Ph.D. Dissertation

Reliable Extraction of Realistic 3D Facial Animation Parameters from Mirror-reflected Multi-view video clips

Student: I-Chen Lin
Advisor: Ming Ouhyoung, Ph.D.

Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan

June 2003

Abstract

With the rapid development of facial animation and facial motion analysis, the necessity of motion capture techniques increases dramatically. However, existing motion capture devices are still very expensive and have specific limitations.

In this dissertation, an accurate and inexpensive procedure for estimating 3D facial and lip motion trajectories from mirror-reflected multi-view video is proposed. Two plane mirrors are located near a subject's cheeks and a single digital video camcorder is utilized to capture markers' front and side view images on a face simultaneously without special synchronization mechanisms. A novel closed-form linear algorithm is proposed to reconstruct 3D positions from real vs. mirrored point correspondences, where the extrinsic environment parameters do not need to be calibrated in advance.

Since nice symmetric properties of mirrored objects are exploited, our computer simulations and expected error estimation manifest that the proposed 3D position estimation approach is more robust against noise, more accurate and simpler than general-purpose stereovision approaches by a linear algorithm or maximum likelihood optimization. In our experiments, a root mean square (RMS) error less than 2mm in 3D space can be reached while we use only 20 arbitrary point-corresponding pairs to evaluate the orientations and locations of mirror planes.

For 3D facial motion extraction, our proposed procedure can track markers semi-automatically under normal light conditions. Adaptive Kalman predictors and filters are employed to improve the tracking stability and to conjecture the occluded markers' positions. The motion tracking can be fully automatic with fluorescent markers illuminated by ultraviolet(UV) "blacklight blue"(BLB) lamps. For the problems of missing marker and false marker detection as well as false tracking, we employ the spatial coherence on face surfaces and the temporal coherence in motion to judge, rectify and compensate false tracking trajectories automatically. More than 300 markers on a subject's face and lips are tracked from 30 fps video clips. This system will be extended for real-time tracking from live video in the near future. The estimated 3D facial motion data have also been

practically applied to our facial animation system.

In addition, a web-enabled talking head is also proposed, where facial animation is driven by natural speech. A speech analysis module is employed to obtain the corresponding phoneme sequence within the input speech, and then they are converted to the MPEG-4 high-level facial animation parameters called visemes to drive a 3D head model performing corresponding facial expressions. The talking head has been developed as plug-ins for web browsers and requires only 6 Kbps to stream high-resolution animation through Internet.

Furthermore, my work was also used in a collaborative project between INRIA of France and National Taiwan University for a French-driven talking head system.

Contents

1. INTRODUCTION 1

1.1 Motivation 1
1.2 Problem Description 4
1.3 Contribution 6
1.4 Overview and Organization 8

2. RELATED WORK 13

2.1 Introduction 13
2.2 3D Structure Reconstruction from multiple views 13
2.3 Facial Motion Tracking 15
2.4 Human face synthesis 16

3. 3D POSITION ESTIMATION FROM MIRROR-REFLECTED MULTI-VIEW VIDEO 19

3.1 The Problem Statement 19
3.2 The Proposed Closed-form Linear Algorithm 20
3.3 Error Estimation for the Algorithm 26

4. 3D MARKER TRACKING UNDER NORMAL LIGHT 35

4.1 Markers' Placement and Equipment Setting 35
4.2 Adaptive Kalman Filter for Stability Improvement 38
4.3 Semi-automatic Marker Motion Tracking 39
4.4 FAP Extraction 44

5. FULLY AUTOMATIC MASS 3D MARKER TRACKING UNDER BLACKLIGHT-UV LAMPS 45

5.1 Introduction 45
5.2 Equipment Setting and Feature Extraction 47

5.3 A Fully-Automatic Tracking Procedure for Mass 3D Markers.....55

6. EXPERIMENTS AND DISCUSSIONS.....75

6.1 Concepts 75
6.2 Error estimation by computer simulation and actual experiment 76
6.3 Discussions 84

7. FACIAL ANIMATION AND APPLICATIONS 87

7.1 Introduction 87
7.2 A Web-enabled Speech-driven Talking Head 87
7.3 Face Synthesis.....94
7.4 Facial Animation driven by Extracted 3D Facial Motion 97

8. CONCLUSION AND FUTURE WORK 101

ACKNOWLEDGEMENTS 103

BIBLIOGRAPHY.....105

APPENDICES

A. THE TWO-VIEW 3D POSITION ESTIMATION ALGORITHM 113

B. HIGH-LEVEL MPEG-4 FACIAL ANIMATION PARAMETERS 115

C. PUBLICATION LIST 117

Chapter 1

Introduction

1.1 Motivation

Human faces may be the most noticeable and expressive part of our bodies. We can identify people from their faces, and we also realize others' feelings and moods from their facial expressions. No matter pursing lips, raising eyebrows, grinning, or even making a delicate expression may reveal certain meanings for us. Moreover, mouth and lip motions are regarded as another major channel in recognition and understanding of spoken language. In M. Breeuwer and R. Plomp's experiment [BREE85], visual speech significantly improves the perception from 23% to 79% when auditory speech is degraded. Since faces are so important, subtle and closely related to our life, synthesizing realistic facial animation is one of the most attractive topics in computer graphics for decades.

With the rapid improvement of hardware and software techniques, computer-generated facial animation starts to play a vital role in various kinds of applications, such as computer graphics based movies, and video games. Recently, we can even see some virtual characters reporting news on television. However, up to now, synthesizing realistic facial animation is still a difficult problem due to our familiarity with human faces. As mentioned above, facial expression is one of our major communication approaches, and we should pay more attention to even minor variations on a face. An observer can easily detect even the slightest flaw. Furthermore, I.S. Pandzic et al. [PAND99] noticed that significant artifacts of mouth motion in facial animation could even worsen the understanding.

Therefore, to realistically mimic facial animation, a synthetic face's behaviors

must precisely conform to those of a real one. However, facial surface points, being nonlinear and without rigid body properties, have quite complex action relations. During speaking and pronunciation, the facial and lip motion variation can be more complicated. Motion trajectories of a point between articulations are also nonlinear and depend on not only current articulations but also preceding and successive ones, which are so called co-articulation effects [COHE93]. An example is shown in Figure 1.1

Performance-driven facial animation [WILL90, GUEN98] provides a direct and convincing approach to handling delicate human facial variations. This method animates a synthetic face using motion data captured from a performer. In modern computer graphics-based movies such as “Final Fantasy”, “Shrek”, and “Toy Story”, optical or magnetic motion trackers are utilized to capture markers’ 3D motion trajectories on a performer’s face. These devices usually track only a limited number of markers; however, the dozen or so markers they can place on facial feature points only sparsely cover the whole face area. Therefore, to derive a vivid facial animation, animators must adjust for the uncovered areas.

On the other hand, many methods proposed to approximate human facial motion by physical dynamic systems or mathematical formulations. Some of the researches try to control face surfaces as bicubic patches [REEV90]. K. Waters and D. Terzopoulos [WATE87, TERZ90] proposed a muscle-based face model with three-layer tissues. Cohen et al. [COHE93] suggested that the weights of transitions between visemes should be overlapping dominance functions with bases of negative exponential functions. Even though these hypotheses try to parameterize complicated facial motion, they encounter critical problems. For example, “What are the parameters’ values?” and “How much error will occur when adopting certain parameter values?”. We can only answer these questions by comparing simulations with measured data from a real human face. However, existing measurement devices such as the optoelectronical motion trackers, though highly accurate, are also quite expensive and pose limitations on the marker number and their placement on surfaces.

Another concern for synthetic faces is from the viewpoint of data transmission. Due to bandwidth constraints, “streaming” high-resolution videos is quite difficult. Model-based video coding approaches, using synthetic faces and talking heads instead of current frame-based videos, are considered to be a good substitute.

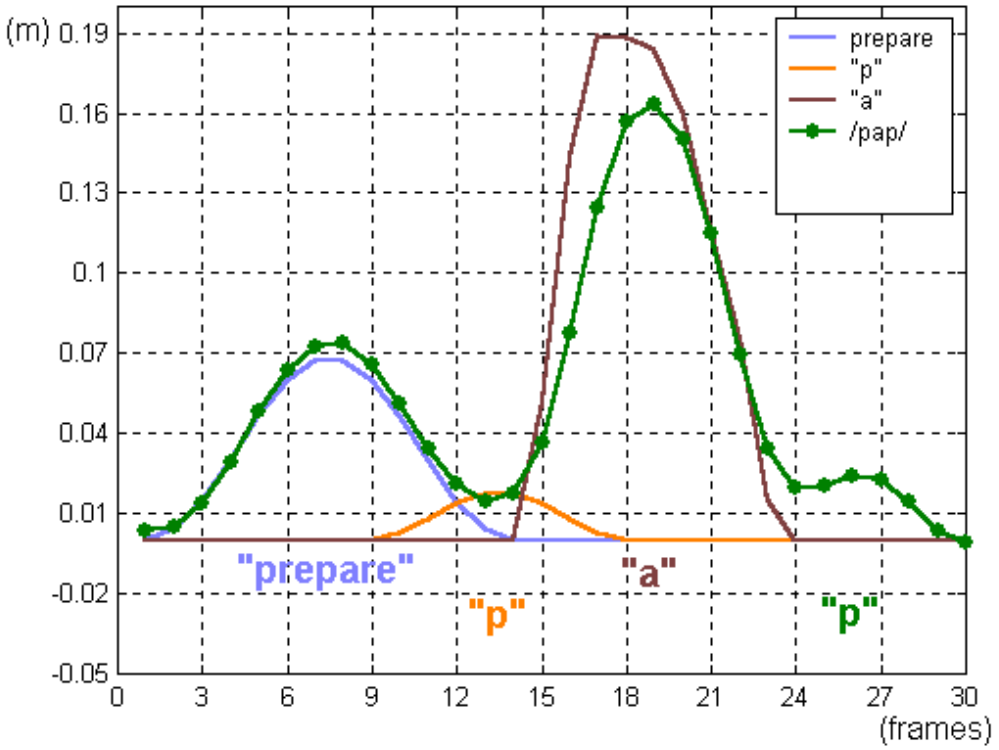


Figure 1.1 Captured lower lip tip motion trajectories of an utterance /pap/ and corresponding visemes “p”, “a” and “preparation” (shown only y-axial motion). The curve with dots is the trajectory of /pap/, the light purple one is “preparation”, the orange one is “p”, and the brown one is “a”. The nonlinear motion transitions between current visemes are dependent on preceding and successive visemes. [The unit in x axis: NTSC frame (1/29.97 sec); the unit in y axis: meter]

1.2 Problem Description

To tackle the problem of acquiring facial motion data, the goal of this dissertation is to develop an accurate and inexpensive procedure to estimate 3D facial animation parameters. The motion estimation can be further divided into two main issues: 3D position reconstruction and motion tracking.

3D position reconstruction

It is well known that we can reconstruct 3D structure from multiple view images [LONG81]. To avoid the estimation error caused by imperfect synchronization between multiple cameras, we adopt mirror-reflected multiview video clips to acquire multiple views simultaneously with only a single video camera, as shown in Figure 1.2.

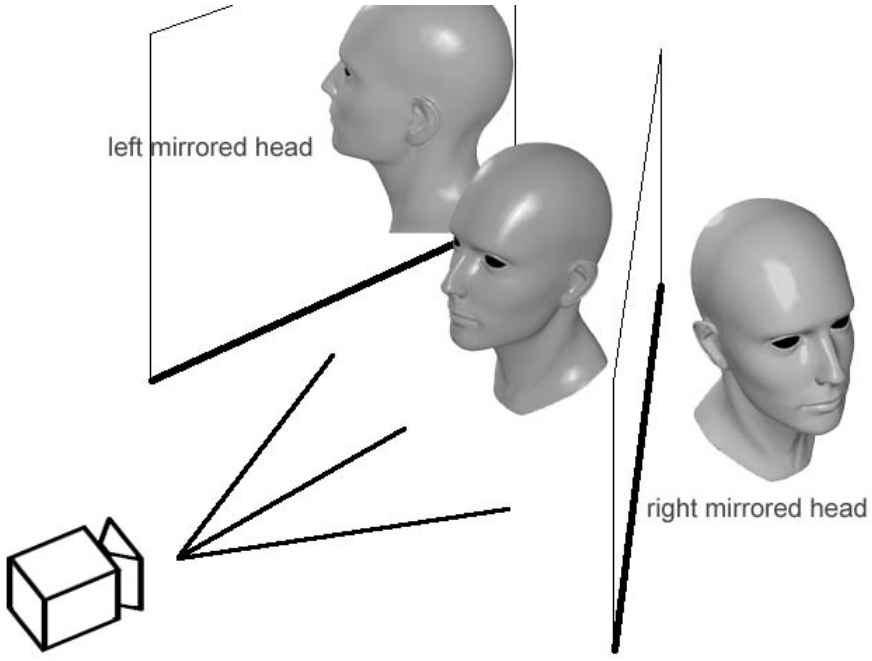


Figure 1.2 A diagram of our capture equipment. Two plane mirrors are placed next to a subject's face, and the front view and mirror-reflected images are captured simultaneously with a single video camera.

Using mirror-reflected multi-view video clips for facial motion estimation has

been mentioned in a few researches [PATT91, BASU97] where the processes are either simplified or some general-purpose stereovision approaches are employed. Nevertheless, in our early trial, general-purpose stereovision approaches, such as the two-view approach [WENG89, WENG93], can easily degenerate due to slight measurement errors of 2D corresponding point pairs. However, we find that there exist nice symmetric properties between front and mirror-reflected objects that can be utilized. With these properties, a novel closed-form linear algorithm is proposed in this dissertation to estimate 3D position robustly and accurately.

Motion tracking

Two kinds of point correspondence, *2D point correspondence between camera views within a frame* and *3D point correspondence between frames* should be estimated in 3D motion tracking. The first issue, 2D point correspondence within a frame, is that we have to find out the correct point correspondence between each view to reconstruct the correct 3D structure in a video clip. After the 3D structure of each frame are reconstructed, the 3D point correspondence between frames should be estimated to recover the 3D motion trajectories of facial surface points. The difficulty of point correspondence estimation will be raised dramatically as the number of feature points increases since the candidates of corresponding point pairs will increase quadratically.

Moreover, various factors will disturb our marker extraction in video clips, and they can roughly be classified as *variation due to projection* and *noise in video*. The problem of variation due to projection is inherent in systems using projected images. For examples, markers' colors vary due to reflective angle change; markers' projected shapes vary in different viewpoints. Sometimes, markers may even be occluded, and this situation is quite critical since it is difficult to detect and compensate occlusion. On the other hand, the problem of noise in video is inherent in modern video camera design. For example, the use of a field as the unit instead of a frame causes the interlaced effects; the sensors of digital cameras, such as CCD (charge-coupled device) truncate the continuous projection image into discrete data. Thermal noise is another concern.

In our work, we propose an efficient procedure for tracking a considerable number of 3D markers, notably more than 200. It can detect and compensate false-detection and missing markers in the tracking fully automatically.

Delivering facial animation across Internet

In addition to facial motion acquirement, another issue in this dissertation is how to deliver facial animation across Internet efficiently. The international standard MPEG-4 [MPEG99], which tries to standardize both natural and synthetic media, includes synthetic faces as a part of visual objects. In the standard, the head model parameters and the controls of facial expressions are defined as a set of MPEG-4 Face Definition Parameters (FDPs) and MPEG-4 Face Animation Parameters (FAPs) respectively.

In the dissertation, a talking head based on MPEG-4 FDPs, FAPs is proposed. We use plug-ins techniques for web browsers to transmit vivid facial animation in a very low bit rate.

1.3 Contribution

This dissertation presents procedures to estimate facial motion accurately from real subjects' faces with off-the-shelf and inexpensive devices. Therefore, we propose extracting 3D facial animation parameters from mirror-reflected multi-view video clips.

Two lighting conditions are used for different requirements of data. In the normal light condition, our tracking system is semi-automatic. This is because the markers' projected colors and skin colors can vary significantly due to different reflective angles, and manual adjustment is required occasionally. However, to acquire the correct facial texture, capturing under normal light is necessary. A derivate research about facial animation with detailed expression mapping is proposed by P.-S. Tu [TU2003].

In order to estimate 3D facial motion fully automatically, a different procedure is proposed for tracking under blacklight-UV lamps. We employ UV-responsive fluorescent markers, and the feature point detection becomes easier and more accurate. In our current experiment, we can fully automatically track 300 markers over 9.2 frames per second on a Pentium 4 3.0G Hz PC, and it is capable of tracking more than 100 markers in real time from live video.

The following are my major contributions:

- **A closed-form linear algorithm for 3D position reconstruction from mirror-reflected multi-view video clips is proposed.**

  Without measurement or calibration of mirrors' positions in advance, the proposed linear algorithm can first reconstruct mirrors' location and orientations from a set of real vs. mirrored point correspondences. 3D positions of feature points can then be evaluated.

- **Comparison and discussion of the proposed method with general-purpose two-view approach is presented.**

  Since nice symmetric properties of mirrored objects are applied, the proposed method is more robust, more accurate and simpler than general two-view approaches for 3D structure reconstruction. Four sets of computer simulation experiments are done and expected errors are estimated theoretically to prove the benefit of the proposed method. The advantages and disadvantages of the two approaches are also discussed.

- **A procedure for 3D motion tracking under a normal light condition is proposed.**

  Adaptive Kalman filter is utilized to improve the stability of marker tracking and an interactive graphical user interface is also provided for manual adjustment.

- **An efficient procedure for automatically tracking a large number of markers under a blacklight-UV light condition is proposed.**

  The proposed procedure can detect and handle the marker false detection and missing marker problems fully automatically during motion tracking. To our best knowledge, there are no other video-based systems up to now that tracked more than 200 facial markers in 3D. Furthermore, seldom systems can automatically capture a large quantity of 3D facial motion trajectories with only a regular PC and a camera.

- **A web-enabled talking head is proposed.**

  The proposed system can "stream" facial animation on Internet with only 6K bits per second and the web-enabled talking head has been further

licensed and improved by Cyberlink, Corp. as a commercial product "Talking show" in 2000 [TALK].

1.4 Overview and Organization

In this dissertation, a complete procedure from (i)3D position and motion estimation, (ii)marker tracking in video clips to (iii)facial animation is proposed. First of all, chapter 2 presents state-of-the-art researches and publications in related areas, which are composed of 3D structure reconstruction, motion tracking and human face synthesis. The benefits and drawbacks of methods derived from different conceptions are also mentioned.

Then, components of the procedure are proposed in following chapters. Figure 1.5 is a flow chart of the proposed work, and it also manifests correlations between chapters. The complete procedure and its components are briefly introduced as follows.

Chapter 3 is the core of the proposed work, where we deduce and propose an algorithm that estimates 3D position from real versus mirrored-point correspondence. Since position extraction of projected markers cannot be entirely exact, measurement noise will degrade the results produced by the algorithm. To evaluate effects of the measurement noise, expected error of the proposed algorithm is calculated theoretically in Section 3.3.

The facial motion tracking procedures are proposed for two lighting conditions. Chapter 4 describes how we semi-automatically track markers' 3D trajectories under a normal light condition. The left tracking flow in Figure 1.3 shows this procedure. The concept and the equipment setup and are specified in Section 4.1 and 4.2, and block-matching-based feature extraction is presented in section 4.3. At last, this section also proposed our approach to estimate facial motion trajectories from 3D candidate sequences reconstructed by the algorithm proposed in Chapter 3.

Chapter 5 presents a fully automatic procedure for tracking a large quantity of fluorescent markers under the blacklight ultraviolet (UV) lighting condition, and the process corresponds to the right tracking flow in Figure 1.3. Similar to the procedure for the normal light condition, Section 5.1 introduces the concept and Section 5.2

mentions equipment setting and the feature extraction process. Since fluorescent markers can emit luminescence when they are illuminated by UV fluorescent light, these special markers are prominent in video. Therefore, the process of fluorescent marker extraction can be more reliably achieved by general computer vision methodology, including conditioning, color labeling, grouping, extracting, etc. When tracking such numerous markers, false alarms and missing problems are much more seriously. How to utilize spatial and temporal correlations of the numerous markers for fully automatic tracking is proposed in Section 5.3.

To compare the proposed method with general-purpose stereovision approaches, we conducted computer simulations and actual experiments for different control factors. The experiment results and discussions are presented in Chapter 6.

In Chapter 7, our work for face synthesis is proposed. We propose a speech-driven facial animation system that animates based on a small set of prototype facial motion parameters in Section 7.2. The system is further developed as a low-bit-rate talking head to efficiently stream facial animation over Internet. Section 7.3 mentions how we construct a synthetic face cloning a real one. The way we use estimated 3D FAPs to drive facial animation is presented in Section 7.4.

Finally, Chapter 8 concludes my research and the future work is described.

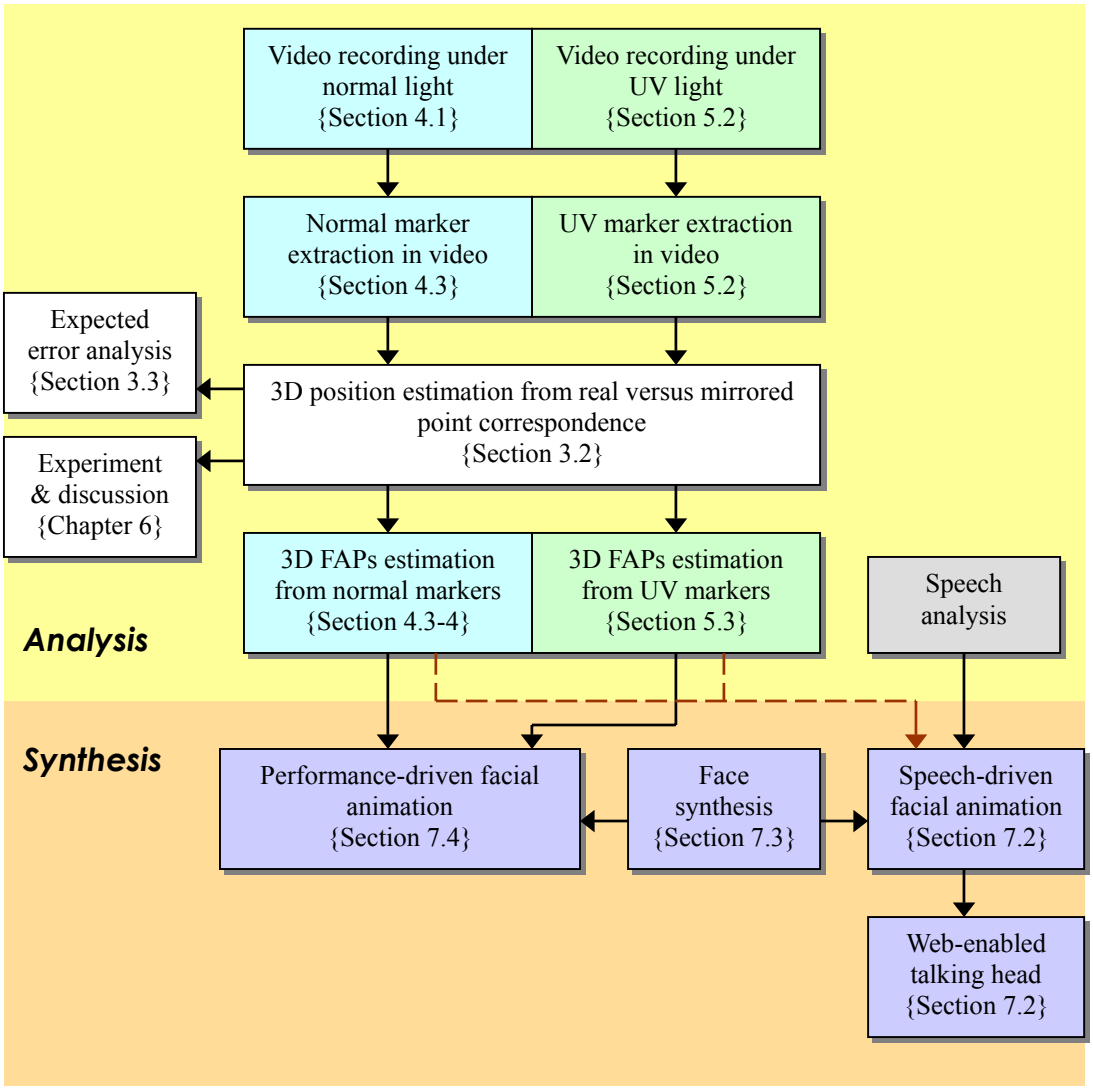


Figure 1.3 The flow chart of the proposed work on analysis and synthesis of realistic 3D facial animation. The corresponding chapter or section of each component is also annotated.

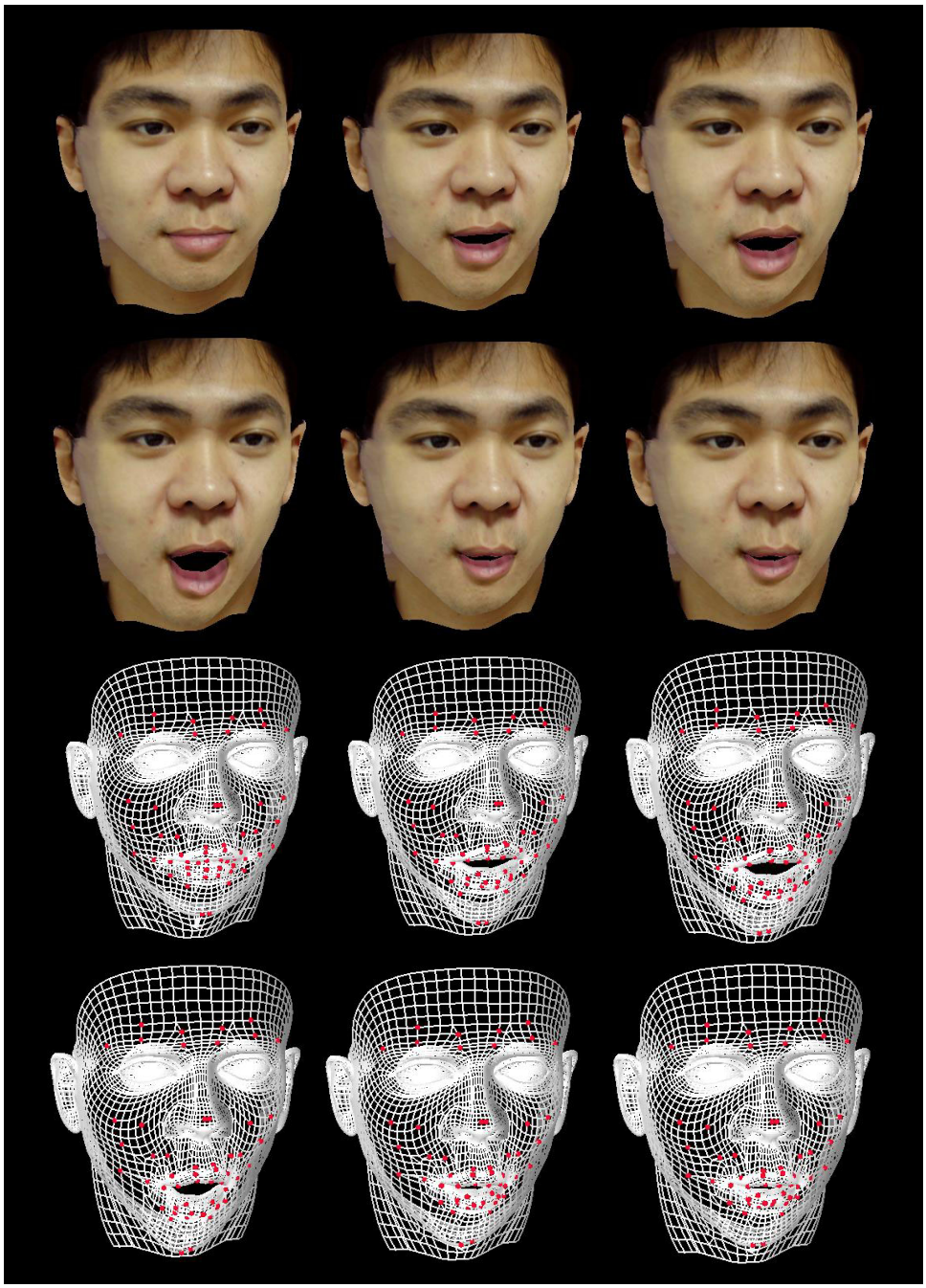


Figure 1.4 The 3D facial motion trajectories estimated with the proposed algorithm for realistic facial animation. The red points in the lower part of the figure represent the estimated markers' 3D positions, and the upper part depicts synthesized facial animation of the pronunciation "O-U".

Chapter 2

Related work

2.1 Introduction

Since the proposed work comprises techniques of 3D structure reconstruction, facial motion estimation and face synthesis, some state-of-the-art researches and statuses in these three domains are introduced in this chapter.

2.2 3D Structure Reconstruction from multiple views

3D structure reconstruction is an essential process for 3D computer vision, e.g. 3D object modeling, 3D object recognition, and 3D motion tracking.

