# Chapter 4

# 4 Facial Feature Extraction

## 4.1 Introduction

In an automatic 3D face synthesis system, facial feature extraction plays a crucial role. The detected features provide the 3D face synthesiser with key attributes, such as the shape, size and orientation information [She03b] [She04b], which are needed in order to make 3D vertex topology of a generic face model fit a specific face. Nevertheless, similar to face detection, automatic facial feature extraction still remains a challenging topic, due to unpredictable factors like lighting conditions, partial occlusions, as well as the variability of facial expressions. Generally speaking, after the face is first detected from the background in a head-and-shoulder image, facial features such as the eyes, eyebrows, mouth and chin, are in turn extracted based on the detected face region. Too many facial features would increase the computational complexity, while too few would result in inaccurate fitting of the face model. In order to precisely extract facial features, various approaches aimed at different sets of facial features have been proposed in a diversity of modalities. The mainstream approaches can be categorised into brightness-based and edge-based algorithms, which are surveyed in section 4.2. In order to implement effective extraction of facial features, section 4.3 to section 4.6 elaborate the proposed and employed methods for extraction of the eyes, eyebrows, mouth and chin, respectively. Also, experimental results in section 4.7 will back up the feasibility of these methods. Finally, section 4.8 summarises the chapter.

## 4.2 Facial Feature Extraction Background

An enormous market demand for applications such as videoconferencing and videophony, has led to considerable investigation of facial feature extraction. Despite the great number of algorithms that have been proposed, automatic facial feature extraction still remains a challenging topic, and requires more robust solutions. Automation has the strict demand of high performance consistency

in each part of the system, so as to be capable of smoothly and robustly coping with various situations. In other words, automation needs minimal operator supervision so as to avoid the abuse of predefined thresholds and training data. However, facial features are geometrically complex, and inconsistent in terms of shape, size and colour. It means that different approaches have to be applied to different facial features. In general, regardless of manual acquisition, the mainstream approaches in facial feature extraction can be categorised into the brightness-based and edge-based algorithms. This section surveys the current state of the art in facial feature extraction. Section 4.2.1 focuses on the brightness-based algorithms which can merely find rough position of the facial features; while section 4.2.2 concentrates on introduction of three edge-based algorithms which are widely used currently.

### 4.2.1 Brightness-based algorithms

Brightness-base algorithms exploit the brightness characteristics of the images to extract facial features. A typical approach of this class of algorithms is to employ the knowledge of the geometrical topology and the brightness characteristics of facial features, such as the eyebrows, eyes and mouth. In [Sob96], the authors exploited the fact that the brightness of the eyes and mouth is lower than the rest of the face. Those facial features in the detected face region are initially enhanced by morphological erosion. Then the topographic grey level relief in both horizontal and vertical directions is utilized to find the facial features. For instance, the first minimum value of the vertical relief from the top to bottom always occurs at the position of the eyes. In addition, two minima appear in the corresponding horizontal direction, which are approximately symmetric in terms of value. With some constraints, the approximate position of these facial features can be located.

Besides the face detection techniques reviewed in the previous chapter, template matching has also been employed in facial feature extraction. *Zhang* used corner templates, derived from three ITU video sequences, *Claire*, *Miss America* and *Michael*, to detect the existence of corners of the eyes and mouth [Zha96]. These templates are roughly adapted to the size of the person's face and rotated towards the orientation of the head. Calculation of the correlation is based on the corner templates and the face image, with the areas that have the maximum correlation value being treated as the corners. Similarly, *Kampmann* used template matching to detect the centres of the eyes and mouth [Kam97a]. Evidently, the disadvantage of template matching is that it cannot accurately cope with variations in scale, pose and shape.

*Cao et al* [Cao02] used a region-growing search to locate the iris centre. This algorithm assumes that the face has already been detected, with known orientation and size. When locating the right

eye centre, an aspect-fixed rectangle is initially placed on the face region with the bottom-left corner of the rectangle pegged at the nose centre. Then the top-right corner of the rectangle is gradually pulled towards the top-right eye, step by step, until the top-right corner of the rectangle reaches the boundary of the top-right eye. The mean of the intensity, calculated within the rectangle, is used to decide whether the boundary of the rectangle touches the eye by comparing the mean with an empirically defined threshold, because the brightness of the eyes is perceivably lower than that of the face surface.

It can be easily seen that the brightness-based methods are only capable of finding the approximate position of facial features, and playing an initial role in facial feature extraction.

### 4.2.2 Edge-based algorithms

Edge-based algorithms target contours of the features, such as those of the mouth, eyes and chin, usually based on the gradient images. When these algorithms are used to extract the facial feature contours, coarse detection of these facial features needs to be performed. In this subsection, three typical edge-based algorithms, *Hough* transform, active contour model, i.e. snakes, and deformable templates, are in turn studied.

#### 4.2.2.1 *Hough* transform

The *Hough* transform is a standard tool in image analysis that allows recognition of global patterns in an image space by recognition of local patterns (ideally a point) in a transformed parameter space. It is particularly useful when the patterns that one is looking for are sparsely digitized, have ``holes'' or the pictures are noisy [Gon02]. The basic idea of this technique is to find curves that can be parameterized such as straight lines, polynomials, circles, etc, in a suitable parameter space. The computational attractiveness of the *Hough* transform arises from subdividing the parameter space into so-called accumulator cells. The basic geometrical components such as straight lines, circles, etc can be found by searching maximum accumulations in the accumulator cells of the correponding parameter spaces.

The *Hough* tranform has been widely applied to facial feature extraction as the contours of some facial features can be viewed as basic geometrical curves. For instance, contours of the irises appear as circles without any occlusion. Many researchers exploit this idea to locate the irises with the *Hough* transfom [Cho93] [Ber98]. However, the *Hough* transform has a drawback that the circle with a larger diameter has a longer contour, and thus will have a higher accumulation value than the circle with a smaller diameter. This could result that the eyelid, an arc with a larger

diameter, is regarded as the iris. Therefore, other constraints are required. More technical details about iris detection using the *Hough* transform will be explained in section 4.3.3. Moreover, the *Hough* tranform of a straight line was also used to locate the mouth [Cho93], because there is a dark seam between two lips that approximately appears a straight line if the mouth is closed.

#### 4.2.2.2 Active contour model

Active contour model, i.e. snake, extensively used in tracking a single deformable object [Ley93], head boundary location [Wai90], and facial feature detection [Hu03], is an energy-minimizing spline guided by external constraint forces and influenced by image forces that pull it toward features such as lines and edges [Kas88]. Snake is governed by a potential energy function, consisting of two force terms: (a) External energy, the data-driven component, which depends on the image data according to a chosen constraint; (b) internal energy, the smoothness-driven term, which imposes the smoothness along the snake [Tse04]. The goal of detecting the object boundary is achieved by minimising the potential energy function. The internal energy can be expressed in terms of B-spline polynomials [Bla98], curvature [Tse04], or the first and second derivatives of a curve, while the external energy term is defined using the negative image gradient.

Snake is renowned for its flexibility in fitting object boundaries. But, this algorithm has the disadvantage of easily converging to the local minimum, because snake is extremely sensitive to noise if the object edges are weak, or the background contains strong edges close to the object boundary. In order to improve its convergence accuracy, the Gradient Vector Flow (GVF) [Xu98], computed as a diffusion of the gradient vectors of a gray-level or binary edge map from the original image, is exploited as the external force, rather than using image gradient directly. The main advantage of GVF is that it can capture a snake from a larger range and force it into concave regions.

Variational approaches are most widely used to solve the energy-minimizing snakes. Nevertheless, variational approaches cannot guarantee a globally optimal solution, require estimation of higher order derivatives of the discrete data, and do not allow direct and natural enforcement of constraints. Moreover, if a contour is not subject to any external forces, it will vanish to a line or point, and if it is not placed close to image boundaries, it will not get attracted. In order to solve these bottlenecks in the variational approaches, *Amini et al* [Ami90] proposed a dynamic programming algorithm. Dynamic programming ensures a globally optimal solution. It is numerically stable, and allows for hard constraints to be enforced on the behaviour of the solution within a natural and straightforward structure. Taking advantage of the inherent discrete nature of the problem, the optimization problem is set up as a discrete multistage decision process and is

(a) (b) (c)

(d) (e) (f)

Figure 4.1 Snake crawling [SNK]
(a) 1st round (b) 5th round (c) 10th round
(d) 15th round (e) 20th round (f) 25th round

solved by a "time-delayed" discrete dynamic programming algorithm. Figure 4.1 shows the process of how a snake crawls, and clings to the object boundary by iteratively applying the dynamic programming algorithm till convergence of the energy. However, dynamic programming cannot prevent snakes from converging to local minima, if the peripheral areas of the object are noisy. Furthermore, snakes rely on other mechanisms to place them somewhere near the desired contour, and always have a problem when boundaries of the object are occluded by obstacles.

#### 4.2.2.3 Deformable template matching

Unlike the ordinary template matching algorithms, deformable templates interact with the image in a dynamic manner. An energy function should be defined that contains terms attracting the template to salient features, such as local peaks, valleys and edges. The parameters are obtained by minimising the energy function. A survey paper about the deformable template algorithms can be found in [Jai98].

Deformable template matching was initially suggested in facial feature extraction by *Yuille et al* [Yui92]. It assumes that the boundaries of the eyelids and mouth lips can be viewed as continuous parabolas. In their method, three *a priori* elastic parabolic curve models are set up for the eyes, the closed mouth and the open mouth. These facial features are defined in terms of edges, peaks and valleys corresponding to different energy functions. The best-fit elastic model is found by minimising a combination of energy functions of the edges, peaks and valleys. In comparison with other approaches, the deformable template matching algorithms have been extensively exploited in various facial feature extractions, and shown their availabilities not only in eyes [Che92] and mouth detection [Mir97] [Rab97] [Zha97] [Yin02], but also in chin [Kam97b] and even nose detection [Yin01]. More details on the deformable templates matching concerning various facial features will be studied later in this chapter.

Compared with the algorithm of the active contour model mentioned above, deformable template matching can guarantee a complete shape of the object boundary when occlusion occurs. Although this method has good experimental performance in extracting facial features, one shortcoming of this approach is that it must be initialized in the proximity of the interesting features. Furthermore, the deformable template matching is quite sensitive to articulation of the image. So, facial features must have a clear presence in the image. Therefore, this method is not applicable to very low resolution images.

