國立交通大學

資訊學院資訊科技（IT）產業研發碩士班

碩 士 論 文

使用視訊模型從單一影像自動產生有動態嘴形動作的虛擬人臉之研究

A Study on Automatic Creation of Virtual Faces with Dynamic Mouth Movements from Single Images Using Video Models

研 究 生：黃巧均

指導教授：蔡文祥　教授

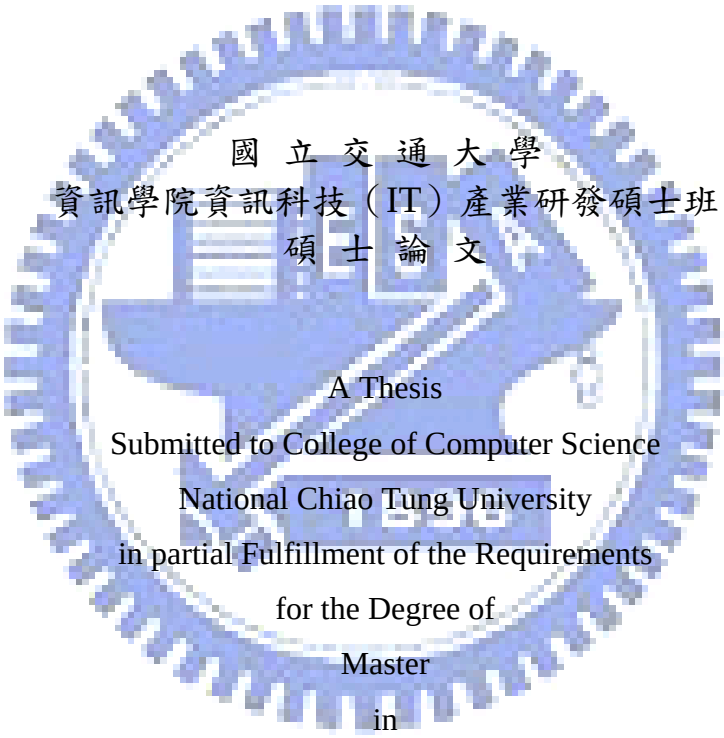
中 華 民 國 九 十 八 年 六 月

使用視訊模型從單一影像自動產生有動態嘴形動作的虛擬人臉之研究

A Study on Automatic Creation of Virtual Faces with Dynamic Mouth Movements from Single Images Using Video Models

研 究 生：黃巧均　　Student：Chiao-Chun Huang

指導教授：蔡文祥　　Advisor：Wen-Hsiang Tsai



國 立 交 通 大 學

資訊學院資訊科技（IT）產業研發碩士班

碩 士 論 文

A Thesis

Submitted to College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Industrial Technology R & D Master Program on

Computer Science and Engineering

June 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年六月

使用視訊模型從單一影像自動產生
有動態嘴形動作的虛擬人臉之研究

研究生: 黃巧均　　指導教授: 蔡文祥 博士

國立交通大學資訊學院產業研發碩士班

摘要

本論文提出一個由單一影像自動產生有動態嘴型動作的虛擬人臉之系統。此系統包含了三個流程：視訊模型分析、臉部特徵點追蹤、虛擬人臉產生。由本系統產生的動態虛擬人臉系列與所輸入單一影像中之人臉皆相同。為了產生虛擬人臉，我們提出一個含二十六個特徵點的嘴形模型。首先我們分離事先錄製的視訊模型之語音成分，再將視訊模型分解成多張連續影像。之後，再半自動地取得所輸入單一影像及視訊模型的第一張影像中的臉部特徵點。接著，我們提出了兩種嘴部狀態和三種閉嘴嘴形，以及一個影像對應技術，並用以分析視訊模型中的嘴型動作及追蹤其臉部特徵點。此技術使用相關係數求得各特徵點之最佳對應位置，並依不同嘴部狀態而動態改變視窗大小。為了取得正確的臉部特徵點，每當偵測到閉嘴嘴形時，我們即校正特徵點至正確位置。接著我們利用一個形變(morphing)技術讓所輸入單一影像及視訊模型中之嘴形動作同步化，使得虛擬人臉看起來像與視訊模型中的人一樣說出相同的話。而我們所指定的臉部控制點亦能調整虛擬人臉之嘴部大小及下巴位置，使得人臉在講話過程中看起來更加地自然。良好的實驗結果證明了本論文所提方法之可行性。

A Study on Automatic Creation of Virtual Faces with Dynamic Mouth Movements from Single Images Using Video Models

Student: Chiao-Chun Huang **Advisor: Dr. Wen-Hsiang Tsai**

Industrial Technology R & D Master Program of CS Colleges
National Chiao Tung University

ABSTRACT

In this study, a system for automatic creation of virtual talking faces with dynamic mouth movement using a single image of a human face and a video model of a real talking face is proposed, which includes three processes: video model analysis, feature point tracking, and virtual face creation. The dynamic virtual face series created by the system is the same as the input image. First, a mouth model of 26 feature points is proposed for virtual face creation. Two mouth states and three closed-mouth shapes are proposed for video analysis to obtain mouth movements in the real-face video model. For feature point tracking, an image matching technique using correlation coefficients with dynamically changed window sizes is proposed. The window sizes are changed according to the mouth states. A technique for correction of the feature point locations of a closed mouth is proposed. A mouth shape morphing technique is used for synchronizing the mouth shapes of the input image with the video model, yielding the effect that the created virtual faces look like speaking the same words as the person in the video model. A concept of assigning facial control points is applied to create the virtual faces with scaled mouth sizes. Good experimental results show the feasibility and applicability of the proposed method.

ACKNOWLEDGEMENTS

The author is in hearty appreciation of the continuous guidance, discussions, support, and encouragement received from his advisor, Dr. Wen-Hsiang Tsai, not only in the development of this thesis, but also in every aspect of her personal growth.

Thanks are due to Mr. Tsung-Yuan Liu, Mr. Chih-Jen Wu, Mr. Che-Wei Lee, Mr. Guo-Feng Yang, Mr. Chun-Pei Chang, Miss Shu-Hung Hung, Miss Chin-Ting Yang, Mr. Jian-Yuan Wang, Miss Mei-Fen Chen, and Mr. Yi-Chen Lai for their valuable discussions, suggestions, and encouragement. Appreciation is also given to the colleagues of the Computer Vision Laboratory in the Institute of Computer Science and Engineering at National Chiao Tung University for their suggestions and help during her thesis study.

Finally, the author also extends her profound thanks to her family for their lasting love, care, and encouragement. She dedicates this dissertation to her parents.

CONTENTS

LIST OF FIGURES

Chapter 1

Introduction

1.1 Motivation

In recent years, people are used to communicate and share multimedia files through the computer network. With the development of the high-speed Internet, more and more people upload video clips and share them through blogs, emails, and websites such as YouTube. Also, people can now watch high-quality videos online.

The contents of videos are of wide variety, including videos for teaching, life recording, security monitoring, etc. Some people just want to share their experiences and wish not to show up in the video, so that they may try to record voices only or use cartoon-like faces instead of showing their own faces in the transmitted video. However, human faces and speeches created artificially in such kinds of videos are still unnatural.

It is usually desired to create more human-like faces which make virtual-face related videos friendlier. This topic is called *virtual talking face creation*, and many researches on this topic concentrate on how to create more realistic faces. A virtual talking face can be used to reflect facial expressions, including emotional looks and mouth movements.

Before virtual face creation, it needs to extract facial feature points from the

image frames in a given video so that we can control the feature points to generate different kinds of virtual expressions. Real-time systems detect the feature points in the first frame of videos, and track them in the other frames. In this way, we can have the feature points of each frame and can create sequential virtual faces with motions. Also, traditional systems create virtual faces by the use of matching input voices (or texts) and visemes with reference data in models. Such voice and text analyses usually are sensitive to noise in the recording environment.

In this study, we want to design an automatic system for creating virtual faces with dynamical mouth movements. And we will not deal with voice and text analyses but only use facial image information. The input to this system is a facial image, and the output is an image sequence representing a human head talking process.

1.2 Survey of Related Studies

In this section, several virtual face creation techniques are reviewed in Section 1.2.1. And an image matching technique based on the correlation coefficient measure is reviewed in Section1.2.2. And several morphing techniques are reviewed in Section 1.2.3 finally.

1.2.1 Review of Related Studies

Many studies about virtual face creation have been conducted. Generally speaking, there are two main approaches to it, but both of them should gather the feature points of facial images before creating virtual faces with different kinds of expressions.

The first approach needs to define some viseme types and phoneme combinations, which have mapping relations. And it needs to analyze voices or texts for mapping the analysis result to one of the phoneme combinations. Because each of

them has a relation to one viseme, we can get a corresponding viseme type. Then the shape of facial images is warped to the shape of the corresponding viseme, and a cartoon-like face with the viseme shape is generated by a computer graphic technique [1-10].

Chen and Tsai [1] designed a system to generate cartoon faces automatically by the use of facial feature point detection, speech analysis, and curve drawing techniques. Talking cartoon faces are generated from image sequences. The Video Rewrite designed by Bregler, Covell, and Slaney [6] is a system proposed to rewrite videos with audios. It automatically labels the phonemes in training data and in new soundtracks. Its video models are defined by mapping phonemes in the soundtrack to the training data, which include chin, mouth, and jaw. To rewrite the video, it combines the video model with the original video. Cosatto [7] presented a system that produces photo-realistic computer animations of a person talking in general text. In Lin *et al*. [8], a lifelike talking head system was proposed. The talking head is driven by speaker independent speech recognition. In Nedel [9], the use of a speech recognition technique to segment the lip features extracted from a video on a phoneme by phoneme basis was proposed. MikeTalk, presented in Ezzat and Poggio [10], is a text-to-audiovisual speech synthesizer which converts input text into an audiovisual speech stream. It morphs every corresponding viseme to acquire a smoothing transition result.

The second approach has two differences from the first approach. One is not to implement the voice and text analyses. The other is the use of expression mapping instead of phoneme mapping. The basic process is to map a facial image to an expression, warp or morph it to the corresponding expression, and draw a cartoon-like face [11-12].

A novel method for generating performance-driven, “hand-drawn” animation in

real time is presented in Buck *et al*. [11]. Given an annotated set of hand-drawn faces for various expressions, their algorithm performs multi-way morphing to generate real-time animation that mimics the expressions of a user.

Zhang *et al*. [12] provided a way for automatic synthesis of the corresponding expression image which has photorealistic and natural looking expression details.

1.2.2 Review of Image Matching Technique by the Use of Correlation Coefficients

Gonzalez and Woods [13] introduced an image matching technique via the use of the correlation coefficient. It finds a subimage $w(x,y)$ within an image $f(x,y)$ where the size of *f* is bigger than *w*. The correlation coefficient is defined as

$$\gamma(x,y)=\frac{\sum_s\sum_t\left[w(s,t)-\bar{w}\right]\sum_s\sum_t\left[f(x+s,y+t)-\bar{f}(x+s,y+t)\right]}{\left\{\sum_s\sum_t\left[w(s,t)-\bar{w}\right]^2\sum_s\sum_t\left[f(x+s,y+t)-\bar{f}(x+s,y+t)\right]^2\right\}^{\frac{1}{2}}}, \qquad (1.1)$$

where $\bar{w}$ is the average value of the pixels in w, $\bar{f}$ is the average value of f in the region coincident with the current location of w.

As *x* and *y* vary, *w* moves around inside the area of *f*. The best match position has the maximum value of γ.

1.2.3 Review of Morphing Techniques

Beier and Neely [15] proposed two transformation techniques for morphing: *single line pair* and *multiple line pairs*. They defined a coordinate mapping from a destination image pixel *X* to a source image pixel *X′* with respect to a line *PQ* in the destination image and a line *P′Q′* in the source image, respectively. As shown in Figures 1.1 and 1.2, let the value *u* be the position along the line *PQ*, and *v* be the

distance from PQ to the image pixel X. Figures 1.3 and 1.4 show some examples of these transformations which are described as follows.

(1) *Transformation of single line pair* ---

For each pixel X in the destination image, perform the following steps.

(i) Find the corresponding values of u and v according to the following equations:

$$u = \frac{(X - P)\cdot(Q - P)}{\|Q - P\|^2}; \tag{1.2}$$

$$v = \frac{(X - P)\cdot Perpendicular(Q - P)}{\|Q - P\|}, \tag{1.3}$$

where *Perpendicular*() returns the vector perpendicular to, and of the same length as, the input vector.

(ii) Find X' in the source image for these values of u and v according to the following equation:

$$X' = P' + u\cdot(Q' - P') + \frac{v\cdot Perpendicular(Q' - P')}{\|Q' - P'\|}. \tag{1.4}$$

(iii) Set the mapping *Destination Image*(X) = *Source Image*(X').

(2) *Transformation of multiple line pairs* ---

For each pixel X in the destination image, perform the following steps.

(i) For each line P_iQ_i, perform Steps (ii) to (vi);

(ii) Find the corresponding values of u and v according to Equations (1.2) and (1.3).

(iii) Find X' in the source image based on these values of u and v, and the line $P_i'Q_i'$ according to Equation (1.4).

(iv) Calculate the displacement $D_i = X - X'$.

(v) Find the shortest distance *dist* from X_i to each line.

(vi) Compute the following weight where a, b, and p are user-defined constants :

$$weight = \left[\frac{length^p}{(a + dist)} \right]^b . \tag{1.5}$$

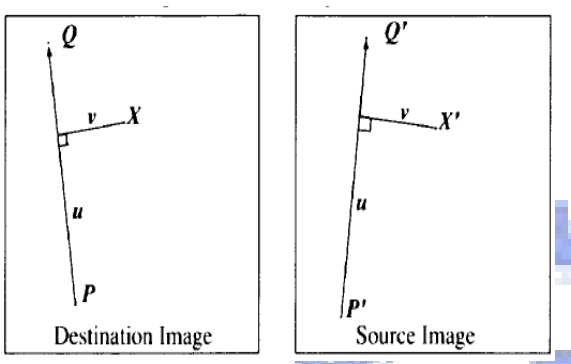


Figure 1.1 Single line-pair transformation.

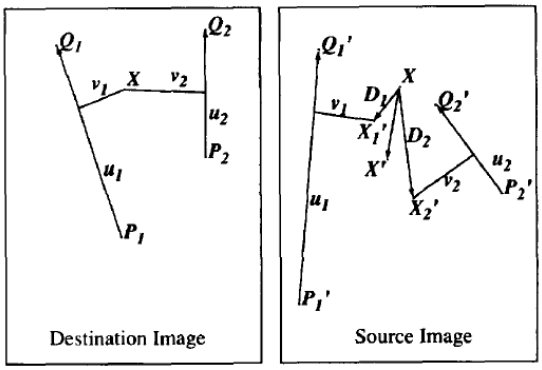
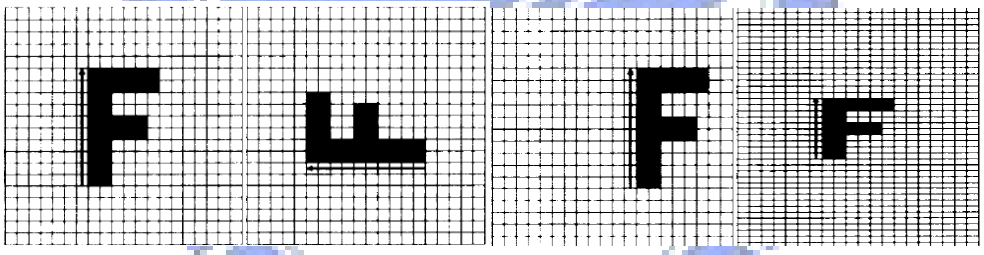
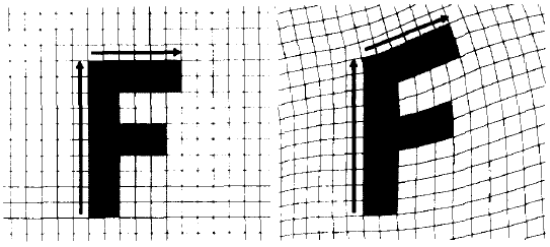


Figure 1.2 Multiple line-pair transformation.



(a) (b) (c) (d)

Figure 1.3 Single line-pair transformation. The original image is in (a), and the line is rotated in (b), translated in(c) and scaled in (d).



(a) (b)

Figure 1.4 Multiple line-pair transformation. (a) The original image. (b) An example of using two line pairs.

Gomes, *et al.* [14] proposed a *bilinear transformation* to warp unit squares into quadrilaterals, as shown in Figure 1.5, and an *inverse bilinear transformation* to warp quadrilaterals to unit squares. We will describe the details in Chapter 4.

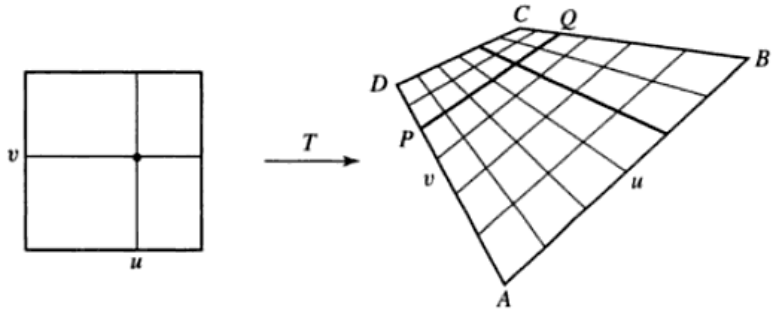


Figure 1.5 Bilinear transformation scheme.

1.3 Overview of Proposed Method

An overview of the proposed method is described in this section. First, some definitions of terms used in this study are introduced in Section 1.3.1. And several assumptions made for this study are listed in Section 1.3.2. Finally a brief description of the proposed method is given in Section 1.3.3.

1.3.1 Definitions of Terms

The definitions of some terms used in this study are as follows.

(1) Neutral Face: MPEG-4 specifies some conditions for a head in its neutral state [16] as follows.

1. The gaze is in the direction of the Z-axis.
2. All face muscles are relaxed.
3. The eyelids are tangent to the iris.

4. The pupil is one third of the iris diameter.
5. The lips are in contact.
6. The line of the lips is horizontal and at the same height of the lip corners.
7. The mouth is closed and the upper teeth touch the lower ones.
8. The tongue is flat and horizontal with the tip of the tongue touching the boundary between the upper and lower teeth.

In this thesis, a face with a normal expression is called a *neutral face*.

(2) Neutral Facial Image: A neutral facial image is an image with a frontal and straight neutral face in it.

(3) Facial Features: In the proposed system, we care about several features of the face, including hair, face, eyebrows, eyes, nose, mouth, and ears of each facial image.

(4) Facial Action Units (FAUs): Facial Action Coding System (FACS) [18] defines 66 basic Facial Action Units (FAUs). The major part of FAUs represents primary movements of facial muscles in action such as raising eyebrows, blinking, and talking. Other FAUs represent head and eye movements.

(5) Facial Expression: A facial expression is a facial aspect representative of feeling. Here, facial expressions include emotions and lip movements. Facial expressions can be described as combinations of FAUs.

(6) FAPUs: Facial animation parameter units (FAPUs) are the fractions of the distances between some facial features, like eye separation, mouth width, and so on.

(7) Real-Face Video Model: Each real-face video model has a person in it, either male or female, whose talking progress is recorded by a camera. The models are used to create final image sequences.

(8) Real-Face Model Control Points: These points are some of the 74 feature

points of the real-face model. They are used to control many features of the models, like eyebrow raising, eye opening, mouth movement, head tilting, and head turning. In this study, they used to control mouth movements.

(9) Mouth Region: a mouth region is a part of faces, which nears the mouth. Some variable facial features may be pasted onto it to form final image sequences.

1.3.2 Assumptions

In the proposed system, real-face video models are captured by a camera. In a real situation, it is not a simple task to track real faces which have a smooth contour. We must make some assumptions and restrictions in this study to reduce the complexity, which are described as follows.

(1) The lighting of the environment is constant.

(2) The face of a video model always faces the camera and is located in the middle of the field of view of the camera.

(3) The head in a video model does not move quickly.

(4) The video models and facial images have good resolutions (higher than 640×480).

(5) The percentage of a face area in a facial image is over 70%.

(6) The mouth in a video model has a sharp contour.

(7) The speech is spoken with a medium speed.

1.3.3 Brief Descriptions of Proposed Method

In this study, the proposed system includes six main processes: video recording, feature point locating, feature point tracking, mouth shape morphing, mouth region extraction, and virtual face creation.

