines.

Face recognition

In the second module, we compare the recorded data with those in a database and output the identification.

1.6 Thesis Organization

The remainder of this paper is organized as follows. After preprocessing the face image, then we propose a novel representation for the face in Chapter 2. The face recognition method is proposed and described in Chapter3. Chapter 4 describes experimental results and their analysis. Finally, Chapter 5 presents some conclusions and suggestions for future work.

Chapter 2 Face Representation

Generally, pixels intensities are used to describe a human face. But In this section, we propose a new feature representation for a human face. First, we transform a face image into an edge map. Because a human face is variable in 3-D, we find the invariant parts of the face under variations in orientation. Then we remove some edge points in an edge map to reduce the processing time.

2.1 Preprocessing

To extract the features from a face image, we transform the face image into an edge map. We will first normalize the properties such as face size, direction and so on. To avoid the disturb noise and preserve the main edges in an edge map, we smooth the face image prior to edge detection.

2.1.1 Normalization

The size of a face image is normalized to a standard size first. Second, face rotation should be adjusted [14]. We also gather statistics of the eye size in the face database, including the width and height. Note that the faces in the ORL and IIS databases we have already been normalized.

2.1.2 Smoothing

In order to resolve the conflict between noise elimination and edge degradation, we adopt an edge preserving smoothing method [24].

The procedure of edge preserving smoothing [24] is as follows:

(1) Rotate an elongated bar mask around a point (x,y).

(2) Detect the position of the mask for which the variance of the gray level is minimum.

(3) Give the average gray level of the mask at the selected position to the point (x,y).

(4) Apply steps (1) to (3) to all points in the picture.

(5) Iterate the above process until the gray levels of almost all points in the picture do not change.

Figure 2.1.1 shows a face image after applying edge preserve smoothing.

(a) **(b)**

Fig. 2.1.1 (a) Original face image (b) Face image after applying edge preserve smoothing

2.1.3 Edge detection

We apply Laplacian edge detector on the face image. The Laplacian L(x, y) of the pixel I(x,y) is defined as follows:

$$L(x,y)=\frac{\partial^2 I}{\partial x^2}+\frac{\partial^2 I}{\partial y^2}$$

After we apply the Laplacian detector, the zero-crossing detector is used to obtain the edge map. Figure 2.1.2 shows a face image and its edge map after applying Laplacian Edge detector.

(a) **(b)**

Fig. 2.1.2 (a) Original face image (b) Face image after applying Laplacian edge detector.

2.2 Vertical Line Sampling

To overcome the problem of variations in orientation we adopt the view-based approach [16].

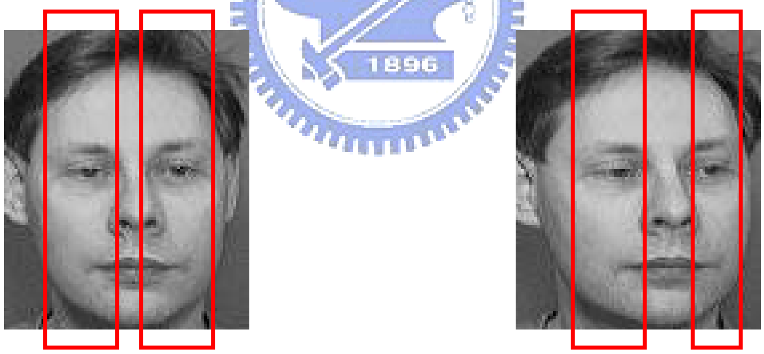
We assert that eyes, mouth corners and face shape are important parts for distinguishing humans. These parts are most invariant parts of the face under variations in orientation. Rama Chellappa has reported that the nose plays an insignificant role [23] and nose information is unstable while a face is in different orientations. We would not use nose information. Figure 2.2.1 shows an example with eyes, mouth corners and face shape circled.

Fig. 2.2.1 Eyes, mouth corners and cheek shape for face distinguishment.

To extract significant information for distinguishing humans, we define two strips, called eye strips, on the human face. Each strip contains an eye, mouth corner and cheek, as shown in figure 2.2.2.

To extract the eye block, we perform the following procedure:

1. Extract the eye location [14], which is defined by the center point of one of the upper horizontal lines on the face
2. Find the edge points whose distances to the eye location are less than a prespecific threshold. The bounding rectangles of these edge points are called the eye region.
3. Segment the eye block between the left and right borders of the eye region. Since we have two eyes, we can obtain two eye blocks in each face image.



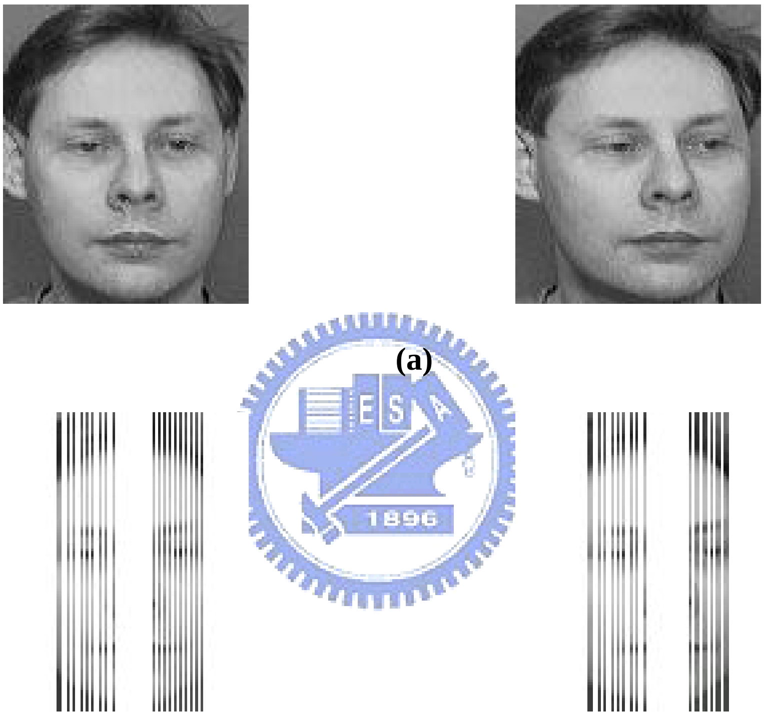
(a)

(b)

Fig. 2.2.2 (a) A face under small variation in orientation; blocks are eye strips including forehead, eyes, mouth corners and face shape.

(b) The information of two eye strips segmented in a face under small variation in orientation.

Since the widths of eye blocks for different face images may be different, we take a fixed number of sampling lines in each of the eye strips. Figure 2.2.3 shows the 10 sampling line in each eye block of Figure 2.2.1.



(b)

Fig. 2.2.3 (a) A source images and (b) its 10 sampling lines in each eye strip.

Let W be the width of an eye strip. The ith sampling line L_i is defined as

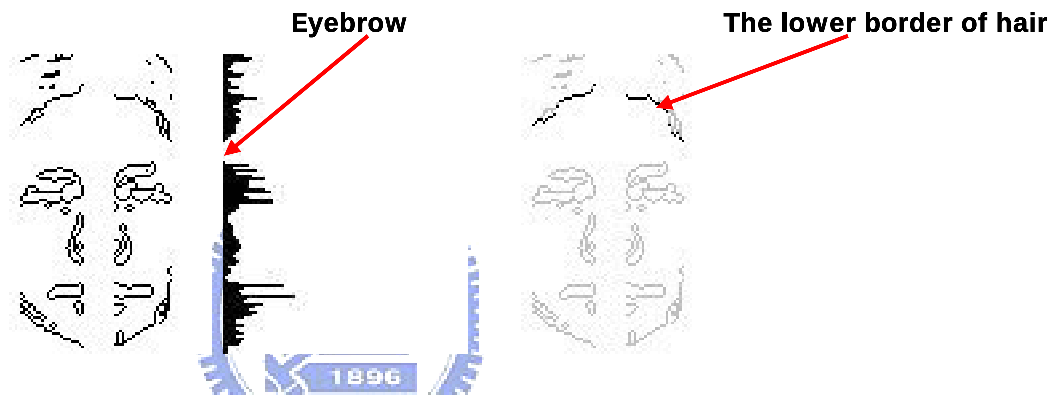
$$L_{i+1} = round(R + L_i) \quad i = 0,\ldots, N-1$$

where N is the numbers of sampling lines, and R the average interval between sampling line

$$R = W / N$$

Hence we can obtain a fixed number of sampling lines from different face images. Different numbers of sampling lines would affect the recognition rates. We test the effect of different numbers of sampling lines in section 4.2.

The lower hair border is also used in the following recognition engine. To find the lower hair border, we first locate the eyebrow. We find the eyebrow at the valley of vertical projection of edges above the eye location in the two eye strips. Then the lower hair border could be found by locating first edges above the eyebrow. Figure 2.2.4 shows an example of the lower hair border.



(a) (b)

Fig. 2.2.4 (a) The eyebrow located at the valley of vertical projection of edges above the eye location in two eye strips.
(b) The lower hair border formed as the first edge points above the eyebrow.

2.3 Edge Points Removal

To make the processing time quicker, we will remove several edge points. There are two kinds of edge points to remove: unreliable edge points and close edge points.