## 4.3 Edge-based Eye Extraction

As agreed previously, feature extraction of each facial feature is comprised of two steps, location of the features, followed by extraction. The eyes are one of the most salient features on the face in terms of their grey-level characteristic and geometrical topology. To extract the detail of the eyes, edge-based deformable template matching is no use when image resolution is not high enough. In this section, a proposed eye feature extraction approach is elaborated, consisting of eye location, feature extraction and iris extraction.

### 4.3.1 Eye location

From observation, textures around and inside the eyes is complicated. This makes the eyes different from other peripheral areas of the face. With this idea in mind, an edge-based method is proposed. Gradient-based edge detection is carried out based on the luminance image using the *Sobel* gradient operators to create an edge map of the luminance image. Figure 4.2 (b) shows the

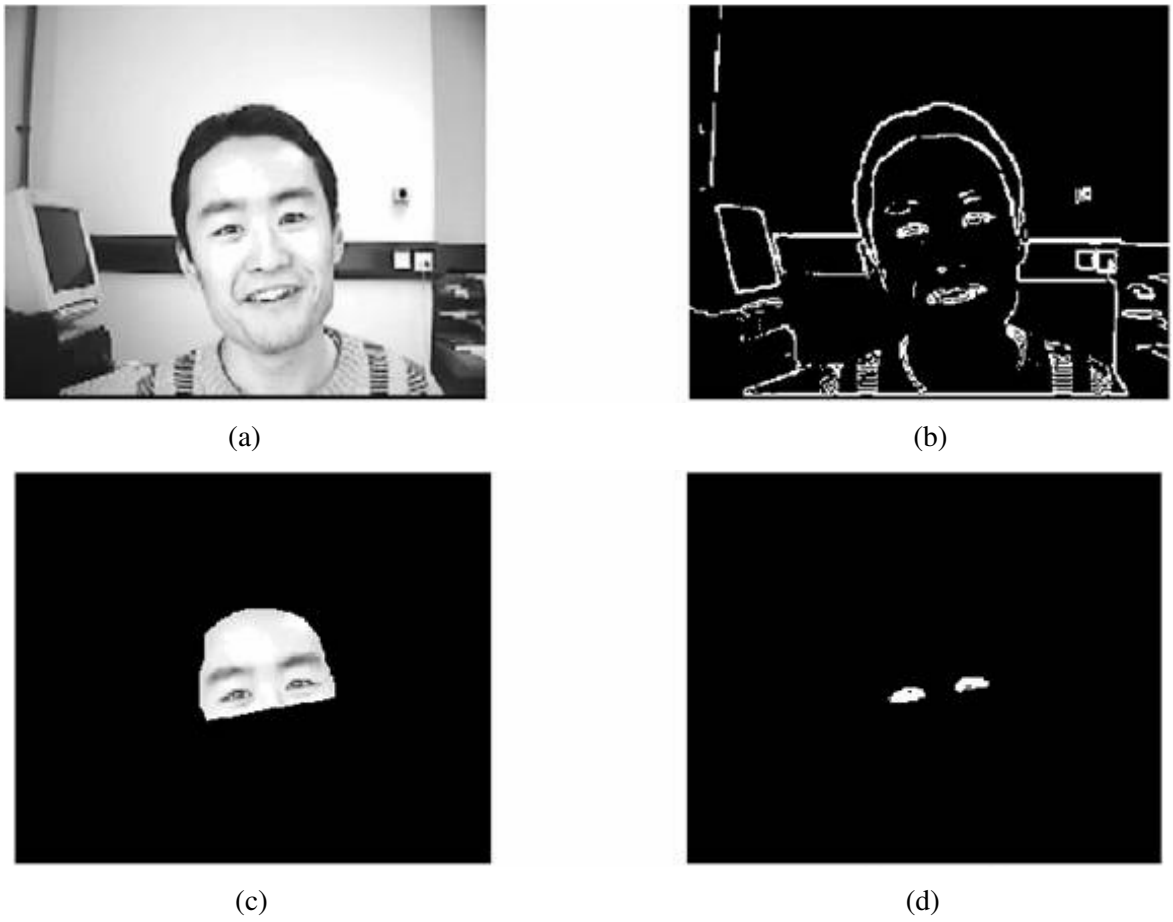


Figure 4.2 Eyes segmentation
(a) Luminance image (b) Edge image
(c) Search area (d) Eye candidates

edge map of the luminance image in Figure 4.2 (a). As can be seen, the eyes are isolated after applying edge detection.

The search area can be narrowed into the upper half of the face. Furthermore, the search area must be rotated as the head tilts. A rough estimate of the face orientation can be found by calculating the central moments of the detected face using the previously introduced face detection algorithm. The $(p, q)$ order central moments are expressed as

$$\mu_{p,q} = \sum_{x} \sum_{y} (x - \bar{x})^p (y - \bar{y})^q \tag{4.1}$$

where $(\bar{x}, \bar{y})$ is the centroid of the detected face. So the orientation of the face is defined as:

$$\theta = \frac{1}{2}\tan^{-1}\left[\frac{2\mu_{1,1}}{\mu_{2,0} - \mu_{0,2}}\right] \tag{4.2}$$

where $\theta$ indicates the angle between the major axis (the long axis of the face if considering the face as an ellipse) of the face and the horizontal coordinate axis. Figure 4.2 (c) shows the search area for the eyes. The edge information in the search area is smoothed by morphological filters. Then the remaining objects in the search area are labelled, and abandoned if their size is less than 40% of the maximum object. Figure 4.2 (d) shows the most likely candidates.

To find the first eye, the main uncertainty is the areas above the eyes and below the eyebrows. Since the edges of the eyebrows are obvious, whereas the areas underneath the eyes are insensitive to edge detection because of the absence of any facial feature in the area. Therefore, scanning along the head inclination, from bottom to top, left to right, the object first detected must be one of the eyes. The position of the second eye can be discovered as the two eyes are mutually and geometrically symmetrical, with respect to the major axis of the face, and can be traced along the line,

$$y - y_c = K\,(x - x_c) \tag{4.3}$$

where $K = -1/\tan\theta$, and $(x_c, y_c)$ is the coordinate of the first eye centroid. The tracing direction is determined by the following rule:

- If $x_c < \bar{x}$, then the left eye in the image has been found. The tracing direction should face towards the right;

- If $x_c > \bar{x}$, then the right eye in the image has been found. The tracing direction should face towards the left.

The line in Figure 4.3 (a) between the two objects illustrates the tracing trajectory, which is perpendicular to the major axis of the face. The object that has the same colour as the tracing trajectory line is the first detected eye. More technical detail about the above proposed eyes location algorithm can be found in [She02].

### 4.3.2 Feature extraction

Since the edge-based deformable template matching algorithms are related to a few strength-dependent curves, they have a high demand on pattern articulation. Therefore, the edge-based

(a) (b)

Figure 4.3 Discovery of 2nd eye and extraction of feature points
(a) Search trajectory of second eye (b) Extracted eye feature points

algorithms are sometimes not applicable to those images with very low resolution, like those images in the QCIF format. As the conventional deformable template matching method cannot robustly be applied to some small pattern extraction problems such as the eyes, we proposed a solution that employs only the parabolic templates which are pegged on the boundaries of detected eyes. This eye boundary extraction method is described as follows:

1. Four feature points, the left and right corners, and the upper and lower middle points of the eyelids, are in turn extracted from the previously extracted eyes [She02].

2. After two eye corners and two middle points on the eyelids have been located for each eye, two parabolas are applied to the detected eyes in light of the position of the four feature points, with the parabolas ending at the two eye corners, and touching with the upper and lower middle points in each eye, respectively.

Figure 4.3 (b) shows the extracted feature points of two eyes using the proposed algorithm while Figure 4.4 (a) shows the extracted eye contours.

### 4.3.3 Iris extraction using *Hough* transform

With the exception of the case when eyes are closed, the contour of the iris can be viewed as a circle embedded within the eye. Therefore, the method of extracting the irises is based on the circular *Hough* transform with edge detection [Gon02].

(a) (b)

Figure 4.4 Eye feature extraction
(a) Extracted eye contours (b) Extracted irises

The equation of a circle can be expressed as:

$$(x - x_c)^2 + (y - y_c)^2 = r^2 \tag{4.4}$$

where $(x_c , y_c)$ is circle centre, while $r$ is the circle radius. Applying the *Hough* transform results in a 3D parameter space with a large number of cubelike cells and accumulators of the form $A(x_c , y_c , r)$, where three axes are $x_c$ , $y_c$ and $r$, respectively. The procedure is to scan the iris search area, which is narrowed into a rectangle formed by the four extracted feature points on each eye (Figure 4.3 (b)), on the edge map of the luminance image. When reaching an edge pixel, $x_c$ and $y_c$ are incremented to solve for the radius $r$ that satisfies the equation (4.4), updating the accumulator $A(x_c , y_c , r)$. By the end, indices of the maximum accumulator in the 3D parameter space should specify the centre and radius of the eye.

Theoretically, a comparison can be carried out based on the above accumulated count. But, in fact, a circle with a large radius has a long contour, always resulting in a larger count than a circle with a smaller radius. Thus, to avoid it, all the accumulators in the 3D parameter space have to be normalised by dividing the corresponding perimeters. Nevertheless, it raises another problem that a small circle may have the same score as the large one. So we propose to normalise all the accumulators as follows:

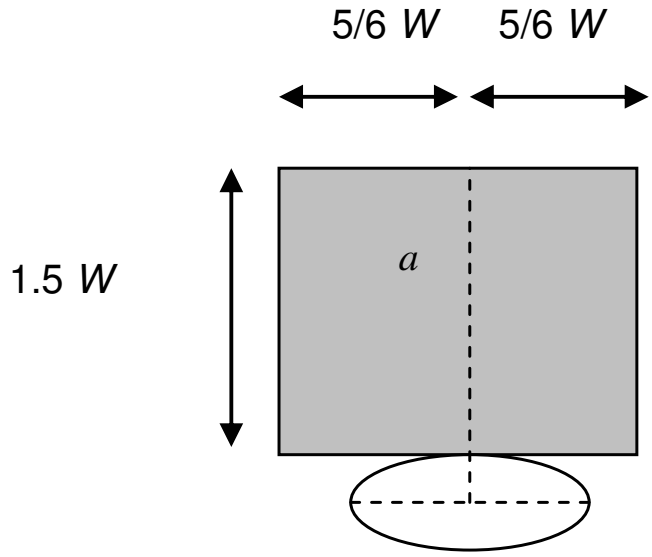



Figure 4.5 Eyebrow search area

$$S = \frac{v^2}{2\pi \times r} \tag{4.5}$$

where $v$ indicates values of the accumulators, and $S$ represents the final score of each accumulator.