The first process is video recording, from which we can get a real-face video model. Secondly, we locate manually the feature points in the first frame of a given video and in an input facial image. The reason why we locate them manually is that it is not easy to detect automatically the feature points of a mouth with a smooth curve edge. Then, we track the feature points from one frame to the next. After the tracking process, we have all feature points of each frame of the video model, so we can morph the input facial image to every frame of the video model and obtain an image sequence.

The process of mouth region extraction mainly takes the bottom face part out from the image sequence below the nose and removes the skin of the neck part. At the last step, the mouth regions and the input facial image are integrated to create the result.

1.4 Contributions

Some major contributions of this study are listed as follows.

(1) A system for automatic creation of virtual faces with mouth movements is proposed.

(2) A system for creating virtual faces without voice and text analyses is proposed.

(3) A technique using a facial image to fit a sequence of other facial images is proposed.

(4) Some techniques for tracking feature points are proposed.

(5) A technique for correcting feature points is proposed.

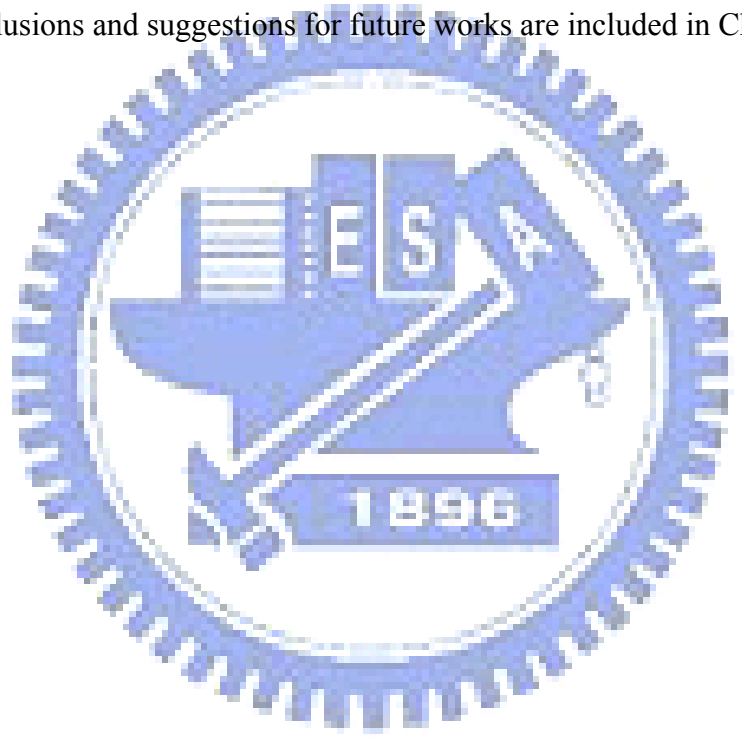
(6) A technique to create virtual teeth and tongues for closed mouth facial images is proposed.

(7) A technique to generate dynamic mouth movements is proposed.

(8) Several new applications are proposed of the proposed system and implemented.

1.5 Thesis Organization

The remainder of the thesis is organized as follows. Chapter 2 describes an overview of the proposed technique for virtual face creation. Chapter 3 presents the proposed technique for tracking facial feature points automatically. Chapter 4 describes the proposed technique of creation of virtual faces with dynamic mouth movements. In Chapter 5, some experimental results and discussions are described. Finally, conclusions and suggestions for future works are included in Chapter 6.



Chapter 2
Overview of Proposed Method for Virtual Face Creation

2.1 Idea of Proposed Method

The virtual face creation system proposed in this study is like a black box. The input to it is a single facial image and the output is a facial image sequence. In other words, it is like to make a human face in a single image to laugh or talk in an artificially-created image sequence or video. We use real-face video models to achieve this goal, so the input image will do the same mouth movements as the models. The system is described in more detail in the following.

First, we propose a technique to analyze video models to get the mouth movement information. Because some mouth movements have quite different mouth shapes from others, such as those of “u” and “o,” it is not easy to conduct image matching for such mouth movements. Besides, image matching has another problem which occurs when a closed mouth is opening or when an opened mouth is closing, that is, the teeth will appear or disappear alternatively to interfere with the correctness of image matching. So we propose a novel image matching technique to deal with such a problem of interference coming from changed mouth-shape and teeth appearances.

After creating a virtual mouth by a morphing technique, it may be bigger or smaller than the mouth of the input image which has a closed mouth. For example, the

virtual mouth will be bigger than the mouth of the input image when a person in the video model says the letter “a” for which the mouth is opening and the chin is moving down. If we just paste the mouths on the input image, the resulting image will have clear edges at the pasted mouth boundary. So we propose a technique to extract the mouth region and smooth its edges before integrating the mouth with the input image.

In this chapter, the techniques proposed to achieve the goals mentioned above are described. First, a review of Chen and Tsai [1] constructing a face model adapted from [16] is given in Section 2.2. Construction of a mouth model and uses of mouth features based on the adapted face model are described in Section 2.3. Finally, a technique is proposed to create virtual faces from sequential images, which is given in Section 2.4. More detailed descriptions of the involved steps of the techniques will be described in Chapters 3 and 4.

2.2 Review of Adopted Face Model

Chen and Tsai [1] proposed a method to generate cartoon faces automatically from neutral facial images. Before cartoon face generation, a face model with facial feature points was defined first. Ostermann [16] specified the 84 feature points and the facial animation parameter units (FAPUs) of the face model used in the MPEG-4 standard, as shown in Figures 2.1 and 2.2. However, this face model is not suitable for cartoon face drawing. Chen and Tsai [1] defined accordingly an adapted face model with 72 feature points by adding or eliminating some feature points of the face model used in the MPEG-4. Also, some FAPUs were specified according to the MPEG-4 standard. An illustration of the proposed adapted face model is shown in Figure 2.3.

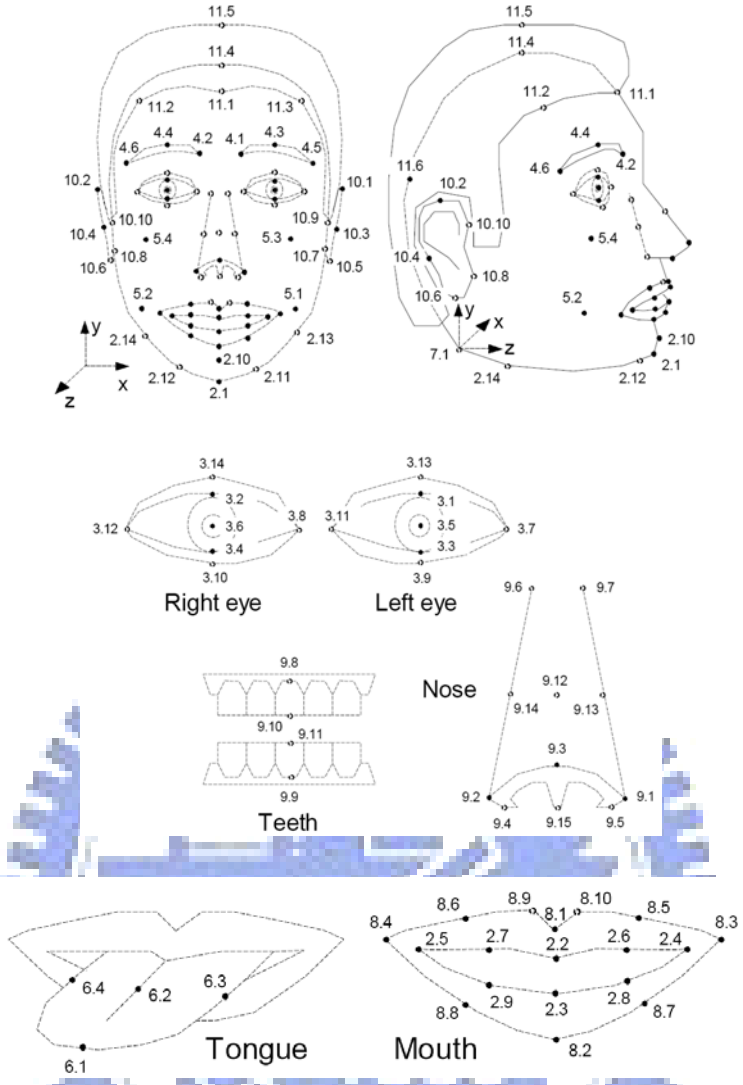


Figure 2.1 84 feature points in MPEG-4.

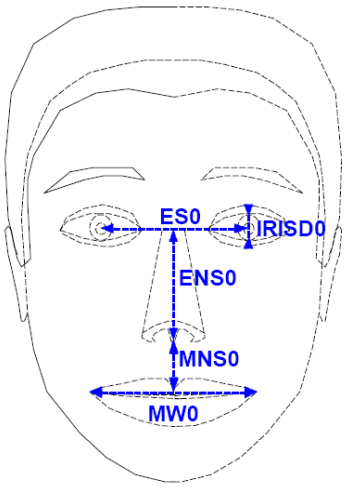
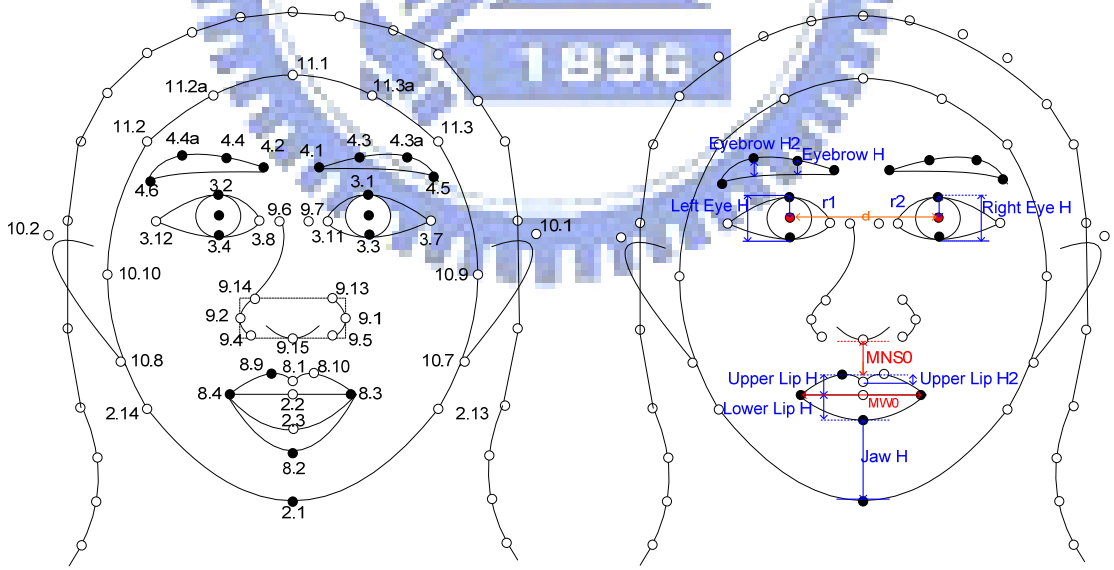


Figure 2.2 FAPUs in MPEG-4.

Chen and Tsai [1] assigned some feature points as *control points* to control facial expressions of the cartoon face. These control points are also called *face model control points* in this study, which are listed as follows.

1. *Eyebrow control points*: there are 8 control points in both eyebrows, namely, 4.2, 4.4, 4.4a, 4.6, 4.1, 4.3, 4.3a, and 4.5.
2. *Eye control points*: there are 4 control points in eyes, namely, 3.1, 3.3, .3.2, and 3.4.
3. *Mouth control points*: there are 4 control points in the mouth, namely, 8.9, 8.4, 8.3, and 8.2, by which other mouth feature points are computed.
4. *Jaw control point*: there is one control point in the jaw, namely, 2.1, which is automatically computed by the position of the control point 8.2 and the value of the facial animation parameter *JawH*.



(a) (b)

Figure 2.3 A adapted face model. (a) Proposed 72 feature points. (b) Proposed FAPUs in Chen and Tsai [1].

2.3 Construction of Mouth Model and Uses of Mouth Features

Construction of the mouth model based on the above-mentioned adapted face model is introduced in Section 2.3.1. Uses of mouth feature regions are illustrated in Section 2.3.2. And uses of mouth control points are illustrated in Section 2.3.3.

2.3.1 Construction of Mouth Model Based on Adapted Face Model

The use of mouth feature points helps us to create virtual faces. It also helps us to find the mouth feature regions which are defined by groups of mouth feature points. Also, the use of mouth feature points can compress the large volume of image files into meaningful points. Before locating positions of feature points, we must define a model to make the feature points meaningful.

In this study, we propose a mouth model based on the face model in the MPEG-4 standard and the adapted face model used in Chen and Tsai [1] by adding and eliminating some feature points. The inner mouth feature points including 2.7, 2.2, 2.6, 2.9, and 2.8 are used in the proposed mouth model. And we define some additional points to make the bottom lip smoother, which include P84_88, P88_82, P82_87 and P87_83, shown as orange dots in Figure 2.4. Furthermore, in the proposed mouth model, we add some additional points again to help morphing, which are marked as blue dots in the entire proposed model shown in Figure 2.5.

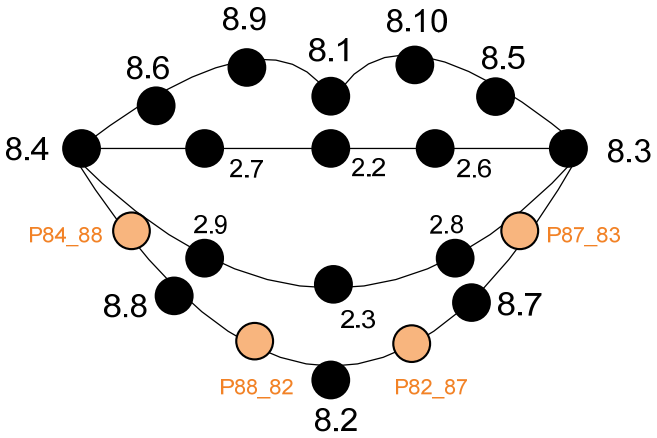


Figure 2.4 Mouth Feature Points used in the proposed method

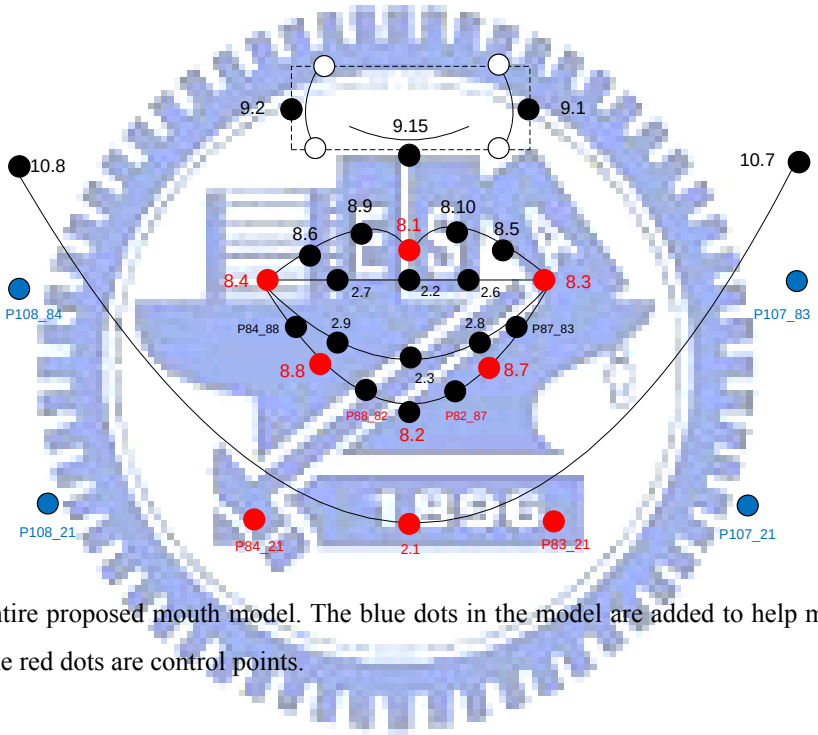


Figure 2.5 Entire proposed mouth model. The blue dots in the model are added to help morphing, and the red dots are control points.

2.3.2 Mouth Feature Regions

The use of mouth feature regions tells us the feature information such as the position, size, and range. An example of the bottom part of a virtual face is shown in Figure 2.6(a), which was created by using an Angelina Jolie's photo as the input image. As shown in Figure 2.6(b), the mouth region to be pasted on the input image is composed of the skin region, the lip region, and the teeth region, as shown in Figures 2.6(c) through 2.6(d). Mouth movements affect the range of the skin region, the size

of the lip region, and the size of the teeth region. The teeth information is obtained from the video model.

(a) (b) (c) (d) (e)

Figure 2.6 Mouth feature regions used in the proposed method. (a) Bottom part of a virtual face. (b) The mouth region. (c) The skin region outside the mouth. (d) The lip region. (e) The teeth region.

2.3.3 Mouth Control Points

Some feature points are treated as control points which can decide the size of a mouth and the range of a mouth region. By controlling the positions of the control points, a virtual face will have mouth movements and look like being able to talk. In this study, we propose a technique to reassign the positions of these points to achieve this goal, shown as red dots in Figure 2.5.

2.4 Virtual-Face Creation Process from Sequential Images

Figure 2.7 illustrates a flowchart of the stages of proposed virtual face creation from sequential images. First, a neural facial image and the first frame of a real-face video model are used as inputs to a *feature point locator*. After the work of feature point location is accomplished by the locator, the remaining frames of the video

model and the feature points of the first frame are used as inputs to a *feature point tracker*.

Then, the feature point tracker tries to extract the feature points of the remaining frames of the video model. Here the problems we mentioned in Section 2.1 are found to happen often when a closed mouth is opening or when an opened mouth is closing. So we propose to detect the states of the mouths, including the two above-mentioned states: the opening state and the closing state, and an unchanged state meaning that the mouth size in the current frame is same as that in the previous frame. Then, we use the information of the mouth states to change the matching area dynamically to reduce incorrect matching results. The area changing technique is called *window size editing* in the following.

When an opened mouth is shrunk gradually to be a closed mouth, the positions of the feature points of the inner upper mouth part sometimes will become different from those of the bottom inner mouth part. So we propose a technique to detect closed-mouth shapes and move the positions of the feature points of the inner mouth part to certain correct positions we want.

We also propose a technique to track feature points in a frame according to the image information in the previous frame. If the feature points in the previous frame are located on wrong positions, the tracker will track the wrong points in the remaining frames in the video model. Feature point correction so is necessary to make sure that the positions of the feature points are all correct; otherwise, feature point tracking will fail, according to our experimental experience.

The *virtual face creator* we propose will then divide and morph the mouth shapes to get the bottom part of every virtual face. The final step is to extract the mouth region from the virtual face and integrate it with the input image. This completes the proposed process for virtual face creation.