Multiple view directions are required to reconstruct 3D structure from images. Most of modern stereovision-based 3D structure estimation approaches derive from epipolar constraints. These approaches first use corresponding points in images of different viewpoints to estimate the essential matrix. Then, the rotation R and translation t between cameras are decomposed from the essential matrix. Finally, each point's 3D position can be estimated by intersecting casting vectors from the cameras' optic center. [LONG81, ZHAN92, ZHAN95, WENG93, HUAN94] provide a good reference or discussion on estimating 3D structure or the essential matrix from images.

Using multiple cameras simultaneously is a common approach to acquire multiple views. Since each camera can have different parameters, this will cause

different distortions on each image. Therefore, before reconstruction methods mentioned above are performed, captured images have to be undistorted and normalized in advance. Generally, two kinds of camera parameters, *extrinsic parameters* and *intrinsic parameters*, are mainly concerned. Extrinsic parameters are the rotation and translation that related the world coordinate to the camera coordinate system; intrinsic parameters, also called the camera model, comprise lens distortions, the focal length, the principal point, etc. R.Y. Tsai's method provides a paradigm in camera calibration [TSAI87]. Recently, J. Heikkilä and O. Silvén's camera calibration procedure [HEIK97, BOUG] and Z. Zhang's flexible calibration method [ZHAN00] are widely utilized.

Besides multiple cameras, placing mirrors in a scene is another way to acquire multiple views. A.R.J. François et al [FRAN03] proved that 3D reconstruction from a single perspective view of a mirror symmetric scene is geometrically equivalent to reconstructing the scene with two cameras symmetric to the unknown 3D symmetry plane. The advantage of utilizing mirrored views is that only one camera is necessary, and errors caused by imperfect calibration between cameras can be avoided. However, the locations or orientations of mirrors have to be estimated in this case.

In the research proposed by H. Mitsumoto et al. [MITS92], they recovered the plane symmetry using the vanish point. D.Q. Huynh [HUYN99] proposed an affine reconstruction method from a monocular view with a symmetry plane. In his method, he reconstructed a 3D object via solving the epipole with a symmetry plane constraint. In our proposed method, 3D positions are reconstructed via estimating the mirror plane from projected corresponding points.

Even though, Huynh's and our method took different points of views in the beginning, we found that Huynh's solution for the restricted epipole is equivalent to our mirror plane estimation. Huynh's work focused on the problem solving of 3D reconstruction with a symmetry plane and discussed the advantage of non-linear computation. On the other hand, our work not only estimates 3D positions but also take advantage of the perfectly-synchronized property between multiple mirrored views to track 3D facial motion. Furthermore, we did computer simulations and deduced the theoretical expected errors to manifest the outstanding benefits of 3D position and motion estimation from mirror-reflected video clips compared to two-view algorithms, where multiple cameras are applied. We also discuss the

advantages and disadvantages between the proposed method and two view approaches.

In addition to plane mirrors, J. Guckman and S.K. Nayar [GLUC99, GLUC02] presented stereo sensors using a single camera with various combinations of mirrors, e.g. two spherical mirrors, two convex mirrors, and four planar mirrors.

2.3 Facial Motion Tracking

Depending on applications, different devices and sensors are used for facial motion tracking. T. Goto et al. [GOTO01] proposed a simple procedure to roughly extract motion of feature points on a bare face from video. FaceStation developed by Eyematic Interface Inc. [FACE] can also automatically locate and track facial features from a video camera. These kinds of systems can provide user-friendly interfaces for exaggerated and expressive facial animation. Nevertheless, while an application requires accurate 3D facial motion or requires motion of points besides distinct facial features, e.g. points on cheeks or on the forehead, conspicuous markers are usually necessary to adhere to a subject's face.

About 3D facial motion tracking from multiple cameras, an optoelectronic system, e.g. Optotrak [OPTO, HAVE96], uses optoelectronic cameras to track infrared-emitting photodiodes on a subject's face. Since the root mean square (RMS) error of this system can be as low as 0.1mm in horizontal and vertical and 0.15mm in depth, such an instrument suffices for research demanding high accuracy such as facial biomechanics or co-articulation effect analysis. However, each diode needs to be powered by wires, which may interfere with a subject's facial motion.

Video-based systems that apply passive markers avoid this problem. For example, the VICON series [VICO] uses six to 24 specifically designed cameras with resolution 1280x1024 pixel^2 and frame rate 60 to 1000 fps to capture markers' motion in visible or infrared spectrums. This kind of costly motion capture system is popular in the computer graphics industry for movies or video games. They usually make use of protruding spherical markers for easiness of shape analysis, but these markers don't work well for lip surface motion tracking because people sometimes tuck in or otherwise obstruct lip surfaces. Besides, the extracted motion of protruding markers is not the exact motion on a face surface but the motion at a

small distance above the surface.

In addition to capturing stereo videos with multiple cameras, E.C. Patterson et al. [PATT91] proposed using mirrors to acquire multiple views for facial motion recording. They simplified the 3D reconstruction problem and assumed a plumb camera and vertical mirrors. S. Basu et al. [BASU97, BASU98] employed a front view and a mirrored view to capture 3D lip motion. In their work, they regarded the mirrored view as a flipped image of a virtual camera and applied a general-purpose stereovision approach to estimating 3D lip motion. We also apply mirrors for acquirement of new images with different view directions. However, Our algorithm proves simpler yet more accurate because it conveniently uses nice symmetric properties of mirrored objects.

Some devices and researches take other concepts to estimate 3D motion or structure. For example, the ShapeSnatcher system [SHAP, KALB01] projects grids onto a face, and therefore it can extract 3D shape and texture from a single image.

2.4 Human face synthesis

In general, the framework of a synthetic face, the controls of facial expression, and the driven events for facial animation are three principal considerations in a facial animation system.

Researches for the framework of a synthetic face can be approximately classified into three categories: 2D-mesh-based, 3D-polygon-based, and image-sample-based. The 2D-mesh-based approach is the most easily controlled and computationally effective design. Only a single image texture and a face mesh are required to construct a synthetic face [PERN98]. The main disadvantage is that the view directions of a 2D face are limited and it is difficult to be combined into a 3D graphical environment. Most researches adopted the 3D-polygon-based approach to avoid problems mentioned above. Modeling and controlling a 3D face is much more delicate. Laser scanners such as those produced by Cyberware Corp. [CYBE] can acquire a precise 3D face shape with texture mapping. W. Lee et al. [LEE99] applied a semi-automatic approach, which modeled a 3D face model based on the orthogonal view images of a person. In F. Pighin and others' work [PIGH98], photographs taken from different view directions were integrated to construct a

delicate face model with view-dependent texture mapping. V. Blanz et al. [BLAN99] established an excellent system to build a personalized 3D head model from only a single face image by statistic information of human heads.

Image-sample-based systems synthesize faces and facial expressions by metamorphing between several photographic images. The morphing technique proposed by T. Beier and S. Neely [BEIE92] made an impressive animation of transitions between different faces in Michael Jackson's MTV "Black or White". "Video Rewrite" proposed by C. Bregler et al. [BREG97] synthesized video realistic facial animation by combining image samples of a face and mouth according to input phonemes. E. Cosatto et al. [COSA00] further decomposed the samples into smaller facial parts and formed a sample space. These made the synthetic process with more flexibility and efficiency. Up to the present, the image-sample-based approach could be the most realistic one among all the approaches, but it suffers the same disadvantages of the 2D-mesh-based approach, where the view directions are limited. This problem can be solved by the view morphing technique [SEIT96]. Nevertheless, a large database of image samples or heavy computation are indispensable. In addition, it is difficult to apply the sample data to others' faces.

For facial expression synthesis, a muscle-based approach imitates anatomy of human faces, which controls expressions by adjustment of interior muscles. K. Waters [WATT87] developed a dynamic face model with linear muscles and sphincter muscles. In D. Terzopoulous and K. Waters' research [TERZ90], a face tissue model of a three-layer structure was proposed to simulate skin, subcutaneous tissue, and muscles. The muscle-based approach is conformed to the facial action coding system (FACS) [EKMA78] and is suitable to model exaggerative expressions. From F.I. Parke and K. Waters' words [PARK96] "FACS seems complete for reliably distinguishing actions of the brows, forehead, and eyelids. FACS does not include all of the visible, reliably distinguishable actions of the lower part of the face", human faces and lips are so subtle that an approximate model can still hardly simulate many fine variations.

Since the exterior face shape is the main concern in computer animation, the feature-point driven approach simulates facial expression by controlling the feature point on a synthetic face surface directly. This kind of approach assumes the exterior face shape as several parametric surfaces, such as bicubic surfaces [REEV90], or radial-basis functions [NIEL93, PIGH98]. The advantage of feature-point driven

facial expression is that facial expressions can be generated intuitively from motion captured data or manual adjustment. On the other hand, it requires a lot of control points to synthesize those subtle facial motions.

As mentioned in Chapter 1, performance-driven, text-driven and speech-driven facial animations are three major approaches to drive synthetic faces. The performance-driven approach synthesizes facial animation directly from captured motion data. Guenter et al. [GUEN98] produced a remarkably lifelike facial animation by abundant motion captured information. They recorded 182 dot markers' positions and facial textures on a subject's face on 30 frames per second. A text-driven talking head translates each input word into visemes and performs animation by interpolating visemes according to input time stamps. To produce voices of a text-driven talking head, a text-to-speech (TTS) module synthesizes auditory speech according to time stamps synchronously. A speech-driven face system is quite similar to the text-driven one but translates visemes from input natural speech instead. The benefit of a speech-driven face system is that it uses natural speech as output voice and does not suffer the unnatural synthetic voice as that in the text-driven one. However, the faithfulness of facial animation depends on how accurate the input speech is recognized. A common problem in text-driven and speech-driven facial animation is the motion transition function between visemes. M.M. Cohen and D.W. Massaro [COHE93] introduced several hypotheses. The voice puppetry proposed by Brand [BRAN99] further applied the Hidden Markov Model (HMM) to approximate facial motions driven by various audio features. T. Ezzat et al. [EZZA02] proposed a multidimensional morphable model (MMM) to synthesize novel facial motion trajectories based on a small set of prototypes.

There are some other related researches on synthetic faces. A wavelet-based method for prototyping facial textures and transforming the age of facial images is presented by B. Tiddeman et al. [TIDD01]. Z. Liu et al. [LIU01] proposed synthesizing delicate details on a face with expression ration images (ERI). Noh and Neumann [NOH01] presented a method to retarget facial motions and preserve the relative features of original facial animation.

Chapter 3

3D Position Estimation from Mirror-reflected Multi-view video

3.1 The Problem Statement

As the conceptual diagram shown in Figure 3.1, a mirrored image can be regarded as a "flipped" image taken by a "virtual camera", which is in a distinct view direction comparing to the real one. For the detailed proof of the relation between real and virtual cameras, we refer to the reference [FRAN03]. With two mirrors next to a subject's face, we can simultaneously acquire three facial images from different viewpoints and can also avoid the problem of data synchronization among different cameras.

Before we use these images for 3D position estimation, orientations and locations of mirror planes must be estimated in advance. Some research [ZHAN98] required explicit measuring these properties. Explicit measurement is not user-friendly and reliable, or precise devices must be employed. Hence, accurate methods to directly handle the whole process from image sequences are necessary.

In some related researches [BASU97, BASU98], 3D positions of the aforementioned situation were estimated by modified general-purpose stereovision approaches, which estimate the affine transformation (rotation $\boldsymbol{R}$, translation $\boldsymbol{T}$) between two cameras from the essential matrix [LONG81, WENG89, WENG93, HAR95]. After evaluating the rotation and translation between two cameras, the 3D position of a target can then be approximated from intersection of cast rays from optical centers of different cameras.

However, there are some nice properties of mirrored images that can be utilized to deduce a more accurate 3D reconstruction algorithm for mirror-reflected multi-view images. We present our approach in the following section.

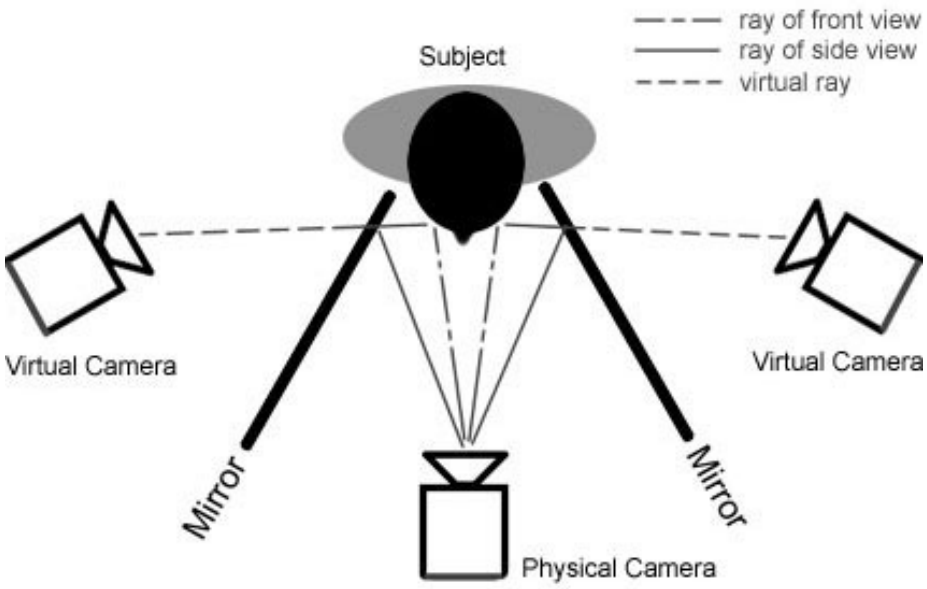


Figure 3.1 The conceptual diagram of "virtual cameras". Properties of a virtual camera, including intrinsic and extrinsic parameters, are symmetric to those of an actual camera with respect to a mirror plane.

3.2 The Proposed Closed-form Linear Algorithm

In this section, we introduce our algorithm for 3D position estimation in condition of one mirror, which can be easily extended to two mirrors' condition. We assume that input images have been normalized by camera calibration processes [HEIK97, BOUG, ZHAN00], and we also assume that real and mirrored markers' projected positions and correspondences have been extracted. The details of marker extraction, tracking and recovering point correspondences under normal light and blacklight conditions are presented in Chapter 4 and 5.

With the point correspondences, now, we can calculate markers' 3D positions by first evaluating mirrors' orientations and locations in the camera coordinate

system and then estimating markers' 3D positions as a minimization problem.

In the first step, we assume plane mirrors and use only the image data within the mirrors' range. A mirror's location and orientation can be represented using a plane equation:

$$ax + by + cz = d \tag{3.1}$$

$\boldsymbol{u} = (a, b, c)^t$, $\|\boldsymbol{u}\| = 1$, where u is the plane's unit normal and vector $\boldsymbol{u}$ has two possible directions. Without loss of generality, we take the direction of $c < 0$. In the following discussion, we assume that I is the camera film's image plane and f is the focal length. (If images are undistorted and normalized according to the normalized camera model, f is 1.0.) We assume the camera's lens center O to be the origin in the coordinate system, and the camera's line of vision, also called the optic axis, is the positive z axis.

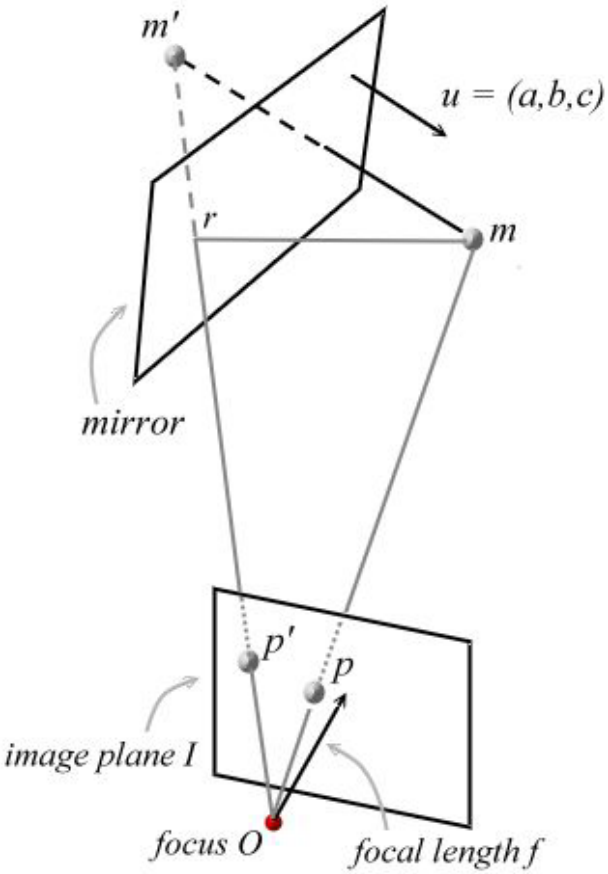


Figure 3.2 The geometric representation of a physical point m, the reflected virtual point m', and the projection points p, p'.

In Figure 3.2, m_i is the actual 3D position of marker i, $m_i = (x_{mi}, y_{mi}, z_{mi})^t$, and m_i' is the 3D position of virtual marker i in the mirrored space, $m_i' = (x'_{mi}, y'_{mi}, z'_{mi})^t$.

p_i is the projection of m_i on I, $p_i = (f\frac{x_{mi}}{z_{mi}}, f\frac{y_{mi}}{z_{mi}}, f)^t = (x_{pi}, y_{pi}, z_{pi})^t$,

p_i' is the projection of m_i' on I, $p_i' = (f\frac{x'_{mi}}{z'_{mi}}, f\frac{y'_{mi}}{z'_{mi}}, f)^t = (x'_{pi}, y'_{pi} z'_{pi})^t$,

where (x_{pi}, y_{pi}) and (x'_{pi}, y'_{pi}) are the estimated markers' 2D positions.

Mirror properties dictate that

$$m_i' = m_i + ku \tag{3.2}$$

where k is a scale value. Vectors m_i, m_i', $\boldsymbol{u}$ are coplanar, and thus

$$m_i' \cdot (u \times m_i) = 0 \tag{3.3}$$

" • " is the dot product and "×" is the cross product.

From Equation 3.3, we reformulate in terms of p_i, p_i',

$$\frac{z'_{mi}}{f} p_i' \cdot \left[u \times \left(\frac{z_{mi}}{f} p_i \right) \right] = 0 \tag{3.4}$$

and simplify it as

$$(p_i')^t U p_i = 0, \text{ where } U = \begin{bmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{bmatrix}. \tag{3.5}$$

We can then represent Equation 3.5 in terms of $\boldsymbol{u}$ as

$$\begin{bmatrix}(y_{pi}-y'_{pi})f & (-x_{pi}+x'_{pi})f & (x_{pi}y'_{pi}-y_{pi}x'_{pi})\end{bmatrix}\begin{bmatrix}a\\b\\c\end{bmatrix}=0 \tag{3.6}$$

By collecting Equation 3.6 for each marker correspondence, we can form a matrix $\boldsymbol{M}$,

$$Mu = 0, \text{ where } M=\begin{bmatrix}(y_{p1}-y'_{p1})f & (-x_{p1}+x'_{p1})f & (x_{p1}y'_{p1}-y_{p1}x'_{p1})\\(y_{p2}-y'_{p2})f & (-x_{p2}+x'_{p2})f & (x_{p2}y'_{p2}-y_{p2}x'_{p2})\\\vdots & \vdots & \vdots\\(y_{pn}-y'_{pn})f & (-x_{pn}+x'_{pn})f & (x_{pn}y'_{pn}-y_{pn}x'_{pn})\end{bmatrix} \tag{3.7}$$

The mirror might not be perfectly flat, however, and we should also allow for noise in marker shape and position on image plane *I*. We therefore apply the least square method to estimate the vector $\boldsymbol{u}$ with the least error. It's well known that the solution of

$$\min_u \|Mu\|, \text{ subject to } \|u\|=1 \tag{3.8}$$

is the eigenvector corresponding to the smallest eigenvalue of the matrix $\boldsymbol{M^tM}$.

Another mirror property is symmetry:

$$(m'_i-\Theta)=H_u(m_i-\Theta) \tag{3.9}$$

where Θ is an arbitrary point on the mirror plane ***Mirror***; $H_u=(I_{3x3}-2uu^t)$ is the Householder matrix, and $\boldsymbol{I_{3\times3}}$ is the identity matrix. We choose $\Theta=(0,0,\frac{d}{c})^t$ and deduce the equation

$$\begin{bmatrix}\left(\frac{2a^2-1}{2f}\right)x_{pi}+\left(\frac{ab}{f}\right)y_{pi}+ac & \frac{x'_{pi}}{2f}\\\left(\frac{ab}{f}\right)x_{pi}+\left(\frac{2b^2-1}{2f}\right)y_{pi}+bc & \frac{y'_{pi}}{2f}\\\left(\frac{ac}{f}\right)x_{pi}+\left(\frac{bc}{f}\right)y_{pi}+\frac{2c^2-1}{2} & \frac{1}{2}\end{bmatrix}\begin{bmatrix}z_{mi}\\z'_{mi}\end{bmatrix}=d\begin{bmatrix}a\\b\\c\end{bmatrix} \tag{3.10}$$

From Equation 3.10, we see that once we have determined vector $\boldsymbol{u}$, z_{mi} and z'_{mi} are proportional to variable d. The value d can be determined by comparing estimated data with a reference ruler in the real world.

Thus, along the above-mentioned steps, unit vector $\boldsymbol{u}$ is first estimated by Equation 3.8, and then the position $[x_{mi}, y_{mi}, z_{mi}]^t$ of each marker or feature point can be reconstructed from the depth information solved by a least square method in the form

$$\min_z \|Gz - du\|, \text{ where } z = (z_{mi}, z'_{mi})^t \tag{3.11}$$

based on singular value decomposition (SVD) or QR factorization. We refer readers unfamiliar with numerical matrix computation to the reference [GOLU96].

Moreover, to reduce the influence of marker position estimation errors in the front view image, we simply mirror the virtual marker m'_i back to the actual world, set as m''_i,

$$m''_i = H_u^{-1}(m'_i - \Theta) + \Theta \tag{3.12}$$

and take $m'''_i = \dfrac{(m_i + m''_i)}{2}$ as the 3D position of marker i.

To more accurately estimate m''', we can also apply nonlinear maximum likelihood optimization that minimizes the location variation on an image plane to improve the estimated mirror normal $\boldsymbol{u}$. However, a mirror plane's useful properties mean the vector $\boldsymbol{u}$ estimated by a linear algorithm is sufficiently accurate. In our simulation, maximum likelihood optimization improved less than 2 percent of the root mean square (RMS) 3D position error under quite noisy circumstances.

The whole algorithm is organized as follows:

The proposed algorithm

Note: real vs. mirrored point positions and correspondences are extracted in advance. The reconstructed 3D positions will be more accurate if the point positions are undistorted and normalized by camera calibration processes. (f=1, if the normalization is applied)

■ **Mirror plane *u* estimation (*ax+by+cz=d*)**

Known: (x_{pi}, y_{pi}) and (x'_{pi}, y'_{pi}) are the estimated positions of marker *i* in real and mirrored images.

Unknown: *a*, *b*, *c*

1. Form matrix ***M***,

$$M = \begin{bmatrix} (y_{p1} - y'_{p1})f & (-x_{p1} + x'_{p1})f & (x_{p1}y'_{p1} - y_{p1}x'_{p1}) \\ (y_{p2} - y'_{p2})f & (-x_{p2} + x'_{p2})f & (x_{p2}y'_{p2} - y_{p2}x'_{p2}) \\ \vdots & \vdots & \vdots \\ (y_{pn} - y'_{pn})f & (-x_{pn} + x'_{pn})f & (x_{pn}y'_{pn} - y_{pn}x'_{pn}) \end{bmatrix}$$

2. $\min_u \|Mu\|$, subject to $\|u\| = 1$

$u = (a, b, c)^t$

■ **3D position estimation from real vs. mirrored point correspondence**

Unknown: d, z_{mi} and z'_{mi} depth of actual and virtual marker *i*.

Take an initialization of d_{tmp}. (d_{tmp} = 8.0 in our setting)

1. For each marker *i*,

$$\min_z \|Gz - d_{tmp}u\|, \text{ where } z = (z_{mi}, z'_{mi})^t$$

$$G = \begin{bmatrix} \left(\frac{2a^2-1}{2f}\right)x_{pi} + \left(\frac{ab}{f}\right)y_{pi} + ac & \frac{x'_{pi}}{2f} \\ \left(\frac{ab}{f}\right)x_{pi} + \left(\frac{2b^2-1}{2f}\right)y_{pi} + bc & \frac{y'_{pi}}{2f} \\ \left(\frac{ac}{f}\right)x_{pi} + \left(\frac{bc}{f}\right)y_{pi} + \frac{2c^2-1}{2} & \frac{1}{2} \end{bmatrix}$$

Evaluate m_i and m'_i, $m_i = (\frac{z_{mi}}{f}x_{pi}, \frac{z_{mi}}{f}y_{pi}, z_{mi})$,

$m'_i = (\frac{z'_{mi}}{f}x'_{pi}, \frac{z'_{mi}}{f}y'_{pi}, z'_{mi})$

2. Estimate m''', a refinement of m :
   For each marker i,

   $$m_i'' = H_u^{-1}(m_i' - \Theta) + \Theta,$$

   where $H_u = (I_{3x3} - 2uu^t)$, and $\Theta = (0,0,\frac{d_{tmp}}{c})^t$

   $$m_i''' = \frac{m_i + m_i''}{2}$$

3. Estimate d,
   calculate the scale $s = \frac{A \text{ measured from calibration objects}}{A \text{ estimated from } m'''}$
   where A can be the perimeter of specific markers on calibration objects, or average distance between specific markers, etc.

   $d = s \times d_{tmp}$.

4. For each marker i,
   Its estimated 3D position is $s \times m_i'''$

Note: for facial animation, a scaled position is sufficient. Step 3 and 4 can be ignored.

3.3 Error Estimation for the Algorithm

To evaluate stability of an algorithm, magnitude of output estimation errors according to input source errors is a useful criterion. In this section, we try estimate the expected error of the proposed algorithm theoretically. Error estimation by computer simulations is then presented in Chapter 6.

In our proposed algorithm, inputs are projected point positions and point correspondence. Input source errors, therefore, include feature detection errors, spatial quantization errors, point mismatching and camera miscalibration. Among them, feature detection errors can result from variation due to projection, as

mentioned in Section 1.2, or inaccuracy of tracking methods; the digitization process of a CCD camera causes spatial quantization errors. We sum up these errors and model them as random variables with Gaussian distributions.

For error estimation, we follow the work proposed by J. Weng et al. [WENG89, WENG93], where the expected error is calculated instead of a worst case bound. That is because their algorithm and ours both apply a large amount of data (>30 point pairs) in a least square manner and the variance of the error distribution in the solution is small comparing to the image coordinate system. The worst case bound is too large and it is almost never reached.

■ **The expected error in mirror plane *u* estimation**

We assume that the camera have been calibrated therefore, $f = 1$. n is the number of point corresponding pairs.

Algorithm:

$\min_u \|Mu\|$, subject to $\|u\| = 1$,

where $M = \begin{bmatrix} (y_{p1} - y'_{p1}) & (-x_{p1} + x'_{p1}) & (x_{p1}y'_{p1} - y_{p1}x'_{p1}) \\ (y_{p2} - y'_{p2}) & (-x_{p2} + x'_{p2}) & (x_{p2}y'_{p2} - y_{p2}x'_{p2}) \\ \vdots & \vdots & \vdots \\ (y_{pn} - y'_{pn}) & (-x_{pn} + x'_{pn}) & (x_{pn}y'_{pn} - y_{pn}x'_{pn}) \end{bmatrix}_{n\times 3}$

$\boldsymbol{u} = (a, b, c)$ is the eigenvector corresponding to the smallest eigenvalue of $\boldsymbol{M^tM}$

The expected error:

We denote $x_{pi}(\varepsilon)$, $y_{pi}(\varepsilon)$, $x'_{pi}(\varepsilon)$, $y'_{pi}(\varepsilon)$ are noise-corrupted inputs to the algorithm.

$x_{pi}(\varepsilon) = x_{pi} + \delta_{xpi}$, where x_{pi} is the actual projected position of marker i, and δ_{xpi} is the noise; as well, $y_{pi}(\varepsilon) = y_{pi} + \delta_{ypi}$, $x'_{pi}(\varepsilon) = x'_{pi} + \delta_{x'pi}$, $y'_{pi}(\varepsilon) = y'_{pi} + \delta_{y'pi}$.

Since the noise δ_{xpi}, δ_{ypi}, $\delta_{x'pi}$, $\delta_{y'pi}$, etc. are small perturbation, we ignore

the higher order terms. In the following deduction, only the first order perturbation are taken into account and we use the sign "≅" to indicate that it is a linear approximation.