The proposed normalisation equation is similar to one used in [[Cho93]. Instead of using the diameter, the perimeter can enhance the algorithm precision. Figure 4.4 (b) shows the extracted irises using the circular *Hough* transform.

## 4.4 Gradient-based Eyebrow Extraction

The search area for the eyebrows can be narrowed by the fact that the eyebrows are permanently over their corresponding eyes. The rectangular shadow in Figure 4.5 illustrates the search area for an arbitrary eyebrow. The ellipse under the rectangle represents the eye. *W* denotes the longest distance between the left and right corners of the eyes. The vertical dashed line *a*, is the perpendicular bisector of the two eye corners. Moreover, the search area should be chosen to be tilted like the corresponding eye, and the orientation of the search area can be derived from the corresponding eye. Figure 4.7(a) shows the narrowed eyebrow search areas.

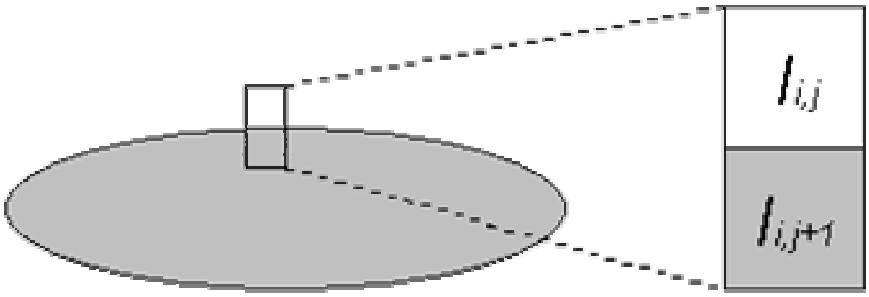



Figure 4.6 Illustration of two vertically adjacent pixels on the eyebrow border

Due to the low intensity of the eyebrows, an automatic eyebrow segmentation approach is proposed as follows. Let $I_{i,j}$ denote the intensity of a certain pixel within the narrowed search area, where the subscripts $i$ and $j$ indicate the horizontal and vertical coordinates of the image plane, respectively. The downward intensity gradient at $I_{i,j}$ can be formulised as $g_{i,j} = I_{i,j} - I_{i,,j+1}$. Then the maximum gradient value $G_i$ on the current column is sought because the place where the maximum gradient arises must be on the border of the eyebrows. Figure 4.6 illustrates two vertically adjacent pixels on the eyebrow border, where the ellipse indicates the eyebrow. Once the boundary position of the current column is discovered, we can determine the threshold value of the column by averaging the intensities of the two adjacent pixels as:

$$T_i = \frac{1}{2}\left(I_{i,j} + I_{i,j+1}\right) \tag{4.6}$$

To normalise the threshold, the mean of the determined thresholds on all the columns of the eyebrows is finally calculated:

$$T = \frac{1}{n}\sum_{i=i_0}^{i_{n-1}} T_i \tag{4.7}$$

where $n$ denotes the number of calculated columns, and $T$ is the finalised threshold value used in the eyebrow segmentation. Figure 4.7 (b) shows the segmented results of the eyebrows in Figure 4.7 (a). Similar to eye feature extraction, four eyebrow feature points are in turn extracted for each eyebrow, showed in Figure 4.7 (c).

(a) (b) (c)

Figure 4.7 Eyebrow feature extraction
(a) Eyebrow search areas (b) Binarised eyebrows (c) Extraction result of eyebrows

## 4.5 Staged Deformable Template Matching for Mouth Extraction

Mouth extraction is still challenging, due to some unpredictable factors such as lighting condition, occlusion, and the variability of face expressions. In comparison with other methods, the deformable template algorithm, initially reported by *Yuille* [Yui92], is one of the most effective ways to locate mouth contours and has attracted a substantial amount of attention. However, *Yuille*'s templates must be manually initialised in the vicinity of the mouth. Moreover, this algorithm makes no mention of how to distinguish whether the mouth is open or not. To implement an automatic system, *Chow et al.* [Cho93] utilised a *Hough* transform to locate the line segment between the two mouth corners by using the fact that the intersection of the lips appears dark when the mouth is closed. Nonetheless, their method is only capable of dealing with the closed mouth. *Zhang* applied corner templates to locate the mouth [Zha96]. But it is well known that mouth corner templates are unable to cope with variation in scale. *Zhang* also suggested a criterion of whether the mouth is open or closed, by calculating the number of candidate intersections, i.e. some intersections of lip contours and the perpendicular bisector of mouth corners [Zha98]. But this criterion is greatly dependent on the articulation of the detected edges. Furthermore, the issue of how to estimate these intersection candidates remains unresolved.

In fact, the challenges of using the deformable templates to extract mouth features can be attributed to the following: template positioning, criterion of mouth being open or closed, and selection of strength constraints. To solve these problems, the rest of this section addresses an automatic mouth extraction algorithm, with Staged Deformable Templates (SDTs). Before the templates are applied, a template positioning method based on mouth segmentation using the chrominance Cr is conducted, as well as a criterion for detecting an open mouth.

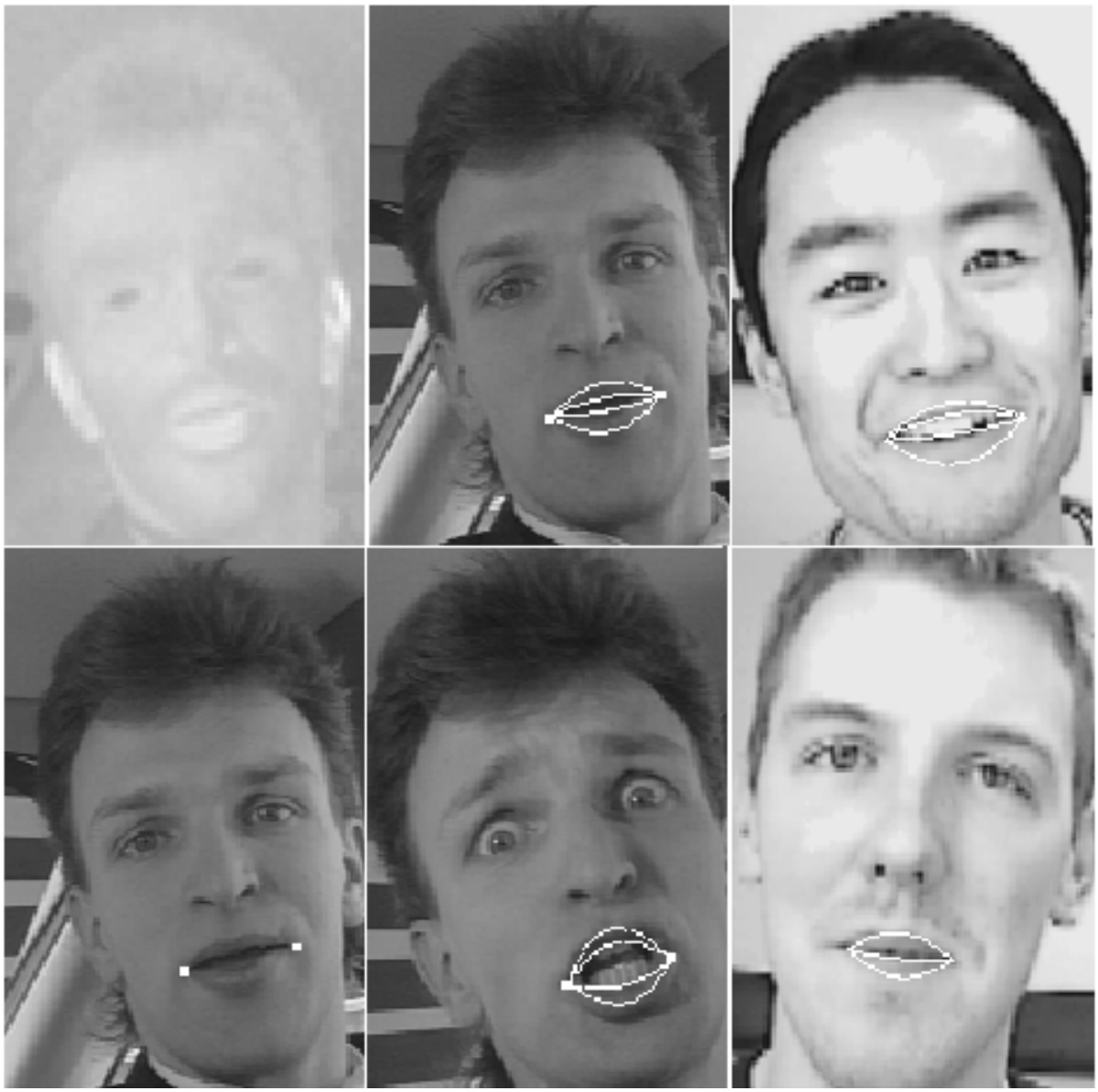


Figure 4.8 Template positioning and extracted mouths

### 4.5.1 Template positioning

Mouth corners are used as the two reference points for placing the mouth templates. This raises an issue of how to segment the mouth, and then locate the corners. For mouth segmentation, the search area can be narrowed into the lower half of the face. Besides, the mouth lip region is rather sensitive to the chrominance component, Cr, and always has local peak value in Cr because the mouth lips appear red (top left image of Figure 4.8). Thus, based on the above knowledge, a heuristic thresholding scheme [Gon02] is adopted:

1. Select an initial estimate for $T$

2. Segment the image using $T$. This will produce two groups of pixels: $G_1$ consisting of all pixels with gray level values $> T$ and $G_2$ consisting of pixels with values $\leq T$

3. Compute the average Cr values $\mu_1$ and $\mu_2$ for the pixels in regions $G_1$and $G_2$

4. Compute a new threshold value: $T = (\mu_1 + \mu_2)/2$

5. Repeat steps 2 through 4 until the difference in $T$ in successive iterations is smaller than a predefined threshold $T_0$.

Initially $T_0$ is set to zero, and the estimate for $T$ is chosen as a median of the search area, since the desired mouth is small compared to the background of the search area. To remove noise, a morphological opening process is applied to the binarised image, followed by a labeling process. The largest candidate is regarded as the mouth, middle points of the leftmost and rightmost of which are considered as the mouth corners. The bottom left image of Figure 4.8 shows the extracted mouth corners with a *carphone* video frame.