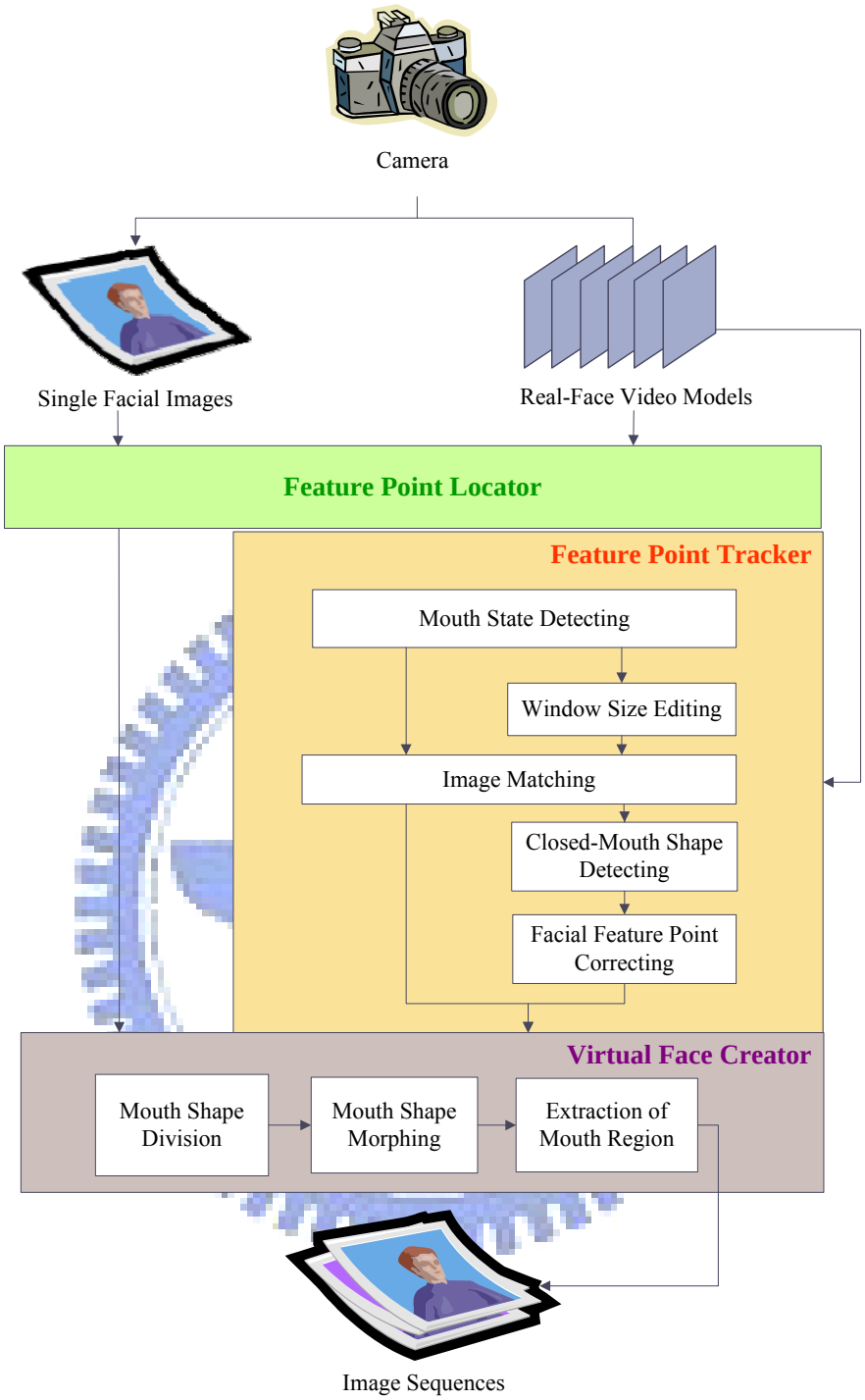


Figure 2.7 Stages of proposed virtual face creation from sequential images.

Chapter 3
Tracking of Facial Feature Points

3.1 Idea of Proposed Techniques

As mentioned in Section 1.2.2, during tracking of facial feature points, suppose that a subimage w at coordinates (s, t) within an image f is processed. Then, the moving range of w inside f is taken to be $[2s+1, 2t+1]$ in this study. The region of w is called a *content window*, and the moving range is called a *search window*. We propose in this study an image matching technique using the mouth movement information to change the size of the content window and the search window. Applying this technique, we can solve the interference problem of changed mouth-shape and teeth appearances mentioned in Section 2.1.

In this chapter, the necessity of changes of content and search window sizes and correction of facial feature point positions are explained in Section 3.1.1 and Section 3.1.2, respectively. Finally, the proposed method for tracking facial feature points is described in Section 3.1.3.

3.1.1 Necessity of Changes of Window Sizes

Because the mouth shapes are not all the same during a human's talking process, the content window sometimes will include insufficient or too much information for image matching. Two other reasons for using different window sizes for each feature point are that the teeth will interfere the matching process in the tracking of some feature points and that the movement ranges of some feature points are different. So a

window size adaptation technique is proposed.

Examples of using the changed and unchanged window sizes are shown in Figure 3.1: Figures 3.1(a) and 3.1(b) are results of applying a constant window size, and Figures 3.1(c) and 3.1(d) are those of applying dynamically changed window sizes. We can find that by the former scheme the points are tracked erroneously to stay at the same position, as shown in Figure 3.1(b), and that by the latter scheme the points are tracked correctly to be at the edge of the mouth, as shown in Figure 3.1 (d).

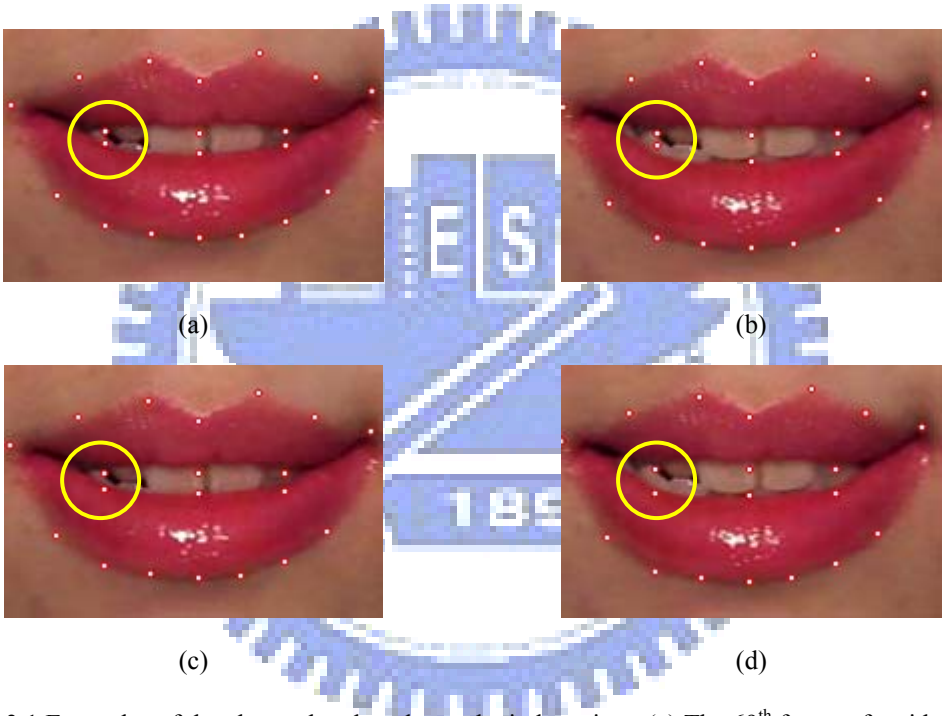


Figure 3.1 Examples of the changed and unchanged window sizes. (a) The 69th frame of a video using a constant window size. (b) The 72th frame of a video using a constant window size. (c) The 69th frame of a video using dynamically changed window sizes. (d) The 72th frame of a video using dynamically changed window sizes.

3.1.2 Necessity of Corrections of Facial Feature Point Positions

When a person in a video model says “a-u” as shown in Figure 3.2, we can find that the mouth is shrinking and the inner upper mouth part has more and more

wrinkles. Another finding is that the outer upper mouth part is brightening. One thing deserves to be mentioned is that the skin of the inner mouth part will be revealed so that the points of the inner upper mouth part looks like moving up, as shown in Figures 3.2(a) through 3.2(d).

Due to such changing image information, including the shape, brightness, and texture, the image matching is unreliable; therefore, we must correct the positions of feature points when the mouth of a video model has the shapes of "a" and "o." A wrong matching result is shown in Figure 3.2(e) from which it is seen that after connecting the points, the mouth shape becomes an opened one, but it is in fact a closed mouth. After applying the proposed correction technique, the points of the inner mouth part are located on correct positions, as shown in Figure 3.2(f).

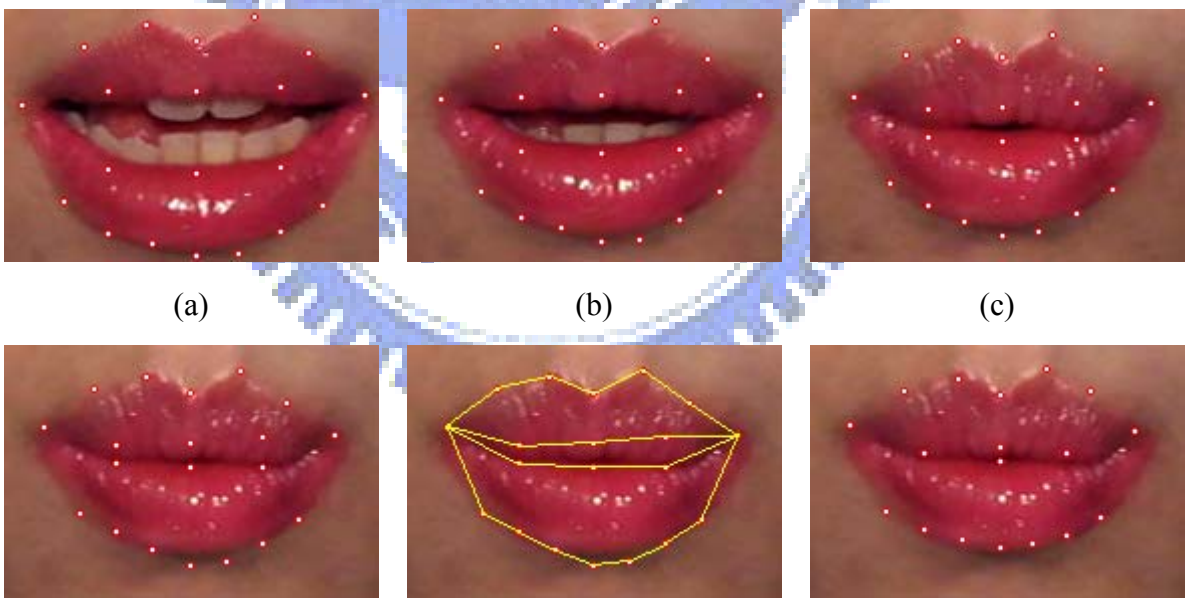


Figure 3.2 Facial feature point tracking result of mouth shape of a person saying "u." (a) Tracking result of 34th frame of a video. (b) Tracking result of 37th frame of a video. (c) Tracking result of 40th frame of a video. (d) Tracking result of 43th frame of a video. (e) Connecting the points in the 43th frame of a video. (f) The 43th frame of a video after correction using proposed method.

3.1.3 Tracking Process

In the proposed method, we track the facial feature points in the frames using the *size changing information* of a mouth, which is acquired from the difference between the size of the mouth in the current frame and that of the previous frame. The changing information represents the mouth movements so that we can know the *mouth states*. Then, we edit the size of the content window and the search window, and correct the positions of the feature points according to the mouth states. The flowchart of the proposed feature point tracking process is shown in Figure 3.3.

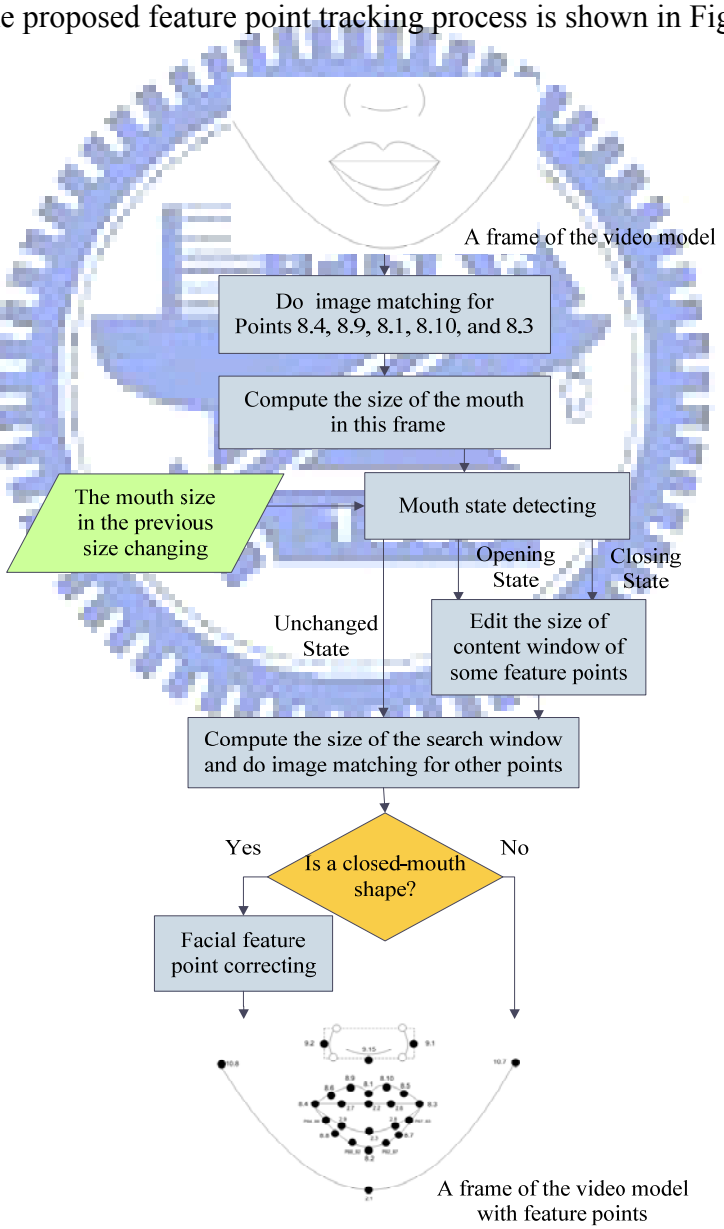


Figure 3.3 Flowchart of the proposed feature point tracking method.

3.2 Definition of Mouth States Using Mouth Size Changing Information

We propose to use the facial animation parameter units *MW0* and *MH0*, which are the width and the height of a mouth, to represent mouth movements, as shown in Figure 3.4. First, we define some mouth states to indicate how the mouth moves. We only care about some frames, in which, the size of the mouth is different from that of the previous frame. These frames are called *changed frames*.

The width difference *wDiff* of the mouth of the current frame from that of the previous frame, and the height difference *hDiff* of the mouth of the two frames, are used to represent the changed size of the mouth. Two states we define for use in the proposed technique are: *opening state* and *closing state*, and they are described in the following.

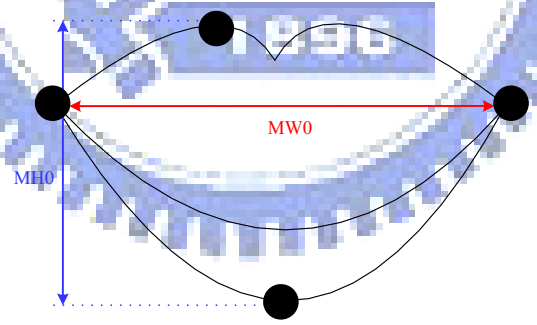


Figure 3.4 The FAPUs in the proposed system.

3.2.1 Mouth states

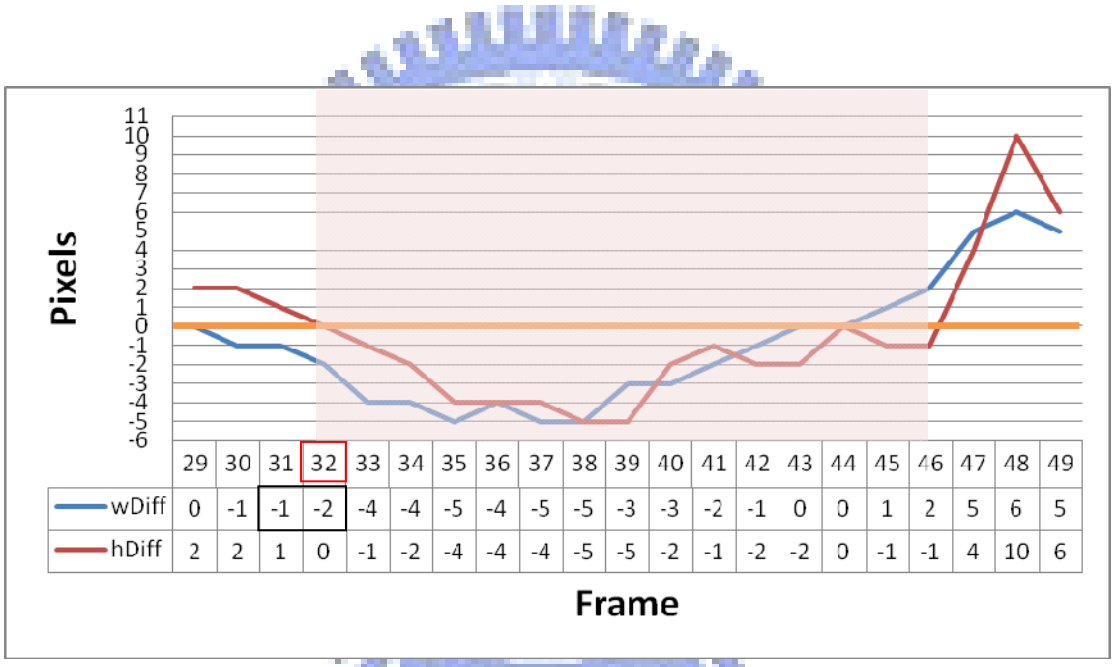
The opening state represents that a mouth is opening. The criteria for judging an opening state are that *hDiff* of the current changed frame is larger than zero, and that *hDiff* of the previous changed frame or *wDiff* of the current frame is larger than zero.

The closing state represents that a mouth is closing. The criteria for judging a

closing state are that one of *wDiff* and *hDiff* of frames, including the current changed frame and the previous changed frame, is smaller than zero.

According to these criteria, we can label states to every frame. A line chart for illustrating this is shown as Figure 3.5, where the 32th through 46th frames are assigned the closing state.

For example, if the 32th frame is the currently-processed frame and if we compare the values *wDiff* of the 31th frame and the 32th one, then according to the previously-mentioned criteria the 32th frame is assigned the closing state.



Frame	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49
wDiff	0	-1	-1	-2	-4	-4	-5	-4	-5	-5	-3	-3	-2	-1	0	0	1	2	5	6	5
hDiff	2	2	1	0	-1	-2	-4	-4	-4	-5	-5	-2	-1	-2	-2	0	-1	-1	4	10	6

Figure 3.5 A line chart of the frames of the closing state from the 32th through the 46th frames of the video model.

3.2.2 Detection of Mouth states

We compare *wDiff* and *hDiff* of the current frame with those of the last changed frame which are denoted as *pre_wDiff* and *pre_hDiff*. In other words, *wDiff*, *hDiff*, *pre_wDiff*, and *pre_hDiff* are the *mouth size changing information*.

Based on the previously-mentioned criteria, the detail of the proposed technique for mouth state detection is described in the following algorithm.

Algorithm 3.1. Detecting the mouth states using mouth size changing information.

Input: A video model V_{model} and locations L_{fp} of the feature points of the first frame of V_{model}.

Output: The mouth states S of every frame.

Steps:

1. For every frame $F_{current}$ of V_{model}, perform the following steps with the initial value of S set *none*.

   1.1 For points 8.4, 8.9, 8.3, and 8.2, apply an image matching technique to extract their corresponding points of $F_{current}$ using L_{fp}, and then update L_{fp} according to the locations of these extracted points of $F_{current}$.

   1.2 Compute *MW0* and *MH0* of $F_{current}$ in the following way:

   $MW0 = 8.3.x - 8.4.x$;

   $MH0 = 8.2.y - 8.9.y$.

   Then, denote *MW0* and *MH0* of $F_{previous}$ as *MW0′* and *MW0′*.

   1.3 Calculate the difference of the mouth size between frames $F_{previous}$ and $F_{current}$ by the following way:

   $wDiff = MW0 - MW0'$;

   $hDiff = MH0 - MH0'$.

2. Assign a mouth state to S by comparing *wDiff*, *hDiff*, *pre_wDiff*, and *pre_hDiff* in the following way:

   if $wDiff = 0$ and $hDiff = 0$, then S is unchanged;

   if $wDiff > 0$ and $hDiff > 0$, then set S = *Opening state*;

   if $hDiff > 0$ and $pre\_hDiff > 0$, then set S = *Opening state*;

   if $wDiff < 0$ and $pre\_wDiff < 0$, then set S = *Closing state*;

   if $hDiff < 0$ and $pre\_hDiff < 0$, then set S = *Closing state*.

3. Update *pre_wDiff* and *pre_hDiff* with *wDiff* and *hDiff* if both of *wDiff* and *hDiff* are

not equal to 0.

For example, if *wDiff* and *pre_wDiff* are both larger than zero, it means that the mouth is opening horizontally.

3.3 Image Matching Using Correlation Coefficients Using Dynamically Changed Window Size

The details of using dynamically changed window sizes are described in this section. The origin P of the content window is set at the center of the window, and the origin of the search window is at the left top. The distances from P to the four borders of the content window are taken to be $[S_{start}, S_{end}, T_{start}, T_{end}]$, as shown in Figure 3.6. The content widow moves around and inside the search window of image f. The range the content window can move is taken to be $[X_{start}+X_{end}, Y_{start}+Y_{end}]$. The center of the search window has the same coordinates as those of P.