2.3.1 Unreliable Edge Points

The distance between two edge points in a sampling line is taken as the "feature" in our recognition system, to be discussed in the next section. Therefore, unreliable edges should be removed in order to reduce “false” distances which may cause error. The unreliable edge points include the following types:

1. **Edge points above the lower hair border**

Edges may occur in hair area because of illumination differences. Record these edge distances would affect the result of face recognition. We have already known the location of eyes and eyebrows. To remove the edge points above the lower hair border, we could eliminate the edge points above the eyebrows except the first edge points

2. **Edges formed by chin shadow**

Because shadow often exists below the face, edges between shadow and skin would be detected. Because shadow color is often darker than those of skin and face chin, we could eliminate the edge points at the lower location in a face image.

2.3.2 Close Edge Points

If edge points are very close in a vertical line, we can also remove all of them except one to reduce the processing time. The method is to remove all but one of the edge points among which the distance is less than a threshold. Figure 2.3.2 shows an example of unreliable edge points and close edge points.

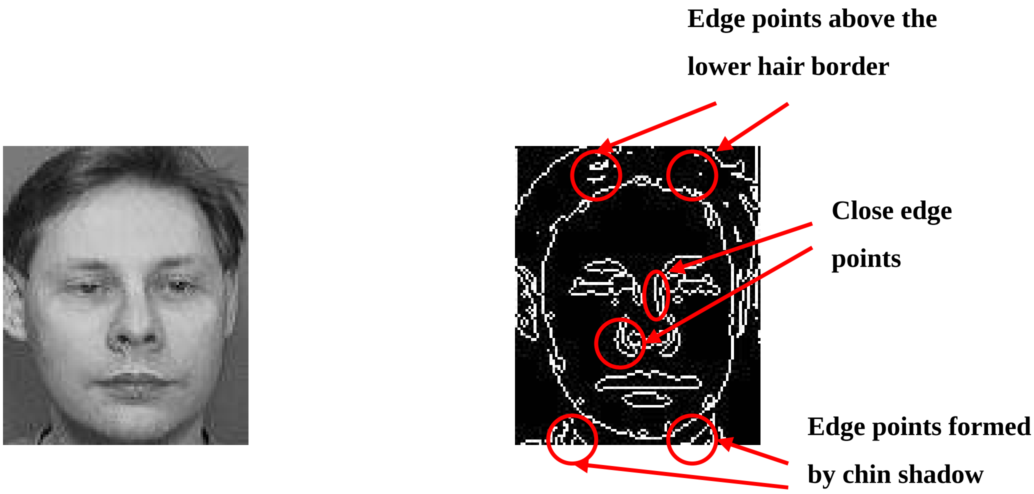
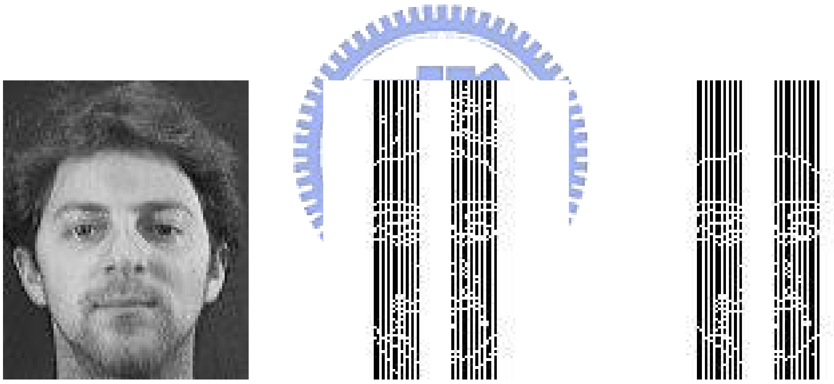


Fig. 2.3.1 Two types of unreliable edge points and close edge points in a face.

Figure 2.3.2 shows an example after unreliable edge points are removed. Note the edge map here is the image after being applied edge detection and being sampled.



(a) **(b)** **(c)**

Fig.2.3.2 (a) A source image

(b) The edge map before removing unreliable edge points.

(c) The edge map after removing unreliable edge points.

Figure 2.3.3 shows that different gap cause different face edge images for recording.

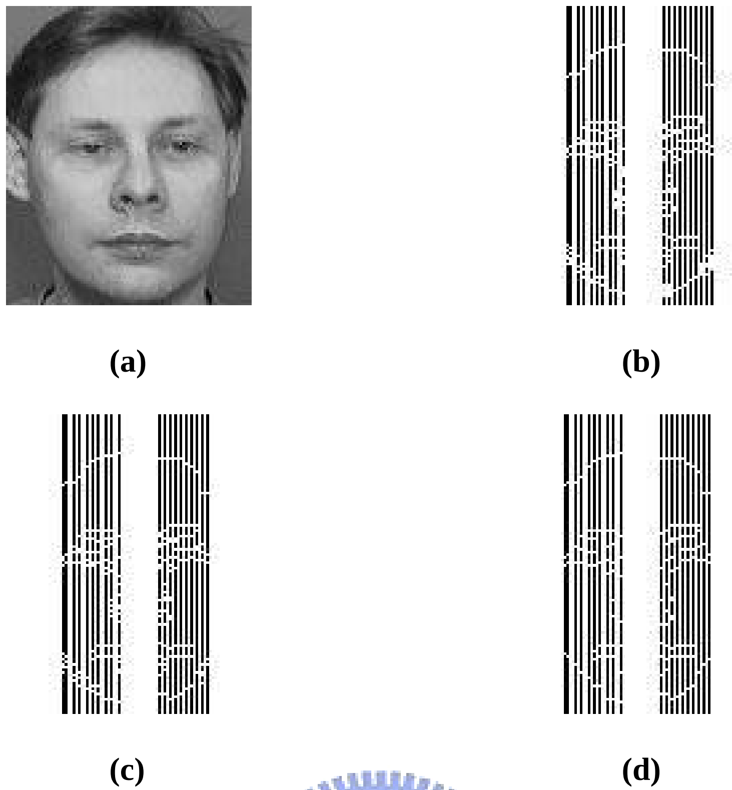


Fig. 2.3.3 (a) A source image

(b) Edge map

(c) Edge map after removing one of the edge point with distance threshold 2.

(d) Edge map after removing one of the edge point with distance threshold 4.

2.4 Face Feature Description

There are few researches recognizing faces using edge maps [12] [15]. Takacs [15] proposed a shape comparison method using the modified Hausdorff distance which is shown as below:

$$\mathrm{d}(\mathrm{a},\mathrm{B}) = \max(\mathrm{I}\min_{\mathrm{b}\in N_B^a}\|\mathrm{a}-\mathrm{b}\|, (1-\mathrm{I})\mathrm{P})$$

$$\mathrm{h}(\mathrm{A},\mathrm{B}) = \frac{1}{N_a}\sum_{a\in\mathrm{A}}\mathrm{d}(\mathrm{a},\mathrm{B})$$

$$\mathrm{H}(\mathrm{A},\mathrm{B}) = \max(\mathrm{h}(\mathrm{A},\mathrm{B}),\mathrm{h}(\mathrm{B},\mathrm{A}))$$

Although the method is interesting, it requires that all matching pairs fall within a given neighborhood of each other in consistency. Besides the Hausdroff distance uses only the spatial information of an edge map [12].

Edges distances are extracted in sampling lines as features in our system. Those not only consist of the structural information such as the size of the eye but also preserve invariance under limited variations in orientation. Figure 2.4.1 shows that edge distances encode implicitly the different structural information for each person. Figures 2.4.2 and 2.4.3 show their invariance under limited variation in orientation.

The edge distances are recorded from top to bottom in a sampling line and from left to right in an edge map. Note we only record two neighbor edge distances in vertical. Edge distances in the horizontal direction are not significant because a human face has higher variations in the horizontal direction. Figure 2.4.4 shows several sequences of edge distances recorded in an edge map.

Edge distances in a sampling line are regarded as a sequence. Hence, an image is transformed into a series of sequences after recording the distances in sampling lines. Face recognition could be achieved by comparing the sequences.