### 4.5.2 Criterion for detecting an open mouth

Discrimination of whether the mouth is open or closed generally affects the selection of the mouth templates. The possible appearance of teeth usually interferes with discrimination, so our proposed criterion is based on the chrominance component, Cr, because of its sensitivity to the red-appearing mouth lips, and its insensitivity to the teeth. On the Cr image, if the mouth is open, there is a valley inside (the top left image of Figure 4.8). But when the mouth is closed, the valley disappears. Therefore, according to this characteristic on the Cr image, a pixel $P(x, y)$ with minimum intensity value on the perpendicular bisector of the mouth corners is explored with $x_m < x < x_M$, where $x_m$ and $x_M$ are extrema of vertical coordinates of the segmented mouth. Our criterion is to compare $P(x, y)$ with a predefined threshold $T$. If $P(x, y)$ is less than $T$, the mouth is considered open; otherwise, the mouth is closed.

### 4.5.3 Mouth contour extraction

In contrast with the Conventional Deformable Templates (CDTs), these SDTs can suppress the

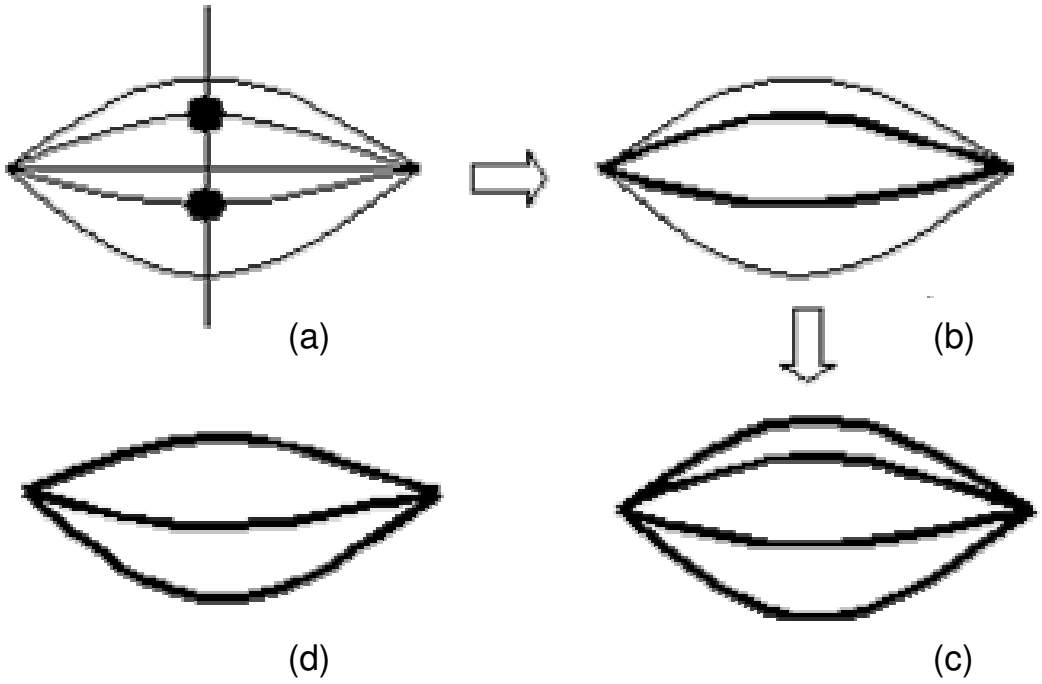



Figure 4.9 Illustration of the staged mouth templates

In an open state:

(a) Stage 1. Middle points extraction of inner lip contours

(b) Stage 2. Inner lip contour extraction

(c) Stage 3. Outer lip contour extraction

In a closed state:

(d) The mouth-closed template

occurrence of mismatch especially if one of the mouth contours fails to be extracted. To make the deformation process more robust, contour extraction is broken down into three stages. Figure 4.9 a, b and c depict these stages of the proposed algorithm in a mouth-open state, analysed as follows.

*(1) Middle point extraction of inner lip contours:* To avoid the effect of teeth when the mouth is open, a method of calculating only gradients of the pixels on the perpendicular bisector of mouth corners is developed, which is unlike the existing methods of taking edge information along the inner lip contours into account. As stated before, when viewing the Cr component of the face, a valley emerges if the mouth is open. Therefore, the calculation will be based on Cr. However, as the mouth is closed, the valley disappears in Cr, whereas another valley arises between the lips in luminance. So the calculation will be based on the luminance. From the top of the mouth to the

pixel right before the pixel $P(x, y)$, the gradients between the current and the next pixels, along the perpendicular bisector of mouth corners are evaluated. The pixel with a maximum gradient value is treated as the middle point of the inner upper lip contour. Similarly, the middle point of the inner lower lip contour can be extracted along the opposite direction.

*(2) Inner lip contour extraction:* With the extracted middle points, each of the inner lip contours can be extracted, and formulated in terms of a parabola.

*(3) Outer lip contour extraction:* Only two elaborately selected strength constraints are required:

$$E_t = E_e + E_r \tag{4.8}$$

where $E_e$ and $E_r$ are the edge strength and the region strength, respectively. To eliminate the impact of teeth, the edge image is derived from the chrominance, Cr, by using gradient-based edge detection with *Sobel* gradient operators. For the same purpose, calculation of the region strength is also based on Cr. The outer lip contour template deforms outwards so that the best-fit outer lip contours are matched by minimising the strength function (4.8).

A mouth-closed template is similarly defined (Figure 4.9 d), with the fact that point $P(x, y)$ is directly regarded as the unique, shared middle point of the overlapped inner lip contours. Some results are shown in the rest of Figure 4.8, which involve some typical expressions as well as the cases that the mouth is open and closed (the bottom right image).

## 4.6 Dynamic Chin Extraction

There are a variety of approaches existing in chin detection. Due to the narrow shadow between the face and neck, the mainstream methods are edge-based, such as active contour model [Hua92] [Rud96], deformable template matching [Kam97b], curve fitting [Got02] and the GVF [Hu03]. All these methods need robust estimation for the initial position of the chin contour, normally implemented by deformable template matching with, typically, the template of a parabola or a combination of few parabolas. However, such a template gives an inaccurate estimate when rotation of the head exceeds a certain range. In this section, an idea of 3D deformable template matching is proposed and implemented, which enables 3D rotation of the chin template by adding 3D data into a conventional 2D deformable chin template. In other words, this mechanism activates the template in a 3D manner, and makes the extraction more precise even if the head rotates. Section 4.6.1 is dedicated to conduct such a paradigm, where 3D chin rotation is

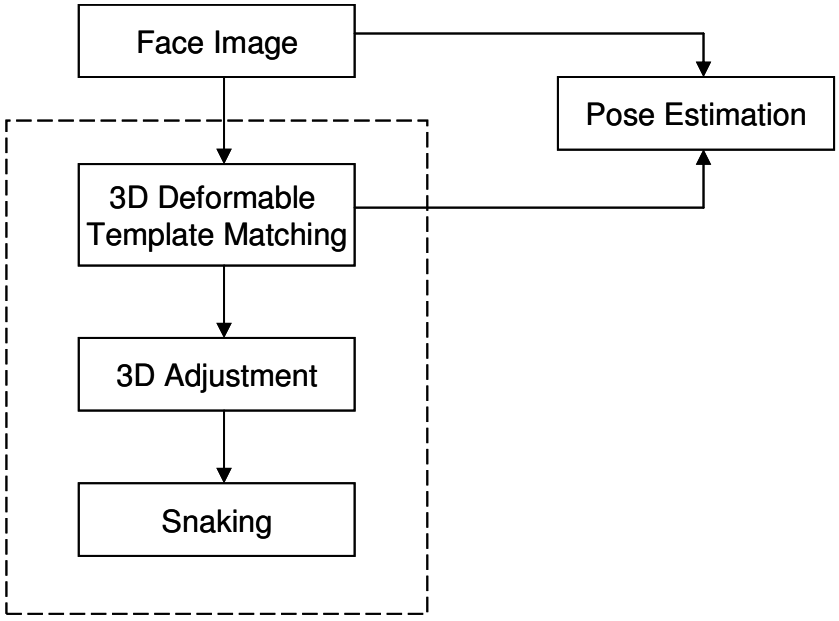



Figure 4.10 Flowchart of dynamic chin extraction

implemented by using the 3D affine transformation parameters, estimated by a single-view based pose estimation algorithm elaborated in Section 5.2.2 of this thesis. Since the operation of pose estimation normally produces estimated data, to ensure that the 3D chin template is more accurately matched, an adjustment process is imposed, which is introduced in section 4.6.2. This 3D deformable template matching algorithm is developed based on the mechanism of the conventional 2D deformable template matching algorithm. As early reviewed in this chapter, the deformable template matching algorithm normally results in a regular extraction of the chin contour according to the configuration of the template. However, as the head rotates, the chin contour in the face image no longer appears regular. So in order to achieve a best match, the active contour model, i.e. Snake is employed because of its flexibility in fitting object boundaries. The nuts and bolts will be introduced in Section 4.6.3. Figure 4.10 illustrates the flowchart of the proposed dynamic chin extraction algorithm, where the three major steps of chin extraction are underlined within a dashed rectangle. Compared with others, the proposed dynamic chin extraction algorithm can achieve more accurate results by taking advantage of the following:

- 3D deformable template matching enables chin template rotation so that the chin template can rotate following head rotation.

- The stage of 3D adjustment ensures the refinement of deformable template matching, and provides an optimized position to place the snake in the next phase.

(a) (b)

(c) (d)

Figure 4.11 Chin detection comparison
(a) Using conventional 2D deformable template (b) Using 3D deformable template
(c) 3D adjustment (d) Using snake

- Snaking ultimately secures accurate extraction of the chin.