We propose to edit the *distance values*, including S_{start}, S_{end}, T_{start}, T_{end}, X_{start}, X_{end}, Y_{start}, and Y_{end}, to achieve the goal of changing sizes of the content window and search window.

After changing these distance values, we can use them as parameters to the previously-mentioned image matching technique in Section 1.2.2. We compute a value of γ each time the content window moves one pixel, so we have to compute $(X_{start}+X_{end}) \times (Y_{start}+Y_{end})$ times in a session of content search. And Equation (1.1) can be written as follows:

$$\gamma(x,y)=\frac{\sum_{s=S_{start}}^{S_{end}}\sum_{t=T_{start}}^{T_{end}}\left[w(s,t)-\overline{w}\right]\sum_{s=S_{start}}^{S_{end}}\sum_{t=T_{start}}^{T_{end}}\left[f(x+s,y+t)-\overline{f}(x+s,y+t)\right]}{\left\{\sum_{s=S_{start}}^{S_{end}}\sum_{t=T_{start}}^{T_{end}}\left[w(s,t)-\overline{w}\right]^2\sum_{s=S_{start}}^{S_{end}}\sum_{t=T_{start}}^{T_{end}}\left[f(x+s,y+t)-\overline{f}(x+s,y+t)\right]^2\right\}^{\frac{1}{2}}}. \quad (3.1)$$

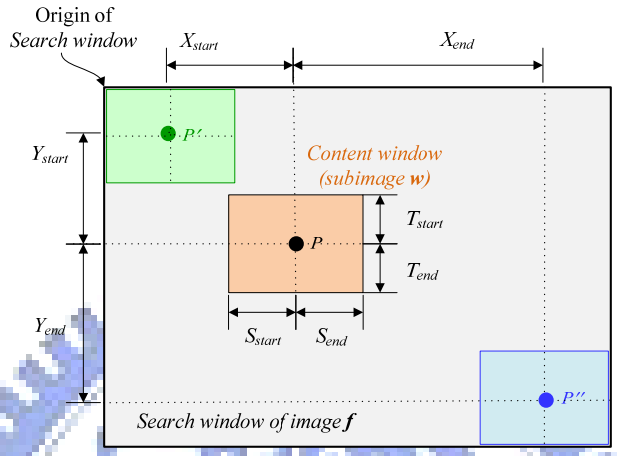


Figure 3.6 An mechanics of image matching using dynamically changed window size.

3.3.1 Initial Search Window Size and Content Window Size

The resolution in our video models is 640×480. The initial content size is set to be 35×35, and the initial search window size is set to be 41×41. An illustration of initial windows is shown in Figure 3.7. In addition, we define two variables *addX* and *addY* by the points 2.2 and 8.1 of the first frame of the video model, which can be added with or assigned to the distance values. More specifically, we assign the initial distance values, the value of *addY*, and that of *addX* in the following way:

(1) $Window_{search}$= the width of the search window;

(2) $Window_{content}$= the width of the content window;

(3) S_{start}, S_{end}, T_{start}, and T_{end} = ($Window_{content}$ − 1) / 2;

(4) X_{start}, X_{end}, Y_{start}, and Y_{end} = ($Window_{search}$ −1) / 2;

(5) $addX = Upper\ lip\ H = 2.2.y - 8.1.y$;

(6) $addY = addX \times 2$.

And we specify the initial values of the distance values of the inner-mouth feature points by the following way:

(1) T_{start} of point 2.2 = $addY$;

(2) T_{end} of points 2.7, 2.2, and 2.6 = 0;

(3) T_{start} of points 2.9, 2.3, and 2.8 = 1;

(4) T_{end} of points 2.9 and 2.8 = $addY$;

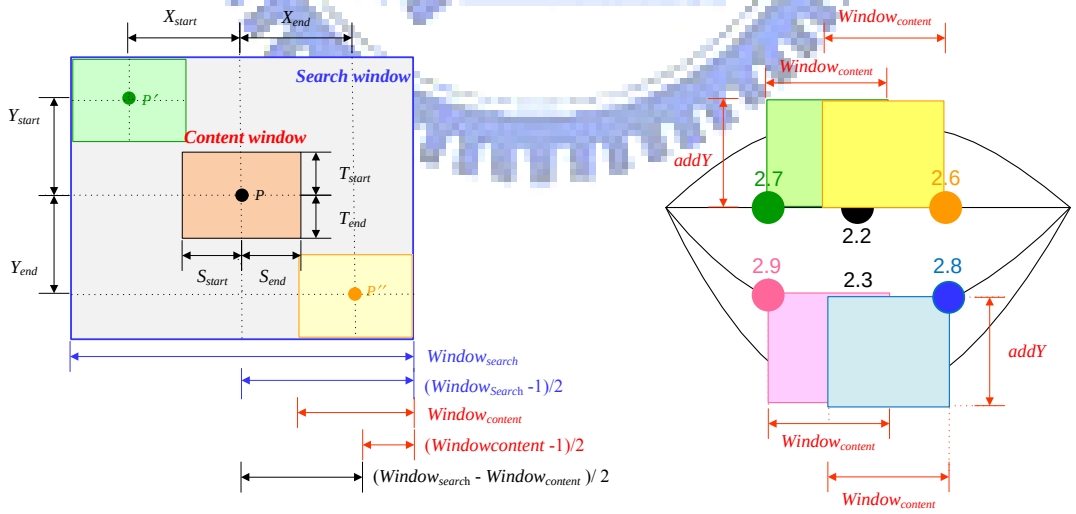
(5) T_{end} of point 2.3 = $addX \times 3$;

(6) S_{start}, of point 2.7 = 1;

(7) S_{start} of point 2.6 = $Window_{content} - 1$;

(8) S_{end} of point 2.7 = $Window_{content} - 1$;

(9) S_{end} of point 2.6 = 1.



(a) (b)

Figure 3.7 An illustration of initial window size. (a) Initial search window size. (b) Initial content window size.

3.3.2 Content Window Size of Opening State

In an opening state, we wish the inner upper mouth part to contain more corner information, so we enlarge the height of their content windows. And we hope the inner bottom mouth to contain more lip information, so we move their content windows to the center mouth and move the center P to the edge of the content windows, as shown in Figure 3.8. Because the input facial image is a neutral facial image with a closed mouth which is going to open, the initial state is set to the opening state. We specify the distance values of the inner-mouth feature points by the following way:

(1) T_{start} of points 2.7 and 2.6 = $addY$;

(2) S_{start} of point 2.9= 1;

(3) S_{end} of points 2.9 = $Window_{content}$ – 1;

(4) S_{start} of point 2.8= $Window_{content}$ – 1;

(5) S_{end} of point 2.8 = 1.

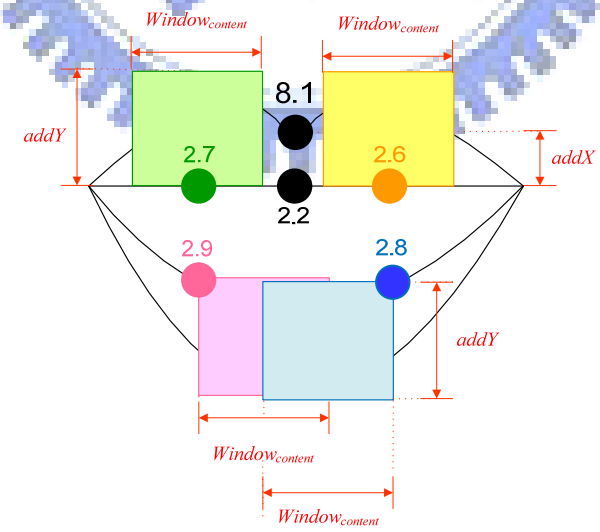


Figure 3.8 An illustration of content window size of opening state.

3.3.3 Content Window Size of Closing State

In a closing state, the desire content window size is opposite to that for an opening state. We wish the inner mouth to contain less skin information, so we reduce the height of the content window of the inner upper mouth part and move the content window of the inner bottom mouth part back to the initial position, as shown in Figure 3.9. We specify the distance values of the inner-mouth feature points by the following way:

(1) T_{start} of points 2.7 and 2.6 = *addX*;

(2) S_{start}, of point 2.9= ($Window_{content}$ – 1) / 2;

(3) S_{end} of points 2.9 = ($Window_{content}$ – 1) / 2;

(4) S_{start} of point 2.8= ($Window_{content}$ – 1) / 2;

(5) S_{end} of point 2.8 = ($Window_{content}$ – 1) / 2.

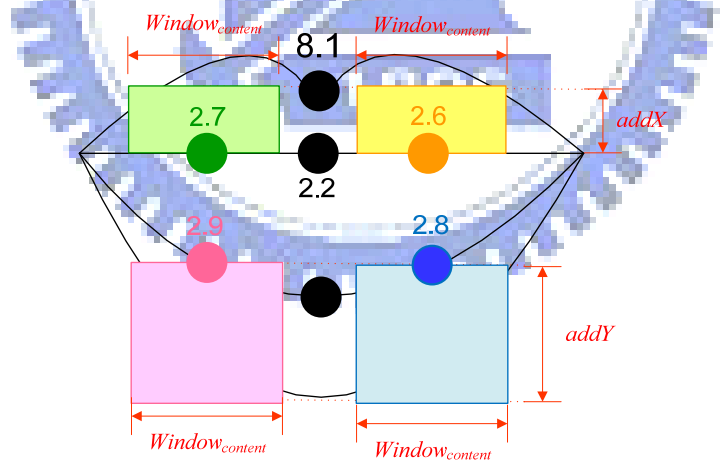


Figure 3.9 An illustration of content window size of closing state.

3.3.4 Balancing Feature Point Position by Changing Search Window Size

A mouth has symmetrical feature points in a mouth model, but not in *real-face*

video models. If we do not adjust the positions, the virtual face creation will create a virtual face with a crooked mouth, according to our experimental experience. We propose in this study an *adaptive* image matching technique to make feature point locations to be symmetric in position.

We wish the content window to move only in a vertical way, as shown in Figure 3.10, with the vertical move range being from P' to P''. In order to move vertically, we set the distance values of X_{end} equal to that of X_{start} so that the width of the search window is equal to the width of the content window.

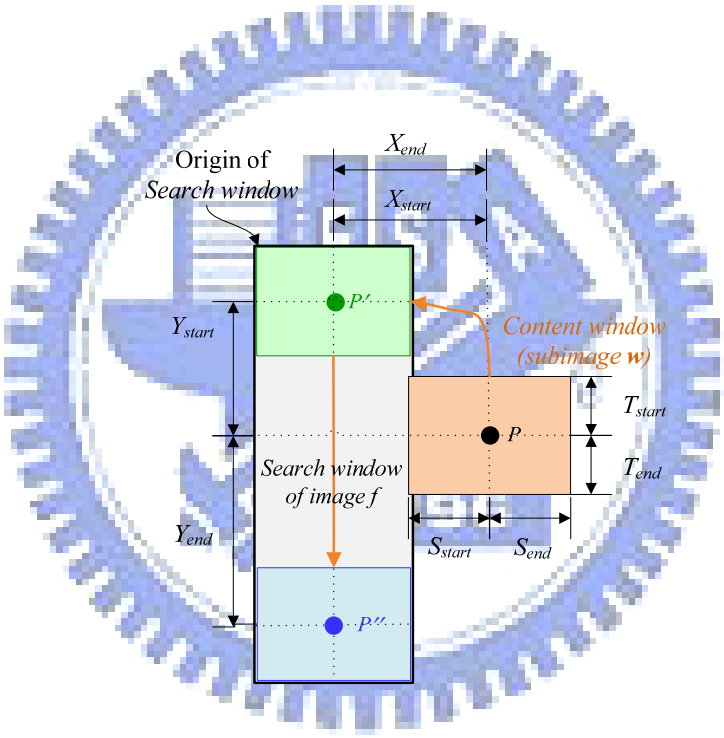


Figure 3.10 Illustration of balancing feature point positions by changing search window size.

First, we extract the positions of points 8.4, 8.9, 8.1, 8.10, and 8.3, as shown in Figure 3.11(a). And set the X_{start} of points 2.2, 2.3, and 8.2 to be 8.1.*x*, as shown in Figure 3.11(b). Second, we set the X_{start} of other points in the following way, as shown in Figure 3.11(c) through Figure 3.11(e).

(1) Set X_{start} of points 2.7, 2.9, and 8.8 = *Average* (8.4.*x*, 8.1.*x*);

(2) Set X_{start} of points 2.6, 2.8, and 8.7 = *Average* (8.1.*x*, 8.3.*x*);

(3) Set X_{start} of point 8.6 = *Average* (8.4.*x*, 8.9.*x*);

(4) Set X_{start} of point 8.5 = *Average* (8.10.*x*, 8.3.*x*);

(5) Set X_{start} of point P84_88 = 8.4.*x* + 0.25×*Length* (8.4.*x*, 8.2.*x*);

(6) Set X_{start} of point P88_82 = 8.4.*x* + 0.75×*Length* (8.4.*x*, 8.2.*x*);

(7) Set X_{start} of point P82_87 = 8.2.*x* + 0.25×*Length* (8.2.*x*, 8.3.*x*);

(8) Set X_{start} of point P87_83 = 8.2.*x* + 0.75×*Length* (8.2.*x*, 8.3.*x*).

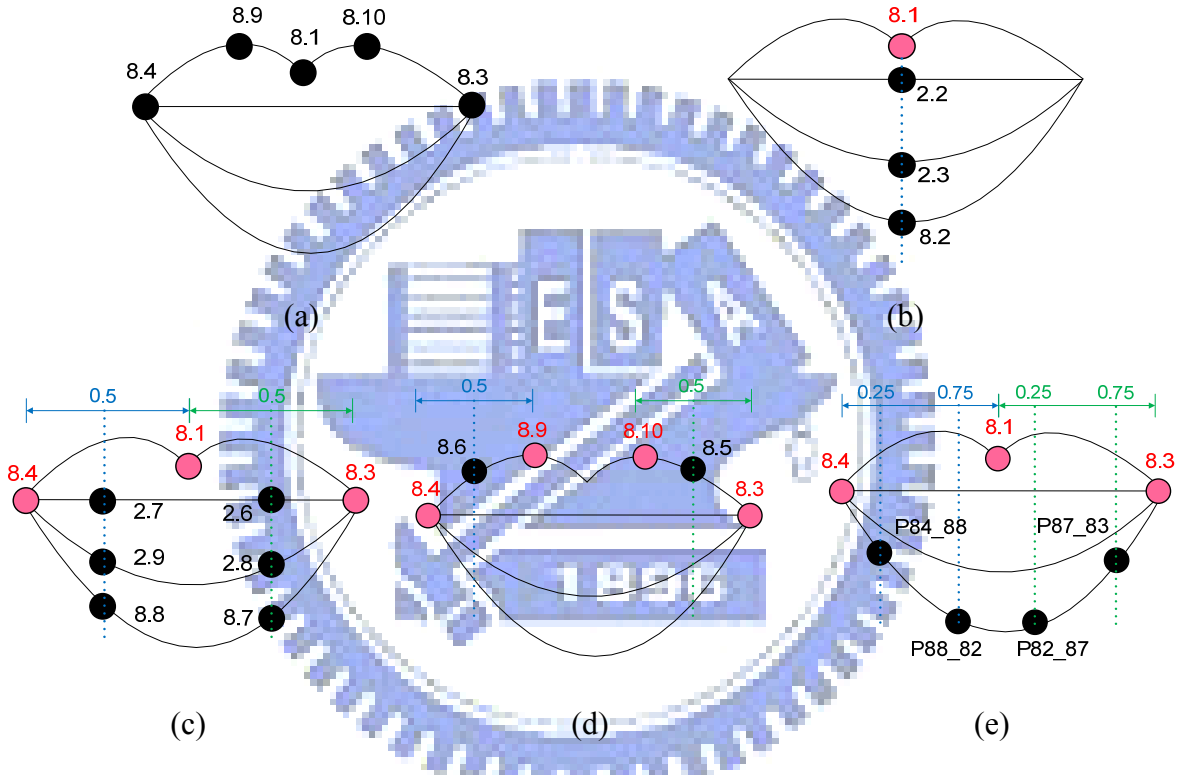


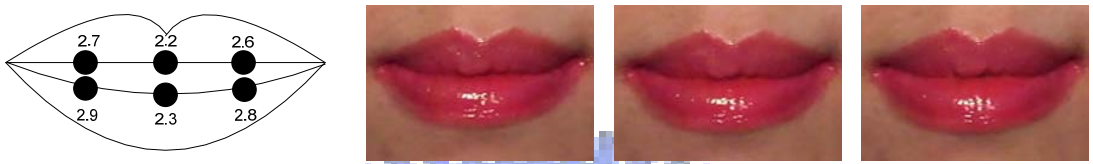
Figure 3.11 A illustration of setting the value of X_{start} and X_{end}.

3.4 Detection of Closed-Mouth Shapes

We detect closed-mouth shapes to correct the feature point positions. Although the mouth seems to be unchanged while the mouth is opening, in fact their shapes in the frames are different from one another, as shown in Figures 3.12(b) through

3.12(d). When a mouth is nearly closed, point 2.7 is closed to point 2.9, and point 2.6 is closed to point 2.8, so are points 2.2 and 2.3, as shown in Figure 3.12(a). At this time, it needs to the correct the feature point positions.

In this study, we define three types of closed-mouth shapes, which will be described in Sections 3.4.1, 3.4.2, and 3.4.3.



(a) (b) (c) (d)

Figure 3.12 An example of closed-mouth shapes. The mouth is opening.

After defining the types of closed-mouth shapes, the next step is to check if correction of feature point positions needs to be done or not. We make the decision for this according to whether frames have closed-mouth shapes or not. The detailed method for detecting closed-mouth shapes is described in the following algorithm.

Algorithm 3.2. Detection of closed-mouth shapes.

Input: A frame F of a video model V_{model}.

Output: A Boolean set $S_{mouth}\{S_1, S_2, S_3\}$ of the frame F, with S_i describing the type of the detected mouth shape .

Steps:

1. Compute the heights h_1, h_2, and h_3 of inner mouth by:

$h_1 = abs\ (2.7.y - 2.9.y)$;

$h_2 = abs\ (2.6.y - 2.8.y)$;

$h_3 = abs\ (2.2.y - 2.3.y)$.

2. Set S_1, S_2 and S_3 in the following way:

$$S_1 = \begin{cases} 1, if\ h_1 \le 1 \\ 0, otherwise \end{cases}; \tag{3.2}$$

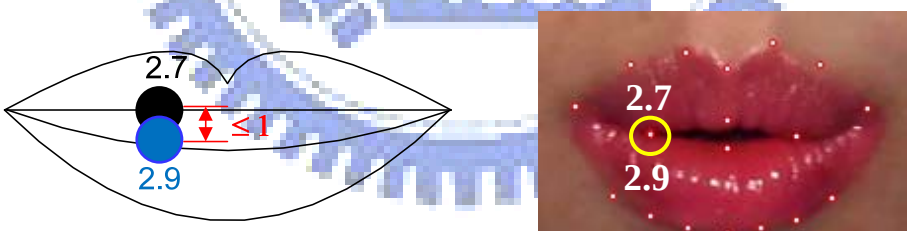
$$S_2 = \begin{cases} 1, if\ h_2 \le 1 \\ 0, otherwise \end{cases}; \tag{3.3}$$

$$S_3 = \begin{cases} 1, if\ h_3 \le 1 \\ 0, otherwise \end{cases}, \tag{3.4}$$

where S_i labeled 1 corresponds to type-*i* closed-mouth shape. If S_1, S_2 and S_3 are all labeled 0, it represents that the mouth does not have a closed-mouth shape.

3.4.1 Type-1 Closed-Mouth Shape

When the distance of points 2.7.*y* and 2.9.*y* is smaller than one, we call this mouth shape as *type-1 closed-mouth shape*, as illustrated by Figure 3.13.



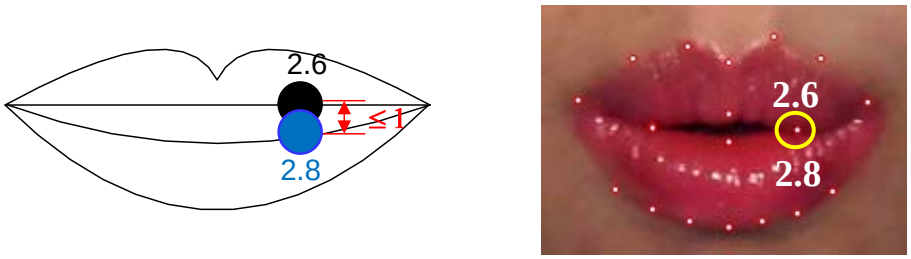
(a) (b)

Figure 3.13 Diagrams of type-1 closed-mouth shape. (a) The left points of inner mouth. (b) An example of type-1 closed-mouth shape.