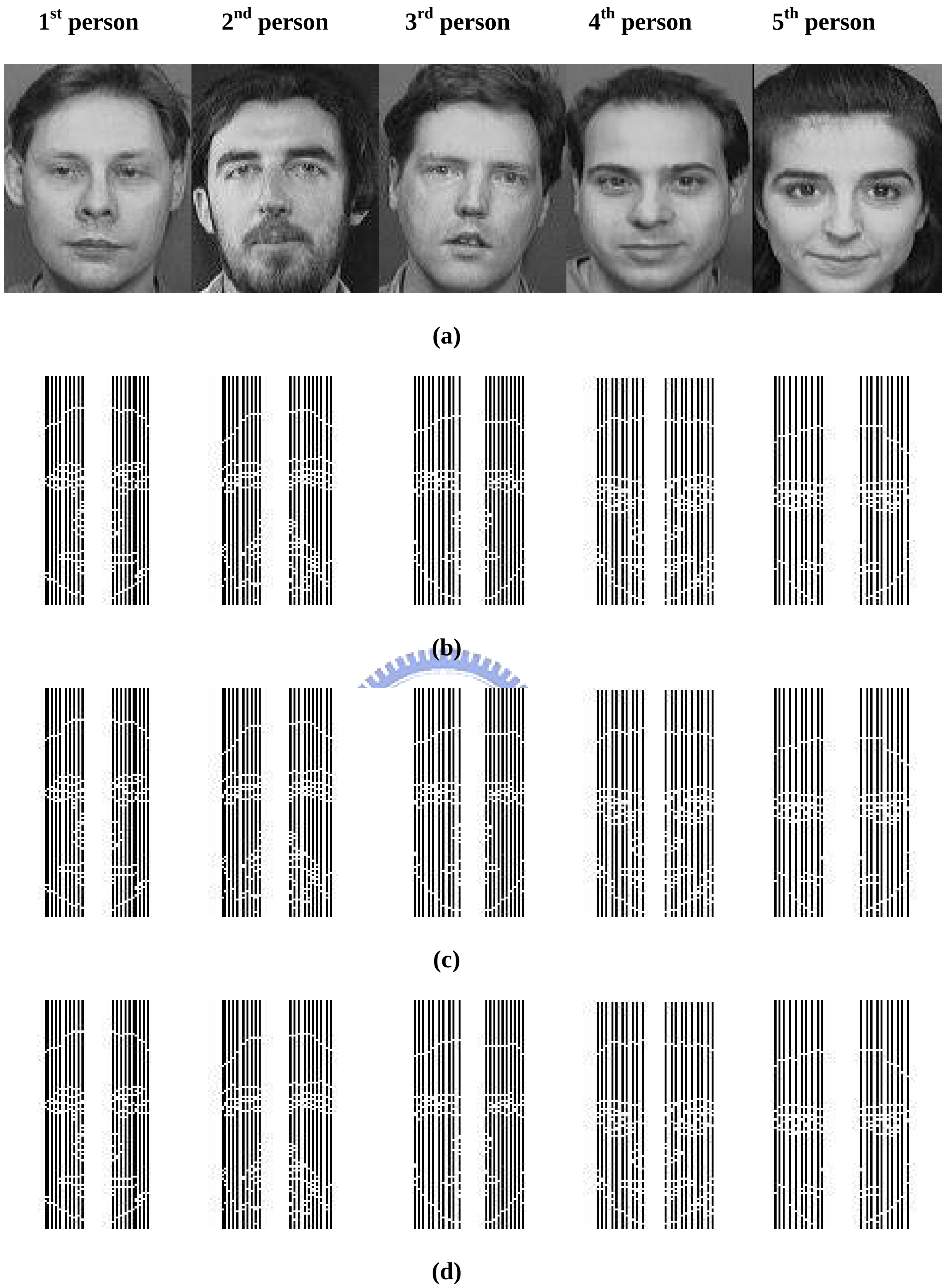


Fig. 2.4.1 (a) Source face images.

(b) Face edge maps in sampling lines.

(c) Different size of eyes.

(d) Different face cheek.

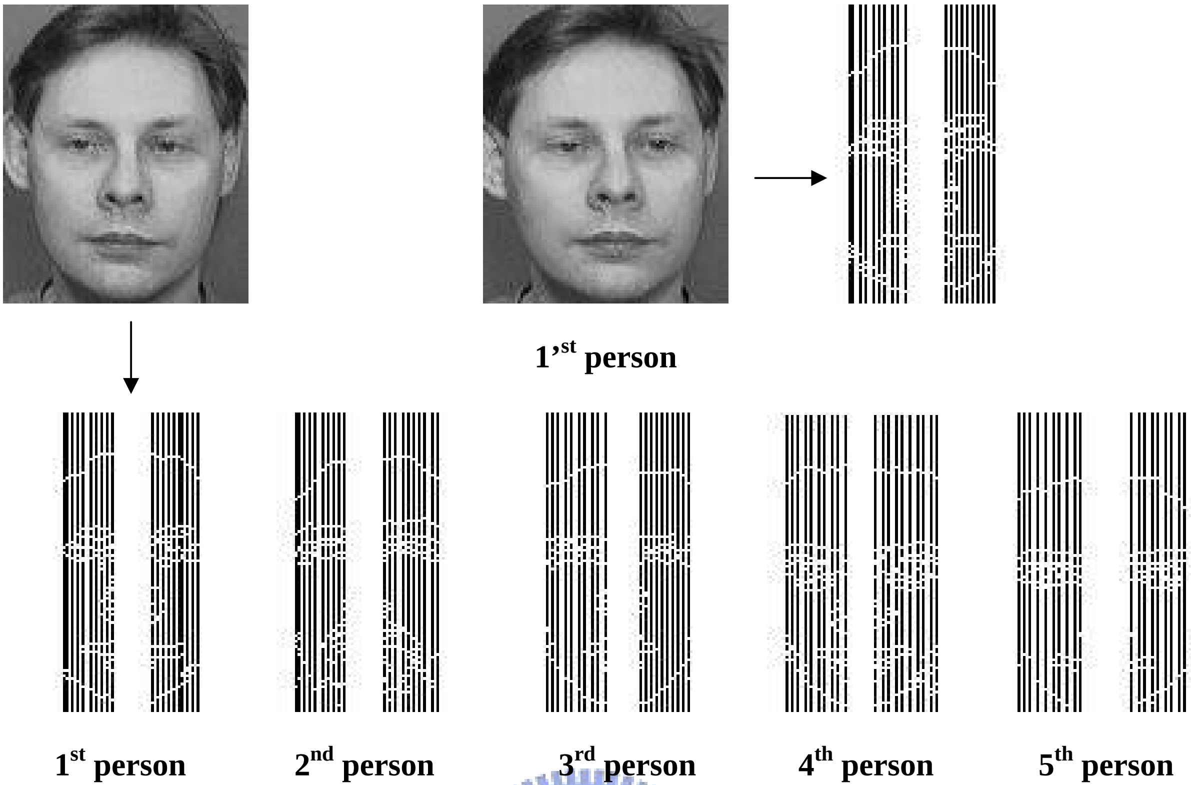


Fig. 2.4.2 Example1 showing similar edge distances under the limited variation in orientation

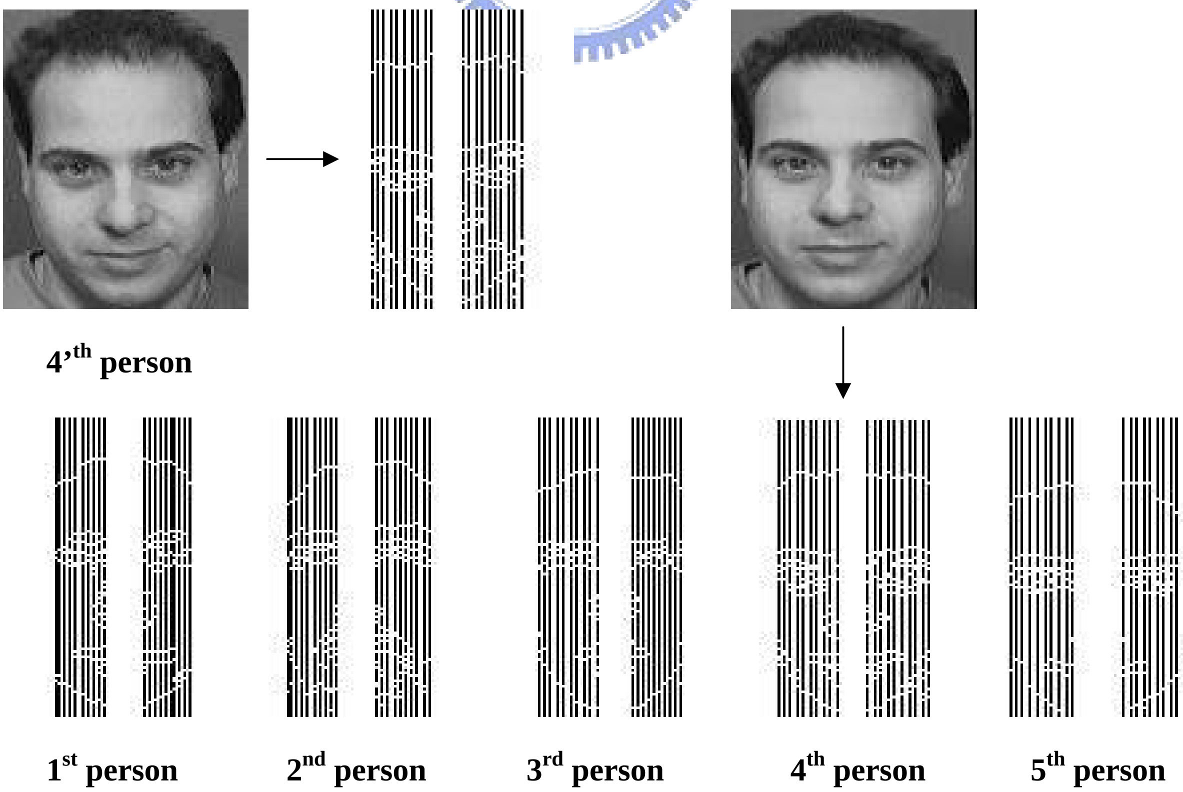


Fig. 2.4.3 Example2 showing similar edge distances under the limited variation in orientation

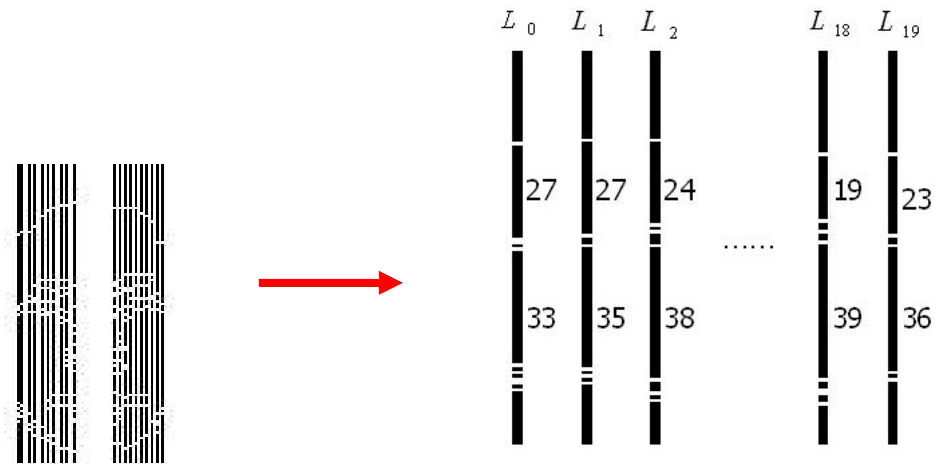


Fig. 2.4.4 Sequences of edge distances in sampling lines.