### 4.6.1 3D model-based deformable chin template matching

Figure 4.11 (a) shows the chin detection result using the conventional 2D deformable template exploited in [Kam97b], where two white spots highlight the two end points of the parabola, which are on the line formed by the two mouth corners. It can be easily seen that the template cannot

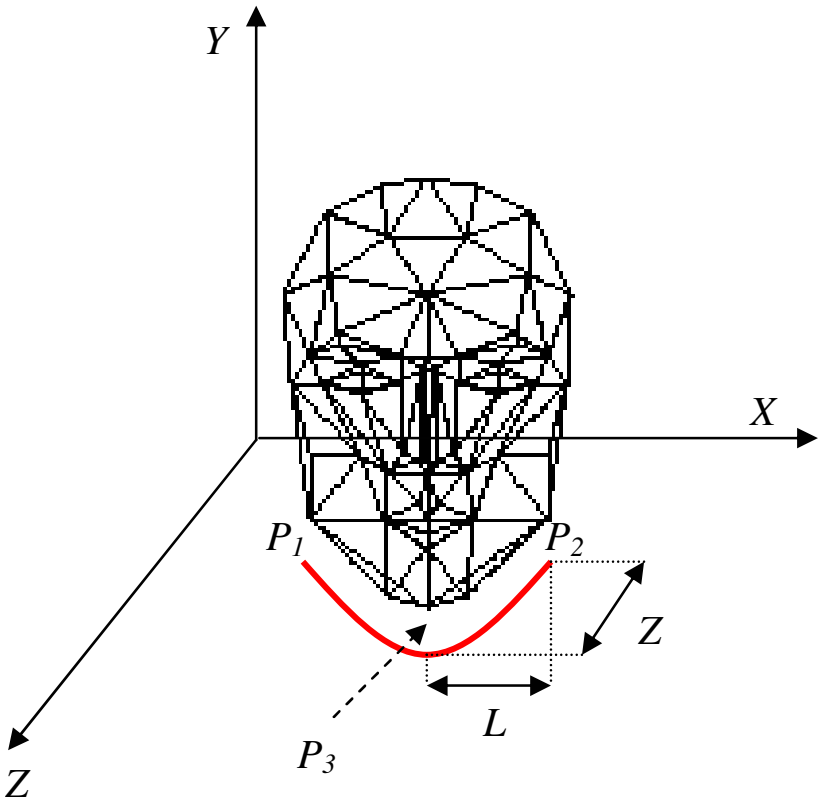



Figure 4.12 3D deformable template of the chin

rotate as the head does. Therefore, a chin template is needed that can not only be vertically deformed, but also 3-dimensinally rotated with the head. Taking inspiration from the concept of 3D face mesh wireframe that can be manipulated in terms of translation, scaling and rotation in the 3D space, if the 3D information of the face is available, then the chin template can also be deformed in the virtual 3D world. This idea forms the basic framework for a 3D deformable template, which is feasible if, and only if, 3D face information can be obtained.

Figure 4.12 illustrates the 3D deformable template of the chin, a red parabola in 3D space, in conjunction with the CANDIDE-3 wireframe, where CANDIDE-3 vertices $P_1$, $P_2$ and $P_3$ represent two corners of the jaw bone and the bottom of the chin, respectively.

The parametric equations of the chin template in 3D space can be formulated as

$$\begin{cases} X = X(s) \\ Y = Y(s) \\ Z = Z(s) \end{cases} \tag{4.9}$$

Thus all 3D coordinates on the chin template can be expressed as functions with respect to the argument $s \in \mathbb{R}$. If $X = s$, then equation (4.9) becomes:

$$\begin{cases} X = X \\ Y = Y(X) \\ Z = Z(X) \end{cases} \tag{4.10}$$

The shape of the template is regulated by

$$Y(X) = H \times \left(1 - \left(\frac{X}{L}\right)^2\right) \tag{4.11}$$

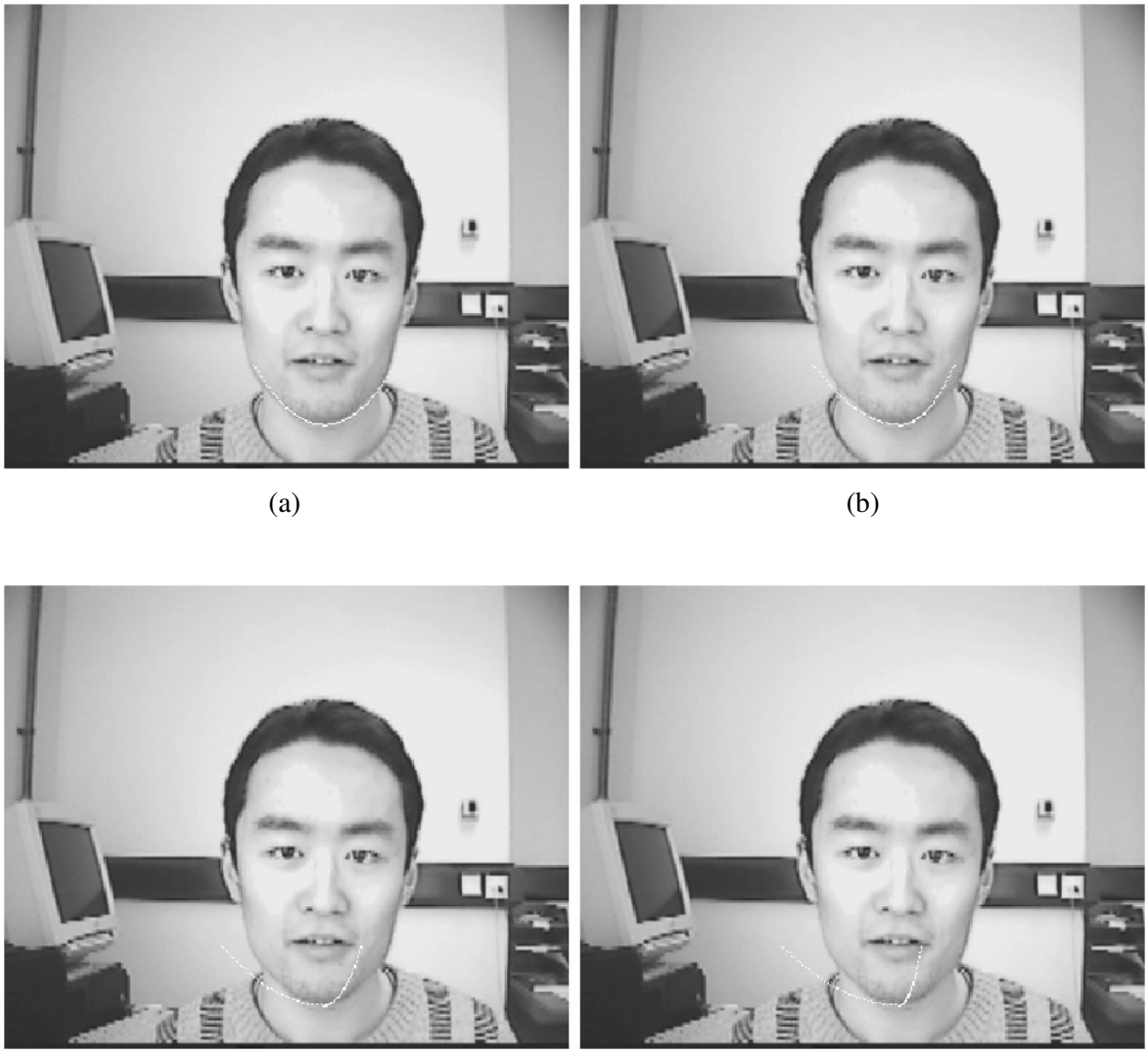
where $H$ and $L$ are the height and half length of the template, more detail of which can be found in Appendix C. Assuming that the depth of the chin template coincides with that of the corresponding points in CANDIDE-3. In other words, the depth of two end points and the tip of the chin template coincide with that of $P_1$, $P_2$ and $P_3$ (See Figure 4.12) in CANDIDE-3. The use of CANDIDE-3 takes advantage of the generality of the face model so that the chin depth can be estimated at the general level. Also, assuming that the depth of the chin template approximately linearly varies from the bottom to two end points (see Figure 4.12). Therefore, the depth of the chin can be determined by linear interpolation between $Z_1$ and $Z_3$, i.e. the depth of $P_1$ and $P_3$:

$$Z(X) = Z_1 + \frac{|Z_3 - Z_1|}{L}\left(L - |X|\right) \tag{4.12}$$

This defines the depth range of the chin template.

Unlike the conventional deformable templates, the 3D deformable template is not only influenced by an intrinsic set of parabola parameters, but also adjusted by 3D affine transformations, such as orientation, scaling and translation of the face. Since the chin contour is part of the face, can be viewed as an independent entity in terms of a parabola, it can be expressed using the 2D orthographic projection model of 3D affine transformation introduced in Appendix A.2:

$$\begin{aligned} I &= T_Y - (\theta_Z \cdot S_X \cdot X + S_Y \cdot Y - \theta_X \cdot S_Z \cdot Z) \\ J &= T_X + (X \cdot S_X - \theta_Z \cdot S_Y \cdot Y + \theta_Y \cdot S_Z \cdot Z) \end{aligned} \tag{4.13}$$



(a) (b)

(c) (d)

Figure 4.13 3D chin template rotation (in radian) with respect to the $Y$ axis.
(a) Neutral position (b) Rotation with 0.3
(c) Rotation with 0.6 (d) Rotation with 0.9

where $(I, J)$ is an image coordinate pair of the projection, while $(T_Y, T_X)$ indicates the mouth centre in the image coordinate system; and $(S_X, S_Y, S_Z)$ and $(\theta_X, \theta_Y, \theta_Z)$ are, respectively, scaling and rotation coefficients. Now the question becomes how to find the parameters $(S_X, S_Y, S_Z)$ and $(\theta_X, \theta_Y, \theta_Z)$, from only a single view image. Both of the scaling coefficients $S_X$ and $S_Y$ are set to 1 because the 3D template in these two directions are in real scale, while methods of evaluating $S_Z, \theta_X, \theta_Y$, and $\theta_Z$ elaborated in section 5.22 are exploited to estimate the four global parameters. Figure 4.13 displays 3D chin template rotation with respect to $Y$ axis. It can be seen that unlike the

conventional parabolic template, the 3D deformable template can interact three-dimensionally with the head in a dynamic manner.

Hence, like in 2D deformable template matching, the 3D template can be deformed to fit the specific chin shape by minimising an energy function. Here only the edge strength is utilised (see equation C.3). Figure 4.11 (b) shows the extracted chin using the proposed 3D deformable template. Compared with the one in Figure 4.11 (a), it is obvious that the barycentre of the chin was yawed with the head.

### 4.6.2 3D adjustment for deformable template matching

As studied in the preceding section, rotation of the 3D deformable chin template is dominated by a group of 3D affine parameters, estimated using a single-view based pose estimation algorithm. Since these parameters obtained by the pose estimation algorithm are ragged, to ensure an accurate match, a 3D adjustment process, i.e. optimization of the extracted chin position, is imposed. Optimization of the extracted chin position is also guided by the same mechanism of 3D deformable template matching that optimizes the chin position by minimising an energy function. The difference is that only one rotation angle is taken into account, i.e. $\theta_Y$, because the major pose estimation error is caused by the estimate of head yaw. The search range for optimization processing is defined as $[\theta_Y - 20, \theta_Y + 20]$ in degree, where $\theta_Y$ is the angle of head yaw obtained from pose estimation. The optimization is carried out by minimising the energy function under variant rotation angles. Figure 4.11 (c) shows the optimized chin contour using the proposed 3D adjustment strategy for deformable template matching. As can be seen, the adjusted chin contour is closer to the real boundary.