3.4.2 Type-2 Closed-Mouth Shape

When the distance of points 2.6.*y* and 2.8.*y* is smaller than one, we call this

mouth shape as *type-2 closed-mouth shape*, as illustrated by Figure 3.14.



(a) (b)

Figure 3.14 Diagrams of type-2 closed-mouth shape. (a) The right points of inner mouth. (b) An example of type-2 closed-mouth shape.

3.4.3 Type-3 Closed-Mouth Shape

When the distance of points 2.2.*y* and 2.3.*y* is smaller than one, we call this mouth shape as *type-3 closed-mouth shape*, as illustrated by Figure 3.15.

(a) (b)

Figure 3.15 Diagrams of type-3 closed-mouth shape. (a) The middle points of inner mouth. (b) An example of type-3 closed-mouth shape.

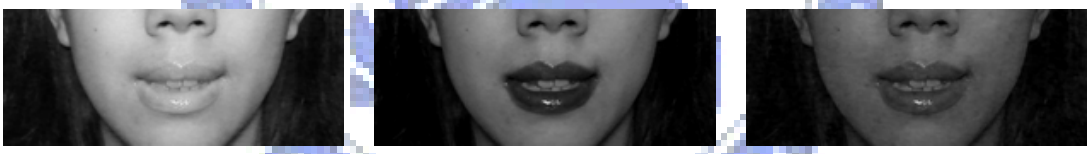
3.5 Correction of Feature Point

Locations of Closed Mouth

Before correction of the locations of the feature points of the closed mouth, we describe the idea of such correction in the green channel in Section 3.5.1. Then we describe how we extract mouth information by edge detection and bi-level thresholding in the green channel in Section 3.5.2. Finally, the proposed correction process is described in Section 3.5.3.

3.5.1 Idea of Correction in Green Channel

Because the green values of the pixels of a mouth are much smaller than those of the facial skin, as shown in Figure 3.16, it is easy to distinguish the mouth from the facial skin and the teeth. We therefore propose using the green channel to extract the mouth information.



(a) (b) (c)

Figure 3.16 The RGB channel images of partial part of 15th frame of a video model. (a) Red-channel image. (b) Green-channel image. (c) Blue-channel image.

3.5.2 Edge Detection and Bi-level Thresholding in Green Channel

The proposed system performs edge detection to check if the mouth has a closed-mouth shape, as described in the following algorithm.

Algorithm 3.3. Edge Detection by applying the sobel operator and bi-level

thresholding in green channel.

Input: A frame F of a video model V_{model} and a threshold value t for edge value thresholding.

Output: A binary image B.

Steps:

1. Take the green-channel image G of F and let $G(x, y)$ denote the green value at pixel (x, y).
2. Detect edges in G by applying the following sobel operator, as shown in Figure 3.17, to implement Equation (3.5) below to get an edge image B_{edge}:

$$S(x,y) = \left|(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)\right| + \left|(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)\right|, \tag{3.5}$$

   where z_5 denotes $G(x, y)$, z_1 denotes $G(x - 1, y - 1)$, and so on.
3. Threshold B_{edge} with t as the threshold value to get a binary image $B(x, y)$ by the following equation:

$$B(x,y) = \begin{cases} 1, & if\ I(x,y) > t; \\ 0, & if\ I(x,y) \le t, \end{cases} \tag{3.6}$$

   where t is a user defined constant (t = 100 in this study).

After the execution of the above algorithm, pixels of $B(x, y)$ labeled 1 correspond to edge pixels.

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

Figure 3.17 Sobel operators.

3.5.3 Correction Process

The final step in feature point tracking is to correct feature points in frames which have closed-mouth shapes. The detail of the correction process is described in Algorithm 3.4 below.

Algorithm 3.4. Correction of feature point positions.

Input: A binary image B generated by Step 1 of Algorithm 3.3, the positions of the feature points of B, and three Boolean values S_1, S_2, and S_3 generated by Algorithm 3.2.

Output: Feature points with correct positions and three Boolean values S_1, S_2, and S_3.

Steps:

1. Let *white_pixels*(p_1, p_2) denote the function for counting the number of white pixels along the line of two points p_1 and p_2.
2. If S_1 = true or S_2 = true, perform the following steps.
   - 2.1 If *white_pixels*(2.7.y, 2.9.y) = 0 or *white_pixels*(2.6.y, 2.8.y) = 0, adjust the coordinates of points by the following way:
     - 2.1.1 point 2.7.y = point 2.9.y;
     - 2.1.2 point 2.6.y = point 2.9.y;
     - 2.1.3 point 2.8.y = point 2.9.y.
   - 2.2 Otherwise, set S_1 = false and S_2 = false.
3. If S_3 = true, perform the following steps.
   - 3.1 If *white_pixels*(2.2.y, 2.3.y) = 0, adjust the coordinates of points by the following way:
     - 3.1.1 point 2.2.y = *Average*(2.2.y, 2.3.y);
     - 3.1.2 point 2.3.y = *Average*(2.2.y, 2.3.y).
   - 3.2 Otherwise, set S_3 = false.

3.6 Experimental Results

Some experimental results of applying the proposed method for tracking feature points are shown in Figure 3.18, from which it can be seen that the proposed method not only can track facial feature points, but also can correct the positions of the feature points so that we can get the correct results.

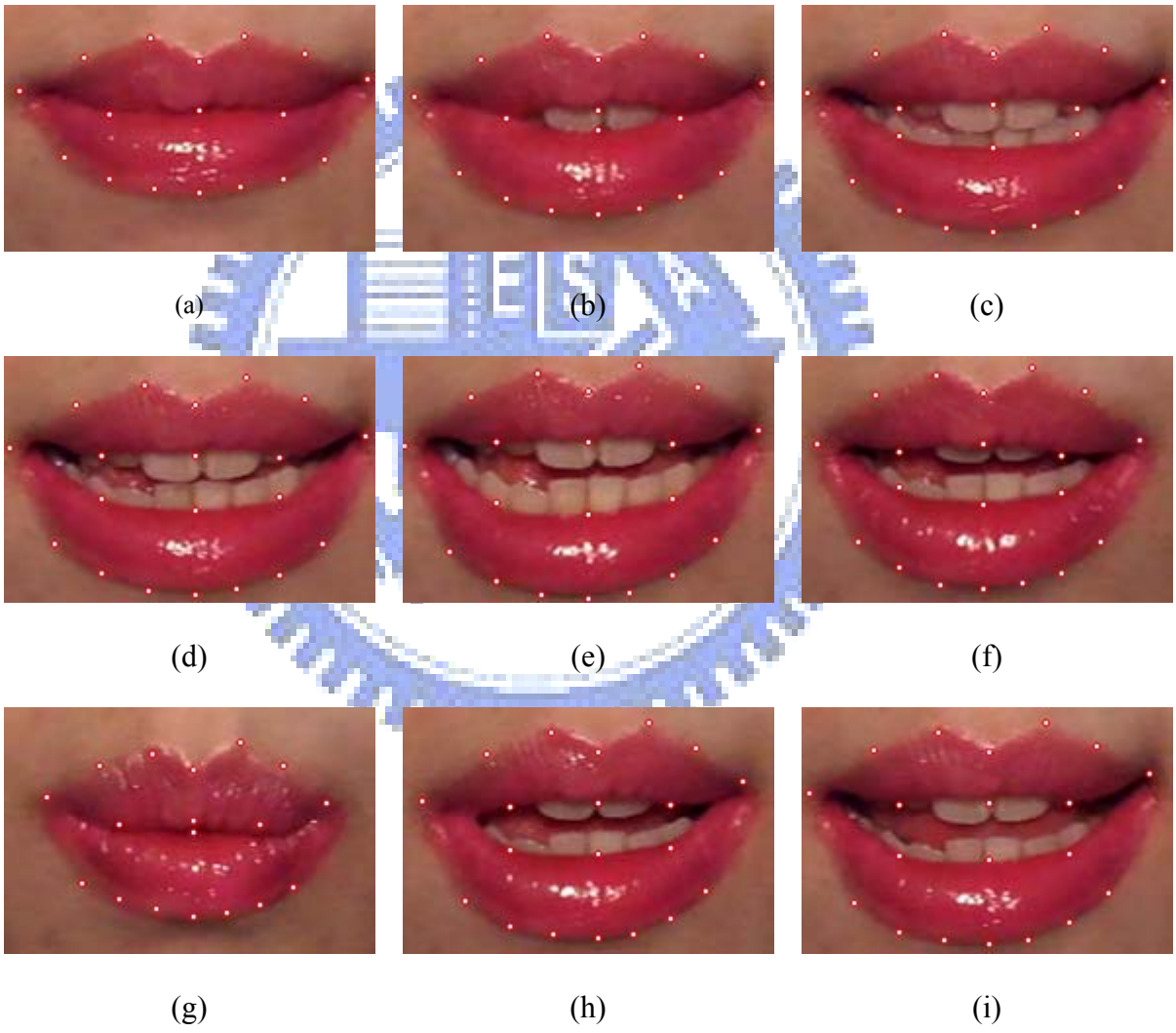


Figure 3.18 A resulting sequence of tracking feature points in a video clip of speaking “everybody” in Chinese.

Chapter 4
Creation of Virtual Faces with Dynamic Mouth Movements

4.1 Idea of Proposed Technique

The main purpose of this study is to enable a person seen in a still input image to say the same words uttered by another person appearing in a video model, that is, to let the input image have the same mouth shapes as those in the video model. In this study, we use a morphing method to *warp* the input image to the frames in the video model to achieve this goal. That is, we divide mouth shapes into quadrilaterals in the input image and do the same in every frame of the video model, and then map every quadrilateral of the input frame to those of the frames of the video model.

In this chapter, the mouth shape division technique we propose is described in Section 4.1.1, and the main steps of the proposed virtual face creation process are described in Section 4.1.2.

4.1.1 Mouth Shape Division

We separate the mouth image into two parts: a mouth part and a skin part which is near the mouth. The mouth part is divided into fourteen overlapping quadrilaterals, and the skin part is divided into thirteen overlapping quadrilaterals, as shown in Figure 4.1.

The way of such divisions is to partition the mouth or skin part into

quadrilaterals according to the mouth features, such as upper lip, bottom lip, and teeth. The quadrilaterals 1 through 4 as shown in Figure 4.1 compose the upper lip, and the quadrilaterals 9 through 14 compose the bottom lip. The reason why we divide the two lips in such ways is that we want the teeth part (quadrilaterals 5 through 8) to be *independent*, because the teeth part is the only part whose image information is obtained from the video model.

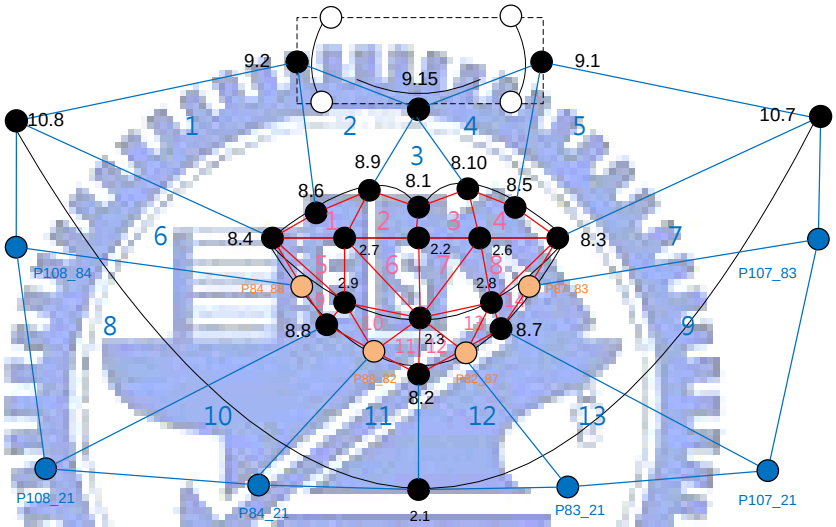


Figure 4.1 Proposed mouth shape division scheme which divides the mouth shape into twenty-seven overlapping quadrilaterals.

4.1.2 Main Steps of Proposed Virtual Face Creation Process

The main steps of the proposed virtual face creation process are mouth shape division, mouth shape morphing, and mouth region extraction, as shown in Figure 4.2.

First, we divide the mouth image into quadrilaterals by the previously-mentioned technique. Then, we use an image morphing technique to let the input image have the same mouth shapes as those in the video model, which is described in Section 4.3.

After the mouth shape morphing, we can get a mouth image sequence. We do not

paste the entire mouth shape onto the input image. Instead, we propose an extraction scheme to extract the *mouth region* more precisely from every frame of the image sequence. To make the mouth region smoother, we fill the gaps and smooth the boundary of the mouth region. Finally, we paste each mouth region onto the input image by some rules described later to get the result.

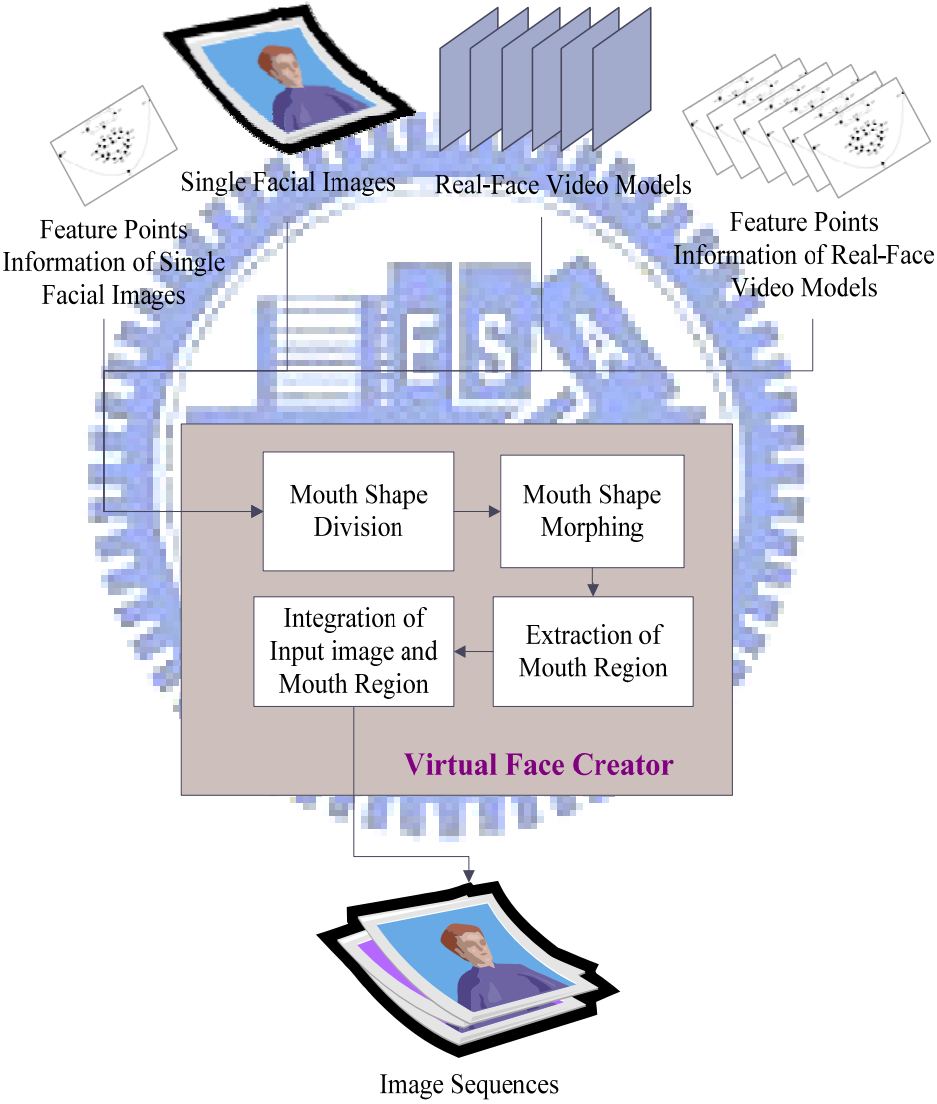


Figure 4.2 The flowchart of proposed virtual face creation from image sequences.

4.2 Creation of Real-Face Video Model

In this study, the real-face video model is recorded with a camera, and a person in it says some speeches. Each of such models in the proposed system is used to enable a person in the input image talk, so we have to extract *talking information* from the videos to convert them into useful video models. The talking information include the mouth shapes, the mouth size changing information, the feature point positions, and the image information in the video models.

In this section, some criteria identified in this study for real-face video model creation are described in Section 4.2.1. The way to locate feature points in real-face images is presented in Section 4.2.2. Finally, the proposed real-face video model creation process is described in Section 4.2.3.

4.2.1 Criteria for Real-Face Video Model Creation

We now describe the criteria for real-face video model creation we propose to make the extraction of the talking information smooth. By the way, some assumptions made for the real-face video model were mentioned in Section 1.3.2.

The frame rate of our video models is 30 frames per second, that is, playing a frame needs about 0.03 seconds. However, the time of typical mouth shape changing during a normal talking process is less than 0.03 seconds. The most important restriction is the talking speed. The speech must be spoken with a medium speed to make the tracking easier.

4.2.2 Locating Feature Points in Real-Face Images

A real-face video model is composed by a real-face image sequence. Because the feature point positions of the first frame of each video model are used to acquire the

feature points of the other frames, we must locate the positions of the feature points of the first real-face image precisely.

The positions of the feature points are the same as the black dots shown in Figure 4.1. Then, the system automatically adjusts the *x* coordinates of these feature points to move them to symmetric positions, as shown in Figures 3.10(b) through 3.10(d). Finally, we manually adjust the *y* coordinates of these feature points to make them fit the edge of the mouth edges.

4.2.3 Real-Face Video Model Creation Process

The proposed real-face creation process includes warping the input image to each frame of the video model for getting the virtual-mouth image, scaling the virtual-mouth image according to the real-face video model, and integrating the virtual-mouth image with the input image, as shown in Figure 4.3. Figures 4.3(c) and 4.3(d) have the same mouth shape as Figure 4.3(b).

During this process, the mouth image may be enlarged and then reduced, or reduced and then enlarged, or unchanged. Enlarging images usually incurs image blurring, and we want to avoid this situation to happen. In this study, the mouth size of the video model and that of the input image have three relations as follows:

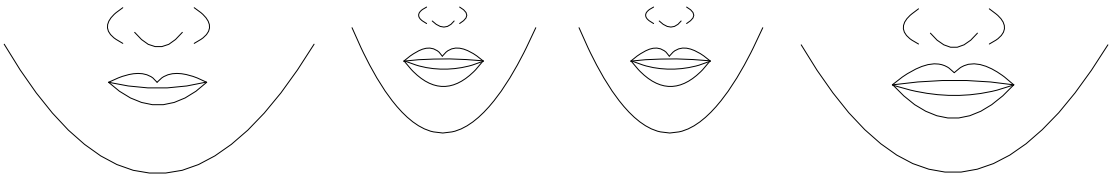
(1) The mouth size of the input image < the mouth size of the video model.

(2) The mouth size of the input image > the mouth size of the video model.

(3) The mouth size of the input image = the mouth size of the video model.

Relation (2) is exactly what we want to avoid; the input image in this relation will be warped to the smaller images of the video model and then scaled back to the large image size. So we propose a way to solve this by adjusting every frame in the video model according to the scale of the mouth width of the input image and the first frame. The mouths in the input image and in the frames should have the same width,

so that the input image does not need to be reduced within the warping process.



(a) (b) (c) (d)

Figure 4.3 The mouth images. (a) The mouth image of the input image. (b) The mouth image of a frame of the video model. (c) The virtual-mouth image is created from (a) by warping it to (b). (d) The virtual-mouth image is a scaled mouth image from (c) and is integrated with (a).

4.3 Mouth Shape Morphing with Bilinear Transformation

We use the *bilinear transformation* and *inverse bilinear transformation* proposed by Gomes, *et al.* [14] to transform between two quadrilaterals, and the details of the bilinear transformation are described in Sections 4.3.1 and 4.3.2, respectively. The proposed mouth shape morphing process is described in Section 4.3.3.