The first step is to calculate $\Delta_M = M(\varepsilon) - M$, which is noise perturbation of the matrix ***M***. Among $M(\varepsilon)$, the third column is

$$x_{pi}(\varepsilon)y'_{pi}(\varepsilon) - y_{pi}(\varepsilon)x'_{pi}(\varepsilon)$$

$$= (x_{pi} + \delta_{xpi})(y'_{pi} + \delta_{y'pi}) - (y_{pi} + \delta_{ypi})(x'_{pi} + \delta_{x'pi})$$

$$= (x_{pi}y'_{pi} + x_{pi}\delta_{y'pi} + \delta_{xpi}y'_{pi} + \delta_{xpi}\delta_{y'pi}) - (y_{pi}x'_{pi} + y_{pi}\delta_{x'pi} + \delta_{ypi}x'_{pi} + \delta_{ypi}\delta_{x'pi})$$

$$\cong (x_{pi}y'_{pi} - y_{pi}x'_{pi}) + \left[(x_{pi}\delta_{y'pi} + \delta_{xpi}y'_{pi}) - (y_{pi}\delta_{x'pi} + \delta_{ypi}x'_{pi})\right] \tag{3.13}$$

After rearrangement, we get

$$\Delta_M = \begin{bmatrix} (\delta_{yp1} - \delta_{y'p1}) & (-\delta_{xp1} + \delta_{x'p1}) & (x_{p1}\delta_{y'p1} + \delta_{xp1}y'_{p1} - y_{p1}\delta_{x'p1} - \delta_{yp1}x'_{p1}) \\ (\delta_{yp2} - \delta_{y'p2}) & (-\delta_{xp2} + \delta_{x'p2}) & (x_{p2}\delta_{y'p2} + \delta_{xp2}y'_{p2} - y_{p2}\delta_{x'p2} - \delta_{yp2}x'_{p2}) \\ \vdots & \vdots & \vdots \\ (\delta_{ypn} - \delta_{y'pn}) & (-\delta_{xpn} + \delta_{x'pn}) & (x_{pn}\delta_{y'pn} + \delta_{xpn}y'_{pn} - y_{pn}\delta_{x'pn} - \delta_{ypn}x'_{pn}) \end{bmatrix}$$

(3.14)

and

$$\Delta_{M^t} = M^t(\varepsilon) - M^t$$

$$= \begin{bmatrix} (\delta_{yp1} - \delta_{y'p1}) & (\delta_{yp2} - \delta_{y'p2}) & \cdots & (\delta_{ypn} - \delta_{y'pn}) \\ (-\delta_{xp1} + \delta_{x'p1}) & (-\delta_{xp2} + \delta_{x'p2}) & \cdots & (-\delta_{xpn} + \delta_{x'pn}) \\ (x_{p1}\delta_{y'p1} + \delta_{xp1}y'_{p1} & (x_{p2}\delta_{y'p2} + \delta_{xp2}y'_{p2} & \cdots & (x_{pn}\delta_{y'pn} + \delta_{xpn}y'_{pn} \\ -y_{p1}\delta_{x'p1} - \delta_{yp1}x'_{p1}) & -y_{p2}\delta_{x'p2} - \delta_{yp2}x'_{p2}) & & -y_{pn}\delta_{x'pn} - \delta_{ypn}x'_{pn}) \end{bmatrix}_{3\times n}$$

To calculate the expectation of matrix Δ_{M^t}, we have to calculate expected

values of all the elements in the matrix. For convenience of deduction and presentation, we use $(\Delta_{M^t})_j$ to denote the j^{th} column of the matrix Δ_{M^t} and use suffixes, such as $m\times n$ to represent the dimension of the matrix or vector. Therefore, $\Delta_{M^t} = \left[(\Delta_{M^t})_1 \quad (\Delta_{M^t})_2 \quad \cdots \quad (\Delta_{M^t})_n\right]$. We also rearrange matrix Δ_{M^t} and form a vector $\psi_{\Delta M^t}$ that contains all its elements as:

$$\psi_{\Delta M^t} = \begin{bmatrix} (\Delta_{M^t})_1 \\ (\Delta_{M^t})_2 \\ \vdots \\ (\Delta_{M^t})_n \end{bmatrix}_{3n\times 1}.$$

$$\psi_{\Delta M^t}(\psi_{\Delta M^t})^t = \begin{bmatrix} (\Delta_{M^t})_1 \\ (\Delta_{M^t})_2 \\ \vdots \\ (\Delta_{M^t})_n \end{bmatrix} \left[(\Delta_{M^t})_1^t \quad (\Delta_{M^t})_2^t \quad \cdots \quad (\Delta_{M^t})_n^t\right]$$

$$= \begin{bmatrix} (\Delta_{M^t})_1(\Delta_{M^t})_1^t & \cdots & (\Delta_{M^t})_1(\Delta_{M^t})_j^t & \cdots & (\Delta_{M^t})_1(\Delta_{M^t})_n^t \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ (\Delta_{M^t})_i(\Delta_{M^t})_1^t & \cdots & (\Delta_{M^t})_i(\Delta_{M^t})_j^t & \cdots & (\Delta_{M^t})_i(\Delta_{M^t})_n^t \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ (\Delta_{M^t})_n(\Delta_{M^t})_1^t & \cdots & (\Delta_{M^t})_n(\Delta_{M^t})_j^t & \cdots & (\Delta_{M^t})_n(\Delta_{M^t})_n^t \end{bmatrix}_{3n\times 3n} \tag{3.15}$$

Assume that the errors δ_{xpi}, δ_{ypi}, $\delta_{x'pi}$, $\delta_{y'pi}$ between different points and different views in the image coordinates are uncorrelated and they are zero-mean and have the same variance σ_p^2. Since $\mathrm{E}(\delta_{pi}^2) = \sigma_p^2 + \mathrm{E}^2(\delta_{pi}) = \sigma_p^2$, the expectation of $\psi_{\Delta M^t}(\psi_{\Delta M^t})^t$ is reduced as:

$$\mathrm{E}\left(\psi_{\Delta M^t}(\psi_{\Delta M^t})^t\right)=\mathrm{E}\left(\begin{bmatrix}(\Delta_{M^t})_1(\Delta_{M^t})_1^t & 0 & \cdots & 0\\ 0 & (\Delta_{M^t})_2(\Delta_{M^t})_2^t & & \vdots\\ \vdots & & \ddots & 0\\ 0 & \cdots & 0 & (\Delta_{M^t})_n(\Delta_{M^t})_n^t\end{bmatrix}_{3n\times 3n}\right)$$

$$=\begin{bmatrix}\mathrm{E}(D_1) & & & 0\\ & \mathrm{E}(D_2) & & \\ & & \ddots & \\ 0 & & & \mathrm{E}(D_n)\end{bmatrix},$$

where $D_i=\left[(\Delta_{M^t})_i(\Delta_{M^t})_i^t\right]_{3\times 3}$

$$=\begin{bmatrix}(\delta_{ypi}-\delta_{y'pi})\\ (-\delta_{xpi}+\delta_{x'pi})\\ (x_{pi}\delta_{y'pi}+\delta_{xpi}y'_{pi}\\ -y_{pi}\delta_{x'pi}-\delta_{ypi}x'_{pi})\end{bmatrix}\begin{bmatrix}(\delta_{ypi}-\delta_{y'pi}) & (-\delta_{xpi}+\delta_{x'pi}) & (x_{pi}\delta_{y'pi}+\delta_{xpi}y'_{pi}\\ & & -y_{pi}\delta_{x'pi}-\delta_{ypi}x'_{pi})\end{bmatrix}$$

The expectation of matrix $\boldsymbol{D_i}$ can be reduced as:

$$\mathrm{E}(D_i)=\sigma_p^2\begin{bmatrix}2 & 0 & (-x_{pi}-x'_{pi})\\ 0 & 2 & (-y_{pi}-y'_{pi})\\ (-x_{pi}-x'_{pi}) & (-y_{pi}-y'_{pi}) & \left(x_{pi}^2+(y'_{pi})^2+y_{pi}^2+(x'_{pi})^2\right)\end{bmatrix}_{3\times 3}.$$

(3.16)

While we denote matrix $\boldsymbol{P_i}$ as

$$P_i=\begin{bmatrix}2 & 0 & (-x_{pi}-x'_{pi})\\ 0 & 2 & (-y_{pi}-y'_{pi})\\ (-x_{pi}-x'_{pi}) & (-y_{pi}-y'_{pi}) & \left(x_{pi}^2+(y'_{pi})^2+y_{pi}^2+(x'_{pi})^2\right)\end{bmatrix}_{3\times 3},$$ the expectation of

$\psi_{\Delta M^t}(\psi_{\Delta M^t})^t$ can also be represented as:

$$\mathrm{E}\left(\psi_{\Delta M^t}(\psi_{\Delta M^t})^t\right)=\sigma_p^2\begin{bmatrix} P_1 & & & 0 \\ & P_2 & & \\ & & \ddots & \\ 0 & & & P_n \end{bmatrix}. \tag{3.17}$$

$\mathrm{E}\left(\psi_{\Delta M^t}(\psi_{\Delta M^t})^t\right)$ will later be used for calculating the expectation error of mirror plane normal vector ***u***, but before this step, we have to calculate the corresponding δ_u, the estimation error of ***u***, given a perturbed input matrix $M(\varepsilon)$.

u is the eigenvector of $\boldsymbol{M^tM}$ associated with the smallest eigenvalue. From the theorem "perturbation of eigenvalues and eigenvectors" presented by J. Weng et al. [WENG89][WENG93], the first-order perturbation of ***u*** can be approximated as

$$\delta_u \cong H\Lambda H^t\Delta_{M^tM}u,$$

$$\text{where } \Lambda=\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1/(\lambda_1-\lambda_2) & 0 \\ 0 & 0 & 1/(\lambda_1-\lambda_3) \end{bmatrix}, H=\begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix}. \tag{3.18}$$

In Equation 3.18, λ_1, λ_2 and λ_3 are the eigenvalues of noise-free $\boldsymbol{M^tM}$ in non-decreasing order; $\boldsymbol{h_1}$, $\boldsymbol{h_2}$ and $\boldsymbol{h_3}$ are the eigenvectors associated with λ_1, λ_2 and λ_3. Equation 3.18 can be reformed as:

$$\delta_u \cong H\Lambda H^t\begin{bmatrix}(aI_{3\times3}) & (bI_{3\times3}) & (cI_{3\times3})\end{bmatrix}\psi_{\Delta M^tM},$$

$$\text{where } I_{3x3} \text{ is the 3x3 identical matrix, and } \psi_{\Delta M^tM}=\begin{bmatrix}(\Delta_{M^tM})_1 \\ (\Delta_{M^tM})_2 \\ (\Delta_{M^tM})_3\end{bmatrix}_{9\times1} \tag{3.19}$$

We use $G_u=H\Lambda H^t\begin{bmatrix}(aI_{3\times3}) & (bI_{3\times3}) & (cI_{3\times3})\end{bmatrix}$ to represent data invariant to input noise and Equation 3.19 becomes

$$\delta_u \cong G_u\psi_{\Delta M^tM} \tag{3.20}$$

However, we need to relate $\psi_{\Delta M^t M}$ to $\psi_{\Delta M^t}$. By first-order approximation, it follows that

$$\Delta_{M^t M} \cong M^t \Delta_M + \Delta_{M^t} M = M^t \Delta_M + \Delta_M^t M \tag{3.21}$$

After rearrangement and simplification, the first-order relation between $\psi_{\Delta M^t M}$ and $\psi_{\Delta M^t}$ is as follows:

$$\psi_{\Delta M^t M} \cong \left(\begin{bmatrix} \begin{bmatrix} (M^t)_1 & 0 & 0 \\ 0 & (M^t)_1 & 0 \\ 0 & 0 & (M^t)_1 \end{bmatrix} & \begin{bmatrix} (M^t)_2 & 0 & 0 \\ 0 & (M^t)_2 & 0 \\ 0 & 0 & (M^t)_2 \end{bmatrix} & \cdots & \begin{bmatrix} (M^t)_n & 0 & 0 \\ 0 & (M^t)_n & 0 \\ 0 & 0 & (M^t)_n \end{bmatrix} \end{bmatrix} + \begin{bmatrix} M_{11} I_{3\times 3} & M_{21} I_{3\times 3} & \cdots & M_{n1} I_{3\times 3} \\ M_{12} I_{3\times 3} & M_{22} I_{3\times 3} & \cdots & M_{n2} I_{3\times 3} \\ M_{13} I_{3\times 3} & M_{23} I_{3\times 3} & \cdots & M_{n3} I_{3\times 3} \end{bmatrix} \right)_{9\times 3n} \begin{bmatrix} (\Delta_M^t)_1 \\ (\Delta_M^t)_2 \\ \vdots \\ (\Delta_M^t)_n \end{bmatrix}_{3n\times 1} \tag{3.22}$$

Again, we collect data invariant to input noise and rewrite Equation 3.22 as

$$\psi_{\Delta M^t M} \cong G_{M^t M} \psi_{M^t} \quad . \tag{3.23}$$

From Equation 3.20 and Equation 3.23, we get

$$\delta_u \cong G_u \psi_{\Delta M^t M} \cong G_u G_{M^t M} \psi_{\Delta M^t} \quad . \tag{3.24}$$

Therefore, the covariance matrix of δ_u can be estimated from Equation 3.17 and Equation 3.24, as follows:

$$\Gamma_{\delta u} = \mathrm{E}\left(\delta_u \delta_u^t\right) = \mathrm{E}\left(G_u G_{M^t M} \psi_{\Delta M^t} (\psi_{\Delta M^t})^t (G_u G_{M^t M})^t\right)$$

$$= G_u G_{M^t M} \mathrm{E}\left(\psi_{\Delta M^t} (\psi_{\Delta M^t})^t\right) (G_u G_{M^t M})^t$$

$$= \sigma_p^2 G_u G_{M^t M} \begin{bmatrix} P_1 & & & 0 \\ & P_2 & & \\ & & \ddots & \\ 0 & & & P_n \end{bmatrix} (G_u G_{M^t M})^t \tag{3.25}$$

The Euclidean norm of the expected error of vector u can be estimated from the square root of the trace of the corresponding covariance matrix.

$$\|\delta_u\| \approx \sqrt{\text{trace}(\Gamma_{\delta u})} \tag{3.26}$$

As suggestions in J.Weng and others' work, for the problem of estimating errors in relative depths, it is not evaluated because we just get two observations for each 3D point. The expected error of depth estimation is not really representative or reliable in this case.

Chapter 4

3D Marker Tracking under Normal Light

4.1 Markers' Placement and Equipment Setting

How many feature points and where these feature points should be placed are essential problems for performance-driven facial animation, and the MPEG-4 face object [MPEG99] tries to standardize these problems. 68 MPEG-4 facial animation parameters (FAPs) are defined to control a synthetic face. Except 2 high-level FAPs, 66 FAPs are defined for feature-point motion controls. The advantage of using MPEG-4 FAPs is its convenience to communicate with other MPEG-4 compatible face systems. However, in these 66 facial animation parameters, only 6 of them are parameters in the z-axial direction, and 58 of them are parameters in the x- or y-axial directions. Comparing with human's plentiful and subtle facial expressions, the MPEG-4 facial animation parameters could be insufficient to represent some delicate facial motion accurately.

In our face synthesis system, we separate a face into 11 regions. While regarding each region as a smoothly deformable surface, we find that there are 50 points (10 for lip contours, 12 for the lip surfaces, 10 for the mouth, 8 for cheeks, and 10 for the forehead) on a face, where the variations are empirically the most representative to control the surface deformation. Therefore, we mainly take these 50 positions as feature points to drive 3D facial animation. In other words, 150 facial animation parameters are captured. In order to acquire precise 3D positions and motions of feature points on a subject's face, colorful dot markers are pasted onto

feature points. We use thin markers without protrusion to avoid interfering with natural lip motion. With these markers, tracking of feature-point movement is much easier and more accurate. Figure 4.1 shows our tracking equipment. We place two planar mirrors next to a subject's face and use only one digital video (DV) camera to capture perfectly synchronized images—one frontal view and two mirrored ones, as shown in Figure 4.2. The orientation and location of mirrors can be arbitrary as long as each marker is visible in at least two views. Our proposed algorithm described in Chapter 3 can reconstruct these properties.

As mentioned in Chapter 3, to improve the accuracy of 3D trajectory reconstruction, either captured images or extracted markers' projected positions should be calibrated by a normalized camera model [HEIK97, BOUG, ZHAN00]. For computational efficiency, we suggest that the feature extraction is performed on the original and unnormalized images, but the extracted markers' positions are then normalized for further computation. To avoid redundancy, in the following introduction and discussion, we assume that the camera coordinate system is normalized.

In this chapter, the tracking process is semi-automatic under a normal light condition, but we propose another procedure that is fully automatic and can extend the quantity of markers to more than three hundreds.

Figure 4.1 Equipment setting. Two plane mirrors are placed near a subject's cheeks and a single video camera is applied to capturing front and side view images simultaneously.

Figure 4.2 A mirror-reflected multi-view video clip captured by a digital video camera in 720×480-pixel resolution. (55 markers: 10 for the lip contour, 12 for lip surfaces, 10 for mouths, 8 for cheeks, and 10 for the forehead)

4.2 Adaptive Kalman Filter for Stability Improvement

The Kalman filter is a linear, unbiased, and minimum error variance recursive algorithm to optimally estimate the unknown state of a linear dynamic system from noisy data at discrete time intervals, and it is widely applied to control systems, radar tracking and etc. [BOZI79, ZHAN92]. Here we briefly mention the concept of the Kalman filter, which is used to improve stability of tracking.

Let $s(t)$ denote an M-dimensional state vector of a dynamic system at time t, and the propagation of the state in time can be expressed as a linear equation

$$s(t) = As(t-1) + w(t)\ ,\ t = 1,2,...,T_{limit},$$

where A is a state-transition matrix and $w(t)$ is a zero-mean, random sequence with a covariance matrix $Q(t)$, representing the state model error.

Suppose that a series of measurement $h(t)$ are available, which are linearly related to the state variable as

$$h(t) = Cs(t) + v(t)\ ,\ t = 1,...\ ,T_{limit}.$$

where C is the observation matrix and $v(k)$ denotes a zero-mean, noise sequence, with covariance matrix $R(k)$.

Given the measurement $h(t)$, the state vector can be estimated as

$$s(t) = As(t-1) + K(t)\left[h(t) - CAs(t-1)\right],$$

where the $K(t)$ is so called the Kalman gain matrix. And the $s(t+1)$ can be predicted as

$$s(t+1|\ t) = As(t).$$

In our work, similar to Y. Altunbasak and others' tracking system [ALTU95], we adopt an adaptive Kalman filter to improve the stability of marker tracking in video. We assume the state transition equation to be

$$\begin{bmatrix} s_{px}(t) \\ s_{vx}(t) \\ s_{py}(t) \\ s_{vy}(t) \\ s_{pz}(t) \\ s_{vz}(t) \end{bmatrix} = \begin{bmatrix} 1 & T_{\text{int}} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T_{\text{int}} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T_{\text{int}} \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_{px}(t-1) \\ s_{vx}(t-1) \\ s_{py}(t-1) \\ s_{vy}(t-1) \\ s_{pz}(t-1) \\ s_{vz}(t-1) \end{bmatrix} + \begin{bmatrix} 0 \\ w_{vx}(t-1) \\ 0 \\ w_{vy}(t-1) \\ 0 \\ w_{vz}(t-1) \end{bmatrix} \tag{4.1}$$

where $s_{px}(t)$, $s_{vx}(t)$, $s_{py}(t)$, $s_{vy}(t)$, $s_{py}(t)$, and $s_{vy}(t)$ represent the state values of positions and velocities in x-, y- and z-axial directions at time t respectively. And $w_{vx}(t)$, $w_{vy}(t)$ and $w_{vz}(t)$ represent the change of velocity in x, y and z axial directions respectively over an interval T_{int} with variance $\sigma_{vx}^2(t)$, $\sigma_{vy}^2(t)$ and $\sigma_{vz}^2(t)$.

The relation between measurement and the state vector can be written as

$$\begin{bmatrix} h_{px}(t) \\ h_{py}(t) \\ h_{pz}(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} s_{px}(t) \\ s_{vx}(t) \\ s_{py}(t) \\ s_{vy}(t) \\ s_{pz}(t) \\ s_{vz}(t) \end{bmatrix} + \begin{bmatrix} v_{px}(t) \\ v_{py}(t) \\ v_{pz}(t) \end{bmatrix}, \tag{4.2}$$

where $v_{px}(t)$, $v_{py}(t)$ and $v_{pz}(t)$ represent the position measurement error in x, y and z axis with variance $\sigma_{px}^2(t)$, $\sigma_{py}^2(t)$ and $\sigma_{pz}^2(t)$. $\sigma_{px}^2(t)$, $\sigma_{py}^2(t)$ and $\sigma_{pz}^2(t)$ are variables and can be adjusted according to the confidence of measurement. The details of the Kalman filter and some applications are well described in the reference book [BOZI79, TEKA95].

4.3 Semi-automatic Marker Motion Tracking

In this section, the procedure to estimate 3D marker motion trajectories from mirror-reflected multi-view video clips is presented. We just introduce the process for one-mirror status and the process for another mirror is similar. Figure 4.3 is the

flow chart of our semi-automatic motion tracking under normal light. The detail process consists of the following steps:

Step 1. Initialize parameters. A user is required to manually designate $p_i(1)$, the projected position of actual marker i, and $p'_i(1)$, the projected position of mirrored marker i, in the first video clip (t = 1), for i = 1 … N. N is the amount of markers the mirror covers.

Step 2. Estimate rough mirror positions and orientations relative to the camcorder from real versus mirrored point correspondences assigned in the first frame. Estimate $m_i(1)$, the actual 3D position of marker i, for i = 1 … N.

Step 3. Predict the 3D position at t + 1 as $m_i(t+1|t)$ and generate mirrored position $m'_i(t+1|t)$ for i = 1 … N. Update the time stamp, set $t = t + 1$.

Step 4. Project the actual and mirrored markers back to the image plane I as $p_i(t|t-1)$, $p'_i(t|t-1)$. Within the searching area centered at $p_i(t|t-1)$, find the best γ (for example, γ = 6) 2D projected candidates $pc_{ij}(t|t-1)$ {j = 1... γ} with minimum **ColorCost**, which is L2-norm of color differences in block matching compared to that of $p_i(t-1)$ and $p_i(1)$.
The ColorCost of a position p_x at time t for marker i is

$$\begin{aligned}\text{ColorCost}_{(px)} = {} & weight \times \text{Score}\left(I_t, p_x, I_{t-1}, p_i(t-1)\right) \\ & + (1-weight) \times \text{Score}\left(I_t, p_x, I_1, p_i(1)\right)\end{aligned} \tag{4.3}$$

where

$$\begin{aligned}&\text{Score}\left(I_a, p_a, I_b, p_b\right) = \\ &\sum_{r=-w}^{w}\sum_{c=-w}^{w}\left\|I_a\left((x_{pa}+r),(y_{pa}+c)\right)-I_b\left((x_{pb}+r),(y_{pb}+c)\right)\right\|\end{aligned} \tag{4.4}$$

and *weight* is a user-defined parameter to specify the effect weight between the previous image and the first one.

Repeat this process to find γ candidates $pc'_{ik}(t \mid t-1)$ $\{k = 1... \gamma\}$ of the mirrored part.

Step 5. For each *j* and *k* combination, generate 3D candidates $mc_{ijk}(t \mid t-1)$ from the projected point correspondence of $pc_{ij}(t \mid t-1)$, $pc'_{ik}(t \mid t-1)$ and calculate the cost function

$$\mathbf{Cost}_{ijk} = \alpha\mathbf{DistCost}_{ijk} + \beta(\mathbf{ColorCost}_{ij} + \mathbf{ColorCost}_{ik}), \quad (4.5)$$

$$\mathbf{DistCost}_{ijk} = f\left(\left\| m_i(t \mid t-1) - mc_{ijk}(t \mid t-1) \right\|\right) \quad (4.6)$$

where α and β are user-defined constant values and f is a user-defined monotonically increasing function. Figure 4.4 is a conceptual diagram and detail description of our 2D/3D candidate search.

Step 6. Find the best candidate with the minimum $\mathbf{Cost}_{ijk}$, and adjust the measurement error variances according to $\mathbf{ColorCost}_{ij} + \mathbf{ColorCost}_{ik}$.
Set the best candidate as the measured 3D position and filter it as $m_i(t)$.

Step 7. If $t < T_{\text{limit}}$, {go to step 3} else {manual fine-tuning with GUI tools}

Step 8. Calculate U_{fine}, the mirror's fine positions and orientations, from user-tuned projected point correspondences. Re-estimate accurate 3D markers' motion trajectories by U_{fine} and tuned projected point correspondences.

Adjusting measurement-error variances in Step 6 accords with image similarity. When a marker image is occluded or interfered with by interlace effect or intense specular-lighting noise, the cost function value will be dramatically high, and the measurement error variances will be large. This decreases the Kalman gain. In other words, the impact weights of contaminated measurement data are diminished and the effects of noise or occlusion can be alleviated.

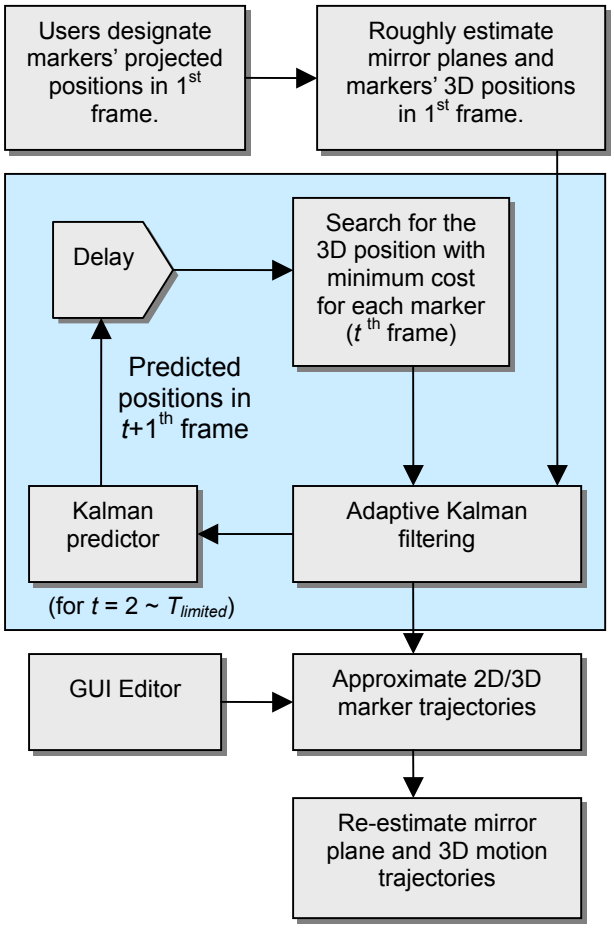


Figure 4.3 The flow chart of semi-automatic markers' 3D motion tracking under normal light.

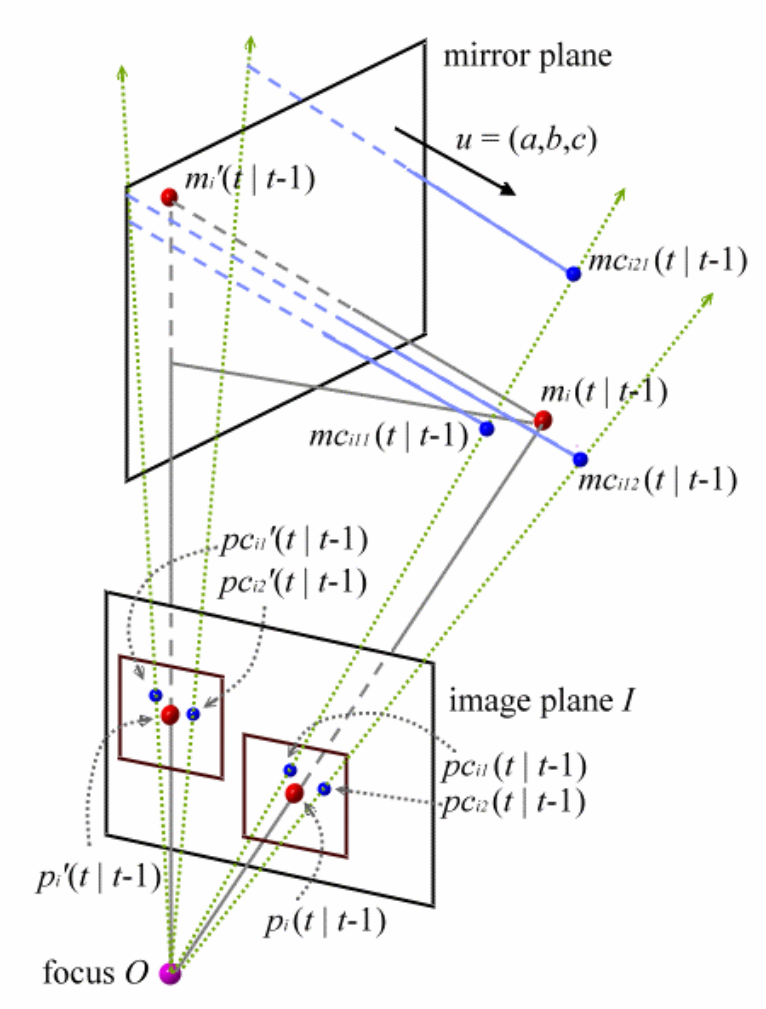


Figure 4.4 A conceptual diagram of 2D/3D candidate finding. As mentioned in Step 4 and Step 5 of the tracking procedure, to estimate $m_i(t)$, $m_i(t \mid t-1)$ is first predicted by an adaptive Kalman filter based on the previous trajectory of m_i and $m_i'(t \mid t-1)$ is also evaluated. A block matching method is applied to search the best γ 2D candidates around projection predicted position $p_i(t \mid t-1)$ and $p_i'(t \mid t-1)$ (γ=2 in this diagram). $mc_{ijk}(t \mid t-1)$ is the 3D position estimated from the 2D candidate pair $\{ pc_{ij}(t \mid t-1),\ pc'_{ik}(t \mid t-1) \}$.