### 4.6.3 Snaking for accurate matching

The deformable template matching algorithm generally results in regular chin contour extraction according to the configuration of the template. But when the head rotates, the appearance of the chin contour in the face image no longer remains regular. As shown in Figure 4.11 (c), the aforementioned method still cannot secure precise match between the template and real chin contour. So the active contour model, i.e. snake is employed because of its flexibility in fitting object boundaries. The parametric representation of a snake is $v(s) = [x(s), y(s)]$, $s \in [0, 1]$. The snake is forced to cling on to the object boundary by minimising its total energy function:

$$E_{total} = \int_0^1 E(v(s))\,ds = \int_0^1 \left[E_{in}(v(s)) + E_{ex}(v(s))\right] ds \qquad (4.14)$$

where $E_{in}$ and $E_{ex}$ give rise to internal energy and external energy, respectively, defined as:

$$E_{in}(v(s)) = \frac{1}{2}\left[\omega_1(s)\left|v_s(s)\right|^2 + \omega_2(s)\left|v_{ss}(s)\right|^2\right] \tag{4.15}$$

$$E_{ex}(v(s)) = -\omega_3\left|\nabla\left(G * I\left(v(s)\right)\right)\right| \tag{4.16}$$

The internal energy term $E_{in}$ imposes a piecewise smoothness constraint on the snake, where $v_s$ and $v_{ss}$ indicate the first and second derivatives of snake $v(s)$ with respect to $s$, and govern the tension and rigidity of the snake, respectively; the external energy term $E_{ex}$ drives the snake to the object boundaries, where $I$ denotes a gray-level image, smoothed by a lowpass *Gaussian* filter $G$, while $\nabla$ is the gradient operator. The magnitude of the above energy terms is controlled by weight $\omega_1$, $\omega_2$ and $\omega_3$ so as to easily adjust the energy strength. This avoids that snake crawling is dominated by any single power. In our case, $E_{in}$ and $E_{ex}$ are restricted to [0, 1] and [−1, 0], respectively.

The dynamic programming algorithm proposed in [Ami90] is employed to extract the chin because dynamic programming ensures a globally optimal solution in a numerical manner. Taking advantage of the inherent discrete nature of the problem, the optimization problem is set up as a discrete multistage decision process, and solved by a "time-delayed" discrete dynamic programming algorithm. Considering the discrete form of equation (4.14) and (4.15), we have:

$$E_{total} = \sum_{i=0}^{n-1} E_{in}(v_i) + E_{ex}(v_i) \tag{4.17}$$

$$E_{in}(v_i) = \frac{1}{2}\left[\omega_{1i}\left|v_i - v_{i-1}\right|^2 + \omega_{2i}\left|v_{i+1} - 2v_i + v_{i-1}\right|^2\right] \tag{4.18}$$

Starting from the initial point on the contour, we can treat the global minimization problem as one at each of a finite set of stages ($i_0$, $i_1$, …, $i_{n-1}$). Thus:

$$E_{total}(v_0, v_1, \ldots, v_{n-1}) = E_0(v_0, v_1) + E_1(v_0, v_1, v_2) + E_2(v_1, v_2, v_3) + \ldots + E_{n-2}(v_{n-3}, v_{n-2}, v_{n-1}) \tag{4.19}$$

where

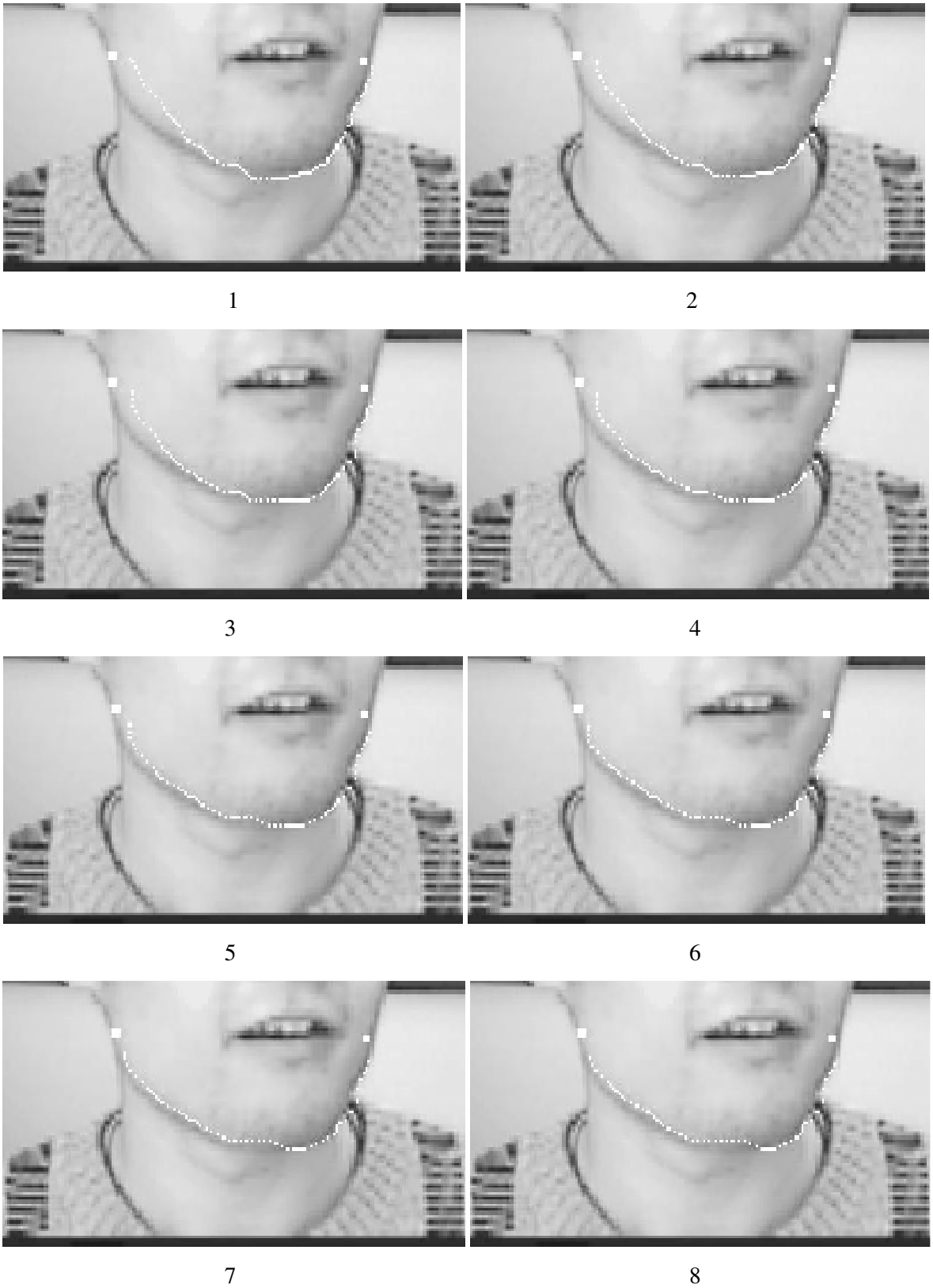



Figure 4.14 Iterations of dynamic programming

$$E_i(v_{i-1}, v_i, v_{i+1}) = E_{ex}(v_i) + E_{in}(v_{i-1}, v_i, v_{i+1}) \tag{4.20}$$

except $E_0(v_0, v_1) = E_{ex}(v_0) + E_{in}(v_0, v_1)$ with only the first derivative embraced in $E_{in}$. Each stage in equation (4.19) can have $m$ possible values. Thus the dynamic programming solution is to find a coherent sequence of optimal energy accumulators $s_i$, $i = 0, 1, \ldots, n\text{-}2$ for each stage. Let

$$\begin{aligned}
s_0(v_0, v_1) &= \min E_0(v_0, v_1) \\
s_1(v_1, v_2) &= \min_{v_0} s_0(v_0, v_1) + E_1(v_0, v_1, v_2) \\
s_2(v_2, v_3) &= \min_{v_1} s_1(v_1, v_2) + E_2(v_1, v_2, v_3) \\
&\vdots \qquad\qquad \vdots \qquad\qquad \vdots \\
s_{n-2}(v_{n-2}, v_{n-1}) &= \min_{v_{n-3}} s_{n-3}(v_{n-3}, v_{n-2}) + E_{n-2}(v_{n-3}, v_{n-2}, v_{n-1})
\end{aligned} \tag{4.21}$$

Combining equation (4.18), (4.20) and (4.21), the general form of the dynamic programming can be obtained

$$s_i(v_i, v_{i+1}) = \min_{v_{i-3}} s_{i-1}(v_{i-1}, v_i) + E_{ex}(v_i) + \frac{1}{2}\left[\omega_1 \left|v_i - v_{i-1}\right|^2 + \omega_2 \left|v_{i+1} - 2v_i + v_{i-1}\right|^2\right] \tag{4.22}$$

An optimal solution is achieved by tracing the minimum energy accumulation through the active contour. The above optimisation process is iteratively performed until the change of the energy accumulation remains stable, that is, the snake has converged to the local minimum.

Figure 4.14 illustrates how the snake converges to the optimal position of the chin (Figure 4.11 (d)) from its initial position shown in Figure 4.11 (c). The numbers under their corresponding images in Figure 4.14 indicate the complete iterations using dynamic programming. As can be seen, the snake crawls towards the chin contour until it clings on to the chin.

## 4.7 Experimental Results

The experimental section consists of three subsections, each of which is dedicated to conduct one test. A comparison of mouth extraction using the CDTs and SDTs is introduced in section 4.7.1, followed by a demonstration of the advance of the proposed dynamic chin extraction algorithm in section 4.7.2. Last, an evaluation of the proposed facial feature extraction algorithms is conducted

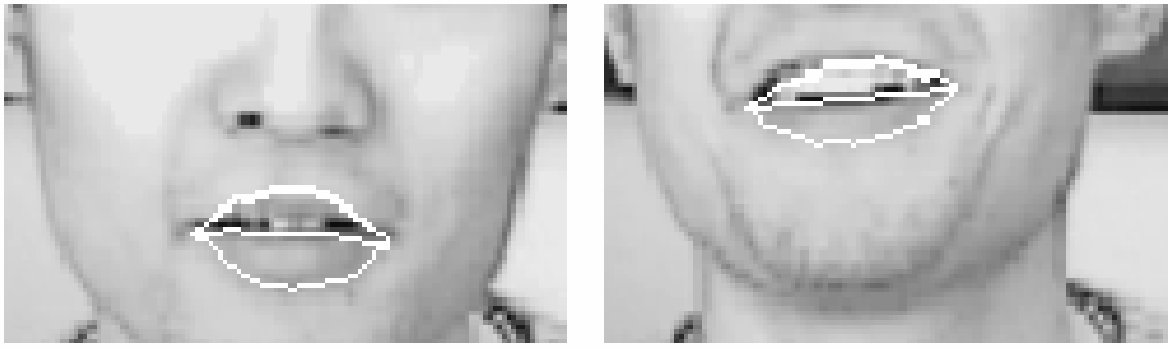


Figure 4.15 Mouth extraction results using CDTs

Figure 4.16 Mouth extraction results using STDs

in section 4.7.3.