4.3.1 Review of Bilinear Transformation

Gomes, et al. [14] proposed the technique of bilinear transformation to warp a unit square to an arbitrary quadrilateral, and this transformation is denoted as $T(u, v)$. They defined $T_1(u, v)$ and $T_2(u, v)$ according to the following equation:

$$T(u, v) = (T_1(u, v), T_2(u, v)), \qquad (u, v) \in [0, 1]^2; \tag{4.1}$$

$$T_1(u, v) = auv + bu + cv + d; \tag{4.2}$$

$$T_2(u, v) = euv + fu + gv + h. \tag{4.3}$$

As shown in Figure 4.4, we know the values of transformations T_1, and T_2 on the vertices of the square to be

$$\begin{aligned} &T_1(0, 0) = a_1,\ T_1(0, 1) = b_1,\ T_1(1, 1) = c_1,\ T_1(1, 0) = d_1; \\ &T_2(0, 0) = a_2,\ T_2(0, 1) = b_2,\ T_2(1, 1) = c_2,\ T_2(1, 0) = d_2. \end{aligned} \tag{4.4}$$

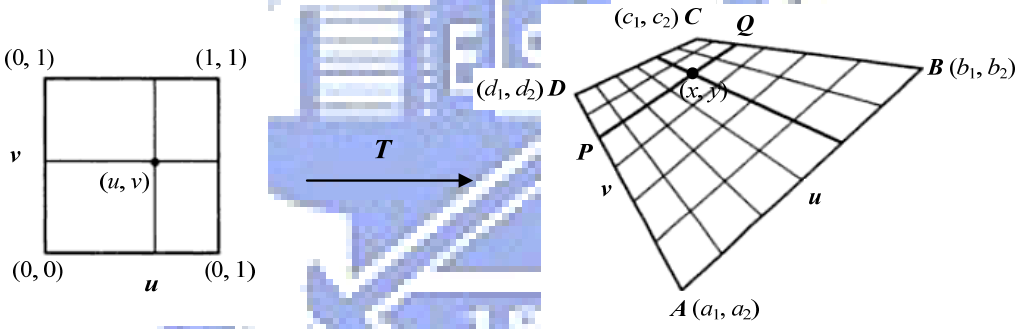


Figure 4.4 The proposed bilinear transformation in Gomes, et al. [14].

Then, we compute the coefficients of Equations (4.2) and (4.3) by substituting $T_1(u, v)$ and $T_2(u, v)$ in Equation (4.4), then we obtain these coefficients with a symbolical expression which is listed as follows:

$$a = c_1 - b_1 + a_1 - d_1; \tag{4.5}$$

$$b = b_1 - a_1; \tag{4.6}$$

$$c = d_1 - a_1; \tag{4.7}$$

$$d = a_1; \tag{4.8}$$

$$e = c_2 - b_2 + a_2 - d_2; \tag{4.9}$$

$$f = b_2 - a_2; \tag{4.10}$$

$$g = d_2 - a_2; \tag{4.11}$$

$$h = a_2. \tag{4.12}$$

We know the coordinates of the four vertices of the quadrilateral *ABCD*, so that we can compute these coefficients and can determinate the bilinear transformation by plugging in these values in Equations (4.2) and (4.3). For every pixel (u, v) in the unit square can find a corresponding pixel (x, y) in *ABCD*.

4.3.2 Review of Inverse Bilinear Transformation

Gomes, et al. [14] also proposed an inverse bilinear transformation which is conducted in a reverse direction from the quadrilateral to the unit square, as shown in Figure 4.5. A pixel *R* in quadrilateral *ABCD* is denoted as *R*(x, y).

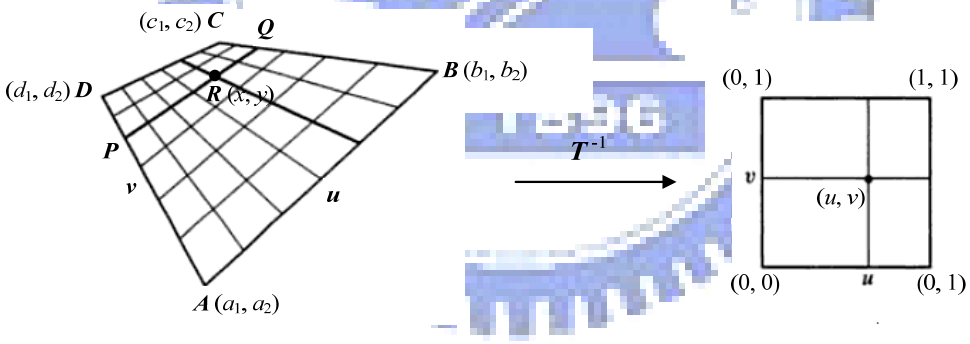


Figure 4.5 The proposed inverse bilinear transformation in Gomes, et al. [14].

Solving for u and v in Equation (4.2), and substituting these in Equation (4.3), we can obtain two equations as follows:

$$(au + c)(fu + h - y) - (eu + g)(bu + d - x) = 0;$$

$$(av + c)(fv + h - y) - (ev + g)(bu + d - x) = 0.$$

And they can be rewritten as follows:

$$Eu^2 + Fu + G = 0;$$
$$Hv^2 + Iv + J = 0, \tag{4.13}$$

where

$$E = af - be;$$
$$F = ex - ay + ah - de + cf - bg;$$
$$G = gx - cy + ch - dg;$$
$$H = ag - ce;$$
$$I = ex - ay + ah - de - cf + bg;$$
$$J = fx - by + bh - df.$$

Then, we can get two solutions of u and v as follows:

$$u = \frac{\left(-F + \sqrt{(F^2 - 4EG)}\right)}{2E}, \quad v = \frac{x - bu - d}{au + c}; \tag{4.14}$$

$$v = \frac{\left(-I + \sqrt{(I^2 - 4HJ)}\right)}{2H}, \quad u = \frac{y - gv - h}{ev + f}. \tag{4.15}$$

4.3.3 Proposed Mouth Shape Morphing Process

Because the quadrilaterals of the mouth we deal with are not unit squares, we use the transformations by Gomes, et al. [14], as shown in Figure 4.6.

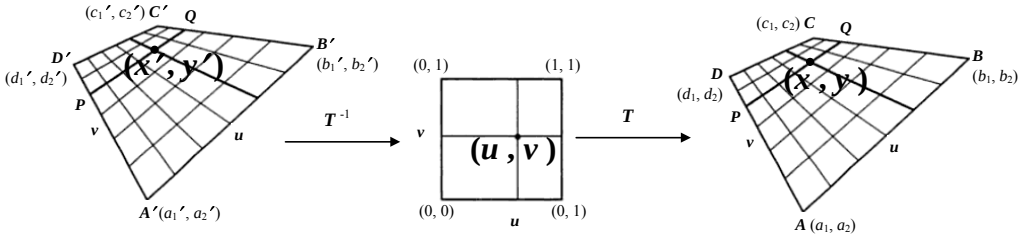


Figure 4.6 The proposed transformations between two arbitrary quadrilaterals in Gomes, et al. [14].

For mouth shape warping, we use a backward bilinear transformation from the quadrilateral $A'B'C'D'$ to the quadrilateral $ABCD$, as described in the following algorithm.

Algorithm 4.1. Bilinear transformation between two quadrilaterals.

Input: A quadrilateral $ABCD$ with color information and the coordinates of the four vertices of the quadrilateral $A'B'C'D'$.

Output: A quadrilateral $A'B'C'D'$ with color information.

Steps:

1. Compute the values of a', b', c', d', e', f', g', and h' of $A'B'C'D'$ by Equations (4.5) through (4.12).
2. Compute the values of a, b, c, d, e, f, g, and h of $ABCD$ by Equations (4.5) through (4.12).
3. For every pixel (x', y') in $A'B'C'D'$, perform the following steps to find the corresponding pixel (x, y) in $ABCD$.
   - 3.1 Compute the values of E, F, G, H, I, and J with a', b', c', d', e', f', g', and h' by Equations (4.13).
   - 3.2 Compute the corresponding (u, v) with E, F, G, H, I, and J by Equations (4.14) and (4.15).
   - 3.3 Compute the corresponding position (x, y) with a, b, c, d, e, f, g, h, u, and v by Equations (4.2) and (4.3).

4.4 Creation of Virtual-Face Image Sequences

In this section, the proposed process for generation of single virtual-mouth images is described in Section 4.4.1. And the proposed technique for scaling mouth sizes by the real-face model is described in Section 4.4.2. The proposed process for extraction of mouth regions from virtual faces is described in Section 4.4.3. The proposed processes for image gap filling and boundary smoothing are described in Section 4.4.4. Finally, the proposed process for creation of virtual-face images is described in Section 4.4.5.

4.4.1 Generation of Virtual-Mouth Images

We morph every quadrilateral of the input image to a corresponding one of each frame of the video model with the previously-mentioned shape division, and an example is given in Figure 4.7. The coordinates of point 2.1 and the additional points which help morphing marked as blue dots in Figure 4.1 are listed as follows:

(1) Point P84_21 = (point 8.4.x − 1, point 2.1.y + 2);

(2) Point P83_21 = (point 8.3.x + 1, point 2.1.y + 2);

(3) Point P108_21 = (point 10.8.x, point P84_21.y);

(4) Point P107_21 = (point 10.7.x, point P83_21.y);

(5) Point P108_84 = (point 10.8.x, point 8.4.y);

(6) Point P107_83 = (point 10.7.x, point 8.3.y);

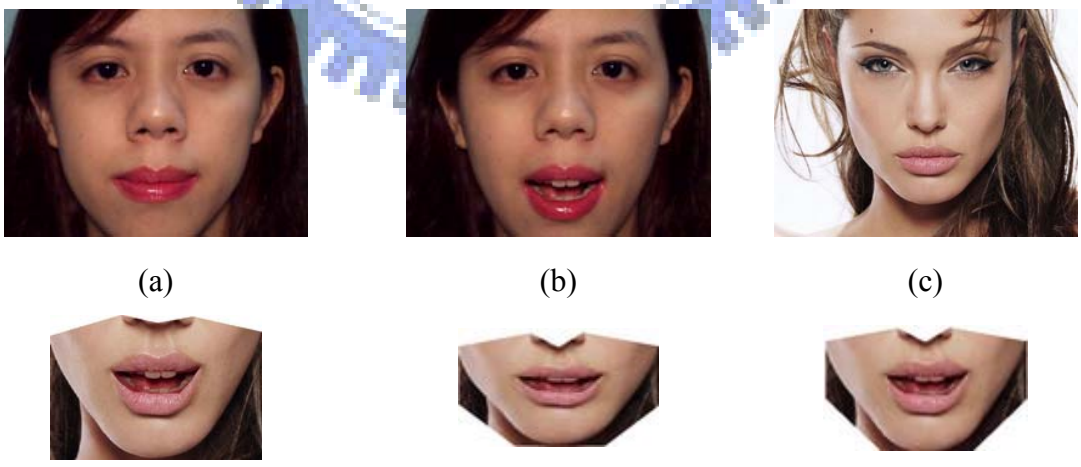
(7) Point 2.1.y =point 2.1.y + 5.

(a) (b) (c)

Figure 4.7 Generation of a virtual-mouth image. (a) An Angelina Jolie's photo as the single input image. (b) The real-face image which is the 50th frame of the video model. (c) The virtual-mouth image.

4.4.2 Scaling of Mouth Sizes by Real-Face Model

After generating the virtual-mouth images, we scale them to fit the input image. The input image has a closed mouth. If we scale the mouth only according to the scale of the mouth size of the input image, the resulting image will be unnatural, as shown in Figure 4.8(e). We propose a technique to scale the mouth size according to the real-face model, as shown in Figures 4.8(a) and 4.8(b), so that the result of the scaled mouth size is more natural as shown in Figure 4.8(f).



(a) (b) (c)

(d) (e) (f)

Figure 4.8 The illustration of scaling mouth sizes. (a) The first frame of the video model. (b) The 85th frame of the video model. (c) A single input image. (d) The virtual-mouth image. (e) The virtual-mouth image scaled by (c). (f) The virtual-mouth image with a scaled mouth.

Also, the chin must have the same vertical movements as that of the mouth, which has three feature points P84_21, P83_21, and 2.1. So, we scale the mouth size and adjust the position of the chin according to the size changing information of the mouth in the current frame and that in the first frame. We compute the scaling rate R_x and R_y of the current frame, and move the points, including points *A*, *B*, *C*, *D*, *E*, *F*, P84_21, P83_21, and 2.1 as shown in Figure 4.9, to the scaled positions which are decided by the scaling rate. The coordinates of *A*, *B*, *C*, *D*, *E*, and *F* are assigned according to those of the mouth control points mentioned in Section 2.2.3.

In the scaling process, the mouth division, as shown in Figure 4.9, is different from that discussed in Section 4.1.1, and the mouth is inside the quadrilaterals *DEBA* and *EFCB*. Here, we reassign the coordinates of some points in the following way:

(1) Point *A* = (point 8.4.x – 3, point 8.9.*y*);
(2) Point *B* = (point 8.1.*x*, point 8.9.*y* + 1);
(3) Point *C* = (point 8.3.*x* + 3, point 8.9.*y*);
(4) Point *D* = (point 8.4.*x*, point 8.2.*y*);
(5) Point *E* = (point 8.2.*x*, point 8.2.*y* + 3);
(6) Point *F* = (point 8.3.*x*, point 8.2.*y*);
(7) Point P108_21.*y* = point P88_82.*y*;
(8) Point P107_21.*y* = point P82_87.*y*;
(9) Point P84_21.*x* = point 8.8.*x* – 2;
(10) Point P83_21.*x* = point 8.7.*x* + 2.

(4.16)

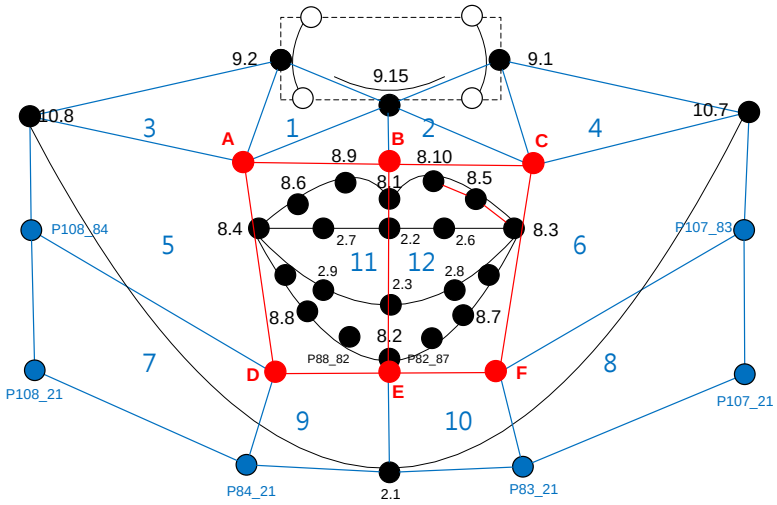


Figure 4.9 Proposed mouth shape division scheme used to scale the mouth size, which divides the mouth shape into 12overlapping quadrilaterals, including quadrilaterals *DEBA* and *EFCB*.

A mouth has two kinds of movement: the vertical movement M_y and the horizontal one M_x. M_x and M_y are computed by the scaling rates and the size of the mouth (described later). The vertical movement of the bottom lip is usually larger than that of the upper lip, so we define the vertical movement of the upper lip to be one third of the value of M_y. The horizontal movements of the left mouth part and the right mouth part are both defined equally to be half of M_x.

If the mouth width *MW0* of the current frame is smaller than that in the first frame in the video model, we reduce the width, the height, and the number of the quadrilaterals which will be transformed, as shown in Figure 4.10(b). Then, the resulting virtual mouth will contain only the mouth and skins which are near the mouth, and have no contour of the virtual face, as shown in Figure 4.10(a). We adjust the positions of the vertices of these quadrilaterals in following way:

(1) Point 10.8 = (point 8.4.x – (8.4.x – 10.8.x) / 3, point P108_84.y – 5);
(2) Point P108_84.x = point 10.8.x; (4.17)

(3) Point 10.7 = (point 8.3.x + (10.7.x − 8.3.x) / 3, point P107_83.y − 5);

(4) Point P107_83.x = point 10.7.x;

(5) Point P84_21.y = point 8.2.y + (2.1.y − 8.2.y) / 4;

(6) Point P83_21.y = point P84_21.y;

(7) Point 2.1.y = point P84_21.y + 2.

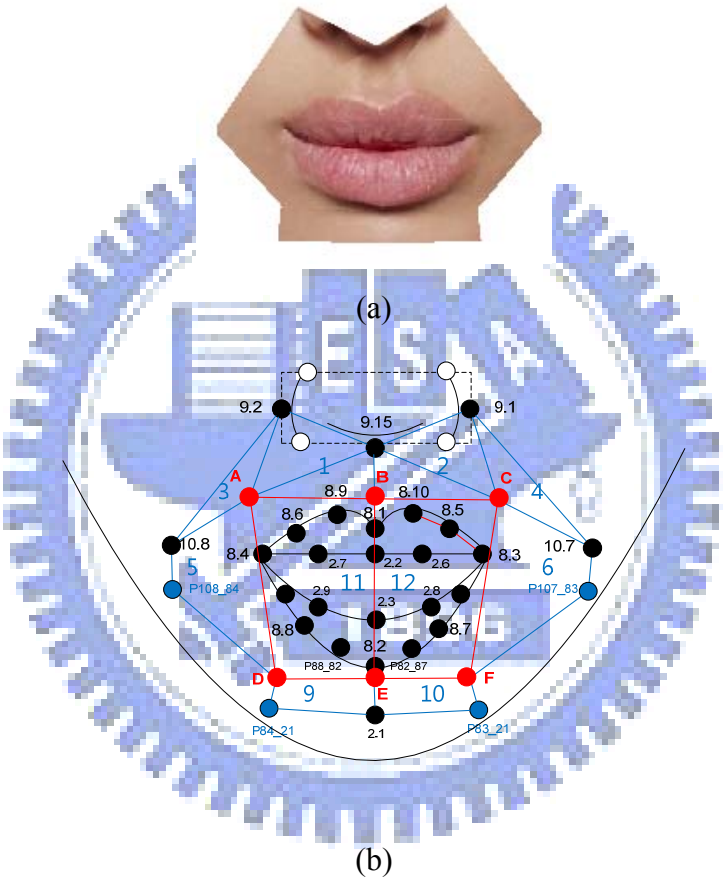


Figure 4.10 Illustration of the scaled mouth shape when the mouth width of the current frame is smaller than that in the first frame in the video model. (a) The virtual-mouth image containing the mouth and the skins near it. (b) Proposed mouth shape division scheme used to scale the mouth size.

We generate the virtual-mouth images with scaled mouths and dynamically moving chins by the following algorithm.

Algorithm 4.2. Generation of virtual-mouth images with scaled mouths and dynamically moving chins.

Input: A virtual-mouth image sequence V_{mouth}, and an input image I.

Output: A virtual-mouth image sequence V_{mouth}' with scaled mouths and dynamically moving chins.

Steps:

1. For each virtual-mouth image V of V_{mouth} except the first frame V_0, perform the following steps.

   1.1 Assign the coordinates of points A, B, C, D, E, F, P108_21, P107_21, P84_21, and P83_21 in V and I by Equations (4.16) for the mouth division.

   1.2 Compute the scaling rates R_x and R_y by the following equations:

$$R_x = \frac{(MW0_V - dMW0)}{dMW0}; \tag{4.18}$$

$$R_y = \frac{(MH0_V - dMH0)}{dMH0}, \tag{4.19}$$

   where $MW0_v$ and $MH0_v$ are the width and height of the mouth in V, and $dMW0$ and $dMH0$ are the width and height of the mouth in V_0.

   1.3 Compute the horizontal movement M_x and vertical movement M_y of the mouth as follows:

$$M_x = \frac{R_x \times MW0}{2}; \tag{4.20}$$

$$M_y = (R_y + 1) \times \frac{MW0 \times MH0_v}{MW0_v} - MH0\,. \tag{4.21}$$

   1.4 Adjust the positions of points A, B, C, D, E, F, P84_21, P83_21, and 2.1 in I

as follows.

1.4.1 Move upward the points in the upper lip, including points *A*, *B*, and *C*, by subtracting M_y /3 from the *y*-coordinates of these points.

1.4.2 Move downward the points in the bottom lip and the chin, including points *D*, *E*, *F*, 2.1, P84_21, and P84_21, by adding M_y /3 to the *y*-coordinates of these points.