4.4 FAP Extraction

In the previous step, markers' 3D motion trajectories have been estimated. However, a subject under test may swing or nod his or her head when speaking and making facial expressions, and thus the motions of 3D markers are composed of not only facial motions but also head motions. To acquire precise facial motion, the head motion must be estimated and removed from 3D motion trajectories.

As mentioned in T.S. Huang and A. Netravalis' review [HUAN94], with 3 non-collinear 3D points, the movement of rigid object can be uniquely determined by a rotation matrix R_{head}, and translation vector T_{head}.

$$ms_s(t) = R_{head}(t) \times ms_s(1) + T_{head}(t) \tag{4.7}$$

where $ms_s(t)$ is 3D position of a specific point s on a rigid object at time t, and where $ms_s(1)$ is the 3D position of point s on a rigid object at the initial time.

Therefore, the 3D data of more than 4 additional markers placed on the performer's ears are regarded as points on a rigid body, and we applied an algorithm for rigid-body motion estimation proposed by K. Arun et al. [ARUN87] to determine the head rotation $R_{head}(t)$ and head translation $T_{head}(t)$ for each video clip. After the rotation and translation of successive time stamps are determined, we can extract 3D facial motion of marker i at time t without head movement as

$$fm_i(t) = R_{head}^{-1}(t)\left(m_i(t) - T_{head}(t)\right) \tag{4.8}$$

where $m_i(t)$ is the original estimated 3D position of marker i at time t.

Chapter 5

Fully Automatic Mass 3D Marker Tracking Under Blacklight-UV Lamps

5.1 Introduction

In Chapter 4, a semi-automatic 3D motion tracking procedure is proposed. User intervention is necessary in the above-mentioned system for two reasons. The first reason is due to difficulty of marker identification. Under a normal light condition, reliable extraction of makers is not easy since markers' colors and projected shapes can change dramatically in different reflective angles, and some facial parts may occasionally be misjudged as markers for the same explanation. To avoid extracting markers, we apply block matching for tracking, it compares the color variation between previous and successive video clips. But, we still have to identify where the markers are in the first frame and manual selection is required. The second reason is due to ambiguity and occlusion in tracking. While applying block matching for tracking, perturbation of markers' reflective colors and projected shapes can make the tracking trajectories "trembling". By chance, it may even make the tracking "derail" to where there is no marker. We utilize adaptive Kalman filters to alleviate these situations. But, occlusion is still the most critical problem to prohibit the tracking method in Chapter 4 from being fully automatic. For example, when our mouths are pouted or greatly opened, the markers below the lower lips vanish in video clips. We also tried to use thresholding in block matching and Kalman predictors to tackle this problem. Notwithstanding, it works satisfactorily only for short-term marker occlusion.

To be fully automatic tracking, some researches employed a generic facial motion model. T. Goto et al. [GOTO01] utilized separate simple tracking rules for eyes, lips, etc. respectively. F. Pighin et al. [PIGH99] proposed to track animation-purposed facial motion based on linear combination of 3D face model bases. In the "voice puppetry"[BRAN99], M. Brand applied a generic head mesh with 26 feature points, where spring tensions are assigned to each edge connection. Such a generic facial motion model can rectify "derailing" tracking trajectories and is beneficial for sparse feature tracking. However, an approximate model can also restrict the feature tracking while a subject does prominent or extraordinary facial expressions.

From another aspect, applying special lights to highlight markers is effective to improve the feature extraction. As mentioned in Chapter 2, active markers, which emit infrared rays, or passive markers, which is of high infrared response, are all widely utilized in industrial motion capture products. E.C. Patterson et al. presented a facial tracking system with a dozen passive markers for ultraviolet (UV) light. In the research of B. Guenter et al. [GUEN98], they track 182 dot markers painted with fluorescent pigments for near UV light. This research not only used special markers and lights to enhance the feature detection, but also take into account the spatial and temporal consistency for reliable tracking.

B. Guenter and others' work [GUEN98] inspired our tracking method proposed in this chapter. We also apply markers with special pigments for blacklight blue (BLB) fluorescent lights to considerably increase the distinctness of markers from others in color space; we also utilize the spatial and temporal coherence to detect and compensate the missing and false-detection problems in tracking. However, the proposed method is more efficient and versatile. It is able to capture more than 300 markers and will be extended to track more than 100 markers from live videos in real time on a regular pc. In Guenter and others' work [GUEN98], a subject's head was required to be immobile due to the limitation of markers' vertical order in their marker matching routine, and therefore, the head movement then must be tracked independently as a postprocess. By contrast, the proposed method is capable of fully automatic tracking both facial expressions and head motions simultaneously.

In Section 5.2, how to detect markers with fluorescent pigments is presented; the tracking procedure that can be fully automatic is proposed in Section 5.3.

5.2 Equipment Setting and Feature Extraction

In order to enhance the distinctness of markers from others in video clips, we apply UV-responsive markers and UV blacklight blue (BLB) lamps. Here, an introduction to these devices and the UV light are briefly mentioned.

Ultraviolet (UV) light represents a section of the light spectrum, extending from the blue end of the visible (400nm) to the x-ray region (100nm). It can activate some materials, such as phosphors, to the luminescence condition. Luminescence is composed of fluorescence and phosphorescence. The main difference between these two conditions is the period of radiation. Fluorescence vanishes but phosphorescence continues for a while as the UV radiation stops. UV light is further divided into three subsections, UV-A, UV-B, and UV-C. UV-A light is the longest wavelength (400nm-315nm) and the lowest energy among three subsections and is also referred to as “black light”. Since the black light is less harmful comparing to the most aggressive component UV-B, it is usually used to detect counterfeit money in banks or for special effects in nightclubs or theaters.

In the proposed tracking system, we also utilize the fluorescent phenomenon to emphasize markers in video clips. Markers are covered with fluorescent pigments and blacklight blue lamps are used to excite fluorescence of markers. The fluorescence is visible in the visible light spectrum, and therefore, we don’t need special attachment lenses for light filtering. Figure 5.1 is a photo of the proposed tracking equipment taken under normal light and Figure 5.2 is a video clip captured by a digital video camera, where fluorescence of markers is excited by UV light. For further introduction of UV light and luminescence, please refer to the bibliography [ILLU85].

Figure 5.3 is an example and the flow diagram of our marker extraction method. As shown in the original video clip (Figure 5.3(a)), owing to application of UV-responsive markers and blacklight blue lamps, markers are prominent comparing to others in video clips; therefore, the automatic feature extraction is more reliable and more feasible than feature extraction in the normal light condition. We mainly follow the methodology of connected component analysis in computer vision, which are composed of thresholding, connected component labeling and region property measurement, but we also slightly modify the implementation for computational efficiency. With the modification, our system can be extended to

real-time tracking for live videos on a regular pc.

Since the intensity of UV-responsive markers is much higher than that of others, to exclude pixels that have less probability of marker projection, the first stage is color thresholding. Thresholding distinguishes pixels with higher R, G, and B values from pixels with lower values. Figure 5.3(c) is a color-thresholded image, and pixels that pass the threshold are displayed in white. The threshold is determined empirically.

In many feature extraction systems, mathematical morphology operations, such as dilation, erosion, opening, closing, etc., are performed before or after color thresholding, but our system are not. That is because thresholding works satisfactorily in most cases; the most difficult case, interlaced scan lines as shown in Figure 5.3(g), can be solved more efficiently by merging nearby connected components.

The second stage is color labeling. In our experiment, we collected six UV-responsive markers that are pink, yellow, green, white, blue, and purple when illuminated by normal fluorescent lamps, but there are only four typical colors, pink, blue-green, dark blue and purple while illuminated by blacklight blue lamps (as listed in Figure 5.3(d)). Hence, four classes of colors are adopted and each color class comprises dozens of color samples. A selection tool is provided to select these color samples from training videos. To classify the color of a pixel in video clips, the nearest neighborhood method (1-NN) is applied. To diminish the classification error resulting from intensity variation, the matching operation work on a normalized color space (*nR*, *nG*, *nB*), where

$$nR = \frac{R}{(R+G+B)}, \quad nG = \frac{G}{(R+G+B)}, \quad nB = \frac{B}{(R+G+B)}$$

and (*R*, *G*, *B*) is the original color value. In general, the more color samples is in a color class, the more accurate color class of a pixel is classified. For real-time or near real-time applications, four color samples in each color class are sufficient. Figure 5.3(e) is the color-labeled image represented by typical colors in color classes.

Connected component labeling is the third stage in our feature extraction. It groups connected pixels with the same color label number as a component and we adopt 8-connected neighbors. Several connected components algorithms were proposed, for examples, iterative algorithms, the classical algorithm, space-efficient

two-pass algorithms, etc. [HARA92]. These algorithms are general-purpose and take into account all circumstances of connection. Nevertheless, in our case, a marker's projection is smaller than a radius of 5 pixels, and thus, the process of connected component labeling can be much simplified. We modify the classical algorithm as the following C-like pseudo-code:

```
void PreliminaryCCL()
{
    //initialization
    for( c =0; c < color_class; c++) {
        newLabel[c]=0;
    }//for i
    //labeling
    for( i = 0 ; i < I_height; i++) {
        for( j=0; j < I_width; j++) {
            cl = ColorLabel[i][j];
            if(IsValidColorGroup(cl) {
                A = PrecedingNeighbors(i, j, cl);
                if(IsEmpty(A) {
                    CCL[i][j] = newLabel[cl];
                    newLabel[cl]++;
                } else {
                    CCL[i][j] = MIN(A);
                }//else
            }//if
        }//for j
    }//for i
}//void PreliminaryCCL()
```

In the pseudo-code, the result of preliminary CCL are placed in the array *CCL*,

and the function **PrecedingNeighbors(*i*, *j*, *cl*)** for the pixel at (*i*, *j*) which collects valid *CCL* values at (*i*-1, *j*-1), (*i*, *j*-1), (*i*+1, *j*-1), and (*i*-1, *j*). The **PreliminaryCCL,** unlike the classical algorithm, checks only preceding neighbors. Not all 8-connected components can be labeled as the same group by **PreliminaryCCL** since we do not utilized a large equivalent class table for transiting label numbers as in the classical one. But the inconsistency is local and can easily solve in our next stage. A result image processed by **PreliminaryCCL** is shown in Figure 5.3(f).

After the process of preliminary connected component labeling, there are still redundant connected components caused by interlaced fields of video, incomplete connected component labeling or noise (as shown in Figure 5.3(g)). The fourth stage is to refine the connected components to make each extracted components as close as the actual markers' projection. Since markers are placed evenly on a face and the shortest distance between two markers of the same color class is less then diameter of a dot marker, nearby connected components should belong to the same marker. Therefore, the first two kinds of redundant connected components can be simply tackled by merging components of a distance less than the markers' average diameter. For the redundant components caused by noise, we suppress them by removing connected components less than four pixels. Figure 5.3(h) is the refinement of Figure 5.3(g); Figure 5.3(i) is the extracted markers' projection.

The extracted connected components are still not equivalent to the actual markers' projection. Responsive colors of a fluorescent marker can still vary due to changes of view direction. This may result in erroneous classification of color classes and cause missing or redundant extraction of projected marker. Besides, the position of an extracted marker is also disturbed by noise.

To compensate the imperfect feature extraction, the following section proposes a procedure to automatically track 3D motions of mass markers with missing and false-detection in feature extraction.

Figure 5.1 The tracking equipment for the blacklight condition. The photo is taken under normal light. Two “Blacklight Blue”(BLB) lamps are placed in front of a subject and mirrors. The low-cost special lamps are coated with fluorescent powders, and it can emit long wave UV-A radiation to excite luminescence.

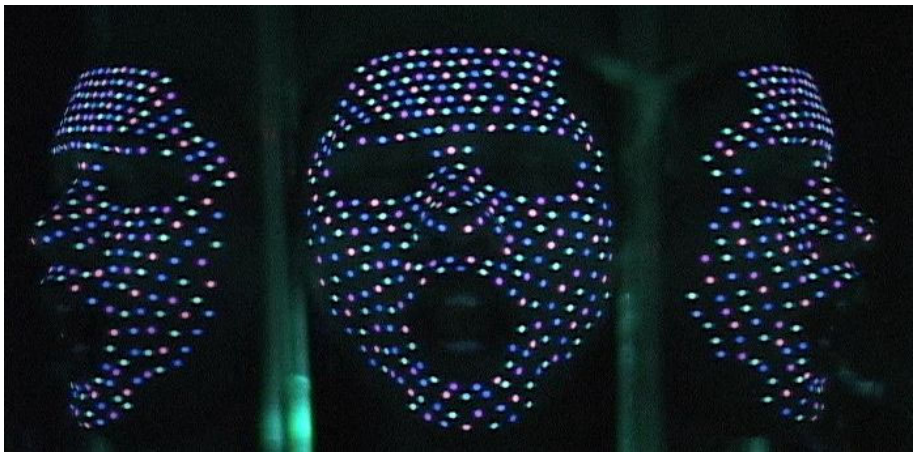
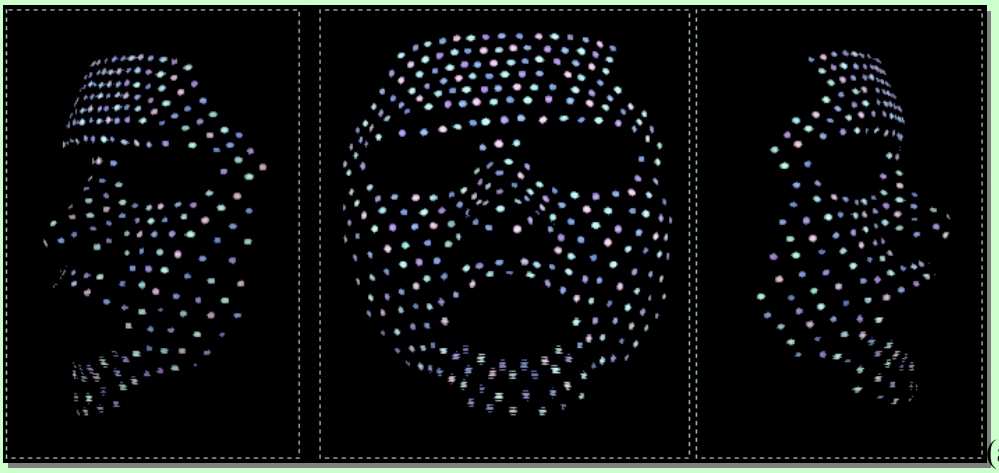


Figure 5.2 A captured video clip of fluorescent markers illuminated only by UV “Blacklight Blue” lamps. The fluorescence is visible in the visible light spectrum and no special lens is required for filtering. (300 markers are evenly pasted upon a subject’s face.)

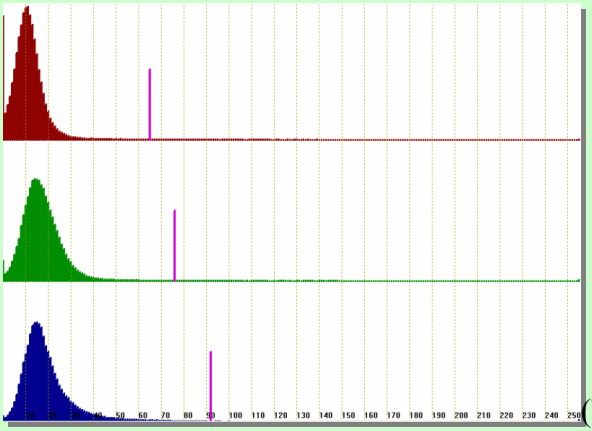
Feature extraction of UV-responsive markers

- An original video clip composed of a front and two side views:



(a)

The RGB color histogram:

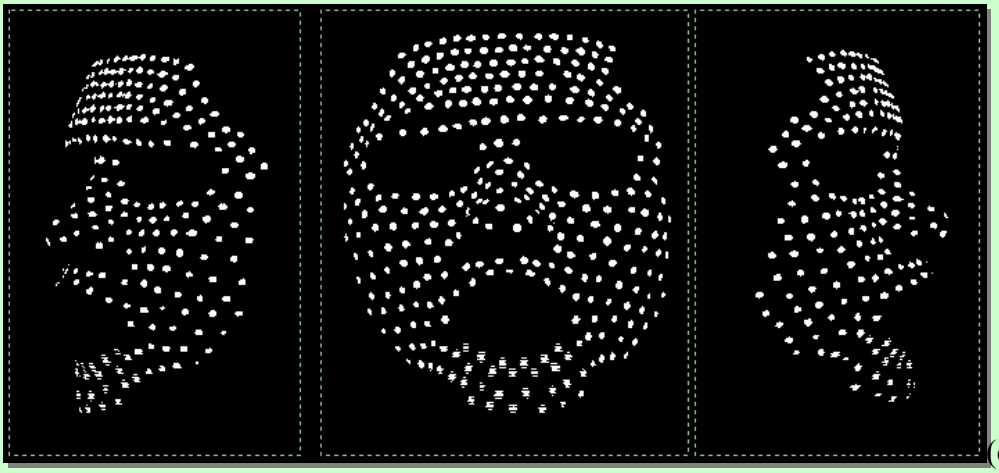


(b)

Color thresholding:

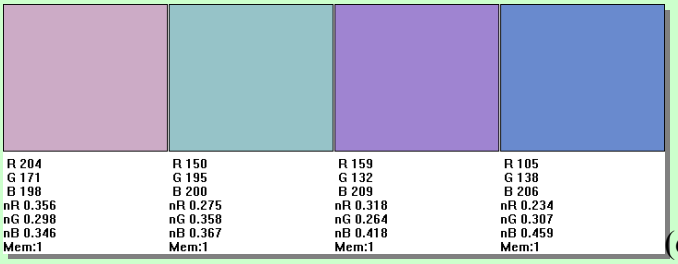
RGB (65,76,92)

- Valid pixels that passed the color threshold:



(c)

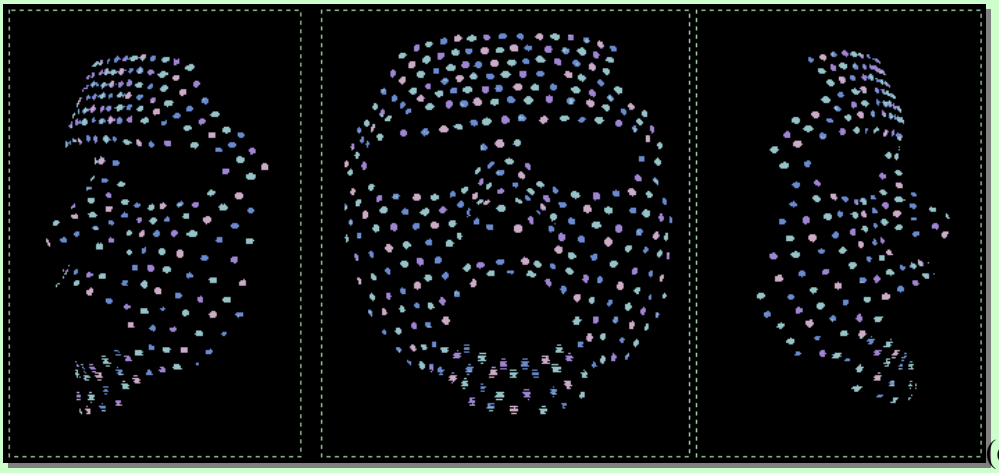
n classes of color samples: (n=4, in this case)



(d)

Color labeling by the nearest normalized color sample.

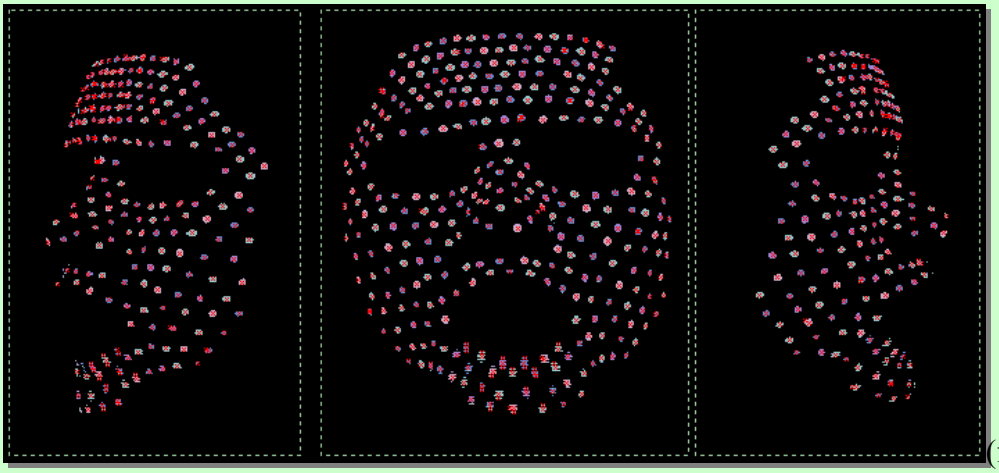
■ The color-labeled image represented by typical colors of color classes:



(e)

Efficient grouping by a simplified connected component labeling method.

■ The preliminary connected component labeled image:
(A red cross represents the center of a connected component)



(f)

Redundant connected components caused by interlaced effects, incomplete connected component labeling or noise:

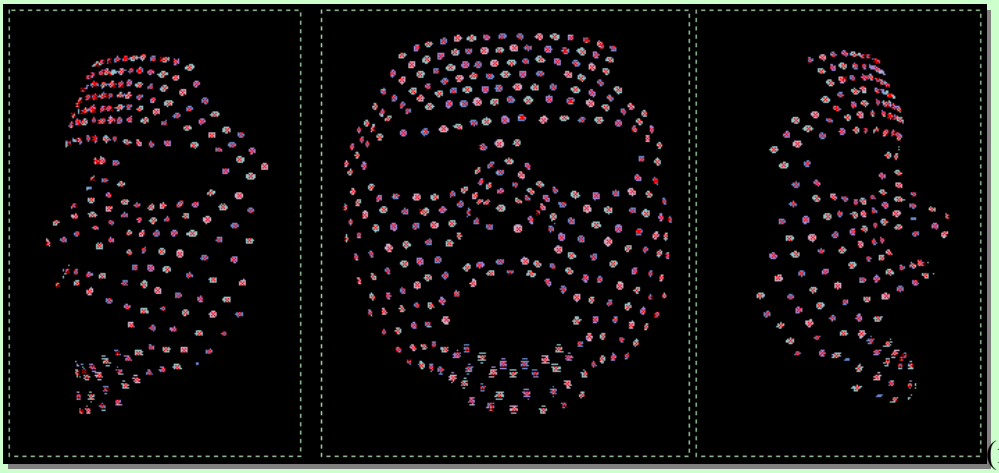
(g)

Refinement of connected components: (Merging nearby components of the same color class and removing those with pixels less than 4)

(h)

Refinement of connected components

■ Extracted projected positions of UV-responsive markers:
(A red cross represents the center of an extracted markers' projection)



(i)

Figure 5.3 An example and the flow diagram of feature extraction of UV-responsive markers. The process starts from color thresholding, color labeling, connected component grouping to refinement.

5.3 A Fully-Automatic Tracking Procedure for Mass 3D Markers

Fully automatic tracking multiple target trajectories over time is an important problem, called the "multitarget tracking problem", in radar surveillance systems [BUCK00]. With only measurement error and false detection, this problem is equivalent to the minimum cost network flow (MCNF) problem. The optimal solution is feasible and the computation complexity is $\mathbf{O}(N^3 \log NC)$, where N is the number of nodes in network and C is the maximum value of the coefficients among edges [CAST90, WOLF89]. Nevertheless, when measurement errors, missing detection and false alarms all occur in tracking, time-consuming dynamic programming, etc. are required to estimate approximate trajectories and the tracking results can degenerate seriously as the number of missing detection slightly increases [BUCK00]. As mentioned in the previous section, even though fluorescent markers and blacklight lamps are used to enhance the distinctness of markers and to improve the steadiness of markers' projected colors, missing and false detection are still unavoided in the feature extracting process.

Fortunately, markers' motion on a facial surface is unlike that of targets tracked in radar systems. Targets in the general multitarget-tracking problem are moved independently and consequently only the prior trajectory of a target can be utilized to conjecture the target's movement from detected candidates over time. By contrast, points on a face surface have not only earlier information but also spatial coherence within the current time stamp. Except for the mouth, nostrils and eyelids, most parts on a face are continuous surfaces, and position and motion of a facial point are similar to those of its neighbors. With this additional property, automatic diagnosis of missing and false detection becomes feasible and the computation is more efficient.

Figure 5.4 is the flow chart of the proposed tracking procedure for the UV light condition. Here, we first present issues encountered in 3D motion tracking of mass UV-responsive markers and our proposed solutions.

- **Equipment setting**

In the first step, equipment setting, two mirrors, and two UV-blacklight blue lamps are placed in front of a video camera as shown in Figure 5.1. As mentioned in previous chapters, the camera's intrinsic parameters are first estimated by camera calibration methods [HEIK97, ZHAN00], and we adopt a well-organized camera

calibration library developed by J.-Y. Bouguet [BOUG]. All operations in the following steps are performed in the normalized camera coordinate based on the evaluated camera parameters. The mirror planes orientations and locations are then estimated by the proposed method introduced in Chapter 3. All the equipments should be fixed stably to avoid re-calibration of device parameters.

■ **Recovering point correspondence in the neutral face**

Initialization of the tracking procedure is to reconstruct the 3D positions of markers in the first frame. In Chapter 4, since the markers in video clips are not distinct enough, user interaction is required to explicitly specify all markers' projected positions and point correspondence. Comparatively, UV-responsive markers are much more distinct and markers' projected positions can be automatically estimated by the method presented in Section 5.2. For efficient recovering point correspondence in the first frame, two ways are utilized for different conditions.

The first approach is to employ 3D range scanned data. Figure 5.5 shows the operation of a 3D laser scanner and Figure 5.6 shows the process to recover point correspondences. First, markers' projected positions are extracted (as shown in Figure 5.6(a)). Then, a user has to manually select n (n>3) corresponding point pairs on the nose tip, eye corners, mouth corners, etc. in the first video clip to form a 3D point set. After corresponding feature points in 3D scanned data are also designated, the affine transformation between 3D scanned data and specified markers' 3D structure can be evaluated by a least square solution proposed by K.S. Arun et al. [ARUN87]. While we extend the vector $\overrightarrow{op_i}$, where o is the lens center and p_i is the extracted projected position of marker i in the frontal view, the intersection of line $\overrightarrow{op_i}$ and 3D scanned data are regarded as the conjectured 3D position of marker i, denoted as m_i. The corresponding point in a side view is then recovered by mirroring m_i and projecting the mirrored one back to the image plane. Due to perturbation of measurement noise, the nearest point of the same color within a tolerant region is regarded as the corresponding point p_i'.

The second approach is to recover point correspondences by evaluating a subject's 3D face structure from rigid-body motion directly. If an object is rigid or not deformable, affine transformation (rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$) resulting from

motion is equivalent to the inversed affine transformation resulting from changes in the coordinate system. And therefore, reconstructing 3D structure from rigid-body motion is equivalent to reconstructing 3D structure from multiple views [ZHAN92, WENG93, HUAN94]. With this property, we require a subject to retain his or her face in a neutral expression and slowly move his or her head toward four directions: right-up, right-down, left-down, and left-up. A preliminary 3D structure of the face can be estimated from markers' projected motion in the frontal view, and point correspondence can then be recovered.

- **Construction of 3D candidates by mirrored epipolar lines**

If N_f and N_s feature points of a certain color class are extracted in the frontal and side views respectively, each point corresponding pair can generate a 3D candidate, and therefore, there are total N_fN_s 3D candidates of the color class. In B. Guenter and others' work [GUEN98], they took all these N_fN_s potential 3D candidates to track N_{mrk} markers' motion. However, in a two-view system, given a point p_i in the first image, its corresponding point is constrained to lie on a line called the "epipolar line" of p_i [ZHAN95, HARA93]. With this constraint, one only has to search features along the epipolor line. The number of 3D candidates decreases substantially and the computation is much more efficient.