Chapter 3 Face Recognition

We have transferred a face image into a series of sequences, where each sequence represents edge distances in sampling lines. In this chapter, we will use these features, i.e., the different series of sequences, to recognize the person using the dynamic programming algorithm. In section 3.1, we will describe how to calculate the minimum cost between two sequences with different lengths. Different lengths of sequences are caused by that noise might be detected or edge points might disappear during the preprocessing stage. In section 3.2., the editing costs are addressed and discussed. Finally, the dynamic programming algorithm for face recognition is presented in section 3.3.

3.1 Sequence Matching

After recording edge distances in sampling lines, a series of sequences represents a face. Each face image is represented by sequences of edge distances, which are the only feature used in our system.

According to our observation, the series of sequences would be “similar” from the same person in limited variations. Note “similar” here means that the length of two sequences and the elements values in two sequences are close. Figure 3.1.1 shows an example that two images taken from the same person have similar sequences.

For a given series of sequence S_1, if the total cost between S_1 and another series of sequences S_2 is smaller than that of S_1 and other series, it means these two images which produce these two series of sequences S_1 and S_2 have the maximum possibility captured from the same person. Hence, face recognition can be performed by calculating the total cost among the sequences.

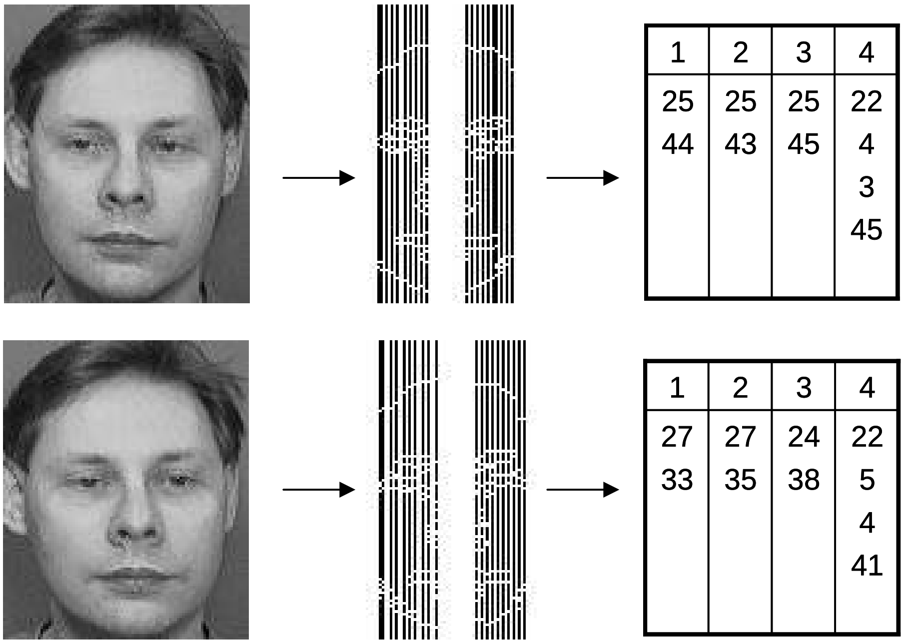


Fig. 3.1.1 Two images from the same person have similar edge sequences.

Because noise might be extracted and edge points might disappear during the preprocessing stage, different lengths of sequences are obtained from the same person practically. Subtracting the two elements in different sequences one by one, i.e. Euclidean distance, will cause mismatch. For example, although two sequences <15, 3, 2, 27> and <16, 5, 25> from the same person, then the cost between these two sequences would be $(1 + 2^2 + 23^2 + 27^2)^{1/2}$.

In this paper, we propose finding the best matching sequence based on an optimization criterion. Some properties of the matching process should be concerned:

1. All elements in a sequence should be matched.

Some useful information, such as the size of eye, might be lost if we discard several elements in the sequences. Therefore, all elements in a sequence should be matched in order to preserve the structural information.

2. All elements in a sequence should be matched in order.

Each edge distance recorded in a sequence represents structural information from top to bottom. If edge distances are matched out of order, structural information might be matched incorrectly; for example, the height of the forehead is probably mismatched to the height of the cheek.

We implement the best sequences matching by dynamic programming. In other words, we consider all cases of subsequences matching based on an optimization criterion in order to find the best matching sequences. Figure 3.1.2 shows an example all possible matching between two sequences <15, 3, 2, 27> and <16, 5, 25>. The detail algorithm would be listed in section 3.2.

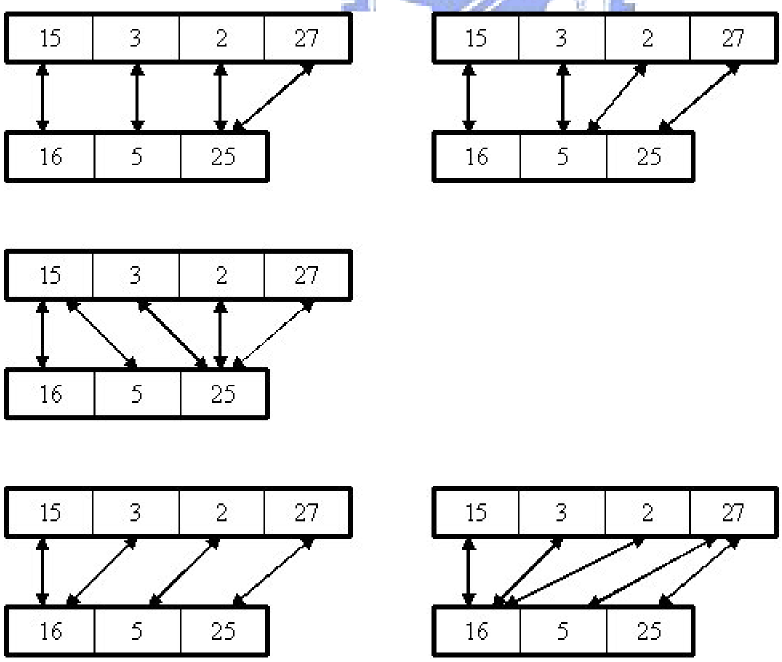


Fig. 3.1.2 All combinations of matching sequences between two sequences <15, 3, 2, 27> and <16, 5, 25>.

3.2 Assignment of Editing Cost

Since a face image is represented as a series of edge distances and matching is performed as string matching, three types of editing will be defined when comparing the two sequences. The three types of editing are listed below.

- **Substitution -** The difference between two elements in two sequences.
- **Split -** An element in a subsequence is matched to several elements in another subsequence.
- **Merge -** Several elements in a subsequence are matched to an element in another subsequence.

We use an example to explain the three types of editing further. Consider two sequences <11, 23, 34> and <10, 10, 15, 35>. The types of editing are shown in Figure 3.2.1.

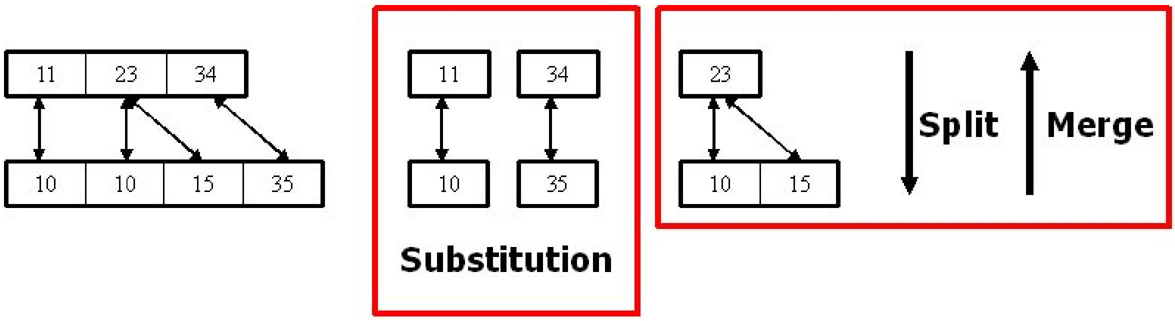


Fig. 3.2.1 Three types of edge distance editing between two series.

After defining the three types of editing, editing costs in our system are assigned as follows.

- **Substitution = absolute difference of two elements**
- **Merge = absolute difference of the sums of two subsequences + editing-penalty*number of merging distances**
- **Splitting is a reverse process of merging. We assign the same editing costs.**

Note the editing penalty, called merge penalty here, is detected according to the experimental results.