1.4.3 Move the point *A* in the left mouth left by subtracting M_x /2 from the *x*-coordinate of *A*.

1.4.4 Move the point *C* in the right mouth right by adding M_x /2 to the *x*-coordinate of *C*.

1.5 Warp the quadrilaterals *DEBA* and *EFCB* in *V* to those in *I* to get a warped virtual-mouth image *V′*.

1.6 If $MW0_v \geq dMW0 \times 0.85$,

warp the quadrilaterals 1 through 10 in *V′* and *I* to get the final scaled virtual-mouth image *V″*;

else, perform the following steps.

1.6.1 Change the sizes of quadrilaterals 1, 2, 3, 4, 5, 6, 9, and 10 by Equations (4.17) to let the composition of these quadrilaterals be a mouth.

1.6.2 Warp these quadrilaterals in *V′* to *I*, and blend quadrilaterals 1, 2, 3, 4, 9, and 10 with α by *alpha blending* to get the final scaled virtual-mouth image *V″*.

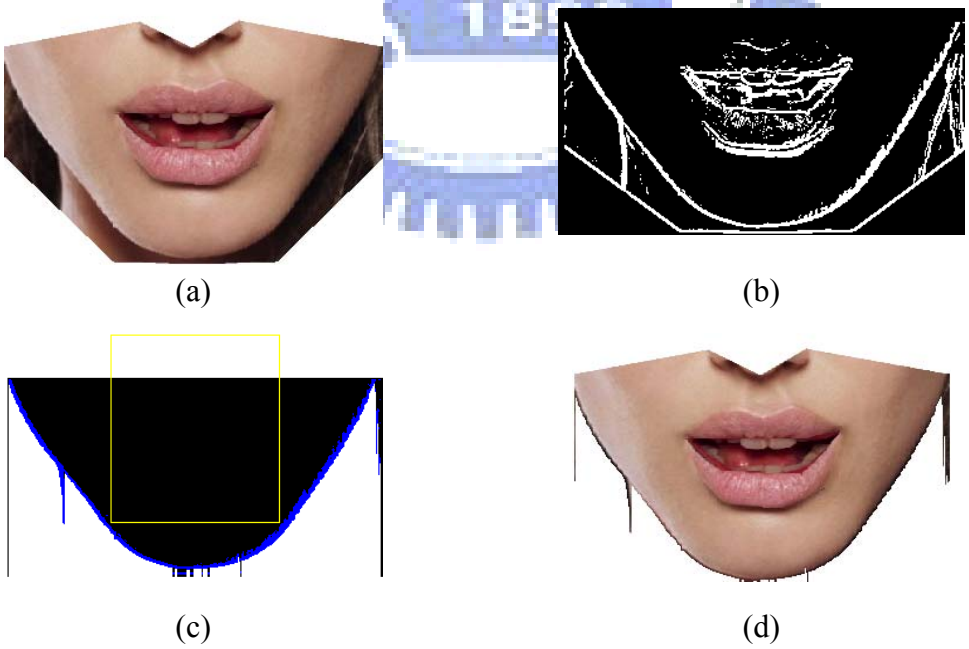
2. Compose the virtual-mouth sequences V_{mouth}' by all *V″*.

4.4.3 Extraction of Mouth Regions from Scaled-

Mouth Images

The virtual-mouth image with a scaled mouth is called a *scaled-mouth image* in this section. We perform the extraction of mouth regions when $MW0_v \geq dMW0 \times 0.85$. In this condition, the scaled-mouth images contain the contours of the faces. We extract the mouth regions R_{mouth} by using a binary image B generated by edge detection as mentioned previously in Section 3.5.2.

We propose a technique to extract the mouth region by observing the color changing information of B, as shown in Figure 4.11(b). Because we only care about the edge of the face, we remove the edge information of the mouth in B by marking them with the black color. The background is black, and the edge is marked with the white color in B. After the extraction, we get an image B' and a mouth region R_{mouth}. In B', the blue part is the edge of the face, and the black part is the facial skins, as shown in Figure 4.11. The blue and the black parts are what we want to keep and they compose the mouth region.



(a) (b)

(c) (d)

Figure 4.11 The facial images. (a) The scaled-mouth image created from the 85th frame of the video model. (b) The image B. (c) The image B'. (d) The mouth region of (a).

We scan the pixels in B in a vertical direction, and notice the color change. The order of the colors must be black, white, and black, and it means that we scan pixels from the face portion to the edge, and then to the neck below the edge. The R_{mouth} contains the edge and the face above the edge. The face sometimes has noise as white dots in B' according to our experimental experience, so we require that the edge have H_{edge} continuous white pixels in B where H_{edge} is a user-defined constant. We extract the mouth region in a region R_1 in B composed by *bUpLf* and *bDnRt*, and we define the range of the mouth by a rectangle R_2 in B composed by *sUpLf* and *sDnRt*, as shown in Figure 4.12. The details of the extraction are described in the following algorithm.

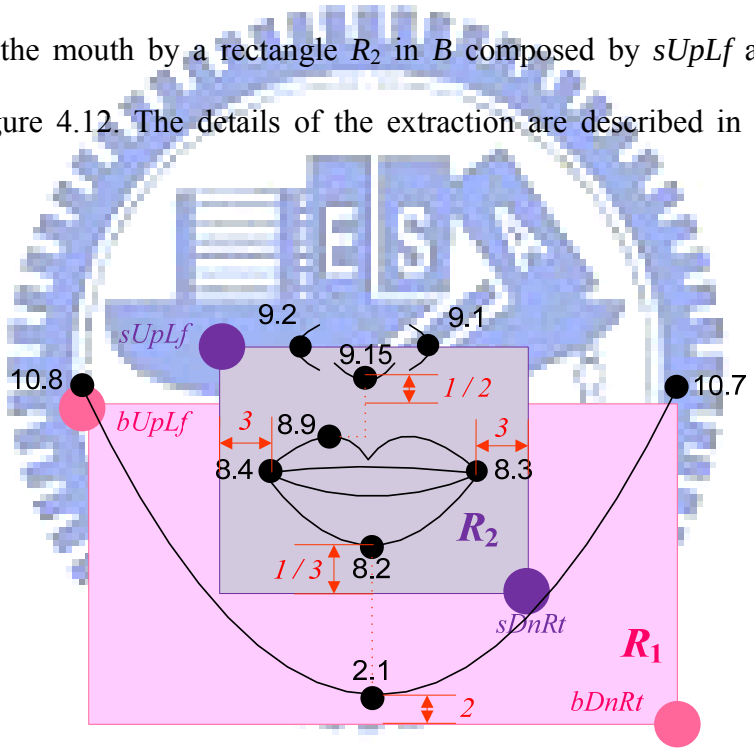


Figure 4.12 Illustration of the range of the mouth and the mouth region.

Algorithm 4.3. Extraction of mouth regions from scaled-mouth images.

Input: A scaled-mouth image V of V_{mouth}' generated by Algorithm 4.2, a binary image B generated by Algorithm 3.3, and a user-defined constant H_{edge} for detecting the edge.

Output: A mouth region R_{mouth} and an image B'.

Steps:

1. Let each pixel P of B have a corresponding pixel P' in B' whose coordinates are the same as those of P'. For each P', set the initial color C' as the color C of P.
2. Remove the edge values of the mouth by changing the pixel colors in the rectangle R_2 in B, as shown in Figure 4.12, to the black color.
3. For each vertical line L in R_1 in B, as shown in Figure 4.12, perform Steps 3.1 through 3.3.
   - 3.1 Scan the pixels in L one by one, and set the initial value of the color C_{pre} of the previously-scanned pixel to be *white*, meaning that C' of the previously-scanned pixel is not in R_{mouth}.
   - 3.2 Set the initial value of the height of the edge H which will be detected to be 0.
   - 3.3 For each pixel P in L, perform the following steps to get an image B' which have three colors, including black, white, and blue.
     - 3.3.1 If C_{pre} = *black* and C =*black*, then set C' =*black*;
     - 3.3.2 If C_{pre} =*black* and C = *white*, then
       set C_{pre} = *white*, C' = *blue*, and H = 1;
     - 3.3.3 If C_{pre} =*white* and C =*black* and $H > H_{edge}$, then
       set C_{pre} = *blue* and C' = C;
     - 3.3.4 If C_{pre} =*white* and C =*black* and $H \leq H_{edge}$, then
       set C_{pre} =*black*, C' = C, and H = 0;
     - 3.3.5 If C_{pre} =*white* and C =*white*, then
       set $H = H + 1$ and C' = *blue*;
     - 3.3.6 If C_{pre} =*blue*, then set C' =*white*.
4. Keep the pixels in V if the colors of them in B' are not white, and compose the mouth region R_{mouth} by the kept pixels.

4.4.4 Gap Filling and Boundary Smoothing

After the extraction of the mouth region, we find a problem: the edge of the mouth region is not smooth, as shown in Figure 4.11(c). In the mouth region, some pixels need to be removed and some kept for creating smooth edges. Before performing Step 4 in Algorithm 4.3, we must smooth the boundary by removing some pixels, shown as the green pixels in Figure 4.13(a), and fill the gaps by keeping some pixels, shown as the red pixels in Figure 4.13(a). In this way, we can get a mouth region with smoother edges, as shown in Figure 4.13(b). The details are described in the following algorithm which is used to replace Step 6 in Algorithm 4.3.

(a) (b)

Figure 4.13 The illustration of gap filling and boundary smoothing. (a) The B_{smooth} image. (b) The mouth region after filling and smoothing.

Algorithm 4.4. Gap filling and boundary smoothing of mouth regions.

Input: A scaled-mouth image V of V_{mouth}' generated by Algorithm 4.2, and an image B' generated by Algorithm 4.3.

Output: A mouth region R_{mouth}' with smooth edges and an image B_{smooth}.

Steps:

1. For each vertical line L in R_1 in B', perform the following steps.

1.1 For each pixel P with the color C in L, perform the following steps.

1.2 Mark the pixels which will be removed or kept by the following steps.

1.2.1 Compare the left pixel P_l of P with the right k pixels from P_{r1} through P_{rk}

of P.

1.2.2 If C is *white* AND P_l is not white AND at least one of P_{rk} is not white, then set C = *green*.

1.2.3 If C is not *white* AND P_l is white AND at least one of P_{rk} is white, then set C = *red*.

1.3 Fill the gaps and smooth the boundary in B' by the following steps.

1.3.1 If the pixels are green or white, then remove them in V.

1.3.2 If the pixels are red or black, then keep them in V.

2. Compose the mouth region R_{mouth}' with smooth edges by the kept pixels.

4.4.5 Creation of Single Virtual-Face Images

Up to now, we have performed the warping, the scaling, the extraction, the filling, and the smoothing operations as described in Sections 4.4.1 through 4.4.4, as shown in Figure 4.14.

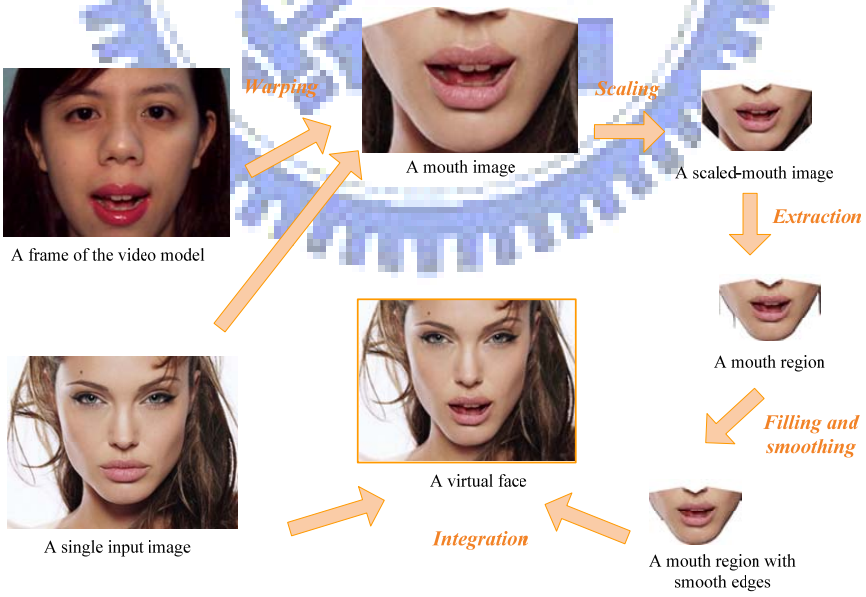


Figure 4.14 Illustration of the virtual face creation.

The last task is to integrate the mouth region with the input image. We paste the

mouth region onto the input image according to the positions of the feature points, and the pasting range is from points *bUpLf.y* to 2.1.*y* horizontally and from points *bUpLf.x* to *bDnRt.x* vertically. Finally, the virtual faces can be created, as illustrated by the center image shown in Figure 4.14.

4.5 Experimental Results

Some experimental results of applying the proposed method for virtual face generation are shown in Figure 4.16. These virtual faces were created using an Angelina Jolie's photo shown in Figure 4.7(a) as the input image and a video of the author of this thesis shown in Figure 4.15 as the video model.

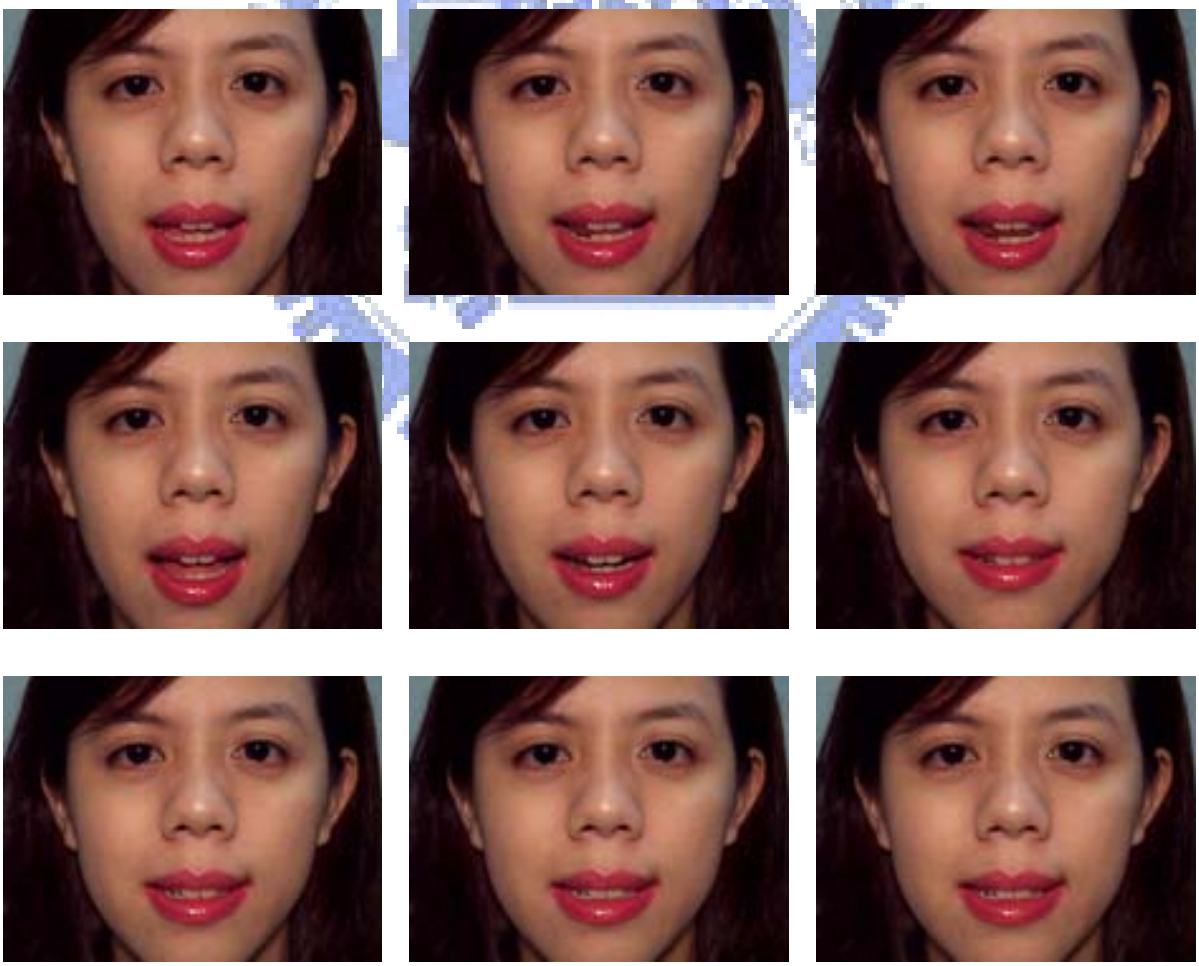


Figure 4.15 A real-face video model of speaking "teacher" in Chinese.

Figure 4.16 A resulting sequence of virtual face creation by using the video model in Figure 4.15.

Chapter 5
Experimental Results and Discussion

5.1 Experimental Results

In this section, we present our experimental results generated by the proposed techniques and some screen shots of our system.

Firstly, a video of this author saying some words was recorded by a camera, and then some frames and the audio data were extracted from it. In Figure 5.1, we show 150 frames extracted from the video.

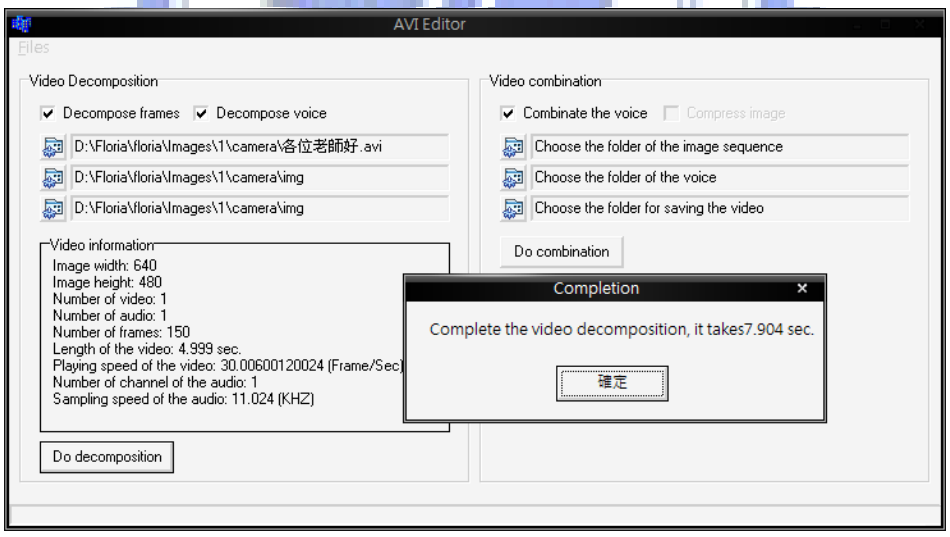
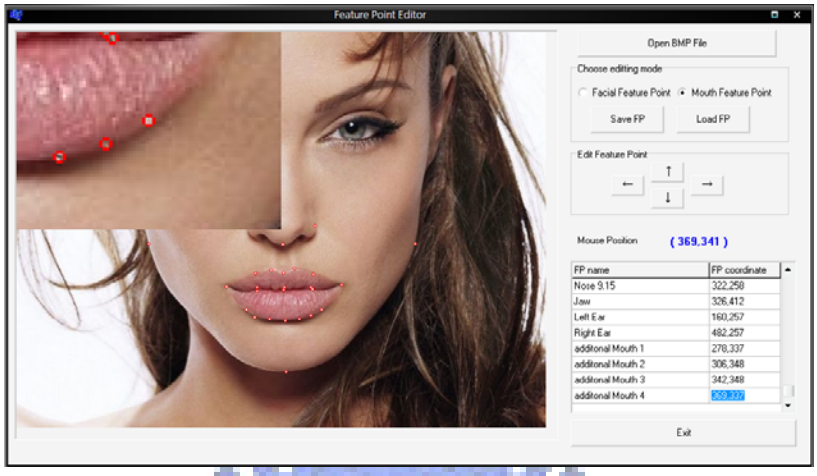


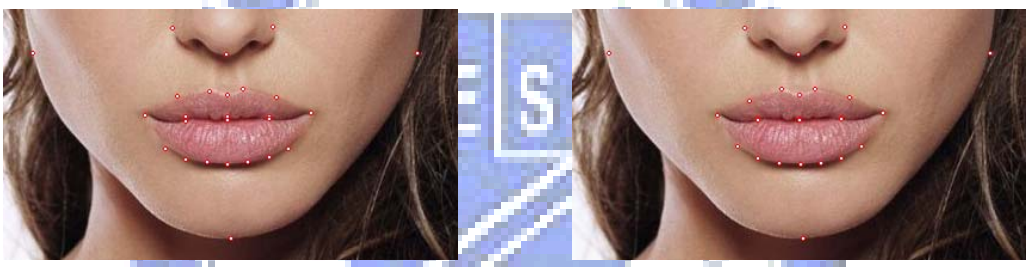
Figure 5.1 Illustration of the 150 frames extracted from the video.