There is a similar constraint in mirror-reflected multi-view images. Since a mirrored view can be regarded as a flipped view from a virtual camera, the constraint is also tenable but flipped. We call this mirrored constraint "mirrored epipolar line". We briefly introduce the concept of the mirrored epipolar line by Figure 5.7. We assumed that p is an extracted feature point, o is the optic center, and p', the unknown corresponding point in the mirrored view, is unknown. Since p is a projection, the actual marker's 3D position, m, must lie on the line l_{op}. According to the mirror symmetry property, the virtual marker's 3D position, m', must lie on l'_{op}, which is a symmetric line of l_{op} with respect to the mirror plane. While a finite-size mirror model is adopted, the projection of l'_{op} is a line segment and it is denoted as $\overline{p'_a p'_b}$. The corresponding point p' then must lie on the mirrored epipolar line segment $\overline{p'_a p'_b}$, or otherwise the marker m is not visible in the mirrored view.

The mirrored epipolar line of a point p can easily be evaluated. From Equation 3.5, where $(p')^t Up = 0$, we expand p and p' by their x, y, and z components and the equation becomes

$$\begin{bmatrix} x'_p & y'_p & 1 \end{bmatrix} \begin{bmatrix} -cy_p + b \\ cx_p - a \\ -bx_p + ay_p \end{bmatrix} = 0 \tag{5.1}$$

and the line

$$(-cy_p + b)x'_p + (cx_p - a)y'_p + (-bx_p + ay_p) = 0 \tag{5.2}$$

is the mirrored epipolar line of p.

Due to the perturbation resulting from noise, the corresponding point may not lie on the mirrored epipolar line exactly. To evaluate potential 3D candidates with noise tolerance, we extend the line k pixels up and down (k=1.5 in our case) to form a "mirrored epipolar band" and search corresponding points within the region between two constraint lines

$$(-cy_p + b)x'_p + (cx_p - a)y'_p + (-bx_p + ay_p) + (cx_p - a)k = 0 \tag{5.3a}$$

and

$$(-cy_p + b)x'_p + (cx_p - a)y'_p + (-bx_p + ay_p) - (cx_p - a)k = 0 . \tag{5.3b}$$

Figure 5.8 shows an example of potential point corresponding pairs generated by the mirrored epipolar constraint; Figure 5.9 shows the 3D candidates generated from the constrained point correspondences.

■ **Head movement estimation and removal**

In Chapter 4, markers' 3D motion trajectories are first reconstructed by a block-matching based search method with adaptive Kalman predictors and filters. For estimation of head movement, specific markers' trajectories are used and facial motion parameters are then evaluated by removing head movement from markers' motion trajectories.

As we have mentioned, markers' 3D motion trajectories comprise both facial motion and head motion. Because the moving range of a head is larger than those of facial muscles, when a subject does facial expression and moves his or her head

concurrently, most of the markers' motion results from head motion. This situation could make the Kalman predictors and filters dominated mainly by head motion but little by facial motion. Our experiments coincide with our intuitive inference. We find that prediction and filtering are more accurate if separate Kalman predictors/filters are applied to head motion and facial motion tracking respectively. And also, our proposed method for finding frame-to-frame 3D point correspondences is more reliable if head motion is removed in advance. Therefore, here, we change our strategy and try estimating and removing head motion before finding frame-to-frame 3D point correspondence.

We define that the head pose in the first frame (t=1) is upright, and the head motion at time t is the affine transformation of the head pose at time t with respect to the head pose at t=1. As we mentioned in Section 4.4, the affine transformation consists of rotation $R_{head}(t)$ and translation $T_{head}(t)$. For automatic head movement tracking, seven specific markers are pasted on locations invariant to facial motion, such as a subject's ears and the concave tip on the nose column. Adaptive Kalman filters are again to alleviate trembles resulting from measurement errors. It is different from the position-velocity state model of Kalman filter for each marker in Chapter 4. Another point of view is taken here.

$R_{head}(t)$ and $T_{head}(t)$ are both three degrees of freedom. $T_{head}(t) = [t_x(t), t_y(t), t_z(t)]$. $R_{head}(t)$ can be parameterized by $(r_x(t), r_y(t), r_z(t))$ in radian.

$$R_{head} = R_z R_y R_x$$

$$= \begin{bmatrix} \cos(r_z) & -\sin(r_z) & 0 \\ \sin(r_z) & \cos(r_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(r_r) & 0 & \sin(r_y) \\ 0 & 1 & 0 \\ -\sin(r_y) & 0 & \cos(r_r) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(r_x) & -\sin(r_x) \\ 0 & \sin(r_x) & \cos(r_x) \end{bmatrix}$$

$$= \begin{bmatrix} c(r_z)c(r_y) & -s(r_z)c(r_x)+c(r_z)s(r_y)s(r_x) & s(r_z)s(r_x)+c(r_z)s(r_y)s(r_x) \\ s(r_z)s(r_y) & c(r_z)c(r_x)+s(r_z)s(r_y)s(r_x) & -c(r_z)s(r_x)+s(r_z)s(r_y)s(r_x) \\ -s(r_y) & c(r_y)s(r_x) & c(r_y)c(r_x) \end{bmatrix}, \qquad (5.4)$$

where R_z, R_y, and R_x are rotation matrices along z, y, and x axes; $c(\ )$ and $s(\ )$ are abbreviations of $cos(\ )$ and $sin(\ )$. We apply Kalman filters to these six parameters $[r_x(t), r_y(t), r_z(t), t_x(t), t_y(t), t_z(t)]$ directly. The process of head motion evaluation is as follows:

Step 1. Users designate specific markers s_i (for i=1...N_{smrk}) for head moting tracking from the reconstructed 3D markers of the neutral face (t=1) and denote the position as $ms_i(1)$.
(*This step can be further extended to be fully automatic by clustering methods, such as the K-means algorithm (k=3), since markers on an ear are close to each other and far from those on the face.*)

Step 2. Initialize adaptive Kalman filters for head motion and set $r_x(0) = r_y(0) = r_z(0) = 0$, $t_x(0) = t_y(0) = t_z(0) = 0$, and $t = 1$.

Step 3. Predict the head motion parameters $r_x(t+1|t)$, $r_y(t+1|t)$, $r_z(t+1|t)$, $t_x(t+1|t)$, $t_y(t+1|t)$ and $t_z(t+1|t)$ by Kalman predictors and then construct $R_{head}(t+1|t)$ and $T_{head}(t+1|t)$ by Equation 5.4.
Increase time stamp $t = t+1$

Step 4. Generate predicted positions of specific markers as

$$ms_i(t|t-1) = R_{head}(t|t-1) \times ms_i(1) + T_{head}(t|t-1) \quad (5.5)$$

and find $ms_i(t)$ by searching the nearest potential 3D candidates of the same color. The search is restricted within a spherical range centralized at $ms_i(t|t-1)$ and of a radius r_{srch}.
If no candidate is found, set the marker ineffective at time t.
(*The potential 3D candidates are constructed by the method in the above subsection.*)

Step 5. Detect false tracking, whose estimated motions are odd comparing to other specific markers; set the markers of odd estimation ineffective at time t.
(*The false tracking detection is presented in the next subsection. We skip the details here*)

Step 6. Estimate the affine tranformation (R_{msr} *and* T_{msr}) of effective specific markers between time t and the 1st frame by the method proposed by K. S. Arun [ARUN87].
Extract $r_{msr\_x}(t)$, $r_{msr\_y}(t)$ and $r_{msr\_z}(t)$ from R_{msr} by Equation 5.4 and extract

$t_{msr\_x}(t)$, $t_{msr\_y}(t)$ and $t_{msr\_z}(t)$ from T_{msr}.

Step 7. Take $[r_{msr\_x}(t), r_{msr\_y}(t), r_{msr\_z}(t)]$ $[t_{msr\_x}(t), t_{msr\_y}(t), t_{msr\_z}(t)]$ as measurement inputs to the adaptive Kalman filter and estimate the output $[r_x(t), r_y(t), r_z(t)]$ $[t_x(t), t_y(t), t_z(t)]$.

Step 8. If $t > T_{limit}$, stop; else goto ***Step 3***.

Operation of the Kalman filter for translation $[t_x(t), t_y(t), t_z(t)]$ is the same as the Kalman filter for each marker in Chapter 4, which takes 3D positions as input and the internal states are positions and velocities. The operation of the Kalman filter for rotation $[r_x(t), r_y(t), r_z(t)]$ is similar but the input is a set of angles and the internal states represent angles and angular velocities. With this improved procedure, the head motion tracking is more reliable and stable comparing to the process in Chapter 4, where specific markers are tracked independently.

Once the head motion at time t is evaluated, an inverse affine transformation similar to Equation 4.8 is applied to potential 3D candidates for head motion removal.

■ **Finding frame-to-frame 3D point correspondence with outlier detection**

In the previous subsection, we have presented how to reconstruct markers' 3D structure in the first frame, how to generate 3D candidates for tracking, and how to remove head motion from motion trajectories. In the subsection, we assume that head motion is removed from potential 3D candidates, and our goal is to track markers' motion trajectories from a sequence of potential 3D candidates frame by frame.

Figure 5.10 is a conceptual diagram of the problem statement. The number of potential 3D candidates in a frame is around 1.2~2.3 times the number of the actual markers. The additional 3D candidates can be regarded as the false detection in the multiple-target tracking problem. If only false detection occurs, the graph algorithms for minimum cost network flow (MCNF) can evaluate the optimal solution. In our case, we employ Kalman predictors and filters to efficiently grasp the time-varying position variation of each marker. However, a marker can "miss" in video clips occasionally. The missing condition results from blocking or occlusion due to view directions, incorrect classification of markers' colors or noise disturbance. While the missing and false detection occur concurrently, a simple tracking method without

evaluation of false tracking would degenerate and the successive motion trajectories could be disordered.

We use an example to explain the serious consequence of false tracking. In Figure 5.11, the marker **B**, is not included in the potential 3D candidates of the third frame, and its actual position is denoted as **B**(3). Based on the previous trajectory, **B'**(3) is the nearest potential candidates with respect to the predicted position. According to this false trajectory **B**(1)→**B**(2)→**B'**(3), the next position should be **B'**(4). Consequently, the motion trajectory starts to "derail" seriously and is difficult to recover. Furthermore, false tracking of a marker may even interference with tracking of other markers. In the example of Figure 5.11, the marker **C** is also undetected in the fourth frame; the nearest candidates with respect to the predicted position is **C'**(4). Unfortunately, **C'**(4) is actually the marker **D** at the fourth frame, denoted as **D**(4). Because each potential candidate should be "occupied" by one marker at most, a misjudgment would not only make the marker **C** but also the marker **D** depart from the correct trajectories.

For detection of false tracking, we take advantage of the spatial coherence of face surfaces, which means a marker's motion is similar to that of its neighbors. Before we present our method, the terms are specified in advance. For each marker, its neighbors are other markers that locate within a 3D distance ε from the marker in the neutral face. For the motion of marker i at time t, we don't use the 3D location difference between time t-1 and t but use the location difference between time t and time 1 instead. We denote $v_i(t) = m_i(t) - m_i(1)$. This is because the former is easily disturbed by measurement noise but the latter is more reliable. The motion similarity between marker i and marker j at time t is the Euclidean distance between two motion vectors $\left\| v_i(t) - v_j(t) \right\|$.

A statistical approach is used to judge whether a marker's motion is a false tracking at time t. For each marker i, we first calculate the similarity of each neighbor and sort them in decreasing order. To avoid the judgment being contaminated by the motions of unknown false tracking of neighbors, only the first α% neighbors are included in the sample space Ω (α = 66.67, in our experiments). We assume that the vectors within the sample space Ω approximate a Gaussian distribution. The averages and standard deviations of x, y, and z components of v_j (for $j \in \Omega$) are then evaluated and denoted as (μ_{vx}, μ_{vy}, μ_{vz}) and (σ_{vx}, σ_{vy}, σ_{vz})

respectively. A tracked motion $v_i(t)$ is valid if it is not far from the distribution of most of its neighbors.

The judgment criterion of valid or false tracking for the marker i is

$$\begin{cases} JF(i,t) \le threshold & \text{, valid tracking} \\ \text{else} & \text{, false tracking} \end{cases} \tag{5.6}$$

and the judgment function is

$$JF(i,t) = \sqrt{S_x^2 + S_y^2 + S_z^2} \tag{5.7}$$

where $S_x = \dfrac{x_{vi} - \mu_{vx}}{\sigma_{vx} + k}$, $S_y = \dfrac{y_{vi} - \mu_{vy}}{\sigma_{vy} + k}$, $S_z = \dfrac{x_{vi} - \mu_{vz}}{\sigma_{vz} + k}$, and $v_i(t) = (x_{vi}, y_{vi}, z_{vi})$.

S_x, S_y, and S_z can be regarded as the divergence of v_i with respect to the refined neighbors Ω along the x, y, and z directions. If the divergences are within the standard deviations, the values are smaller than one; on the contrary, if the divergences are larger, the values increase. In Equation 5.7, k is a small user-defined number. With k in the denominators, we can prevent unpredictable values of S_x, S_y, and S_z when markers are close to their locations of the neutral face.

After we eliminate the false tracking of 3D candidates, a conflicting situation can still exist. Two valid motions that do not share the same 3D candidates could have the same extracted 2D feature points in either the frontal view or the side view. We call this the tracking conflict. To prevent the tracking conflict, we simply evaluate the number of valid motions for each 2D feature point If a 2D feature point is “occupied” by more than one valid motion, we only keep the motion closest to the prediction as a valid motion.

In our experiment, the average number of appearance of false tracking in a frame is 7.45%, and that of tracking conflict is 0.34%.

■ **Conjecturing positions of missing markers**

If a false tracking is detected, the similarity of its neighbors in motion can also be used to conjecture the position or motion of the missing marker. Based on this idea, two interpolation methods are applied to the estimation. The first one is the weighted combination method. For a missing marker i, the motion at time t can be estimated by weighted combination of that of its neighbors and it can be presented

by the equation:

$$v_i = \sum_j \left(\frac{1}{d_{ij} + kc} \right) v_j, \quad \text{for } j \in \text{Neighbor}(v_i) \tag{5.8}$$

where d_{ij} is the distance between m_i and m_j in the neutral face and kc is a small constant to avoid a very large weight when the marker i and j are quite close in the neutral face.

In addition, a RBF (radial basis function) based data scattering method is appropriate for the position estimation of missing markers. The above-mentioned weighted combination method tends to average and smooth the motions of all the neighbors; by contrast, the influence of nearby neighbors is greater in RBF interpolation in general (it depends on the radial basis function). And therefore, more prominent motions can be estimated. Details of the RBF interpolation are introduced in Chapter 7 for face deformation. Since the RBF interpolation is more time-consuming, the weighted combination is adopted for real-time or near real-time tracking. Figure 5.13 is the estimated motion by the proposed method; Figure 5.14 shows the tracking results by a method with Kalman filtering only and by our method with rectification of false tracking.

The complete procedure of automatic UV-responsive marker tracking is listed in the end of this chapter.

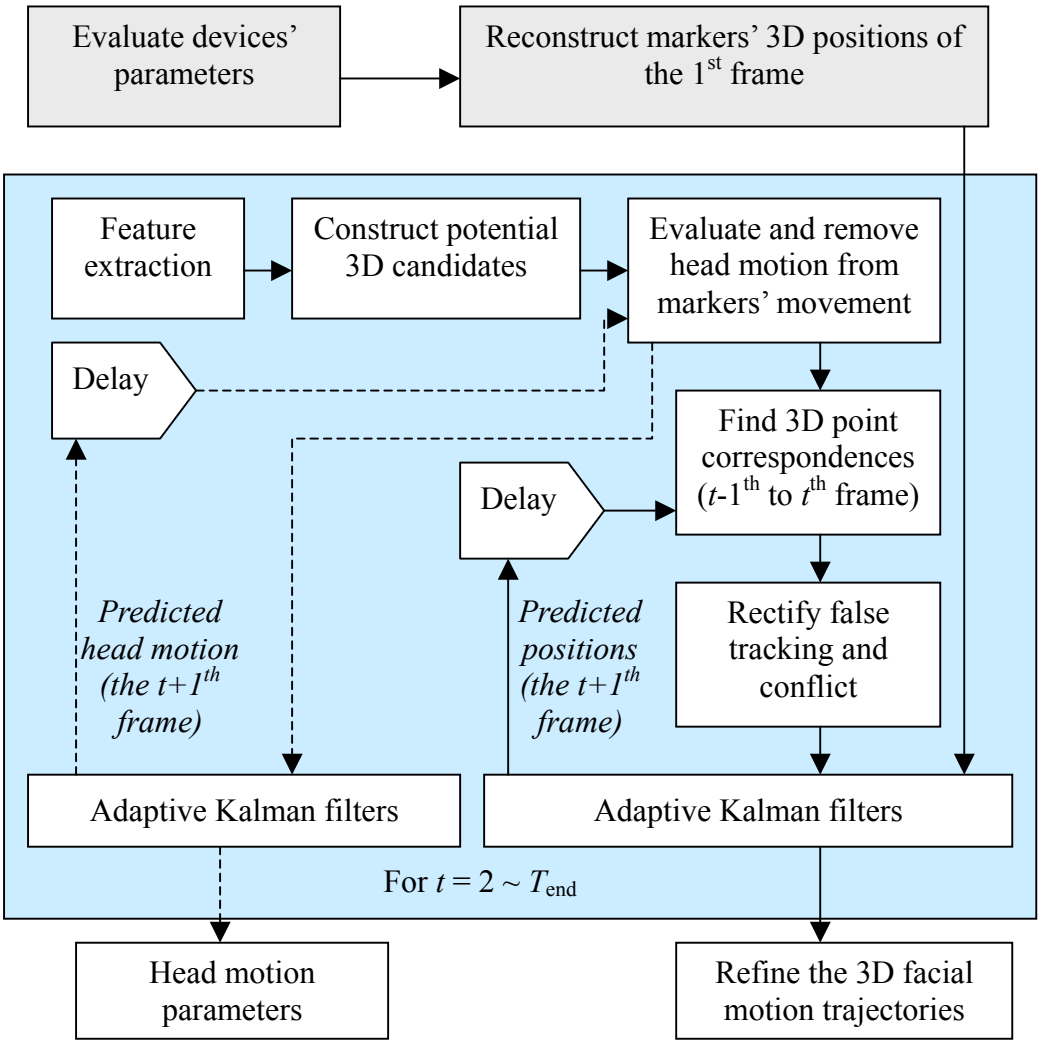


Figure 5.4 The flow chart of our automatic 3D motion tracking procedure for a large number of UV-responsive markers.

Figure 5.5 A 3D laser scanner is employed to acquire depth range images of a subject. We integrate two to three range scans for the 3D face structure of a subject.

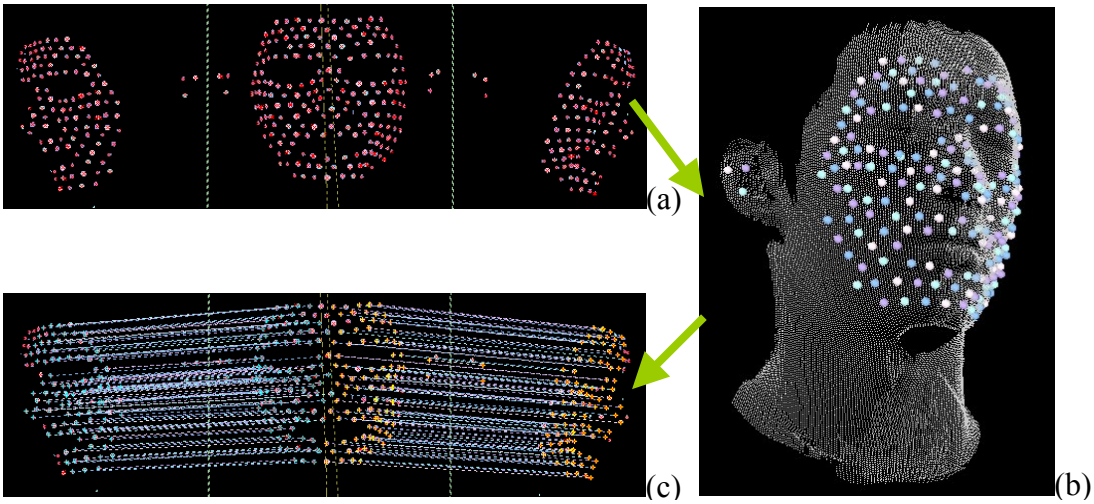


Figure 5.6 Recovering point correspondences with 3D scanned data and RBF interpolation.

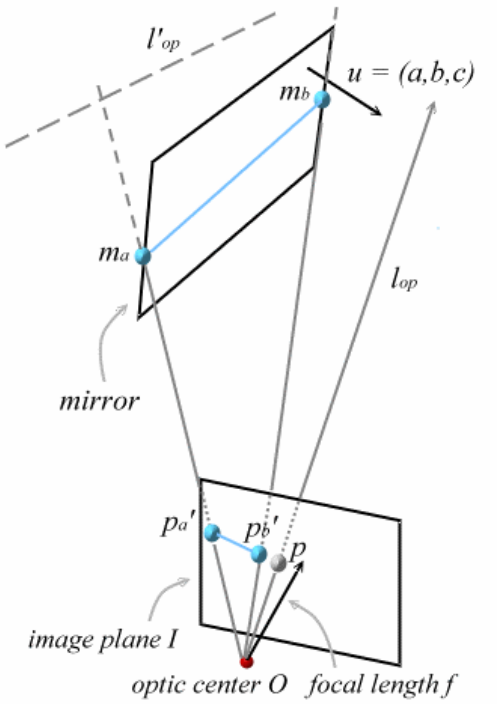


Figure 5.7 A conceptual diagram of "mirrored epipolar line". p is an extracted feature in the frontal view and l_{op} is the line extended by $\overrightarrow{op}$. l'_{op} is the line symmetric to l_{op} by the mirror plane. $\overline{m_a m_b}$ is the projection segment of l'_{op} on the mirror plane. $\overline{p_a p_b}$, the projection of $\overline{m_a m_b}$ on the image plane I, is the mirrored epipolar line segment of p.

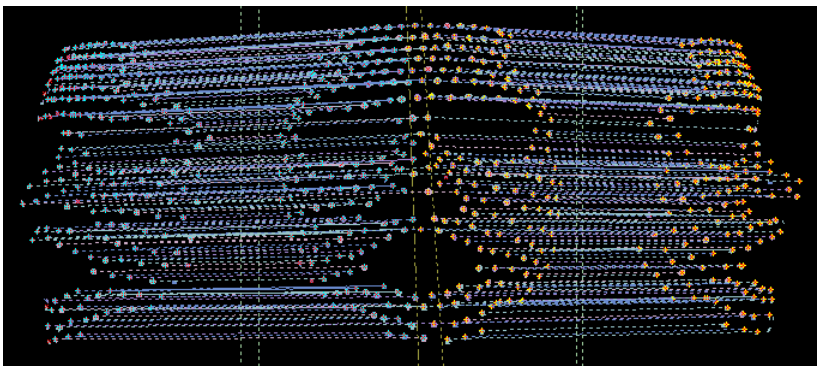


Figure 5.8 Candidates of point corresponding pairs under mirrored epipolar constraints. For each extracted feature in the frontal view, each feature point of the same color that lies within the mirrored epipolar band is regarded as a corresponding point.

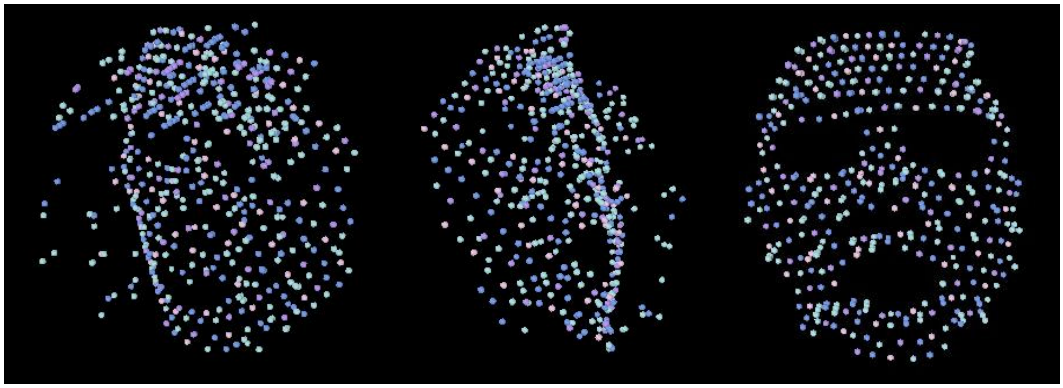


Figure 5.9 Potential 3D candidates generated under the mirrored epipolar constraint and the distance constraint. 3D candidates are first constructed from candidates of point correspondences; those whose positions are out of a bounding box are removed from the list of potential candidates.

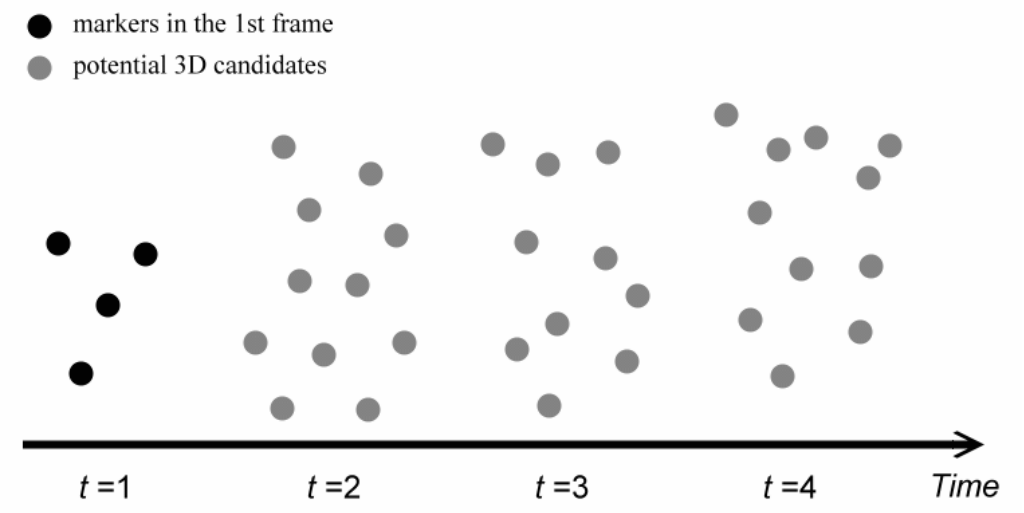


Figure 5.10 A conceptual figure for the problem statement of 3D marker tracking. The markers' 3D positions in the 1st frame are first evaluated. The goal of 3D motion tracking is to find frame-to-frame 3D point correspondence from potential 3D candidates. In our experiments, the number of potential candidates is around 1.2~2.3 times the number of actual markers. (The additional candidates can be regarded as false detection.)

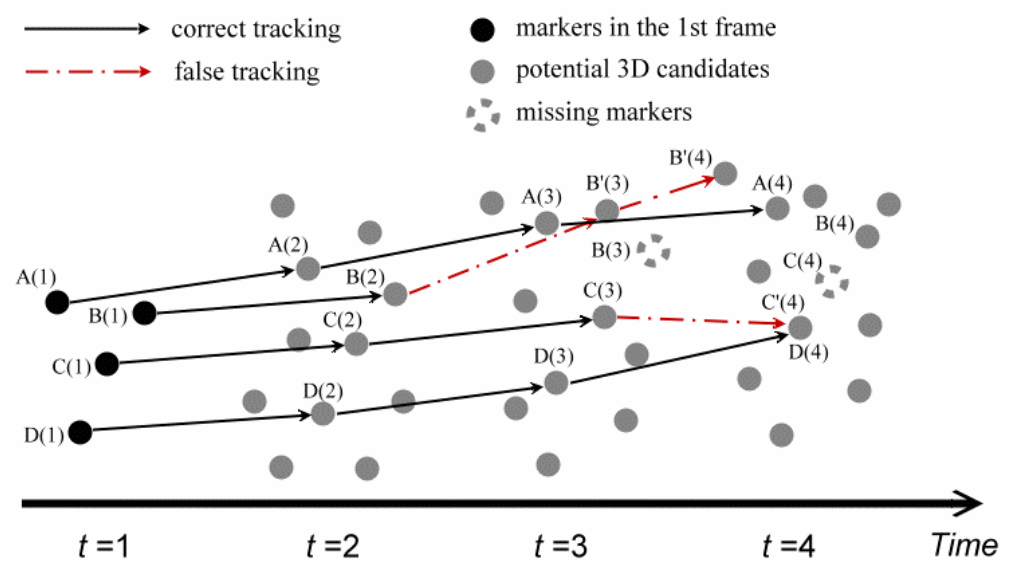


Figure 5.11 An example of false tracking due to missing markers. If there is no facility in judgment on false tracking, the motion tracking can degenerate immediately when missing markers appear.