Although the minimum cost between two sequences could be calculated based on an optimization criterion, the same cost from different sequences might occurred without the merge penalty. For example, the cost between two sequences <11, 23, 34> and <10, 25, 35>, and the cost between two sequences <11, 23, 34> and <10, 10, 15, 35> are the same, as shown in figure 3.2.2.

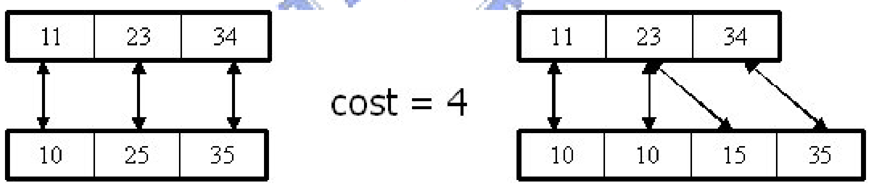


Fig. 3.2.2 An example the same cost from different sequences occurred

So we should add the merge penalty for each merge to weight the cost between two sequences. To continue the example above, when we weight the merge penalty, the minimum cost will be 4 between the two sequences <11, 23, 34> and <10, 25, 35>, and the minimum cost will be 4 + 1 * merge- penalty between two sequences<11, 23, 34> and <10, 10, 15, 35>. Because the value of the merge penalty is greater than zero reasonably, the sequence <10, 25, 35> is closer to the sequence <11, 23, 34> than the sequence <10, 10, 15, 35>.

Different values of merge-penalty would affect the selection of the best sequence with the minimum cost compared to a specific sequence, we use figure 3.2.3 to explain further. If the value of merge-penalty is greater than 1, the sequence <10, 25, 35> will be closer to the sequence <11, 23, 34>. Otherwise, the sequence <10, 15, 8, 32> will be closer to the sequence <11, 23, 34>. We will test and discuss the effects of different values of merge-penalty in our system.

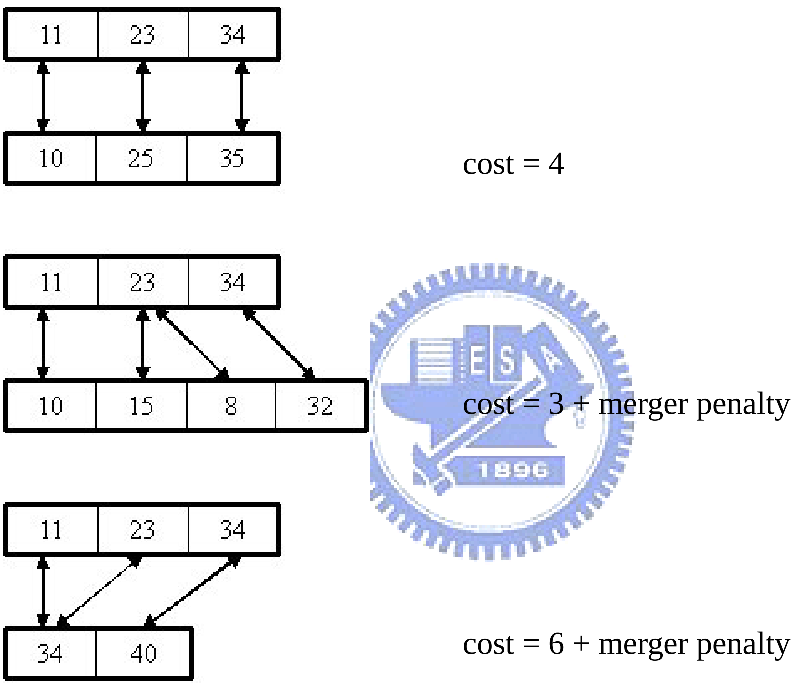


Fig. 3.2.3 The minimum costs for each sequence compared to the sequence <11, 23, 34>.

3.3 Optimization Algorithm

In this paper, we propose an algorithm to calculate the minimum cost between two sequences based on an optimization criterion. Assume that the number of the sampling lines in a face image is ***N***. There are accordingly ***N*** sequences to represent a face image. Let S_y^x be the sequence obtained from the ***y***th sampling line in the ***x***th face image, where ***y***=**1**…***N***.

Assume that two sequences $S_k^i = \langle a_1, a_2, \ldots, a_{t-1}, a_t \rangle$ in the ***i***th face image and $S_k^j = \langle b_1, b_2, \ldots, b_{s-1}, b_s \rangle$ in the ***j***th face image are both obtained in the ***k***th sampling line, where *t* may be equal to *s* or not. We define $\mathrm{cost}_k(l,t,n,s)$ as the cost function between two subsequence $\langle a_l, a_{l+1}, \ldots, a_{t-1}, a_t \rangle \subseteq S_k^i$ and $\langle b_t, b_{t+1}, \ldots, b_{s-1}, b_s \rangle \subseteq S_k^j$ in the ***k***th sampling line.

Then the dynamic programming algorithm to calculate the $\mathrm{cost}_k(l,t,n,s)$ is defined as follows:

$$\mathrm{cost}_k(l,t,n,s) = \begin{cases} \left| b_n + b_{n+1} + \cdots + b_{s-1} + b_s - a_l \right| + Merge\_penalty * (s-n) & if \quad l = t; \\ \left| a_l + a_{l+1} + \cdots + a_{t-1} + a_t - b_n \right| + Merge\_penalty * (t-l) & if \quad n = s; \\ \min\limits_{l \le u \le t,\, n \le v \le s} \left\{ \mathrm{cost}_k(l,u,n,v) + \mathrm{cost}_k(u+1,t,v+1,s) \right\} & otherwise; \end{cases}$$

Unless one of the two subsequences has only an element, the subsequences is split recursively to calculate the minimum cost in all possible combinations. We can obtain the minimum cost between two sequences S_k^i and S_k^j by calculating $\mathrm{cost}_k(1,t,1,s)$.

Accordingly, the total cost between the *i*th face image and the *j*th face image is sum of the costs of matching sequences in all sampling lines, that is,

$$total\_\mathrm{cost} = \sum_{k=1}^{k=N} \mathrm{cost}_k(1,t,1,s)$$

In the following, *i*th face image is called a test image, and the *j*th face image a reference face image in the face database. If the total cost between the test image and a reference face image in the face database is minimum compared to the other reference ones, we say the test image is most similar to this reference face image. These two images are regarded as obtained from the same person; in other words, we identify the test image as the person of the image that is most similar to the test image. The purpose of face recognition was achieved.

For example, there are five reference face images which were obtained from the different persons in the face database, labeled as the 1st reference face image, the 2nd reference face image and so on. Assuming the total cost between the test image and the 3rd reference face image in the face database is minimum compared to the other reference ones, the test person was identified as the 3rd person.

Chapter 4 Experimental result and Analysis

In this chapter, we present and discuss some experimental results. We will compare the recognition rate of different methods. The proposed system is developed on the Pentium 4 2.8G personal computer with 512 MB RAM.

In section 4.1, the briefs of two face database used are introduced. The implementation and the recognition rate of the experiments are given in section 4.2, we also compare the recognition rate of different methods in this section. Finally, we analyze the errors in section 4.3

4.1 Face databases

Two face databases, namely the Olivetti and Oracle Research Laboratory (ORL) [27] and the Institute of Information Science academia sinica (IIS) [28] face databases, were tested in our experiment.

4.1.1 ORL face database

The database consists of 400 images, 10 for each of 40 different subjects. Subjects were asked to face the camera and no restrictions were imposed on expression; only limited side movement and limited tilt were tolerated. The images were manually cropped and rescaled to a resolution of 92 x 112, 8-bit grey levels. Figure 4.1.1 shows some examples in the database.