Next, we located 26 feature points manually, as shown in Figure 5.2(a), and the system will automatically adjust the *x*-coordinates of these points to make them symmetrical horizontally, as shown in Figure 5.2(b). Then, we adjust the

y-coordinates of these points to the correct positions, as shown in Figure 5.2(c).



(a)



(b) (c)

Figure 5.2 Illustration of the feature point positions. (a) The feature points were located by enlarging the image. (b) The horizontally symmetric points. (c) The adjusted feature points.

After feature point locating, we choose a single neutral facial image as an input image, and choose a text file (*.face) which contains the coordinates of the feature points of the input image for virtual face creation, as shown in Figure 5.3.

Then, we select the folder which contains previously-mentioned frame sequence. The system will extract the feature point positions in each frame automatically by tracking them in each frame according to the previous frame using the techniques proposed in Chapter 3. In Figure 5.4, the top frame is the previous frame, in which the feature points are marked as blue dots, and the bottom frame is the current frame, in

which the feature points are marked as red dots.

After extracting the feature point positions of all frames, virtual faces can be created by using the proposed techniques described in Chapter 4. The mouth size information and the mouth state of each frame are shown in the system interface. Figure5.5 shows an intermediate result of the creation process. The right image is the virtual face created from the left top image which is the input image, and the left bottom image is the current frame. The final results are shown in Figure 5.6 created from a video model of speaking the sentence “Good day, every teacher” (in Chinese) with a thresholding value t = 100 and an edge height H_{edge} = 2. Two others result using a Liv Tyler’s photo and a Neng-Jing Yi’s photo as the input images with a thresholding value t = 190 and an edge height H_{edge} = 5 is shown in Figures 5.7 and 5.8.

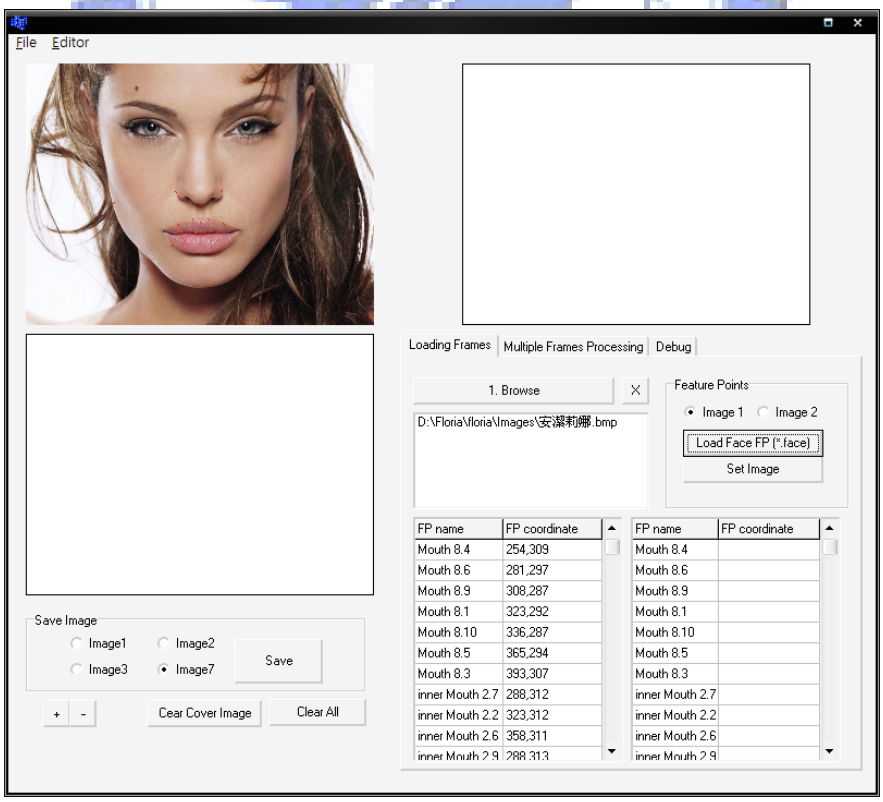


Figure 5.3 Choosing an input image and feature point coordinates of it.

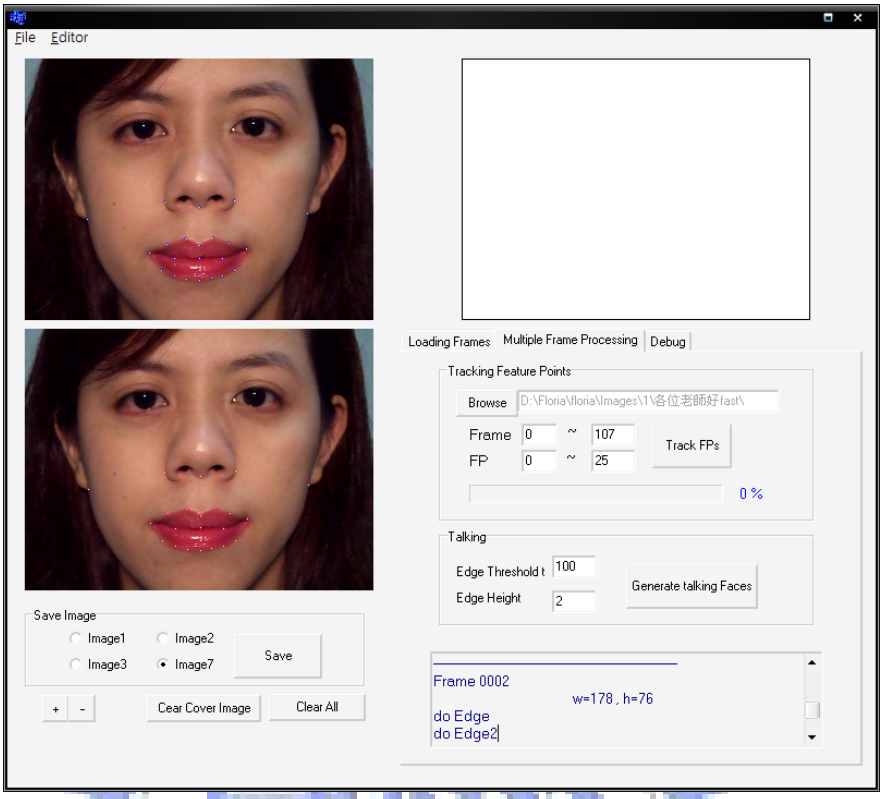


Figure 5.4 The feature point tracking process.

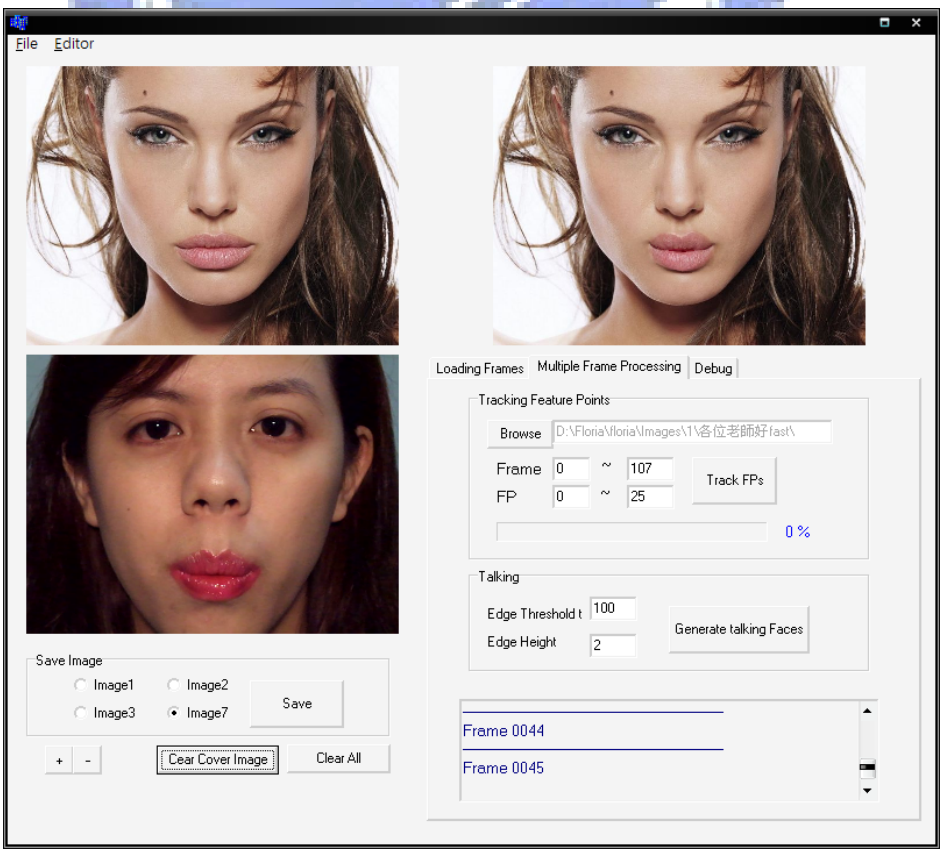


Figure 5.5 The intermediate result of virtual face creation process.

Figure 5.6 The result of virtual face creation process by using Angelina Jolie's photo as the input image.

Figure 5.7 The result of virtual face creation process by using Liv Tyler's photo as the input image.

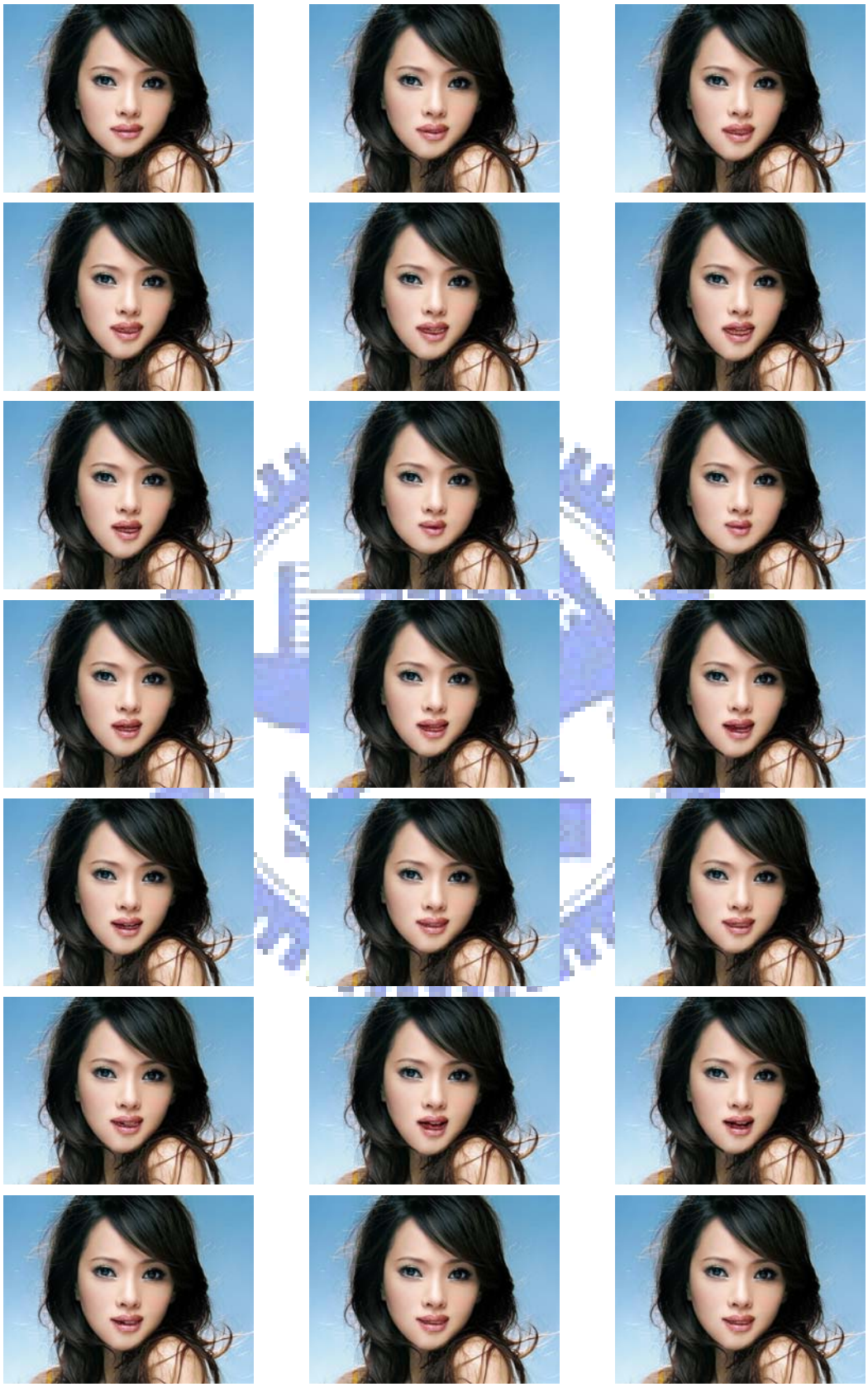


Figure 5.8 The result of virtual face creation process by using Neng-Jing Yi's photo as the input image.

5.2 Discussions

After presenting the experimental results, we would like to discuss some issues in concern as follows.

The first issue is the feature point locating process. It is a most important process of the virtual face creation, and users must concentrate on images which the feature points will be located on during this process. They must locate 26 feature points manually, and then adjust the *y*-coordinates of those points slightly. The system in this study will automatically adjust the *x*-coordinates of those points to be at symmetric positions. Because the virtual faces we create are realistic, the *y*-coordinates must be located accurately to get the optimal results.

The second issue is the feature point tracking process. The proposed tracking technique dynamically fits the mouth shapes in the real-face video models and the window sizes are dynamically changed according to the two mouth states we proposed. Because the image information of closed mouths are different from other mouth shapes, including the shape, brightness, and texture, the correction of feature point locations are proposed. For each frame, when we detect any one of the three closed-mouth shapes we proposed, we correct the feature point locations immediately. By implying the correction, we can ensure that every feature point in each frame of the video models is correct.

Finally, we discuss the virtual-face creation process. In this process, realistic virtual faces are created. We dynamically scale the mouth sizes and adjust the positions of the chins and the mouths according to the mouth shapes of the real-face video models.

The experimental results show that the created virtual faces are natural.

Chapter 6
Conclusions and Suggestions for Future Works

6.1 Conclusions

In this study, a system for automatic creation of virtual faces with dynamic mouth movements has been implemented. We have presented a way to automatically create virtual faces from a given neutral facial image, and the mouths of these virtual faces can move dynamically by feature point tracking and mouth size scaling. The system contains three components, including a feature point locator, a feature point tracker, and a virtual face creator.

By the feature point locator, 26 feature points of the input image and those points of the first frame of the video model are located manually at the accurate and symmetrical positions. Mouth feature regions are defined by groups of these points.

Next, using the feature point tracker, the feature points of each frame can be extracted by an image matching technique proposed in this study. Furthermore, mouth-movement information of the video model is analyzed in this study to get mouth states. The mouth states of each frame are detected for image matching to dynamically change window sizes.

However, correction of the feature point locations need be implemented when detecting a closed mouth. This is achieved by the use of a hierarchical bi-level thresholding technique and an edge detection technique.

Finally, the virtual face creator creates virtual face sequences with dynamical mouth movements by a proposed mouth shape morphing technique. The mouths and the chins of the virtual faces are created to move naturally, and the skins near them are made to look smooth by a morphing technique. And, a mouth size scaling technique is proposed for the synchronization of mouth movements.

The experimental results shown in the previous chapters have revealed the feasibility of the proposed system.

6.2 Suggestions for Future Works

Several suggestions for future researches are listed as follows.

(1) Automatic detection of feature points of a mouth --- For the convenience of using the proposed system, automatic feature point detection can let users skip the feature point locating process and the operation of the proposed system will be easier.

(2) Integration of eye and eyebrow movements --- With the eye and eyebrow movements, created virtual faces will become more vivid and amusing.

(3) Integration of facial wrinkles --- Just like integration of eye and eyebrow movements, virtual faces with wrinkles, such as those on the forehead, along smile lines, and round the eyes, look more natural and lifelike.

(4) Improvement on feature point tracking --- In order to deal with larger videos, the speed of feature point tracking must be faster to have the ability to handle high-quantity videos.

(5) Improvement on mouth shape detection --- In the proposed system, a person in the video model talk with a medium speed, but the talking speed

of people is faster. For wider applications, the mouth shape detection must be improved.

(6) Real-time virtual face creation --- If the facial features can be extracted in real time, the virtual face can synchronize with a speaking person in the proposed system in real time. The proposed system can be used in distance teaching for students to choose a favorite movie star or singer to be the teacher.

References

[1] Y. L. Chen and W. H. Tsai, “Automatic generation of talking cartoon faces from image sequences,” *Proceedings of 2004 Conference on Computer Vision, Graphics and Image Processing*, Hualien, Taiwan, Republic of China, Aug. 2004.

[2] Y. L. Chen and W. H. Tsai, “Automatic real-time generation of talking cartoon faces from image sequences in complicated backgrounds and applications,” *Proceedings of 2006 International Computer Symposium (ICS 2006) - International Workshop on Image Processing, Computer Graphics, and Multimedia Technologies*, Taipei, Taiwan, Republic of China, Dec. 2006.

[3] Y. C. Lin, “A study on Virtual talking head animation by 2D Image analysis and voice synchronization techniques,” *M. S. Thesis*, Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan, Republic of China, June 2002.

[4] C. J. Lai and W. H. Tsai, “A study on automatic construction of virtual talking faces and applications,” *Proceedings of 2004 Conference on Computer Vision, Graphics and Image Processing*, Hualien, Taiwan, Republic of China, Aug. 2004.

[5] Y. F. Chang and W. H. Tsai, “Automatic 2D virtual face generation by 3D model transformation techniques and applications,” *M. S. Thesis*, Institute of Multimedia Engineering, National Chiao Tung University, Hsinchu, Taiwan, Republic of China, June 2007.

[6] C. Bregler, M. Covell, and M. Slaney, “Video rewrite driving visual speech with audio,” *ACM Computer Graphics Proc. SIGGRAPH 97*, Los Angeles, CA, pp. 353-360, Aug. 1997.

[7] E. Cosatto, “Sample-based talking-head synthesis,” *Computer Animation 98 Proceeding*, Philadelphia, PA, USA, pp. 103-110, June 1998.

[8] I-Chen Lin, et al., “A speech driven talking head system based on a single face image,” *Proceedings of the 7th Pacific Conference on Computer Graphics and Applications*, Seoul, South Korea, pp. 43-49, Oct. 1999.

[9] J. P. Nedel, “Integration of speech & video applications for lip synch lip movement synthesis & time warping,” *M. S. Thesis*, Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, Pennsylvania, May 1999.

[10] T. Ezzat and T. Poggio, “Visual speech synthesis by morphing visemes,” *International Journal of Computer Vision*, Vol. 38, issue 1, pp.45-47, June 2000.

[11] I. Buck, et al., “Performance-driven hand-drawn animation,” *ACM SIGGRAPH 2006 Courses*, No. 25, Boston, Massachusetts, July 30 - Aug. 03, 2006.

[12] Q. Zhang, et al., “Geometry-driven photorealistic facial expression,” *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, San Diego, California, July 2003.

[13] R. C. Gonzalez and R. E. Woods, “Digital image processing,” 2nd ed., New Jersey: Prentice-Hall, 2002.

[14] J. Gomes, et al., “Warping and morphing of graphical objects,” San Francisco, CA: Morgan Kaufmann, 1998.

[15] T. Beier and S. Neely, “Feature-based image metamorphosis,” *ACM SIGGRAPH Computer Graphics*, Vol. 26, issue 2, pp.35-42, July 1992.

[16] J. Ostermann, “Animation of synthetic faces in MPEG-4,” *Proceedings of the Computer Animation*, pp.49, June 1998.

[17] T. Goto, et al., “MPEG-4 based animation with face feature tracking,” *Proceedings of the Erographics Workshop on Computer Animation and*

Simulation'99, Milano, Italy, Springer, Wien New York, pp.89-98, Sep. 1999.