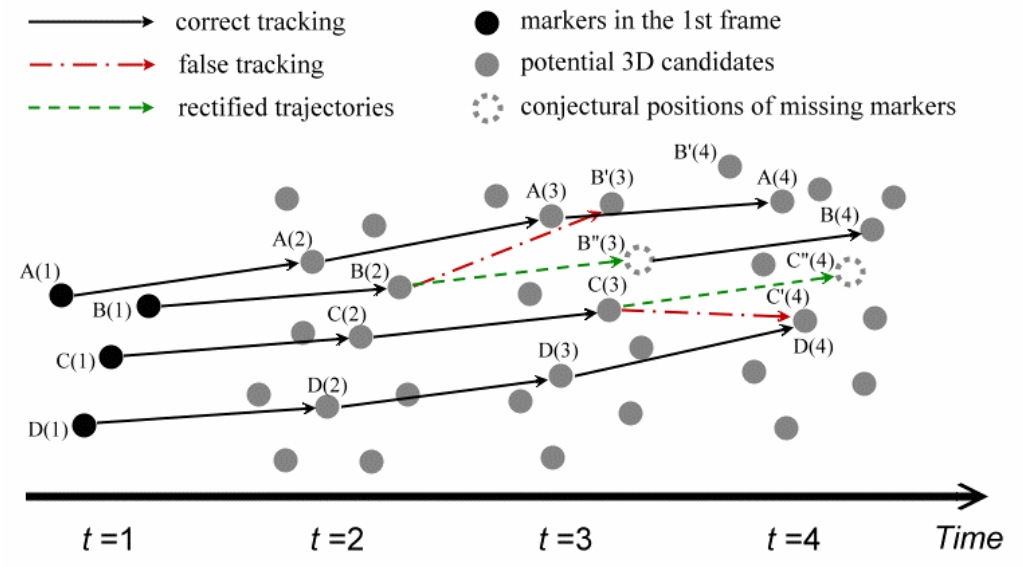


Figure 5.12 The rectified motion trajectories. We utilize the temporal coherence of a marker's motion and the spatial coherence between neighbor markers to detect and rectify false tracking.

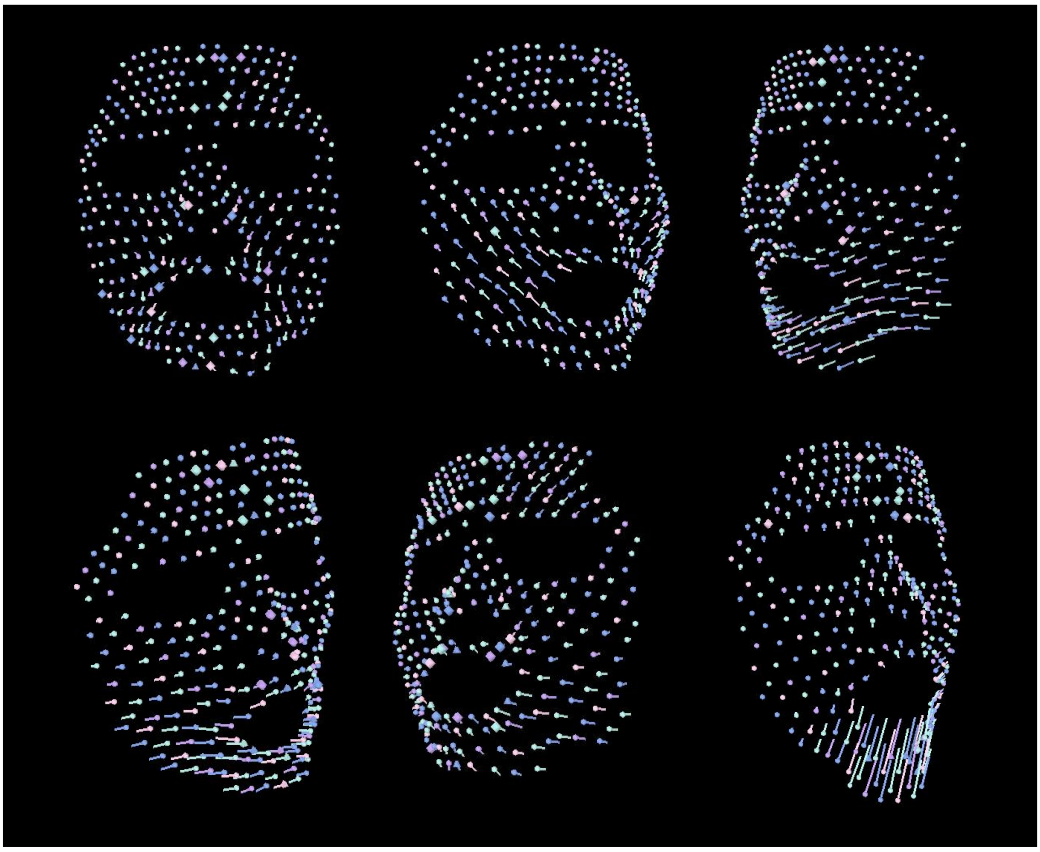


Figure 5.13 The motion flows of facial expressions. The 3D dots are the estimated markers' 3D positions and the colorful line segments are the difference vectors between an expression and the neutral face. The expressions from the upper-left to the lower-right are sadness, joy, the mouth twisting toward the right side, the mouth twisting toward the left side, pout, and the mouth wide opening.

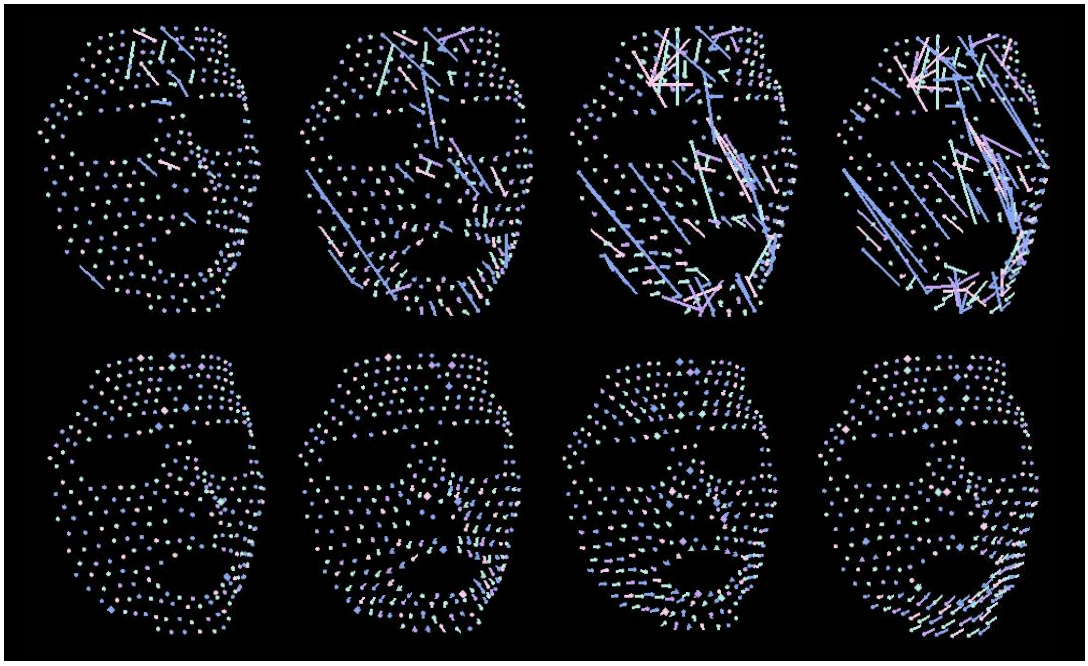


Figure 5.14 The tracking results without vs. with false tracking rectification. The upper part is the result tracked without false tracking detection; the lower part is the result tracked with our rectification method for false tracking. The snapshots from left to right are captured at t=20, t=100, t=200, and t=500. If no detection and rectification mechanism is used for false tracking, the result will degrade dramatically as time goes by.

The motion tracking procedure for mass 3D UV-responsive markers

Step 1. **Initialization of equipments' parameters.**

- □ Evaluate intrinsic parameters of the digital video camera [HEIK97] [ZHAN00]. (*All operations in the following steps are performed in the normalized camera coordinate.*)
- □ Estimate the mirror planes by the proposed method in Chapter 3.
- □ $t = 1$.

Step 2. **Reconstruction of markers' 3D structure in neutral face.**

If a depth range image of the subject is provided {

- □ Recover 2D point correspondences from depth range images generated by a 3D laser scanner.

} else {

- □ Find real versus mirrored point correspondences from a rough 3D face structure generated by rigid-body motion sequences.

}

- □ Estimate accurate 3D structure from real vs. mirrored point correspondences.
- □ For locations without depth images, recover the point correspondences and 3D position by RBF interpolation.
- □ Designate specific markers for head motion estimation (by users or the clustering methods)

Step 3. **Marker extraction in the frame.**

- □ $t = t+1$;
- □ Extract feature points by the method in Section 5.2.

Step 4. **Construction of 3D candidates by mirrored epipolar lines.**

- □ For each feature point in the frontal view, find the potential 2D point correspondence within mirrored epipolar bands.
- □ Estimate potential 3D candidates from potential point correspondence by the method proposed in Chapter 3.

Step 5. **Head motion estimation and removal.**

- □ Estimate 6-DOF head movement by the proposed process.
- □ Apply the inverse head motion to potential 3D candidates.

Step 6. **Finding frame-to-frame 3D point correspondences with outlier detection**

- For each marker *i*, predict $m_i(t \mid t-1)$ by the adaptive Kalman filter.
- Find the 3D candidates closest to $m_i(t \mid t-1)$ and denote it as $m_i(t)$.
- Detect the false tracking and set the false-tracking motion as an invalid one.
- Detect the tracking conflict and set the conflict motion as an invalid one.

Step 7. **Conjecturing positions of missing markers**

- For each missing marker (with an invalid motion at time *t*), estimate its position by the weighted combination or the RBF interpolation from the neighbors' motions.

STEP 8. **KALMAN FILTERING**

- Filter the estimated 3D positions with the adaptive Kalman filter.

If ($t \le T_{limit}$) {

- go to ***Step 3***

} else {

- Terminate the tracking process

}

Chapter 6

Experiments and Discussions

In this chapter, some discussions between the proposed 3D pose estimation approach and common-use stereovision approaches are presented. To compare the accuracy and error tolerance of each approach in various conditions, these approaches were utilized to estimate randomly generated and noise-corrupted 3D point sets in a virtual space. Moreover, an experiment was also performed for actual accuracy evaluation.

6.1 Concepts

Conceptually, estimating 3D position from mirror-reflected multi-view images should prove more robust than methods that estimate 3D position by calculating rotation matrix $\boldsymbol{R}$ and translation vector $\boldsymbol{T}$ between two cameras. Rotation and translation are both three degrees of freedom respectively. In our case, we evaluate the mirror plane normal $\boldsymbol{u}$ and scale d, which has only four DOF. Furthermore, without loss of generality, when we take c, the z-axial component in $\boldsymbol{u}$, in a negative direction and $||u||=1$ (as mentioned in Chapter 3), the degrees of freedom is just three. In general, fewer DOF mean we can use much less information to reach accuracy of the same magnitude.

Also, when estimating $\boldsymbol{R}$ and $\boldsymbol{T}$, we have to first evaluate the essential matrix, which has eight DOF, and then estimate an analogous rotation matrix $\boldsymbol{W}$. However, because $\boldsymbol{W}$ usually doesn't conform to a rotation matrix's properties, we must then further adjust $\boldsymbol{W}$ to fit the properties. We can then evaluate the vector $\boldsymbol{T}$. Each of these steps involves many numerical matrix computations, and errors accumulate

with each step. Therefore, the two-view linear algorithm yields distorted ***R*** and ***T*** estimations, necessitating successive nonlinear optimizations such as maximum-likelihood evaluations. J. Weng et al. discussed error analysis and 3D position estimation and structure reconstruction from stereovision approaches [WENG89, WENG93].

6.2 Error estimation by computer simulation and actual experiment

In order to compare the accuracy and robustness of the proposed approach with general-purpose stereovision approaches, computer simulations are used. The three subject algorithms are as follows:

Method(a). The proposed linear algorithm that reconstructs 3D positions via mirror-plane normal *U* evaluation.

Method(b). A two-view linear approach that estimates 3D positions via evaluating rotation *R*, and translation *T* between physical and virtual cameras.

Method(c). A nonlinear maximum-likelihood optimization that iteratively improves the result of the two-view linear approach.

Method (b)(c) are modified from J. Weng et al's research [WENG89][WENG93] by flipping the part of projected mirrored-reflected images to form the view of a virtual camera.

The four experimental computer simulations are as follows:

(i). Evaluating the relation between accuracy and the number of point correspondences. The conditions and results are shown in Figure 6.1.

(ii). Evaluating the robustness of each algorithm in different noise conditions. The conditions and results are shown in Figure 6.2.

(iii). Evaluating the effect of changes in image resolution for each algorithm. The conditions and results are shown in Figure 6.3.

(iv). Evaluating the effect of changes in mirrors' orientation (changes in included angles of two view directions) for each algorithm. The conditions and results are shown in Figure 6.4.

All these simulations were performed on a numerical computation software tool, MATLAB, Mathwork inc. [MATH].

We used hundreds of 3D point sets as testing objects. They are randomly generated within a 9,000 × 18,000 × 9,000 pixel cube and 40,000 pixels away from the optic center. For the second to the fourth tests, each testing set consists of 60 randomly generated 3D points. The simulated camera has a 720 × 720-pixel charge-coupled device (CCD) array and a 1,500-pixel focal length. Assuming the object is 2 meters away, one pixel length equals 0.05 mm.

For the first test, normal-distributed noise with constant variance was applied to simulating the sum of various kinds of noise disturbance in projection, and the contaminated projecting point data were then truncated to fit pixel grids on the CCD (charge-coupled device) array. Since the noise is random and with mean zero, the effects of perturbation can be alleviated in an overdetermined system. As shown in Figure 6.1, the more point correspondences are employed, the better estimation of 3D positions is. Therefore, to improve the accuracy of a 3D position reconstruction system, one can increase the number of point correspondences. Because the unknown parameters in U evaluation are of fewer degrees of freedom (DOF) then R, T ones, the proposed method can reach the same accuracy by much fewer point correspondences than the general-purpose two-view ones.

In the second test, the number of point correspondences is fixed at 60, and the standard deviations of noise varied from 0 to 3 pixles in both x-and y-axial directions. Similarity to the reason in the previous test, owing to the fewer DOF in unknown parameters, our method is more robust in the same noisy condition among three methods. As well, the third test also manifests that the proposed method can reach the same accuracy with lower resolution than linear or nonlinear virtual camera approaches.

In the fourth test, to evaluate the effects of view directions, we select 28 distinct points on a $^1/_4$ sphere. Then we take the tangent planes with respect to these 28 points as mirror planes. These points are on the cross points of the longitute 15, 30, 45 and 60 degrees and the latitude –45, –30, -15, 0, 15, 30 and 45 degrees. Each mirror orientation has a corresponding structure of real vs. virtual cameras. For example, the scene structure of the mirror plane tangent at the cross point (45^o, 0^o) is equivalent to that of of two cameras with a 90^o included angle in view directions. The result manifests that our method is more accurate that the other two methods for

various view directions. The situations for the other $^{3}/_{4}$ part of a sphere are similar and symmetric to the $^{1}/_{4}$ sphere we used.

We also developed an experiment to evaluate accuracy. We attached 20 markers, each 3 mm in diameter, to the right side of a plastic dummy's face and placed a planar mirror next to the right cheek. To mimic reality, the front and side views of the face's right side only occupied the full image's left half. Because a 3D laser scanner has a measurement error range of less than 0.5 mm, we assumed that it provided exact data. By comparing positions estimated using our method with the 3D scanned data, we found that our method's RMS 3D position error is 1.95 mm. The maximal error of 2.94 mm occurs at a marker position beneath the lower lip.

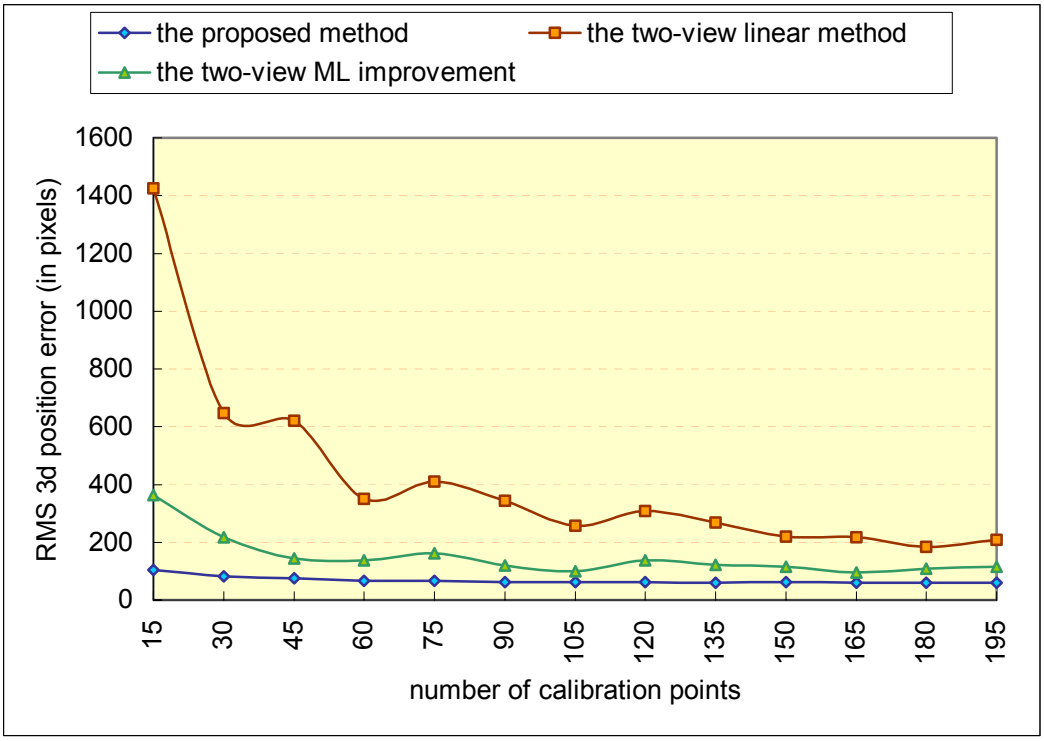


Figure 6.1 Computer-simulated error estimation of three approaches with different numbers of point correspondences. The testing subjects are sets of random-generated 3D points within a 9000×18000×9000 pixel^3 cubic and 40000 pixels away from the lens center. The simulated camera is with a 720×720 pixel^2 CCD array and a 1500-pixel focal length. Gaussian distributed noise (mean = 0, standard deviation = 1 pixel in both x-, y-axial directions) is applied to disturb the projection on the image plane before digitization of the CCD array.

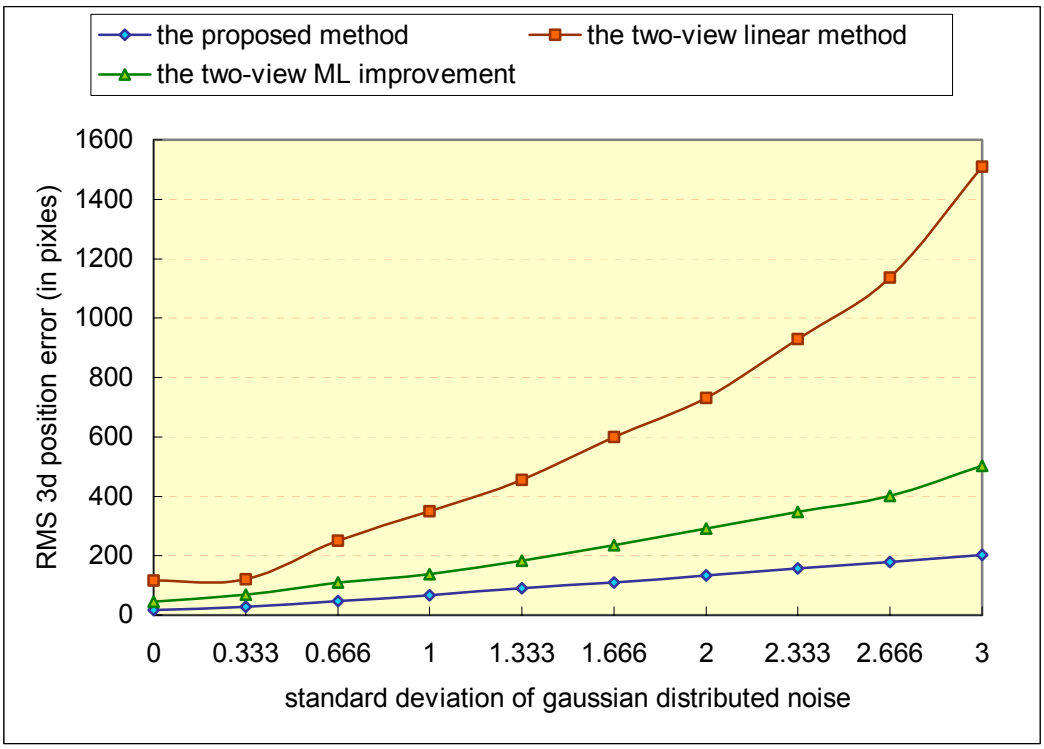


Figure 6.2 Computer-simulated error estimation of three approaches under Gaussian distributed noise of different standard deviations. Every time a testing subject is a set of 60 random-generated 3D points within a 9000×18000×9000 pixel^3 cubic and 40000 pixels away from the lens center. The simulated camera is with a 720×720 pixel^2 CCD array and a 1500-pixel focal length. Gaussian distributed noise (mean = 0) of different standard deviations (in both x-and y-axial directions) is applied to disturb the projection on the image plane before digitization of the CCD array.

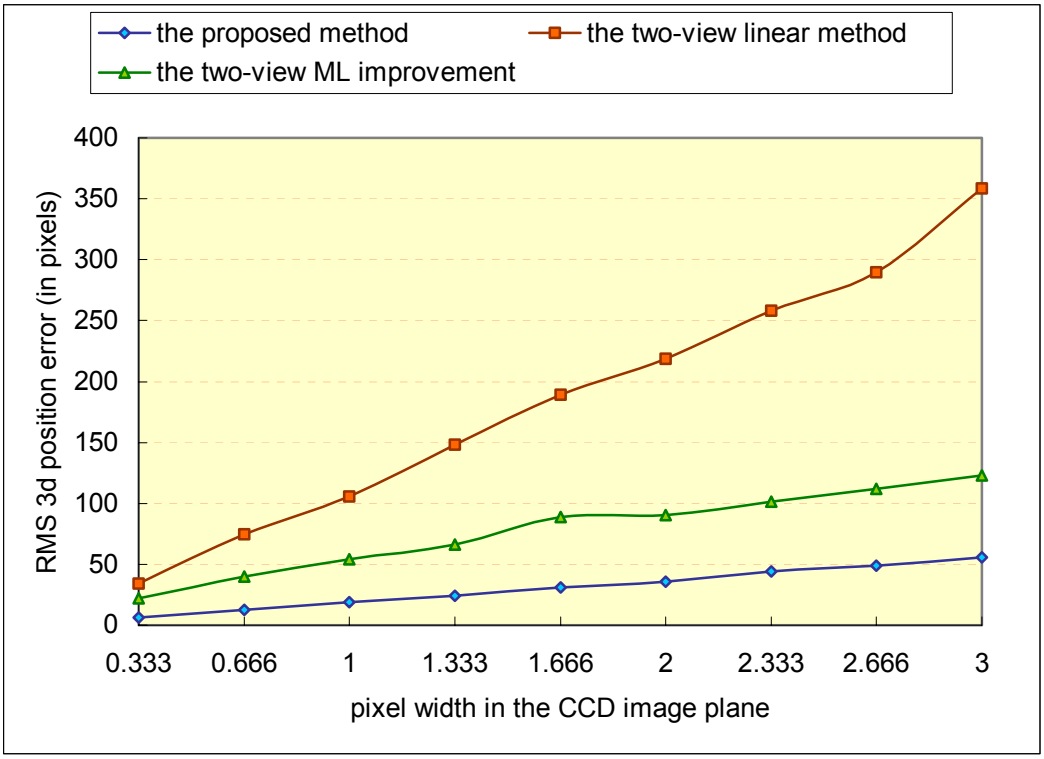
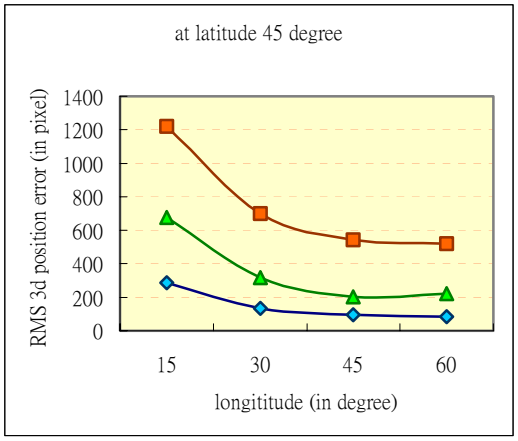
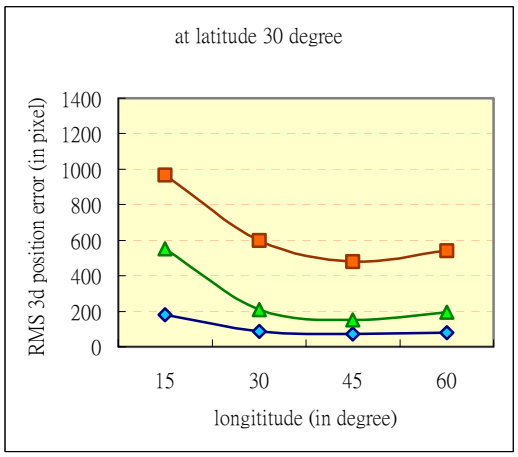
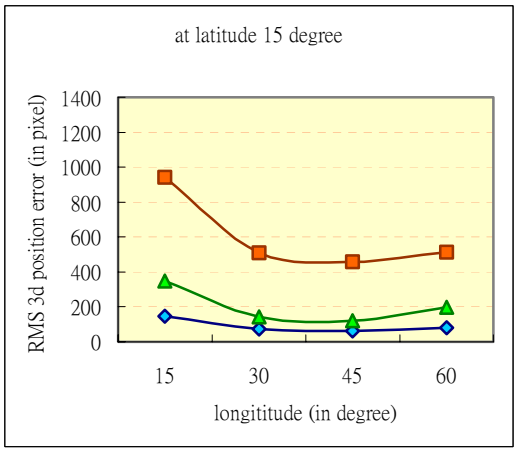
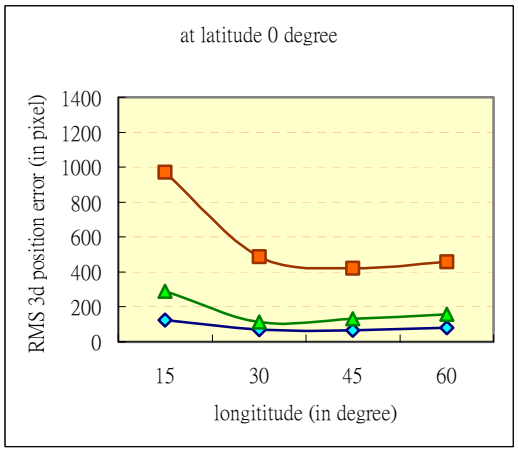


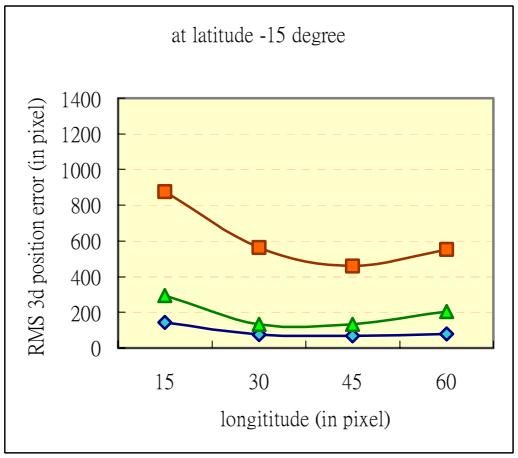
Figure 6.3 Computer-simulated error estimation of three approaches under different image resolutions. Every time a testing subject is a set of 60 random-generated 3D points within a 9000×18000×9000 pixel^3 cubic and 40000 pixels away from the lens center. The simulated camera has a focal length of 1500 pixels and has fixed image plane size but variable pixel widths. At pixel width 1.0, the CCD array is 720×720 pixel^2; at pixel width 0.333, the CCD array is about 2160×2160 pixels. No projection noise is applied.