### 4.7.1 Comparison of mouth extraction using CDTs and SDTs

The feasibility of the proposed SDTs was demonstrated by testing if the SDTs could provide better mouth extraction results over those using the CDTs. Figure 4.15 shows the subjective mouth extraction results using the CDTs. As can be seen, due to the failure to extract the inner upper mouth lips, two upper mouth lip contours are overlapped. Figure 4.16 shows more robust extraction results by using the SDTs. It can be observed that the SDTs can suppress the occurrence of mismatch, especially, if one of the mouth contours fails to be extracted. Moreover, 45 arbitrary images with both the open and closed mouths were employed in the evaluation of mouth extraction. All these test images used had a successful record in face detection before applied. Table 4.1 illustrates the assessment of mouth extraction using the CDT and SDT. If apply the CDT

| | CDT | SDT |
|---|---|---|
| No. of successful extractions | 33 | 38 |
| Extraction Rate | 73.3 % | 84.4 % |

Table 4.1 Mouth extraction assessment

matching algorithm to these 45 images, only 33 can be successfully extracted, while SDT matching can achieve 38 out of 45. Here the successful extraction is defined as all the lip contours are precisely extracted. On the other hand, if the mouth template fails to be accurately placed onto the mouth, or a wrong template is used, the performance will be considered as failure. As can be seen in Table 4.1, the developed SDT algorithm provides a higher mouth extraction rate than the CDT method.

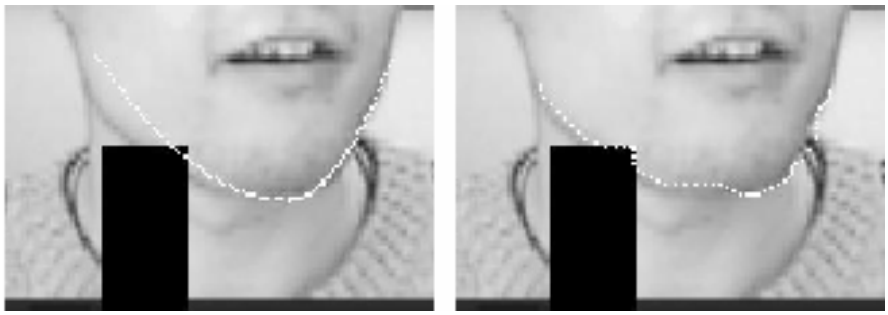
### 4.7.2 Dynamic chin extraction for yawed faces

In this section, the proposed dynamic chin extraction mechanism is tested, typically, using the images with head yaw because the advance of the proposed dynamic algorithm can be easily seen in this case. Figure 4.17 shows the test results of two images with head yaw. The first row shows chin extraction results using the conventional deformable template matching mechanism; while the second row shows the results using the proposed 3D model-based deformable template matching algorithm. It can be observed that the framework of the 2D template is firmly associated with two end points, while the 3D template can easily slip away from them, and rotate along with the head. The third row shows 3D adjustment for deformable template matching. The last row shows the results after applying active contour model based on 3D adjustment. Unlike the conventional deformable template matching, the proposed dynamic mechanism can precisely locate the chin contour, especially in the case when the head yaws.

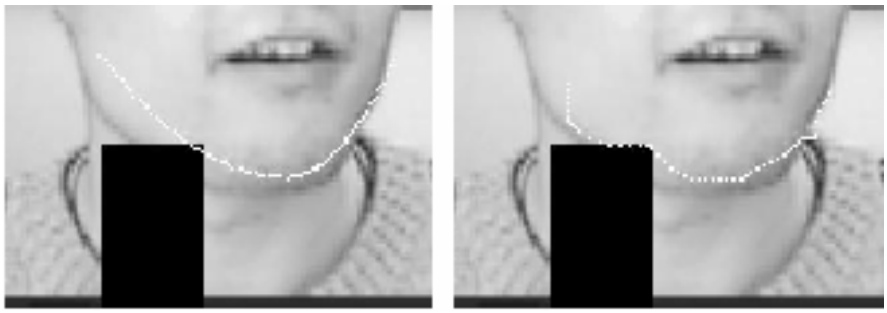
The developed dynamic chin extraction algorithm has also been investigated under the impact of occlusions. Figure 4.18 depicts some test results after conducting an artificial occlusion. The images used in this test come from Figure 4.11 (c) and (d), respectively, the transected views of which are placed in Figure 4.18 (a). As can be seen, the left and right images in Figure 4.18 (a) are chin extraction results after 3D adjustment and snaking, respectively. Then an artificial occlusion was set up and led into the images. The performance of the developed dynamic chin extraction algorithm was assessed as the width of the occlusion was gradually increasing. Figure 4.18 (b), (c) and (d) shows the chin extraction results with occlusion widths of 25, 30 and 35 pixels,

Figure 4.17 Dynamic chin extraction with head yaw

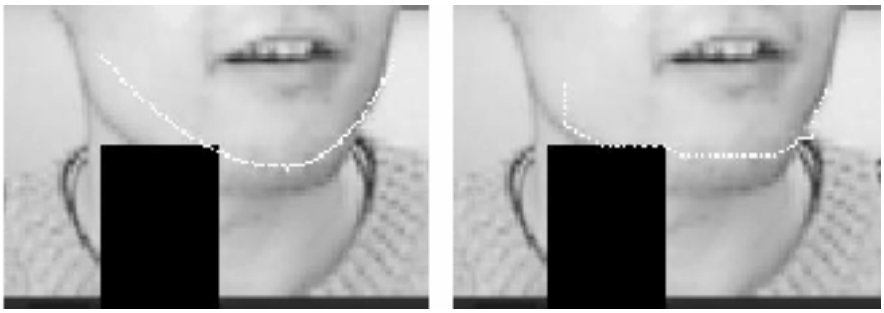
(a)



(b)



(c)



(d)

Figure 4.18 Impact of occlusion to the dynamic chin extraction algorithm
(a) Without occlusion (b) 25-pixel wide occlusion
(c) 30-pixel wide occlusion (d) 35-pixel wide occlusion

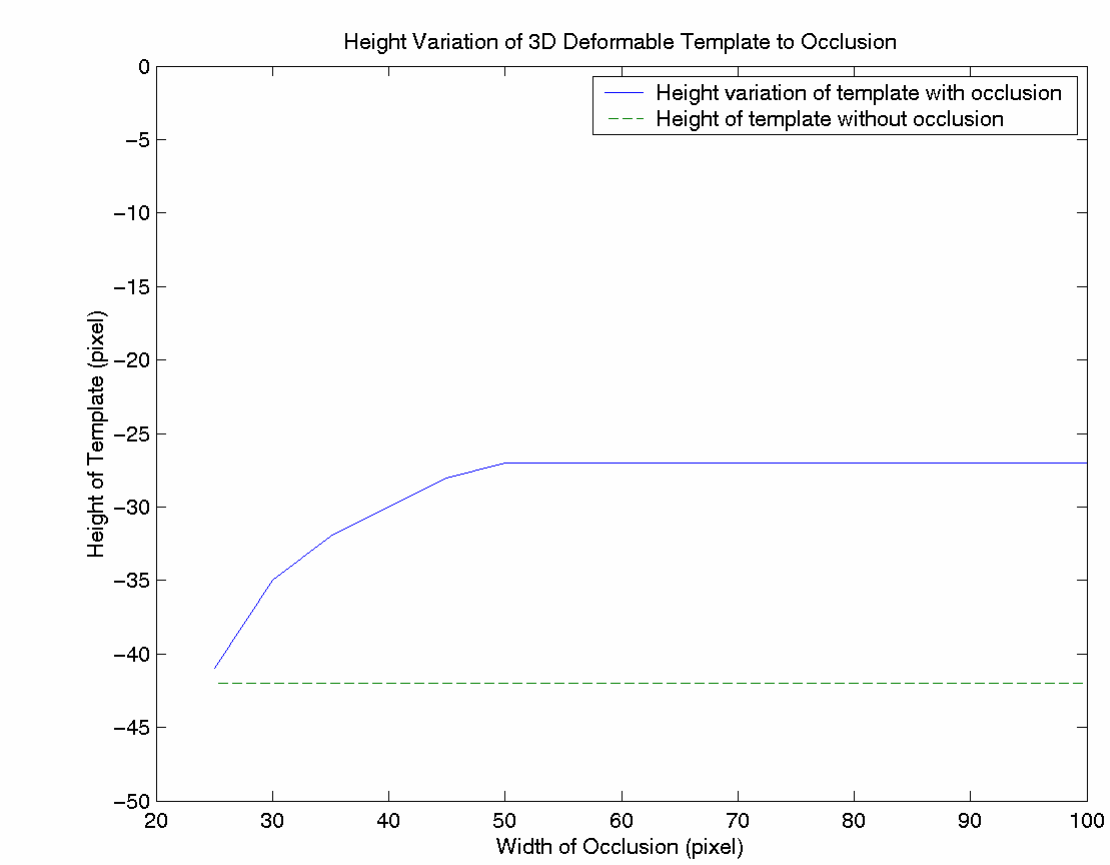



Figure 4.19 Height variation of 3D deformable template to occlusion

respectively. It can easily be seen that when occlusion width is 25 pixels, i.e. Figure 4.18 (b), the developed 3D deformable template matching algorithm is less interfered than snake by the occlusion. This proves that deformable template matching can guarantee a more regular shape compared with snake when occlusion occurs, while snake is easier to cling on to edges. As the occlusion becomes wider and wider, interference becomes more and more evident, as can be observed from Figure 4.18 (c) and (d). Figure 4.19 shows height variation of the 3D deformable template with reference to increasing the occlusion width. The definition of the height of a 3D deformable template can be fined in Appendix B. As can be seen, the height is negative due to the opposite direction of the chin template to the definition. Moreover, the absolute height of the template is decreasing as the occlusion becomes broad, until reaching a certain value. This value implies the place where the coverage of the occlusion involves the major part of the chin, so that the dynamic chin extraction algorithm fails to take effect.