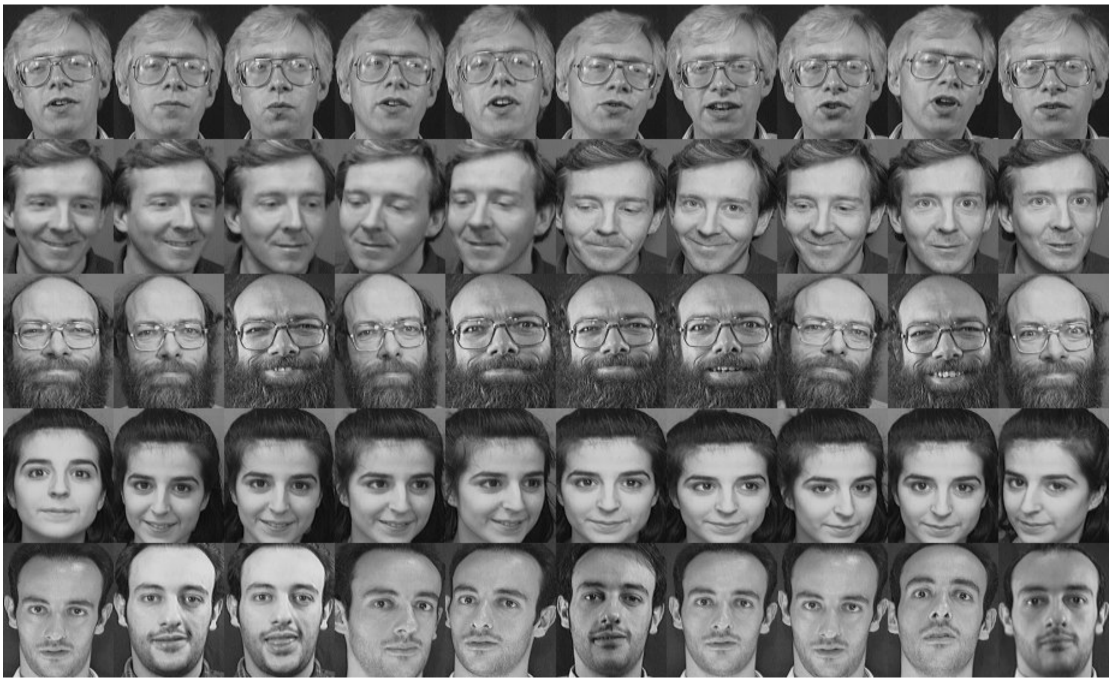


Fig. 4.1.1 Some examples in the ORL face database

4.1.2 IIS face database

The database consists of 3000 images, 30 for each of 100 different subjects. The 30 face images of each subject are composed of 10 frontal face image, 10 right profile face images, and 10 left profile face images. Because we do not consider the profile face in this study, we only adopt the 10 frontal face images for each subject. The total number of face images is thus 1000. No restrictions were imposed on expression for each subject, and the images were manually cropped and rescaled to a resolution of 155 x 175, 8-bit grey levels. Figure 4.1.2 shows some examples in the database.

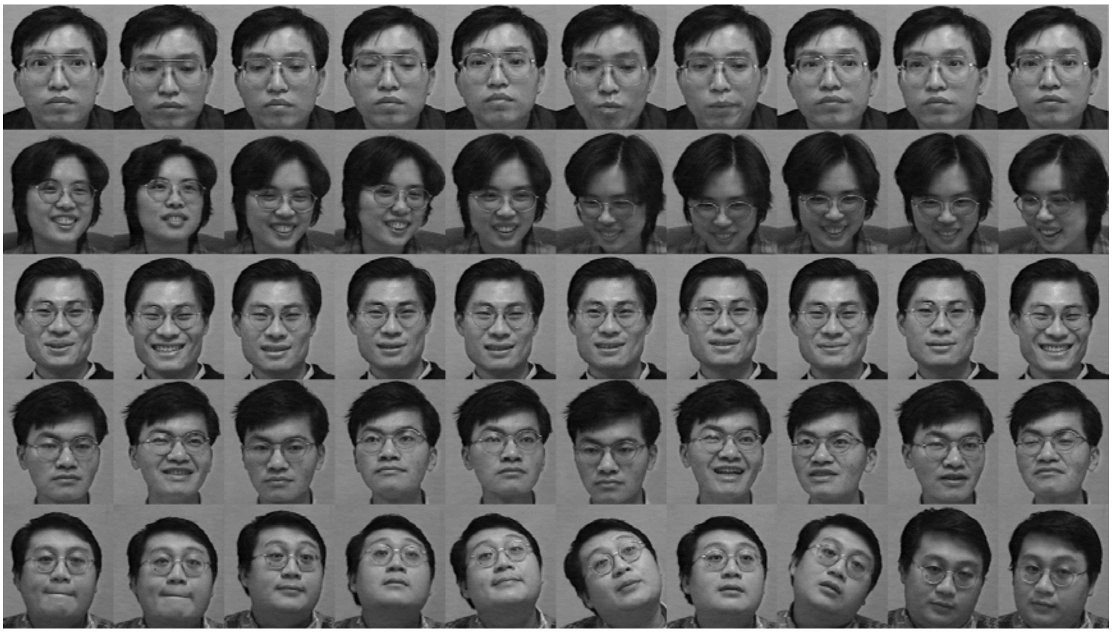


Fig. 4.1.2 Some examples in the IIS face database

4.2 Implementation and Result of Experiments

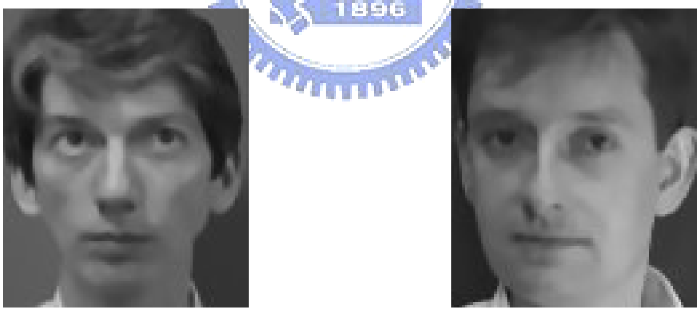
We take five images of each subject as reference images and five for testing in both databases. Hence, a total of 200 training and 200 test images in the ORL dataset and a total of 500 training and 500 test images in the IIS data were used. A test image is counted as correct recognition if the total cost between the test image and one of the training images from the same person is the minimum in among all training images. The results of the recognition rate for different face databases are shown in the Table 4.2.1. We also show three error cases for each face database, as shown in figure 4.2.1 and 4.2.2.

As mentioned in section 3.2, different values of merge-penalty would affect recognition results. Different values of merge-penalty are tested using the ORL face database and shown in figure 4.2.3.

Table.4.2.1 Recognition results for ORL and IIS face databases:

Database	Matching image	Test image	Error	Correct	Recognition rate
ORL (total = 400)	200	200	6	194	97%
IIS (total = 1000)	500	500	44	456	92.1%

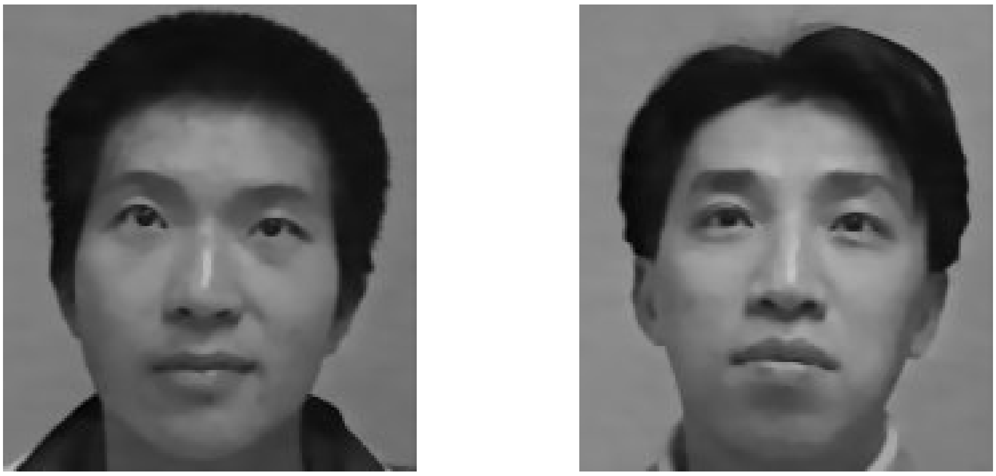
(a)



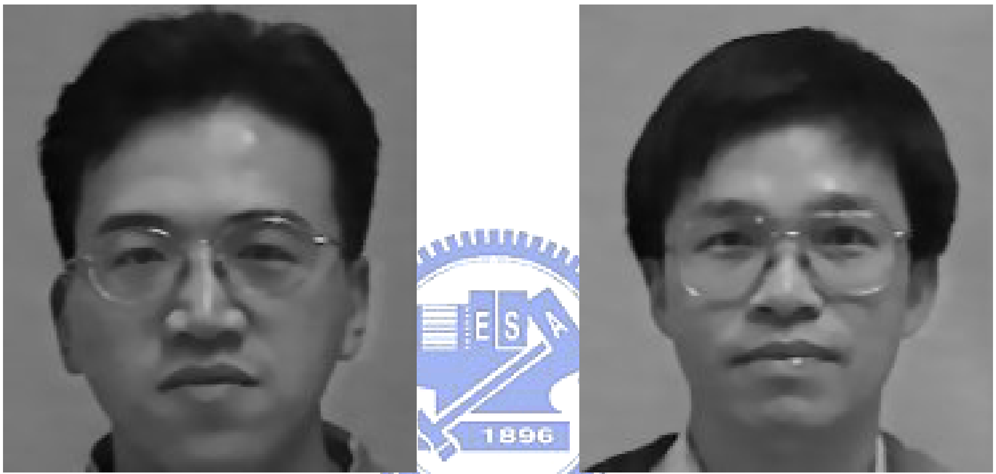
(b)

(c)

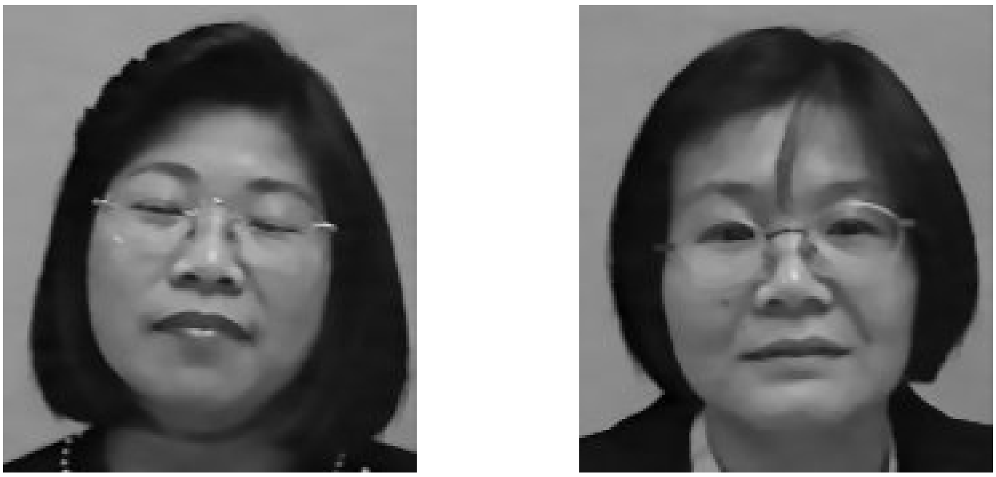
Fig 4.2.1 Three error cases (a)(b)(c) for the ORL face database.