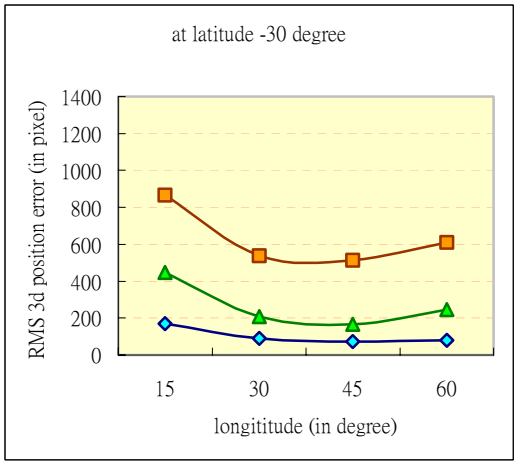

at latitude 45 degree
RMS 3d position error (in pixel)
1400
1200
1000
800
600
400
200
0
15
30
45
60
longitude (in degree)


at latitude 30 degree
RMS 3d position error (in pixel)
1400
1200
1000
800
600
400
200
0
15
30
45
60
longitude (in degree)


at latitude 15 degree
RMS 3d position error (in pixel)
1400
1200
1000
800
600
400
200
0
15
30
45
60
longitude (in degree)


at latitude 0 degree
RMS 3d position error (in pixel)
1400
1200
1000
800
600
400
200
0
15
30
45
60
longitude (in degree)


at latitude -15 degree
RMS 3d position error (in pixel)
1400
1200
1000
800
600
400
200
0
15
30
45
60
longitude (in pixel)


at latitude -30 degree
RMS 3d position error (in pixel)
1400
1200
1000
800
600
400
200
0
15
30
45
60
longitude (in degree)

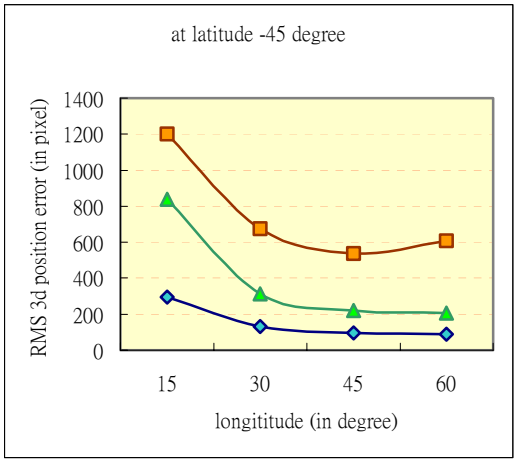


Figure 6.4 Computer-simulated error estimation of three approaches under different orientations of mirror planes (different included angles between two views). The orientations of mirror planes are defined by tangent planes of 28 selected points on 1/4 sphere. These points are on the cross points of the longitute 15, 30, 45 and 60 degrees and the latitude –45, –30, -15, 0, 15, 30 and 45 degrees. Every time a testing subject is a set of 60 random-generated 3D points within a 9000×18000×9000 $pixel^3$ cubic and 40000 pixels away from the lens center. The simulated camera is with a 720×720 $pixel^2$ CCD array and a 1500-pixel focal length. Gaussian distributed noise (mean = 0, standard deviation = 1 pixel in both x-, y-axial directions) is applied to disturb the projection on the image plane before digitization of the CCD array.

6.3 Discussions

■ Advantage

Compared to the commonly used stereovision approaches that adopt two-view images, our approach that estimates 3D positions and motions via mirror plane evaluation from mirror-reflected multi-view images has many advantages.

□ Simplicity and computational efficiency

In our algorithm, evaluating the mirror plane normal U requires solving only one equation by the linear least square evaluation, as shown in Equation 3.8, where the corresponding matrix M is n × 3. With the general-purpose two-view linear algorithm, however, estimating rotation matrix R and unit translation vector T_0 requires processing three linear least square evaluations, and their associated matrices are n × 9, 3 × 3, and 3 × 3. Furthermore, to obtain reasonable results, maximum-likelihood evaluation must be used. Because this optimization process is a kind of nonlinear iterative improvement, more computation results than that with a linear approach. For depth evaluation, both the proposed method and the two-view approach require another least square evaluation for each point correspondence.

□ Accuracy and robustness

Our method has four unknown parameters rather than the six of general-purpose two-view approaches. As mentioned in Section 6.1, the 4 unknown parameters can be further restricted in 3 DOF without loss of generality. We demand less information, such as fewer point correspondences to reach the same accuracy as with stereovision. Our method also has a larger error tolerance.

□ Perfect synchronization and low cost

Multiple-camera approaches face the critical problem of camera synchronization. In facial motion capture, the tip of a subject's lower lip moves down more than 1 cm within 30 ms when quickly pronouncing "pa," for example. When using only video-based synchronization, imperfect synchronization can make the expected measurement error of the lower-lip tip's position more than 0.5 cm. Therefore, accurate motion capture by multiple cameras demands special synchronization devices. In our approach, one camera and two mirrors can simultaneously capture three images of different viewpoints. Perfect synchronization

among multiple views is inherent in our system.

- **Disadvantages**

Our method has two main disadvantages:

- **Restricted measurement range**

Because our method uses a single camera to capture three different views simultaneously, measured targets' motion range must be within the volume of space between two mirrors. Mirrors' orientations and sizes therefore limit the method's applications.

- **Limited image area for each view**

Because our method includes three view images in a snapshot, each view can take up just one-third of the total image area.

However, our third computer simulation (as shown in Figure 6.3) demonstrates that our method offers similar or even better accuracy than the maximum-likelihood optimized two-view approach: it provides identical point correspondence but four times the image resolution (two times both pixel width and height).

- **Applications**

For motion tracking without limited action space, stereovision approaches employing multiple cameras remain flexible and irreplaceable. Synchronization hardware and camera calibration with a large number of point correspondences can probably overcome the disadvantages of difficult synchronization and higher noise sensitivity when using multiple cameras.

Nevertheless, our method provides a good and inexpensive solution in applications where motion ranges are restricted, such as 3D facial motion estimation, or finger or 3D hand gesture tracking. Because our method uses only a single camera and mirrors can reach high accuracy with few point correspondences, it doesn't require heavy calibration. This makes the proposed algorithm adequate also for applications requiring fast or even real-time dynamic calibration.

Chapter 7

Facial Animation and Applications

7.1 Introduction

In this chapter, our proposed work concerning facial animation is presented. Two facial animation systems are developed for different purposes. First, we propose our speech-driven talking head that is appropriate for the Internet in Section 7.2. Since the web-enabled talking head is animated by key-frame interpolation, and is driven by input speech, it requires only a very low bit-rate to deliver lifelike animation. For realistic facial animation, captured faces and motions from subjects are utilized to mimic or reproduce real persons' facial expressions. The processes of head modeling and applying our motion-capture data are presented in Section 7.3 and 7.4.

7.2 A Web-enabled Speech-driven Talking Head

An application that animates facial expressions through speech analysis is presented in this section. To animate facial animation according to input speech, key frames of facial expressions are prepared in advance. A speech analysis module is employed to obtain basic phonemes within the voice data. These extracted phonemes are then converted to the MPEG-4 Facial Animation Parameters (FAPs) to drive the 3D head model with corresponding facial expressions. The approach has been implemented as a real-time speech-driven facial animation system. When applied to Internet, our talking head system can be a vivid web-site presenter, and only requires 6 Kbps with an additional header image (about 30Kbytes in CIF

format, JPEG compressed).

The requirement of this proposed system could be stated as lifelike but low bit-rate animation data over Internet. A 2D image warping technique on a single face image was first employed in our previous talking head system [PERN98]. But the above animation is not very natural in rotation. When developing a system purely based on a 3D model, we can't overcome the problem of hair rendering, which is one of the most difficult issues in real-time computer graphics. Thus, we adopt a two and half dimension head model, which consist of a half-cut 3D model and an image plane (Figure 7.1, Figure 7.2) with a frontal view head image. With this extra image plane, our talking head can exhibit one's hair, neck, and smooth contours. The major advantage of this model is to combine both nice features from 2D mesh and 3D model: simple, vivid, and natural when a small-scale rotation less than 20 degree is applied.

In order to synthesize facial animation according to speech data, we have to know which utterances appear in the input voice data. In addition, the starting and ending time stamps of a certain utterances should also be obtained for synchronization of the mouth shapes with voice data.

For example, in Figure 7.3, it is the PCM data of a Chinese sentence "ni hau ma" spoken by the author. After getting this wave file, our system invokes a speech recognition engine and finds that from *StartTime* to *TimeA* is silence; *TimeA* to *TimeB* should be "ni"; *TimeB* to *TimeC* should be "hau"; *TimeC* to *EndTime* should be "ma". Our system then translates these results into "neutral (from 0 to *TimeA*), "n"-"i" (from *TimeA* to *TimeB*), "h"-"a"-"u" (from *TimeB* to *TimeC*), "m"-"a" (from *TimeC* to *EndTime*) and appropriate key frames are fetched from the expression pool.

Figure 7.4 is the flow diagram of speech-driven facial animation. First, voice data from a speech file or a microphone are fed to a speech recognition engine that helps us conjecture the phonemes within the speech. The engine compares the input voice data with its own database; then reports the most possible utterances and the time stamps of each utterance in the sequence. A mapping table translating utterances to phonetic notations is used to get essential mouth shapes. Therefore, we can get a sequence of basic mouth shapes according to the input speech data. With

this information, facial animation synchronized with the input voice data can be generated. For example, A Mandarin Chinese word “good” pronounced as /hau/ is converted to /h/+/au/, and the corresponding mouth motion is gradually morphed to “h” then transited to ”au”.

Since our purpose is to synthesize facial expressions according to speech data and many mouth shapes of utterances are quite similar to each other, the recognized results don’t need to have high recognition rate. Currently an efficient speech recognition engine can be used to generate facial animation in real time.

In general, the recognition rate for a speaker independent recognition engine is only around 60%. Since many different pronunciations have similar mouth shapes and the real difference is inside the mouth and not visible, the overall “viseme recognition rate” for this talking head system is higher than 90%. Actually, the visual effect is so strong that most people cannot see the difference.

In order to be web-enabled, a talking head system must have characteristics of very low bit-rate, short responsive time, and natural animation, and our speech-driven facial animation system conforms to the requirement exactly. The implementation details of the web-enabled speech-driven talking head are presented as follows.

Facial expressions of the proposed system are controlled by phonetic and emotion data that are key frame numbers and time-slice sequences. Speech data can be encoded by CELP (Code Excited Linear Prediction) coding techniques such as G.723.1. Thus, the bit-rate requirement of our proposed “VR Talk” is very low. To minimize the responsive time and make the animation perform smoothly, we adopt streaming framework with ring buffers to manage the data transmission on Internet. A VRT (VR Talk streaming data) format is also proposed, which includes data of the head model, facial animation control, and encoded speech. This format can be transformed to other streaming data format such as ASF (advanced streaming format) of Microsoft.

The system is separated into two parts: the server side and the client side. In the server side, a VRT file is prepared in advance. Our web-enabled VR Talk player is implemented as a plug-in for web browsers. When a user enter a web page with a link to a VRT file, our plug-in downloads the VRT data in streaming and plays back the speech with corresponding facial animation.

In our self-defined VRT streaming data, images and speech data are major parts of its data size. To reduce the VRT header size, we use the JPEG image coding approach to encoding the facial texture image and backgrounds. At this moment, the display window is of size 256 x 256 pixels. The size of a texture image or a background image is about 15K to 20K bytes, and the size of alpha blending mapping table is about 12K bytes. There are about 900 triangles in the generic head mask. Currently, we just store the triangle data without further encoding, and the triangle data size is about 70K bytes. To sum up, the VRT header size is about 120K bytes. The VRT packets are composed of recognized viseme data, talking head emotion index data, and encoded speech. The animation control data are about 600 bps, and the bit-rate of speech coding standard G.723.1 with silent detection is less than 5.3 Kbps. Thus, the total bit-rate requirement of VRT packet is about 6 Kbps.

Comparing with current video-phone coding techniques such as H.261 and H.263, whose bit-rate is about 40K to 4M bits per second in QCIF format, our proposed web-enabled talking head system can provide a low bit-rate and high-quality tool for video applications on Internet. For the time being, our system is developed on Windows 98/2000. Two kinds of web browsers, Microsoft Internet Explorer and Netscape Navigator are supported. On a Pentium III 500Mhz PC without OpenGL hardware acceleration, the frame rate is about 20 frames per second. However, once the OpenGL hardware acceleration is turned on, the frame rate can reach more than 300 frames per second.

Furthermore, the proposed web-enabled talking head technique has been licensed and improved by Cyberlink Corp., Taiwan in an industrial-academic collaborative project. The proposed VRT file format was also redesigned for Microsoft Advanced Streaming Format (ASF). The ASF format is a frame-based framework. Once a frame is received, the decoder must decompress the frame immediately and the raw data should be sent to the rendering filter at the next step. With this issue, not only the key frames of each viseme but also intermediate frames should be interpreted when data is encoded. In the ASF streaming file derived from VRT format, the facial animation controls contains FAPs only. 9 high-level FAPs (viseme, expression, eye lid motion, and head rotation) defined in MPEG-4 are comprised in the streaming packets. These raw high-level FAPs data are transmitted frame-bye-frame without compression (should be done as specified in MPEG-4 with DCT, or arithmetic coding) and the bit-rate of the animation control stream is about 8.6Kbps.

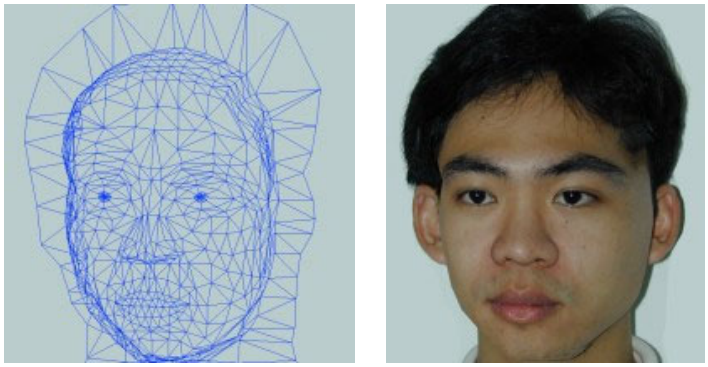


Figure 7.1 The two and half dimensional head model. The left image is the wire-frame display of the model and the right one is displayed with texture mapping.

Figure 7.2 Combination of 21/2D synthetic head with natural scenes. Alpha blending is employed to smooth the borders.

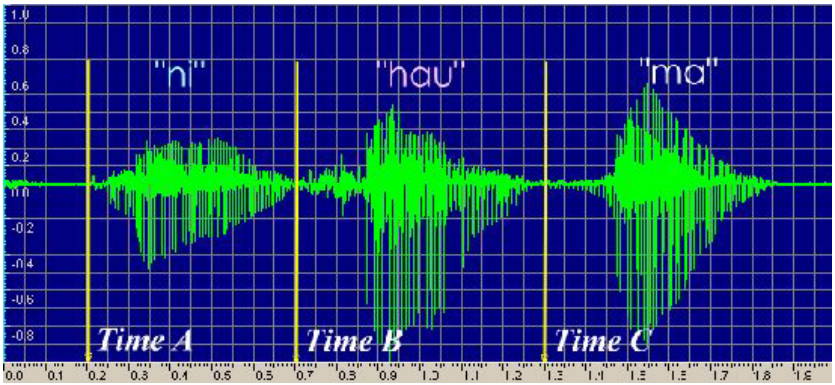


Figure 7.3 The speech data shape of a Chinese sentence pronounced as "*ni hau ma*" (means as "*How are you?*" in English). The vertical lines in the picture are marked to separate three different utterances (words) "*ni*", "*hau*", "*ma*".

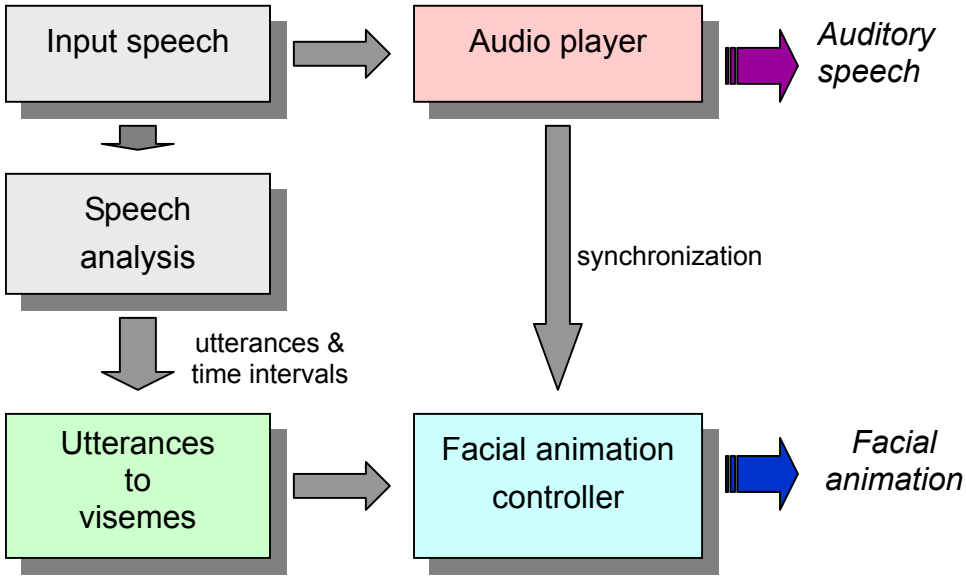


Figure 7.4 The flow chart of speech-driven facial animation.

Figure 7.5 The web-enabled speech-driven talking head "VR-Talk". An emotion index slid bar is provided to control the emotional facial expression, and the background can be changed dynamically.

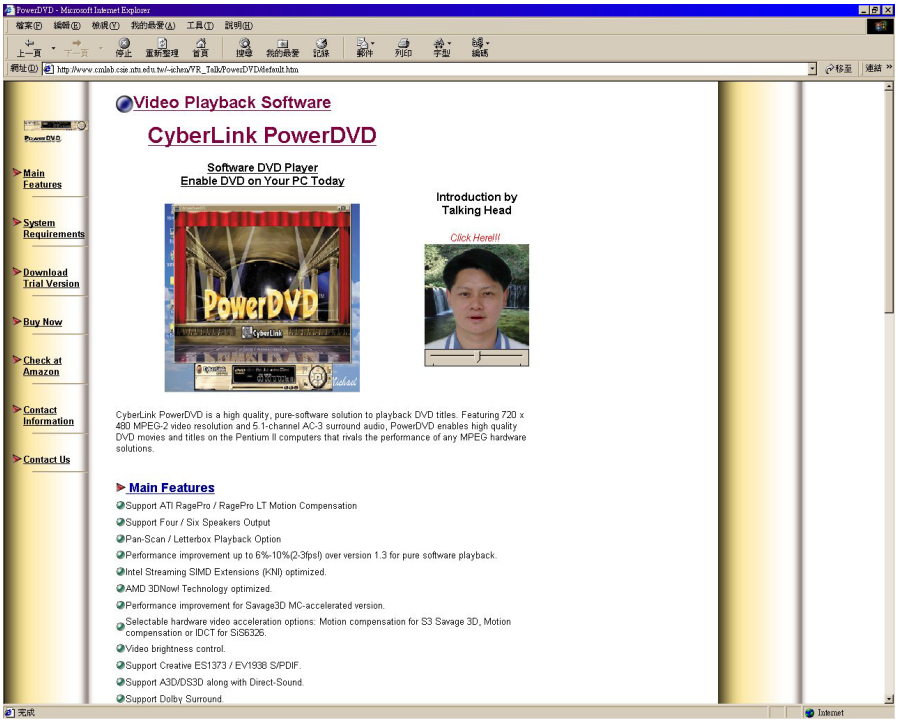


Figure 7.6 The VR-Talk is applied as a web-site narrator.
http:// www.cmlab.csie.ntu.edu.tw/~ichen/VR_Talk/VRTalk_Demo.htm

7.3 Face Synthesis

The proposed approach mentioned in Section 3.2 for 3D position estimation can also be applied to construct a realistic head model. However, a 3D scanner can provide 3D models of error less than 1 millimeter. Thus, we exploit a 3D scanner to acquire 3D head information. Nevertheless, the 3D scanned data cannot be applied for facial animation directly for three main reasons. The first reason is that the topology of a face model generated by a 3D scanner is arbitrary and does not fit the characteristics of human faces; for example, the topology on the lip portion should be distinct from that of the mouth portion. The second reason is that there are always a lot of "holes" in 3D scanned data. The third reason is that the number of polygons generated by a 3D scanner is considerably large, and that is too many for real-time animation. For these reasons, a generic face model with a designed polygon topology is employed and it is personalized for a subject by fitting to the 3D scanned range data.

Figure 7.7(a) is the figure of the generic model, and Figure 7.7(b) is the deformed model. In our current work, to personalized a face from 3D scanned range data, users have to manually specify the corresponding features such as the mouth corners, nose tip, eye corners etc. in the scanned face data. The deformation method we applied is the so-called RBF(radial basis function) based data scattering, which is a smooth interpolation function that can scatter the effects of feature points to non-recorded points. Supposed that m_i is the 3D position of feature point i, m_{oi} is the corresponding point on the generic model, and $u_i = m_i - m_{oi}$ is the displacement. We should construct a function that finds the unknown displacement u_j of unconstrained vertex j from u_i.

In our case, a method based on radial basis functions is adopted to represent the influence of constrained points. We chose $\phi(r) = e^{-r/k}$, where k is a user-defined constant (k=16, in our case). The data scattering function is then of the form

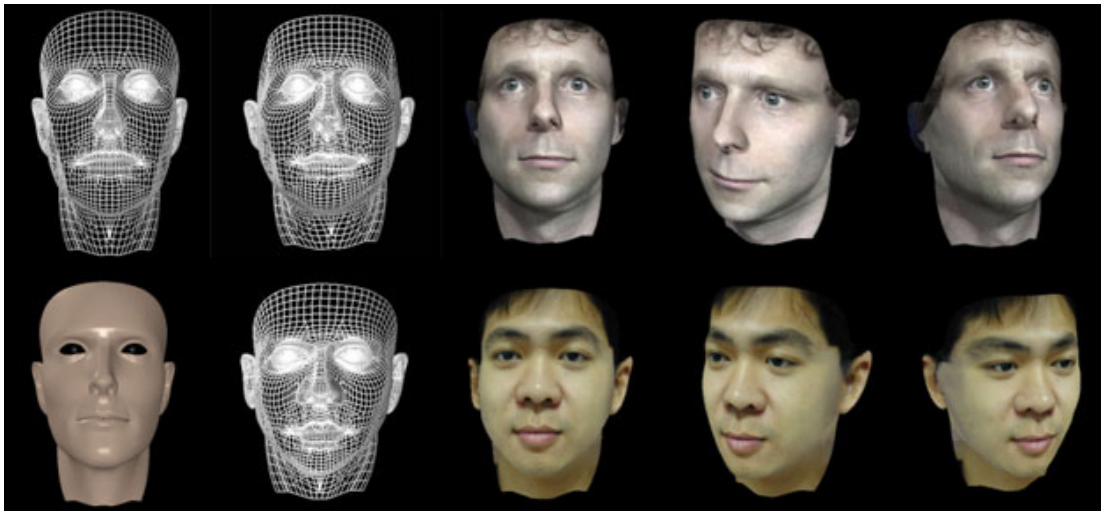
$$f(m) = \sum_i c_i \phi(\|m - m_i\|) + Mm + t \tag{7.1}$$

where m_i is the constrained vertex; low-order polynomial terms M=(M_1, M_2, M_3)^t, t are added as affine basis. Many kinds of functions for $\phi(r)$ have been introduced in Nielson's research [NIEL93].

To determine the unknown coefficients c_i and the affine components M and t, we must solve a set of linear equations that includes $u_i = f(m_i)$, the constraints $\sum_i c_i = 0$ and $\sum_i c_i m_i^t = 0$. In general, if there are n feature point correspondences, we will have n+4 unknowns and n+4 equations with the following form:

$$\begin{bmatrix} \cdot & \cdot & \cdots & & m_{1x} & m_{1y} & m_{1z} & 1 \\ \cdot & e^{-\|m_i - m_j\|/k} & \cdots & & m_{2x} & m_{2y} & m_{2z} & 1 \\ \vdots & \vdots & \cdot & & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \cdots & \vdots & m_{nx} & m_{ny} & m_{nz} & 1 \\ m_{1x} & m_{2x} & \cdots & m_{nx} & 0 & 0 & 0 & 0 \\ m_{1y} & m_{2y} & \cdots & m_{ny} & 0 & 0 & 0 & 0 \\ m_{1z} & m_{2z} & \cdots & m_{nz} & 0 & 0 & 0 & 0 \\ 1 & 1 & \cdots & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \\ M_1 \\ M_2 \\ M_3 \\ t \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{7.2}$$

where $1 \le i, j \le n; m_i = (m_{ix}, m_{iy}, m_{iz})$.



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

Figure 7.7 The personalized 3D face model with texture mapping. There are 6144 polygons and 5902 vertices on the face model. (a) the generic model. (b) the deformed model. (c)~(e) synthetic faces in different view directions.

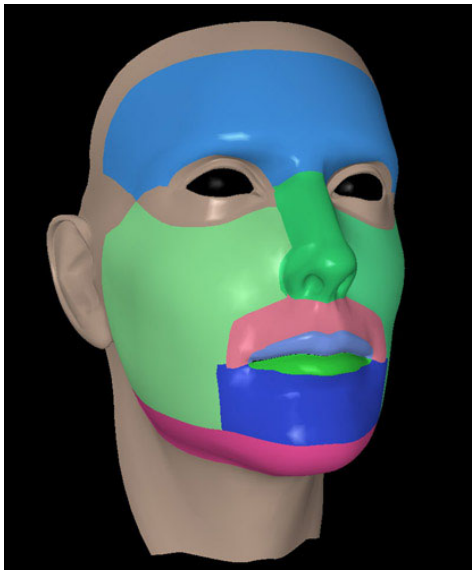


Figure 7.8 The 11 regions of a generic head model: jaw, lower mouth, lower lip, upper lip, upper mouth, left cheek, right cheek, nose, left eye, right eye, and forehead.

7.4 Facial Animation driven by Extracted 3D Facial Motion

A general face is separated into 11 regions: jaw, lower mouth, lower lip, upper lip, upper mouth, left cheek, right cheek, nose, left eye, right eye, and forehead (as shown in Figure 7.8). Control points within a region can only affect vertices in that region, and interpolation is applied to smooth the jitter effect at the boundary of two regions.

These control points consist of motion-capture feature points, "fixed points" and "supplementary hypothetical points". As mentioned in Section 4.1, feature points are the positions where markers are placed. "Fixed points" are the points where the position is always stationary no matter what the facial motion is performed, such as the points near ears and points near the bottom of the neck etc. "Supplementary hypothetical points" are the points difficult to be captured well due to video view point limitation; for example the points of jaw near the ear, etc. We use hypotheses to control the hypothetical points according to related feature points. Eyelids and some of the points on the jaw are hypothetical points. The blinks of eyelids are approximately once per 2.5 seconds as a random process. During blinking, the vertices on the eyelid move downward along the model of the eyeballs.

The action of the jaw is given as the following pseudo code:

If (the current jaw tip is higher than its position in the neutral face) {

Teeth should be clamped together.
Vertices of the jaw, except the neighbor area around the jaw tip, return to their neutral positions.

} else if (the current jaw tip is lower than its position in the neutral face) {

The jaw, which is now a rigid object, rotates and stretches around the hypothesis axes near the ears.

}

The inner lips represent another important and difficult-to-track facial region. A lip's inner surface, hidden behind the outer part in a neutral face, partially appears when the mouth is open. The lower inner lip is especially important when a mouth is puckering or rounding, as it does when pronouncing "u" or "o." At that time, almost half of the lower inner lip becomes visible, and it forms the lower lip's inner contour.

We therefore used a supplementary inner lip model, shown in Figure 7.9. The light green part represents the lower outer lip, driven by feature points in motion captured data, and the dark green part depicts the supplementary inner lip model, which is a modified Hermite surface controlled by outer lip and jaw surface tangent vectors.

After determining the displacement of all control points, a face can be deformed by the radial basis scattering function mentioned in Section 7.3. Once we repeat the above deformation process according to motion capture data frame by frame, we can generate realistic facial animation. Figure 7.10 demonstrates subtle facial motion of our synthetic face. Asymmetric facial expressions such as twisting the mouth can be synthesized because two mirrors are employed to capture the whole face's motion.

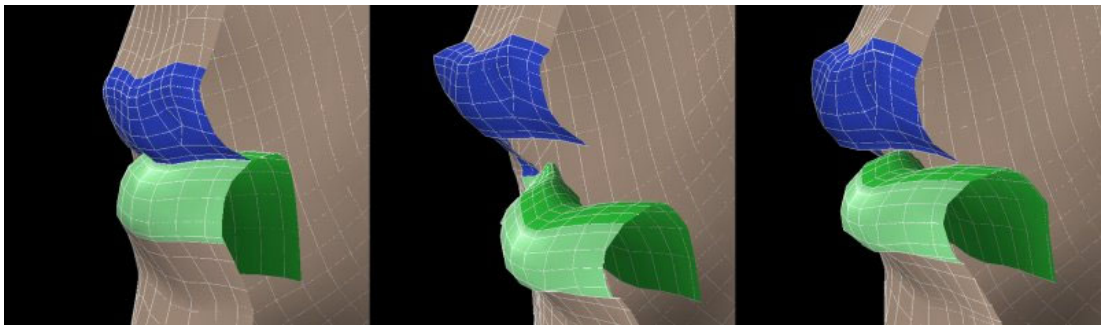


Figure 7.9 A cross-sectional view of lips. The lower lip is composed of the outer lower lip (light green), and supplementary rear lip model (dark green).