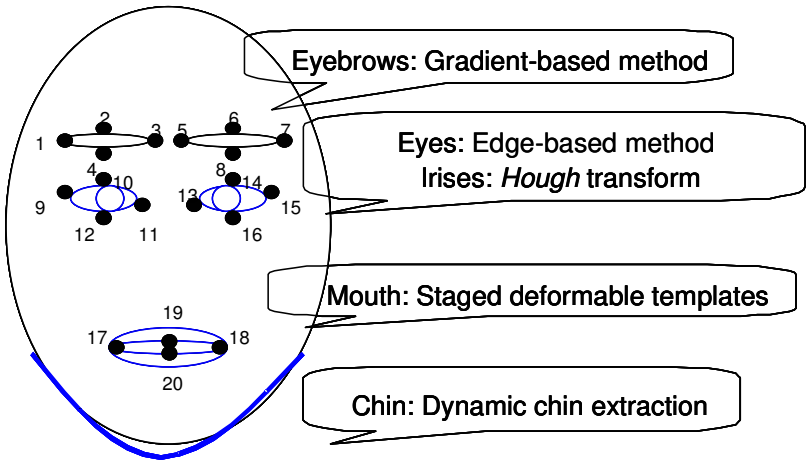



Figure 4.20 Illustration of extracted features and corresponding methods

### 4.7.3 Evaluation of facial feature extraction

A large number of head-and-shoulder images were used in performing the facial feature extraction test. All the test images used had a successful record in face detection before they were employed in the evaluation of facial feature extraction. Figure 4.20 specifies the extraction approaches used, and highlights the corresponding facial features, i.e. the eyes, irises, mouth and chin. However, the irises can only be extracted in the eyes-open state. Figure 4.21 shows some samples from the test. It can be seen that the proposed algorithms can robustly extract the facial features.

Moreover, twenty feature points labeled in Figure 4.20 are exploited to test the accuracy of the proposed facial feature extraction algorithm. The experiment was carried out with three head-and-shoulder video sequences in CIF resolution, as listed in Table 4.2. Seven arbitrary frames were picked up from each video sequence. The average position errors between the manually determined and automatically detected feature points were measured and tabulated in Table 4.2, where the subscripts of $E$ indicate the labeled numbers in Figure 4.20. It can be seen that, the average errors, tabulated in the last column of the table, are acceptably low. Figure 4.22 shows extraction results of these 20 facial feature points by tracking the video sequence *yun*, where the white close loops highlight the detected faces. As can be noted, the feature points can be accurately extracted as well as the face, except the case when self-occlusion occurs, as shown in the bottom right image of the figure.

## 4.8 Summary

Facial feature extraction in 3D face synthesis provides the 3D face synthesiser with a number of key references, i.e. facial features, from which the facial attributes, such as the shape, size and orientation information can be derived. Nevertheless, similar to face detection, automatic facial feature extraction still remains a challenging topic, due to unpredictable factors like lighting conditions, partial occlusions, and the variability of facial expressions. Moreover, too many facial features would increase the computational complexity, while too few would result in the inaccurate aftermath.

In order to precisely extract facial features, various approaches aimed at various facial features have been proposed in a diversity of modalities. The mainstream approaches can be categorised into brightness-based and edge-based algorithms, which have been reviewed in the beginning of this chapter. Brightness-base algorithms exploit the brightness characteristics of the images to search for possible position of the facial features playing an initial role in facial feature extraction, whereas edge-based algorithms target contours of the features, such as those of the mouth, eyes and chin, which are effective complements to brightness-base algorithms. The principle of three widely used edge-based algorithms has been studied.

This chapter has also elaborated some extraction algorithms aimed at different facial features, such as eyes extraction based on edge information, gradient-based eyebrows extraction, staged deformable template matching for mouth extraction, and dynamic chin extraction. All these methods are based on the above mentioned two-step philosophy, i.e. feature location followed by feature extraction. For eye extraction, since the edge-based deformable template matching algorithm is no longer used when image resolution is not high enough, a new method has been implemented, based on the texture complexity of the eyes. Aimed at resolving the existing problem, a staged deformable template matching algorithm has also been developed, consisting of the steps of template positioning, criterion for detecting an open mouth and mouth contour extraction. Moreover, in order to avoid the barriers in conventional deformable template matching for chin extraction, the framework of dynamic chin extraction has been developed, including the stages of 3D model-based deformable template matching, 3D adjustment and snaking for accurate matching. 3D model-based deformable template matching takes a space parabola into account so that the chin template can appropriately rotate as the head yaws or rolls. Then, the accuracy can be guaranteed by both 3D adjustment and snaking. This dynamic mechanism works well especially with head yaw, which conventional 2D deformable template matching always has difficulty to

face up to.

Finally, the feasibility of all the proposed facial feature extraction algorithms has been demonstrated in the experimental section.

| Sequence | Average Position errors between the manually determined and the detected feature points (pel) | | | | | | | | | | | | | | | | | | | Average Error (pel) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $E_1$ | $E_2$ | $E_3$ | $E_4$ | $E_5$ | $E_6$ | $E_7$ | $E_8$ | $E_9$ | $E_{10}$ | $E_{11}$ | $E_{12}$ | $E_{13}$ | $E_{14}$ | $E_{15}$ | $E_{16}$ | $E_{17}$ | $E_{18}$ | $E_{19}$ | $E_{20}$ | |
| *Stewart* | 1.7 | 0.9 | 1.3 | 0.8 | 2.0 | 1.5 | 2.7 | 1.0 | 1.0 | 0.6 | 0.3 | 0.7 | 1.2 | 0.6 | 0.9 | 0.8 | 1.2 | 1.6 | 1.3 | 1.0 | 1.2 |
| *Yun* | 0.8 | 0.3 | 0.2 | 1.0 | 0.2 | 0.2 | 0.3 | 0.4 | 1.1 | 0.9 | 1.0 | 0.5 | 1.2 | 0.9 | 1.1 | 0.6 | 2.0 | 1.9 | 1.2 | 1.7 | 0.8 |
| *Carphone* | 3.4 | 1.0 | 3.6 | 0.9 | 1.3 | 0.4 | 4.1 | 1.5 | 0.8 | 0.6 | 0.7 | 0.4 | 2.7 | 0.7 | 1.3 | 1.4 | 2.4 | 1.5 | 1.2 | 2.5 | 1.6 |

Table 4.2 Position errors of extracted feature points

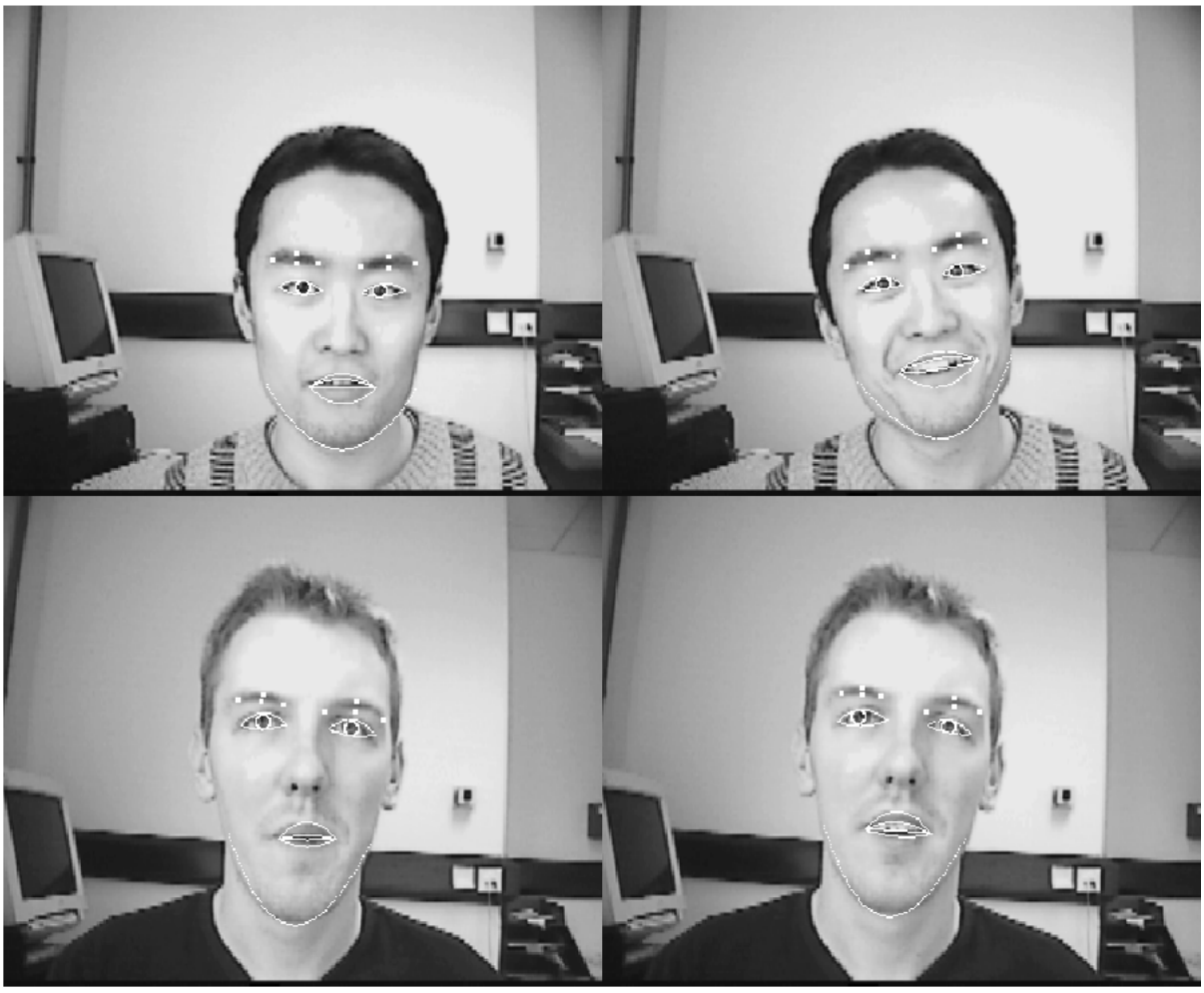


Figure 4.21 Facial feature extraction results

Figure 4.22 Facial feature tracking