(a)



(b)



(c)

Fig 4.2.2 Three error cases (a)(b)(c) for the IIS face database.

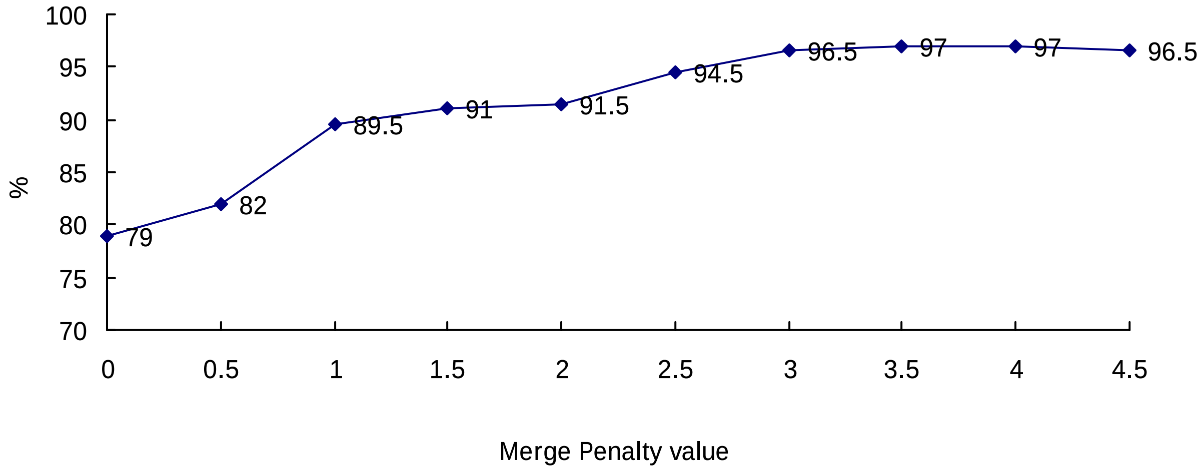


Fig. 4.2.3 Different recognition results obtained from different values of merge-penalty in the ORL face database with edge distance threshold = 1 and 20 sampling lines.

We can see that recognition rate only have about 79% if we do not add the merge penalty. When the value of merge penalty is greater than 1, the recognition rate is up to 90%. The maximal recognition rate is up to 97% at the value of merge penalty equal 3.5. Then recognition rate is growing decreasingly when the value of merge penalty is growing increasingly. That means the proper value of merge penalty required in our system.

Then we discuss the effect of different edge distance thresholds as mentioned in section 2.4. Different edge distance threshold would result in different edge maps in detail. We test different edge distance thresholds using the ORL face database. Figure 4.2.4 shows the results. We can see that recognition rate is decreased slowly when edge distance thresholds increase. That means "detail" edge distances play an important role in our system. Because these detail edges distances enhance different edge distributions from different persons. Although using more detail edge distances are resulted in better recognition rates, more preprocessing time is also required. This is a tradeoff between performance and preprocessing time.

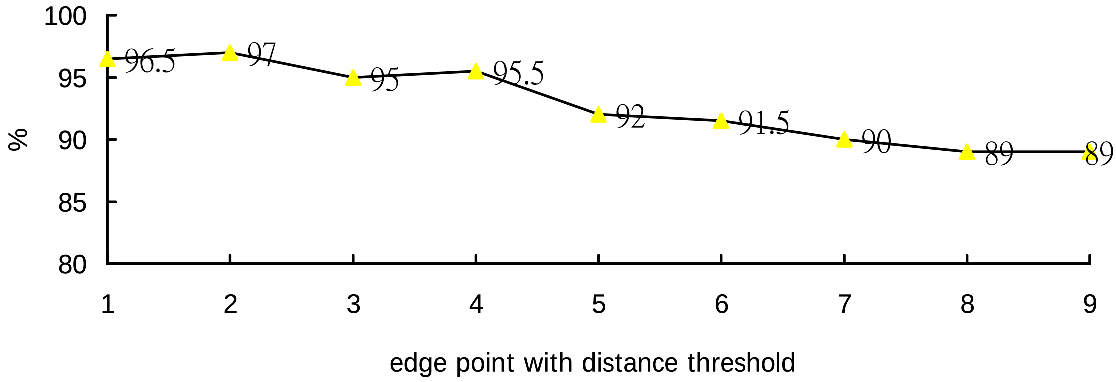


Fig. 4.2.4 Different recognition results obtained from different edge distance threshold in the ORL face database with merge-penalty value =3.5 and 20 sampling lines.

The number of sampling lines also plays an important factor in our system. Figure 4.2.5 shows the results tested by using different numbers of sampling lines in the ORL face database. From Figure 4.2.5, we may conclude when the number of sampling lines is 20, we can obtain the best performance in our experiment.

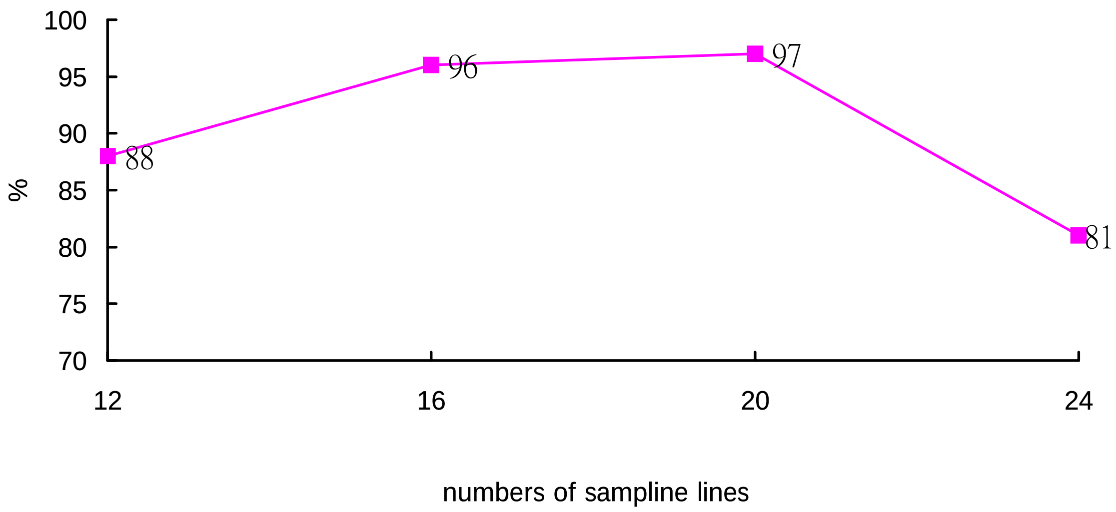


Fig. 4.2.5 Different recognition results obtained from different numbers of sampling lines in the ORL face database with edge distance threshold = 1 and merge-penalty value =3.5.

Although the nose plays an insignificant role according Chellappa [23] reported, we test the performance using the whole face image on the ORL face database. The recognition result is shown as table 4.2.2, which is compared with the result using only two eye strips in the face image.

Table 4.2.2 Comparative recognition rates for the different information used on the ORL face database, where merge-penalty = 3.5 and distance threshold = 2.

	Using the whole face image	**Using only two eye strips**
Recognition rate	95.5%	97%

The recognition rate using the whole face image is close to it using only two eye strips. Because the whole face images include more information, it is required more preprocessing time and capacity using the whole face image. In other words, using only two eye strips in the system not only gets higher performance, but also saves the preprocessing time and capacity. Hence, it seems a better way just using the information in the two eye strips.

Final, we compare our face recognition algorithm to other algorithm. The results are shown in table 4.2.3. The recognition rates of eigenface-based and elastic matching methods are obtained from the reference papers.

Table 4.2.3 Comparative recognition rates for the ORL face database

Classifier Method	**Recognition rate**	**Author**
Eigenface	80% 89.5%	Zhang et al [9] Lawrence [10]
Elastic Matching	80%	Zhang et al [9]
Edge Distance	97%	

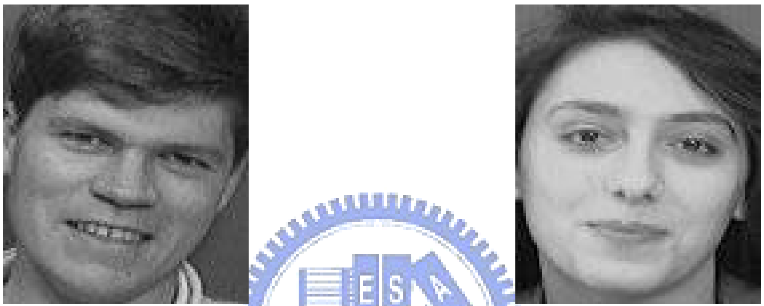
The eigenface-based method did not perform well on the ORL face database because the method was sensitive to the lighting variation according to Zhang [9].On the other hand, a person only had a model to be identified in the system, i.e. view-based method was not adapted, so the performance of the eigenface-based method was also affected by the face under variations in orientation.