Figure 7.10 Synthetic subtle facial expressions of pouting and mouth twisting. Using two mirrors permits capturing asymmetric facial motions.

Figure 7.11 Synthetic subtle facial expressions of joy, sadness, anger, fear, and disgust.

Figure 7.12 The synthetic facial expressions of pronouncing “a-i-u-e-o”.

Figure 7.13 Applying captured facial expressions to others’ face models. The expressions from left to right are smiling, pouting, mouth twisting and eyebrow raising.

Chapter 8

Conclusion and Future Work

In this dissertation, a complete procedure to estimate accurate 3D facial motion trajectories from real and mirror-reflected views of video clips is proposed. No pre-calibration process for locations and orientations of the video camera and mirror planes is required. We discuss the benefits of the proposed 3D position estimation algorithm and compare with general-purpose two view methods. In the two-view algorithms, they reconstruct 3D structure via evaluation of rotation and translation between two views. This is a problem of 6 degrees of freedom (DOF). By contrast, the proposed algorithm uses symmetric properties of mirrored objects and we only require solving a mirror plane equation of 4 unknown variables. It can be further reduced to 3 DOF without loss of generality. Our computer simulations reveal that the proposed procedure can be more accurate and more robust than commonly used two-view algorithms in conditions of input noise or limited calibration information.

Under the normal light condition, our 3D motion tracking procedure, taking advantage of Kalman filters, requires a little user intervention to designate or correct false tracking. However, while tracking under blacklight blue lamps, we utilize the temporal and spatial coherence of mass UV-responsive markers on a face, and the problem of false tracking can be detected and rectified automatically. The proposed procedure can track 188 markers over 12.75 frames per second(fps) and track 300 markers over 9.2 fps on a Pentium-4 3.0GHz PC; it will soon be extended for real-time motion tracking.

The estimated facial motion parameters have been applied to our facial animation system, which can synthesize realistic facial expression with a frame rate of more than 30 frames per second on a Pentium-III 1GHz PC with the Nvidia

Geforce 2 MX OpenGL hardware acceleration. In addition, an application of a speech-driven talking head dedicated to the web environment is also presented. The talking head controlled by MPEG-4 high-level FAPs requires only 6Kbps bit-rate to "stream" lifelike facial animation through Internet. Currently, the web-enabled talking head has been implemented as ActiveX controls and plug-ins for web browsers including Internet Explorer of Microsoft and Navigator of Netscape. All the results are demonstrated in video, online plug-ins or executable programs on the websites:

RFAP: http://www.cmlab.csie.ntu.edu.tw/~ichen/RFAP/RFAP_Intro.htm
VRTalk: http://www.cmlab.csie.ntu.edu.tw/~ichen/VR_Talk/VRTalk_Demo.htm
MFAPExt: http://www.cmlab.csie.ntu.edu.tw/~ichen/MFAPExt/MFAPExt_Intro.htm

For the future work, besides the extension of our facial motion capturing system for live video tracking, the accuracy of feature extraction can be further improved. At this moment, the position of a marker in video clips is simply estimated from the center of its projected region. Some feature extraction techniques, such as refined corner detection, can have accuracy of 0.1 pixels. This will considerably improve the tracking results. Besides, statistical data association approaches, such as joint probabilistic data association, might improve or could be a comparison to our tracking method.

In the current work, we can approximate facial motion by tracking a large quantity of markers. Some subtle variations on a face, such as wrinkles or creases, are also conspicuous, but they are difficult to capture by motion capture techniques. Now, for generating more realistic facial animation, we start to research on the extraction and presentation of the facial expression details.

Acknowledgements

This work is a part of a collaborative project of INRIA, France, and National Taiwan University, Taiwan. We would like to acknowledge Nathalie Parlangeau-Valles, Yves Laprie, Dominique Fohr, and other researchers of the speech group of LORIA, France, for helping with our experiments in French. We'd especially like to thank Michel Pitermann, whose face data we captured for one of the models shown here. This research is in part supported by National Science Council in a project "Content Engineering: Research on MPEG-4/7 Multimedia Technologies" under the grant number NSC91-2622-E-002-040 and by MOE Program for Promoting Academic Excellence Universities under the grant number 89-E-FA06-2-4-8.

Bibliography

[ALTU95] Y. Altunbasak, A.M. Tekalp, and G. Bozdagi, “Simultaneous Stereo-motion Fusion and 3D Motion Tracking”, *Proc. International Conference on Acoustics, Speech, and Signal Processing 1995(ICASSP'95)*, vol.4, pp.2277-2280.

[ARUN87] K. S. Arun, T. S. Huang, and S. D. Blostein, “Least Square Fitting of Two 3D Point Sets”, *IEEE Trans. Pattern analysis and machine intelligence*, vol. 9, no. 5, pp. 698-700, Sept. 1987.

[BASU97] S. Basu and A. Pentland, “A Three-Dimensional Model of Human Lip Motions Trained from Video”, *Proc. IEEE Non-Rigid and Articulated Motion Workshop at CVPR’97*, pp.46-53, San Juan, June 16, 1997.

[BASU98] S. Basu, N. Oliver, and A. Pentland, “3D Modeling and Tracking of Human Lip Motions”, *Proc. International Conference on Computer Vision (ICCV’98)*, pp. 337-343, Bombay, India, Jan. 1998.

[BEIE92] T. Beier, and S. Neely, “Feature-based Image Metamorphosis”, *SIGGRAPH 92 Conference Proceedings*, pp. 35-42. ACM SIGGRAPH, July 1992.

[BLAN99] V. Blanz and T. Vetter, “A morphable model for the synthesis of 3D faces, “, *Proc. SIGGRAPH’99*, pp. 353-360, July 1999.

[BOZI79] S.M.Bozic. *Digital and Kalman Filter*, Edward Arnold Ltd, London, 1979.

[BRAN99] M. Brand. “Voice Puppetry”, *Proc. SIGGRAPH’99*, ACM SIGGRAPH pp.21-28, 1999.

[BREE85] M. Breeuwer, and R. Plomp, “Speechreading supplemented with formant-frequency information for voice speech”, *Journal of the Acoustical Society of America*, 77, pp. 314-317, 1985.

[BREG97] C. Bregler, M.Covell, M.Slaney. “Video Rewrite: Driving Visual Speech with Audio”, *Proc. SIGGRAPH’97*, pp.353-360, 1997.

[BOUG] J.-Y. Bouguet, Camera Calibration Toolbox for Matlab, http://www.vision.caltech.edu/bouguetj/calib_doc/

[BUCK00] K. Buckley, A. Vaddiraju, R. Perry, "A New Pruning/Merging Algorithm for MHT Multitarget tracking", *Proc. Radar-2000*, 2000.

[CAST90] D.A. Castañon, "Efficient Algorithms for Finding the K Best Paths Through a Trellis", *IEEE Trans. Aerospace and Elect. Sys.*, vol. 26, no. 2, pp.405-410, Mar. 1990.

[CHOE01] B. Choe, H. Lee, H.-S. Ko. "Performance-driven Muscle-based Facial Animation", *The Journal of Visualization and Computer Animation*, 12(2): 67-79, May 2001.

[COHE93] M.M. Cohen and D.W. Massaro. "Modeling co-articulation in synthetic visual speech", N.M. Thalmann and D. Thalmann, editors, *Models and Techniques in Computer Animation,* pp. 139-156, Springer-Verlag, 1993.

[COSA00] E. Cosatto and H. P. Graf, "Photo-Realistic Talking-Heads from Image Samples", *IEEE Trans. Multimedia*, vol. 2, no. 3, pp. 152-162, 2000.

[COSA98] E. Cosatto, H.P. Graf. "Sample-Based Synthesis of Photo-Realistic Talking Heads", *Proc. Computer Animation' 98*, pp. 103-110, Philadelphia, Pennsylvania, June 8-10, 1998.

[CYBE] Cyberware Corp. http://www.cyberware.com.

[DUPO00] S. Dupont and J. Luettin. "Audio-Visual Speech Modeling for Continuous Speech Recognition", *IEEE Trans. Multimedia*, vol. 2, No.3, pp.141-149, 2000.

[EKMA78] P. Ekman and W.V. Friesen. *Facial Action Coding System*. Consulting Psychologist Press, 1978.

[EZZA02] T. Ezzat, G. Geiger, T. Poggio, "Trainable videorealistic speech animation", *ACM Trans. Graphics (also in ACM SIGGRAPH'02),* vol. 21, issue 3, pp. 388 – 398, 2002.

[EZZA98] T. Ezzat and T. Poggio, "Miketalk: A talking facial display based on morphing visemes", *Proc. Computer Animation' 98*, pp. 96-102, June 1998.

[FACE] FaceStation, Eyematic Interface Inc. http://www.eyematic.com.

[FRAN03] A.R.J. François, G.G. Medioni, R. Waupotitsch, "Mirror symmetry => 2-view stereo geometry", *Image and Vision Computing*, vol. 21 pp. 137-143, 2003.

[GLUC99] J. Gluckman and S.K. Nayar, "Planar Catadioptric Stereo: Geometry and Calibration", *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR'99)*, vol. 1, 1999.

[GLUC02] J. Gluckman and S.K. Nayar, "Rectified Catadioptric Stereo Sensors", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, No. 2, pp. 224-236, 2002.

[GOLU96] G. Golub, and C. F. Van Loan, *Matrix Computation third edition*, The John Hopkins Univ. Press, Baltimore and London, 1996.

[GOTO01] T. Goto, S. Kshirsagar, N. Magnenat-Thalmann, "Automatic face cloning and animation using real-time facial feature tracking and speech acquisition", *IEEE Signal Processing Magazine*, Vo. 18, No. 3, pp 17-25, May, 2001.

[GUEN98] B. Guenter, c. Grimm, D. Wood, H. Malvar, F. Pighin. "Making Face", *Proc. Computer Graphics (SIGGRAPH '98)*, pp. 55-66, Aug. 1998.

[HARA92] R.H. Haralick, L.G. Shapiro, *Computer and Robotic Vision volume I*, Addison-Wesley Publishing Company, Inc., 1992.

[HARA93] R.H. Haralick, L.G. Shapiro, *Computer and Robotic Vision volume II*, Addison-Wesley Publishing Company, Inc., 1993.

[HART95] R. I. Hartley, "In Defence of the 8-point Algorithm", *Proceedings of IEEE*, pp. 1064-1069, 1995.

[HAVE96] R.M. Havey, T.M. Gavin, A.G.Patwardhan, K.P. Meade, "Methodology -Measurements, Part II: Instrumentation and Apparatus", *Journal of Prosthetics and Orthotics*, vol. 8, no. 2, pp.50-64, 1996.

[HEIK97] J. Heikkilä and O. Silvén, "A four-step camera calibration procedure with implicit image correction", *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, San Juan, Puerto Rico, pp. 1106-1112, 1997.

[HONG02] P. Hong, Z. Wen, T.S. Huang, “Real-time speech-driven face animation with expressions using neural networks”, *IEEE Trans. Neural Networks*, vol. 13, no. 4, pp. 916-927, Jul. 2002.

[HUAN94] T.S. Huang, and A.N. Netravali. “Motion and Structure from Feature Correspondences: A Review”, *Proceedings of the IEEE*, 82(2), pp. 252-268, Feb. 1994.

[HUYN99] D.Q. Huynh, “Affine Reconstruction from Monocular Vision in the Presence of A Symmetry Plane”, *Proc. Intl. Conf. Computer Vision 1999 (ICCV’99)*, pp. 476-482.

[ILLU85] *Illustrated Encyclopaedia of Science and Technology (Traditional Chinese ver.)*, Gruppo Editoriale FABBRI Editori S.P.A, Milan and Kwang Fu Book Co. Taipei, 1985.

[KALB01] G.A. Kalberer, L.V. Gool, “Face Animation Based on Observed 3D Speech Dynamics”, *Proc. Computer Animation 2001 (CA 2001)*, pp. 18-24, Seoul, Korea, Nov. 2001.

[KURA98] T. Kuratate, H. Yehia, and E. Vatikiotis-Bateson, “Kinematics-based synthesis of realistic talking faces”. *Proc. Intl. Conf. on Auditory-Visual Speech Processing (AVSP’98)*, Terrigal-Sydney, Australia, pp. 185-190.

[LEE95] Y. Lee, D. Terzopoulos, and K. Waters, “Realistic Modeling for Facial Animation”, *SIGGRAPH conference proceedings*, pp. 55-62, ACM SIGGRAPH, August 1995.

[LEE99] W. Lee, M. Escher, G. Sannier, N.Magnenat-Thalmann, “MPEG-4 Compatible Faces from Orthogonal Photos", *Proc. Computer Animation (CA’99)*, Geneva, Switzerland. pp. 186-194, May 1999

[LIU01] Z. Liu, Y. Shan, Z. Zhang, “Expressive Expression Mapping with Ratio Images”, *Proc. Computer Graphics (SIGGRAPH 2001)*, pp. 271-276, LA, US. 2001.

[LONG81] H.C. Longuet-Higgins. “A computer algorithm for reconstructing a scene from two projections”, *Nature*, 293:133-135, Sept. 1981.

[MATH] MATLAB, The Mathworks Inc. http://www.mathworks.com

[MITS92] H. Mitsumoto, S. Tamura, K. Okazaki, N. Kajimi, Y. Fukui, “3-D Reconstruction Using Mirror Images Based on a Plane Symmetry

Recovering Method", *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 14, No. 9, Sept. 1992.

[MPEG99] MPEG4 Video "Text for ISO/IEC FCD 14496-2 video", ISO/IEC JTC1/SC29/WG11 N3056, Dec. 1999.

[NIEL93] G.M. Nielson. "Scattered data modeling", *IEEE Computer Graphics and Applications*, vol.13, no.1, pp. 60-70, Jan. 1993.

[NOH01] J.Y. Noh, U. Neumann. "Expression cloning", *Proc. SIGGRAPH 2001*, pp. 277-288, Aug. 2001.

[OPTO] OPTOTRAK, Northern Digital Inc. www.ndigital.com/optotrak.html

[OSTE98] J. Ostermann, "Animation of Synthetic Faces in MPEG-4", *Proc. Computer Animation' 98*, pp.49-51, Philadelphia, Pennsylvania, USA, June 8-10, 1998.

[PAND99] I.S. Pandzic, J. Ostermann, D. Millen, "User evaluation: Synthetic talking faces for interactive services", *The Visual Computer*, vol.15: 330-340, 1999.

[PARK96] F. I. Parke, K. Waters, *Computer Facial Animation*, A K Peters, Wellesley, Massachusetts, 1996.

[PATT91] E.C. Patterson, P.C. Litwinowicz, N. Greene, "Facial Animation by Spatial Mapping", *Proc. Computer Animation '91*, N.M. Thalmann, D. Thalmann (Eds.), Springer-Verlag, pp 31 – 44, 1991.

[PERN98] W.-L. Perng, Y.-K. Wu, M. Ouhyoung. "Image Talk: A Real Time Synthetic Talking Head Using One Single Image with Chinese Text-To-Speech Capability". *Proc. PacificGraphics 98 (PG'98)*, pp. 140-148, Singapore, Oct 1998.

[PIGH98] F. Pighin, J. Hecker, D. Lischinski, R. Szeliski, D.H. Salesin. "Synthesizeng Realistic Facial Expressions from Photographs", *Proc. of ACM Computer Graphics (SIGGRAPH 98)*, pp. 75-84 Aug-1998.

[PIGH99] F. Pighin, R. Szeliski, D.H. Salesin, "Resynthesizing Facial Animation through 3D Model-Based Tracking", *Proc. Intl. Conf. Computer Vison (ICCV'99)*, vol. 1, pp. 143-150, 1999.

[REEV90] W. T. Reeves. "Simple and complex facial animation: Case studies." In

State of the Art in Facial Animation, SIGGRAPH' 90 Course Notes #26, pp. 88-106. ACM, Aug. 1990.

[SEIT96] S.M. Seitz, C.R. Dyer. "View Morphing", *Proc. SIGGRAPH 96*, ACM SIGGRAPH, pp. 21-30.

[SHAP] ShapeSnatcher, Eyetronics Inc. http://www.eyetronics.com.

[TALK] TalkingShow, Cyberlink Corp., http://www.gocyberlink.com.

[TEKA95] A.M. Tekalp, *Digital Video Processing*, Prentice Hall PTR, 1995.

[TERZ90] D. Terzopoulos and K. Waters. "Physically-based Facial Modeling, Analysis and Animation". *Journal of Visualization and Computer Animation*, 1(4): 73-80, March 1990.

[TIDD01] B. Tiddeman, M. Burt, D. Perrett, "Prototyping and transforming facial textures for perception research", *IEEE Computer Graphics and Applications*, vol. 21, no. 5, pp. 42-50, Sept.-Oct. 2001.

[TSAI87] R.Y. Tsai, "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses", *IEEE Journal of Robotics and Automation*, Vol. RA-3, No.4, Aug. 1987.

[TU03] P.-H. Tu, I.-C. Lin, J.-S. Yeh, M. Ouhyoung, "Expression Detail Mapping for Realistic Facial Animation", accepted to appear in *CAD/Graphics 2003*, Macao, China, Oct. 2003.

[VICO] VICON Motion Systems. http://www.vicon.com.

[VOWE] Vowels MRI Database by M. Hasegawa-Johnson, A. Alwan, J. Cha, S. Pizza, K. Haker. http://www.ifp.uiuc.edu/speech/mri .

[WATE87] K. Waters, "A Muscle Model for Animating Three-Dimensional Facial Expresson", *ACM SIGGRAPH'87*, vol.21, pp.17-24, July, 1987.

[WATE94] K. Waters and T. Levergood. "An Automatic Lip-Synchronization Algorithm for Synthetic Faces." *Proceedings of ACM Multimedia*, pp. 149-156, San Francisco, CA, USA, 1994, ACM Press.

[WENG89] J. Weng, T. S. Huang, N. Ahuja, "Motion and Structure from Two Perspective Views: Algorithms, Error Analysis, and Error Estimation", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11. No. 5,

pp. 451-476, 1989.

[WENG93] J. Weng, T. S. Huang, N. Ahuja, *Motion and Structure from Image Sequences*, Springer-Verlag, 1993.

[WILL90] L. Williams. “Performance-driven Facial Animation”. *Computer Graphics (Proc. SIGGRAPH’90)*, 24(4):235-242, Aug. 1990.

[WOLF89] J.K. Wolf, “Finding the Best Set of K Paths Through a Trellis with Application to Multitarget Tracking”, *IEEE Trans. Aerospace and Elect. Sys.*, vol. 26, no.2, pp. 287-296, 1989.

[WU00] J.-R. Wu and M. Ouhyoung, "On Latency Compensation and Its Effects for Head Motion Trajectories in Virtual Environments", *The Visual Computer*, vol. 16, no. 2, pp. 79-90, 200.

[YANG99] T.-J. Yang, I.-C. Lin, C.-S. Hung, C.-F. Huang and M. Ouhyoung. “Speech Driven Facial Animation”, *Proc. Eurographics workshop on Computer Animation and Simulation’99 (CAS’99)*, pp. 99-108, Milan, Italy, Sept. 1999.

[ZHAN92] Z. Zhang, and O. Faugeras, *3D Dynamic Scene Analysis*, Springer-Verlag, Berlin and Heidelberg, 1992.

[ZHAN95] Z. Zhang, R. Deriche, O. Olivier Faugeras, and Q.-T. Luong, “A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry”, *Artificial Intelligence*, 78, pp. 87-119, 1995.

[ZHAN98] Z.Y. Zhang, H.T. Tsui, “3D Reconstruction from a Single View of an Object and Its Image in a Plane Mirror”, *Proc. ICPR 1998*, pp. 1174-1176.

[ZHAN00] Z. Zhang. “A flexible new technique for camera calibration”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11): 1330-1334, 2000.

Appendix A

The Two-view 3D Position Estimation Algorithm

The following algorithm proposed by J.Weng et al. [WENG89][WENG93] is a common-use 3D position reconstruction approach that estimates 3D positions from two images of different view directions.

Without loss of generality, the focal length is set to be the unit of coordinates, and the camera-centered coordinate system is adopted, where the images of different view direction d_1, d_2 is regarded as a rigid-body motion of an object between t_1, t_2. The denotations mainly follow those used in the original paper.

$\mathbf{x_i} = (x_i, y_i, z_i)^t$ is the 3D position of point P_i at time t_1.

$\mathbf{x_i'}=(x_i', y_i', z_i')^t$ is the 3D position of point P_i at time t_2.

$\mathbf{X_i}=(u_i, v_i, 1)^t$ is the projected vector of P_i at time t_1.

$\mathbf{X_i'}=(u_i', v_i', 1)^t = (x_i'/z_i', y_i'/z_i', 1)^t$ is the projected vector of P_i at time t_2.

The two-view linear algorithm :

Step (1). Solving for essential matrix E.

$$A = \begin{bmatrix} u_1u_1' & u_1v_1' & u_1 & v_1u_1' & v_1v_1' & v_1 & u_1' & v_1' & 1 \\ u_2u_2' & u_2v_2' & u_2 & v_2u_2' & v_2v_2' & v_2 & u_2' & v_2' & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_nu_n' & u_nv_n' & u_n & v_nu_n' & v_nv_n' & v_n & u_2' & v_n' & 1 \end{bmatrix}$$

$\min_h \| Ah \| = 1$, subject to $\| h \| = 1$.

$$E = \begin{bmatrix} E_1 & E_2 & E_3 \end{bmatrix} = \sqrt{2} \begin{bmatrix} h_1 & h_4 & h_7 \\ h_2 & h_5 & h_8 \\ h_3 & h_6 & h_9 \end{bmatrix}$$

Step (2). Determining a unit vector T_s with $T^0 = \pm T_s$.

$\min_{Ts} \| E^t T_s \|$, subject to $\| T_s \| = 1$.

if($\Sigma_i (T_s \times X_i') \bullet (E X_i) < 0$), $T_s = -T_s$.

Step (3). Determining rotation matrix R.

$W = [(E_1 \times T_s + E_2 \times E_3)\ (E_2 \times T_s + E_3 \times E_1)\ (E_3 \times T_s + E_1 \times E_2)]$

$\min_R \| R - W \|$, subject to: R is a rotation matrix.

Step (4). Checking $T = 0$, If $T \neq 0$, determine the sign of T^0.

if $\dfrac{\left\| X_i' \times RX_i \right\|}{\left\| X_i' \right\| \cdot \left\| X_i \right\|} \leq \alpha$ for all $i = 1 \sim n$, then report $T \approx 0$.

else if ($\Sigma_i (T_s \times X_i') \bullet (R X_i) > 0$), then $T^0 = T_s$, otherwise $T^0 = -T_s$.

Step (5). If $T \neq 0$, estimate relative depths.

To find $Z_i = \left(\dfrac{z_i'}{\|T\|}, \dfrac{z_i}{\|T\|} \right)^t = \left(\tilde{z}_i', \tilde{z}_i \right)$ by $\min \left\| \begin{bmatrix} X_i' & -RX_i \end{bmatrix} Z_i - T^0 \right\|$.

The nonlinear optimization by maximum likelihood estimation:

The optimization first takes the result of the two-view linear algorithm as an initial guess; then it approximate the R, T by $\min_m \{ \| f(u,m) \| \}$ in a nonlinear least square approach, such as the Levenberg-Marquardt method, or the Gauss-Newton method.

$$f(u,m) = prj(m, y(u,m)) - u,$$

where u is the observed projected position, m is the motion parameters, y(u,m) is the best 3D positions of P, and prj(m, x) is the projected position of the input structure x and motion m.

Appendix B

High-level MPEG-4 Facial Animation Parameters

The followings are the definition of visemes and expressions, which are the high-level MPEG-4 facial animation parameters (FAPs) in MPEG-4 spec part 2: visual [MPEG99].

Table B.1 The definition of MPEG-4 visemes.

viseme_select	phonemes	example
0	none	na
1	p, b, m	put, bed, mill
2	f, v	far, voice
3	T,D	think, that
4	t, d	tip, doll
5	k, g	call, gas
6	tS, dZ, S	chair, join, she
7	s, z	sir, zeal
8	n, l	lot, not
9	r	red
10	A:	car
11	e	bed
12	I	tip

13	Q	top
14	U	book

Table B.2 The definition of MPEG-4 facial expressions.

expression_select	**expression name**	**textual description**
0	na	na
1	joy	The eyebrows are relaxed. The mouth is open and the mouth corners pulled back toward the ears.
2	sadness	The inner eyebrows are bent upward. The eyes are slightly closed. The mouth is relaxed.
3	anger	The inner eyebrows are pulled downward and together. The eyes are wide open. The lips are pressed against each other or opened to expose the teeth.
4	fear	The eyebrows are raised and pulled together. The inner eyebrows are bent upward. The eyes are tense and alert.
5	disgust	The eyebrows and eyelids are relaxed. The upper lip is raised and curled, often asymmetrically.
6	surprise	The eyebrows are raised. The upper eyelids are wide open, the lower relaxed. The jaw is opened.

Appendix C

Publication List

Journal papers

[2002] I-Chen Lin, Jeng-Sheng Yeh, Ming Ouhyoung, “Extracting 3D facial animation parameters from multiview video clips”, *IEEE Computer Graphics and Applications*, vol. 22, no. 6, pp. 72-80, Nov.-Dec. 2002.

[2000] Chien-Feng Huang, I-Chen Lin, Ming Ouhyoung, "High Resolution Calibration of Motion Capture Data for Realistic Facial animation", vol. 11, no. 9, pp.1141-1150, *Journal of Software*, ISSN 1000-9825, Sept. 2000, China Computer Federation.

International conference papers

[2003] Pein-Shien Tu, I-Chen Lin, Jeng-Sheng Yeh, Ming Ouhyoung,, “Expression Detail Mapping for Realistic Facial Animation”, accepted to appear in *Proc. CAD/Graphics 2003*, Macao, China, Oct. 2003.

[2001] I-Chen Lin, Jeng-Sheng Yeh, Ming Ouhyoung, "Realistic 3D Facial Animation Parameters from Mirror-Reflected Multi-View Video", *Proc. Computer Animation (CA 2001)* (ISBN 0-7803-7237-9), IEEE Computer Society, pp. 2-11, Seoul, Korea, Nov. 2001.

[2001] I-Chen Lin, Jeng-sheng Yeh, Ming Ouhyoung, "Extraction of 3D Facial Parameters from Mirror-Reflected Multi-view Video for Audio-Visual Synthesis", *Proc. Intl. Conference on Auditory-Visual Speech Processing*

(AVSP 2001) (ISBN 0-9712714-0-2) pp. 66-71, Aalborg, Denmark, Sept., 2001.

[2000] I-Chen Lin, Chien-Feng Huang, Jia-Chi Wu, Ming Ouhyoung, "A Low Bit-rate Web-enabled Synthetic Head with Speech-driven Facial Animation", *Proc. Eurographics Workshop on Computer Animation and Simulation (CAS 2000)* (ISBN 3-211-83549-0), Springer-Verlag, pp. 29-40, Interlaken, Switzerland, Aug. 2000.

[1999] Ming Ouhyoung, I-Chen Lin, David S.D. Lee, "Web-enabled Speech Driven Facial Animation", (invited speaker), *Proc. Intl. Conference on Artificial Reality and Tele-existence (ICAT'99)*, pp. 23-28, Tokyo, Japan, Dec. 1999.

[1999] I-Chen Lin, Chen-Sheng Hung, Tzong-Jer Yang, Ming Ouhyoung, "A Speech Driven Talking Head System Based on a Single Face Image", *Proc. Pacific Conference on Computer Graphics and Applications (PG'99)* (ISBN 0-7695-0293-8), IEEE Computer Society, pp. 43-49, Seoul, Korea, Oct. 1999.

[1999] Tzong-Jer Yang, I-Chen Lin, Cheng-Sheng Hung, Chien-Feng Huang, Ming Ouhyoung, "Speech Driven Facial Animation", *Proc. Eurographics workshop on Computer Animation and Simulation (CAS'99)*, Springer-Verlag, pp. 99-108, Millan, Italy, Sept. 1999.

[1998] I-Chen Lin, Li-Sheng Shen, Ming Ouhyoung, "Software Speedup of 3D Sound", *Proc. Intl. Symposium on Consumer Electronics (ISCE'98)*, pp. TPA2-17~22, Taipei, Taiwan, 1998.

Patent

專利類別	專利名稱	專利國家	專利號碼	專利發明人	專利權人	期間
(A)發明專利	使用語音與單一影像即時合成動態臉部表情的方法	中華民國	128951	歐陽明,彭偉倫, 林奕成,雷永威等	同左	2001/2/21~2019/4/12

(美國專利已申請正審查中)