The elastic matching method could only deal with position, lighting, and expression variations. However, the ORL face database contains some scale and rotation variations, which are the reasons why the elastic matching method did not perform well on the ORL face database.

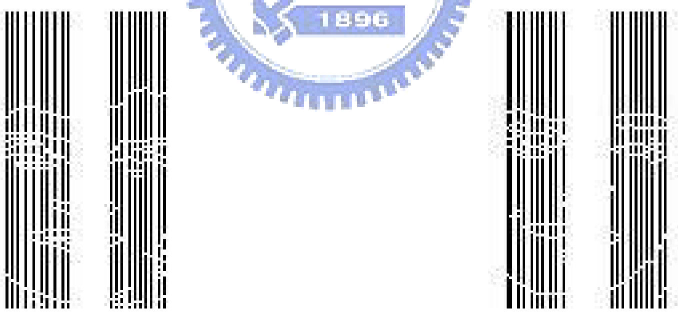
The edge distance method performed well on the ORL face database because the method was designed to solve the problem of the lighting and rotation variation. The edge map which was obtained from the source image after applying the edge detection might overcome the problem of the lighting variation. And we adopt the view-based method to get over the problem of the face under variations in orientation. The recognition result was encouraging.

4.3 Error Analysis

In our experiment, errors could be categorized into two types. The first type of errors is improper edge maps obtained because edge points are removed incorrectly. Figure 4.3.1 shows an example belonging to the first type of errors. The second type of errors may occur when similar edges distributions are obtained from different person, as shown in figure. 4.3.2.



(a)



(b)

Fig. 4.3.1 Imperfect edge maps that error is caused by removing unreliable edges incorrectly. (a) Source images (b) The edge maps after removing unreliable edges.

To remove the unreliable edge points we find the face contour first. However, to find the face contour is a difficult task due to the hair style, glasses and so on. For example, the hair affects the location of lower hair border in figure 4.3.1 and results in the misclassification.

(a)

(b)

Fig. 4.3.2 Two persons with similar edge distribution (a) Source images (b) The edge maps.

We only adopt the information in two eye strips. The other useful information beyond the two eye strips maybe lost. For example, in figure 4.3.2, the two persons have greater differences in their noses and the central part of mouth. However, these two types of information are not adopted in our system. Note it does not mean the nose information is useful for any cases.

In summary, we should address how to record “correct” edge distances to avoid the error occur. We may need more image processing techniques to support our system.

Chapter 5 Conclusion and Future Works

In this paper, we have designed an automatic face recognition system based on edge distances. The system consists of two main phases: face features extraction and face recognition using these features. In the face features extraction stage, because the edge images of objects could be used for object recognition and to achieve similar accuracy as gray-level images, we first preprocess a face image in order to obtain an edge map. Second we find the important parts of a face, which are the most invariant parts of a face. We use two eye blocks to include these parts. Since the widths of the two eye blocks for different face images may be different, we take a fixed number of sampling lines in each of the eye blocks. After removing unreliable edges, we record edge distances in each vertical sampling line.

In the face recognition stage, the sequences of edge distances would be similar with respect to the same person in limited variations, so face recognition can be performed by calculating the distance between the sequences. Since the length of two different sequences from different face images of the same person would not be exactly the same, we match the sequences based on an optimization criterion. Besides, we should add the merge-penalty in matching sequences. Therefore, a person could be identified as the one whose series of sequences has the minimum distance among his own series of sequences.

In our system the dimension of a face image is reduced. Suppose a face image size is X * Y. The size of series of sequences required is only N * C, where N is the numbers of sampling lines and C is the average number of edge distances in a sampling line. It can save the capacity of storage and the processing time by reducing the information of images.

For the future work, some suggestions are listed below.

(1) The value of merge-penalty may be adjusted dynamically using some information such as color around the edges to enhance the performance of the system.

(2) We may find the number of sampling lines based on a mathematics model to differentiate a human more accurately.

(3) Concerning the transition between the sampling lines, we may use other models, for example the hidden Markov model, to obtain better performance.

References

[1] L. Sirovich and M. Kirby, “Low-Dimensional Procedure for the Characterisation of Human Faces,” J. Optical Soc. of Am., vol. 4, pp. 519-524, 1987.

[2] M. Turk and A. Pentland, “Eigenfaces for Recognition,” *J. Cognitive Neuroscience*, vol. 3, no. 1, pp. 71-86, 1991.

[3] M.A. Grudin, “A Compact Multi-Level Model for the Recognition of Facial Images,” PhD thesis, Liverpool John Moores Univ., 1997.

[4] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman, “Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection,“ IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, pp. 711-720, 1997.

[5] L. Rabiner, “A tutorial on Hidden Markov Models and selected applications in speech recognition,” in *Proceedings of IEEE*, vol. 77, pp. 257-286, Feb 1989.

[6] F. Samaria and S. Young, “HMM-based architecture for face identification,” *Image and Computer Vision*, vol. 12, pp.573-583, October 1994.

[7] F. Samaria and A. Harter, “Parametrisation of stochastic model for human face identification,” in *Proceedings of the Second IEEE Workshop on Application of Computer Vision*, 1994.

[8] F. Samaria and F. Fallside, “Face Identification and Feature Extraction Using Hidden Markov Models,” Image Processing: Theory and Application, G. Vernazza, ed., Elsevier, 1993.

[9] S. Lawrence, C. Giles, A. Tsoi, and A. Back, ªFace Recognition: A Convolutional Neural Network Approach,” *IEEE Trans. Neural Networks*, vol. 8, pp. 98-113, 1997

[10] J. Zhang, Y. Yan, and M. Lades, “Face Recognition: Eigenface, ElasticMatching, and Neural Nets,” in *Proceedings of IEEE*, vol. 85, pp. 1,423-1,435, 1997.

[11] L. Wiskott and C. von der Malsburg, “Recognizing Faces by Dynamic Link Matching,” Neuroimage 4, pp. S14-S18, 1996.

[12] Y. Gao and M.K.H. Leung, “Face recognition using line edge map,” *Pattern Analysis and Machine Intelligence, IEEE Trans. on* , vol. 24 , pp. 764-779, June 2002.

[13] F. Y. Shih and C. F. Chuang, “Automatic extractioin of head and facial features,” *Information Science*, vol. 158, pp. 117-130, 2004.

[14] Y. C. Tsao, “Computation of Face Recognizability for human Surveillance” M.D thesis, National Chao Tung Univ.,2004

[15] B. Takacs, “Comparing Face Images Using the Modified Hausdorff Distance,” *Pattern Recognition*, vol. 31, pp. 1873-1881, 1998.

[16] A. Pentland, B. Moghaddam, and T. Starner, “View-Based and Modular Eigenspaces for Face Recognition,” in *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 84-91, 1994.

[17] R. Bruneli and T. Poggio, “Face Recognition: Features versus Templates,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, pp. 1042-1052, 1993.

[18] M. Lades, J. Vorbruggen, J. Buhmann, J. Lange, C. V. D. Malburg, and R. Wurtz, “Distortion invariant object recognition in the dynamic link architecture,” *IEEE Trans. Computer.*, vol.42, no. 3, pp. 300–311, 1993

[19] C. Zhang and F. S. Cohen, “3-D Face Structure Extraction and Recognition From Images Using 3-D Morphing and Distance Mapping,” *IEEE Trans. on Image Preprocessing*, vol 11, no. 11, November 2002

[20] G. C. Feng and Pong C. Yuen, “Recognition of Head-and-Shoulder Face Image Using Virtual Frontal-View Image,” *IEEE Trans. on Systems, Man, and Cybernetics – Part A: Cybernetics*, vol. 30, pp. 871- 883, November 2000.

[21] I. Biederman and J. Gu, “Surface versus Edge-Based Determinants of Visual Recognition,” *Cognitive Psychology*, vol. 20, pp. 38-64, 1988.

[22] V. Bruce et al., “The Importance of “Mass” in Line Drawings of Faces,” *Applied Cognitive Psychology*, vol. 6, pp. 619-628, 1992.

[23] R. Chellappa, C. L. Wilson and S. Sirohey, “Human and Machine Recognition of Faces: A Survey,” in *Proceedings of the IEEE*, vol. 83, pp.705-740

[24] M. Nagao and T. Matsuyama, "Edge Preserving Smoothing," *Computer Graphics and Image Processing*, vol. 9, pp.374-407, 1979

[25] K.Lam and H. Yan, “Locating and extracting the eye in human face images,” *Pattern Recognition*, vol. 29, pp.771-779, August 1995.

[26] S. A. Sirohey and A. Rosenfeld, “Eye detection in a face image using linear and nonlinear filters,” *Pattern Recognition*, vol. 29, pp.1367-1391, 2001.

[27] Olivetti and Oracle Research Laboratory Face Database, http://www.uk.research. att.com:pub/data/att_faces.zip, 2004.

[28] Institute of information Science academia sinica Face Database, http:// smart.iis. sinica.edu.tw/html/face.html, 2004