The Hong Kong Polytechnic University

Department of
Electronic and Information Engineering

Efficient Algorithms for Human Face Modeling

Chow Tze Yin

A thesis submitted in partial fulfillment of the requirements for the Degree of Master of Philosophy

January 2005

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____(Signed)

CHOW TZE YIN (Name of student)

Abstract

The aim of this research is to develop efficient algorithms for modeling face images, which include human face detection, human face tracking, and human face modeling. Human face detection is the first step of all face processing system. In order to achieve a good detection performance level, the system needs to be able to detect the human face accurately and efficiently in a face image. In this thesis, an efficient algorithm for detecting human faces in color images is proposed. The first step in our algorithm is to segment out skin-color regions by using mean-shift algorithm. One of the major challenges to skin-color segmentation is that the human faces may be under poor lighting conditions or under varying lighting over a face region. Our approach considers the distributions of the color components of skin pixels under different illuminations, and the face color regions are identified with the maximum-likelihood technique. Then, the human eyes are detected with color information by using the mean-shift approach. Two eye candidates form a possible face region, which is then verified as a face or not by means of a two-stage procedure with an eigenmask. Finally, the face boundary region of a face candidate is further verified by a probabilistic approach in order to reduce false alarms. Once a face has been detected, it is then tracked in the subsequent video sequence. Facial features are represented using Gabor wavelets, and are then tracked with a modified greedy algorithm. An adaptive template matching method is proposed to adapt the changing appearances during tracking in order to combat the perspective variations of the human face under consideration. The construction of a 3D face model using a similarity measurement is also proposed. Without requiring prior camera calibration, the 3D face model is constructed based on multiple images. An iteration procedure is

proposed to estimate the depth of the 3D face model for a human face. Experimental results show that all of these algorithms can detect, track and construct human faces reliably and efficiently.

Acknowledgements

I would like to express my sincere gratitude to my Chief Supervisor, Dr. K. M. Lam. Without his support, this research work would not have been completed. He also offered me many invaluable ideas and suggestions for writing my thesis.

I would also like to thank all members of the DSP Research Laboratory, past and present. The countless discussions I had with them have proved to be both fruitful and inspiring.

I gratefully acknowledge the financial support of The Hong Kong Polytechnic University through the Research Grants Council of The Hong Kong Special Administrative Region, China (Project No. PolyU5123/01E) which made this research possible by granting me a studentship over the course of my degree program.

Finally, I would also like to take this opportunity to thank the Centre for Multimedia Signal Processing of the Department of Electronic and Information Engineering.

Table of Contents

List of Figures

List of Tables

Author's Publication

The following technical papers have been published or submitted for publication based on the result generated from this work.

Journal papers:

1. Tze-yin Chow, Kin-Man Lam and Kwok-Wai Wong, "An Efficient Color Face Detection Algorithm under Different Lighting Conditions", accepted to appear in *Journal of Electronic Imaging*.

Conference papers:

1. Tze-yin Chow and Kin-Man Lam, "Mean-Shift based Mixture Model for Face Detection in Color Image", presented in IEEE International Conference on Image Processing, pp. 601-604, Singapore.

Chapter 1
Introduction

The objectives of this chapter are to introduce the general concepts of human face modeling and the related processing algorithms, as well as their applications. In addition, a brief introduction to the human face detection algorithm, the face tracking algorithm and 3D face construction techniques proposed and developed in this thesis will be presented.

1.1 Motivation of Human Face Modeling

One of the most challenging and difficult problems in computer graphics and animation is the generation of a realistic 3D human face model, which is essential to computer games, film making, online chat, video conferencing, model-based video coding, etc. Different commercially available laser scanners can be used to capture the 3D structure of an object, which can then be used for 3D object modeling. However, laser scanners are expensive and the data are usually noisy. Prior to the animation of the human face model, hand touch-ups and manual registration are required. As the cost of computers, digital cameras and video camera is relatively low, there is a great interest in producing 3D human face models directly from a sequence of 2D images. For example, voxel-based reconstruction [61], object-based reconstruction [27], and image-based reconstruction [1, 70, 131] are existing methods to recover 3D coordinates from multiple 2D images. However, these methods require accurate camera calibration for the reconstruction. Most of the techniques available nowadays are manually intensive and computationally expensive. Therefore, in order to perform human face modeling, efficient algorithms that can detect and track human

facial feature automatically and construct the corresponding 3D model are indispensable.

1.2 Introduction to Human Face Modeling

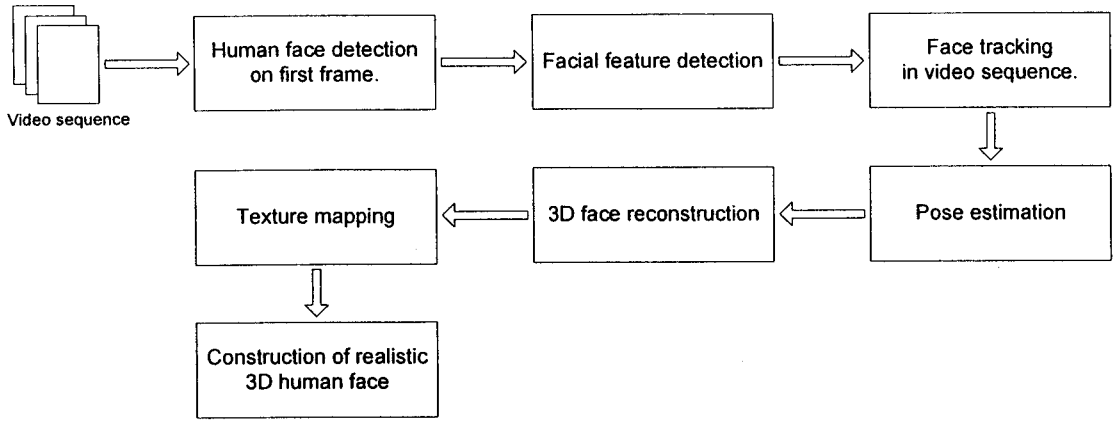


Fig. 1.1. Block diagram of a human face modeling system.

Compared to the use of a laser scanner, a human face modeling system consisting of a computer and a video camera is a cost-effective solution. The goal of the system is to construct a realistic texture-mapped 3D human face [6, 7, 10, 29, 38, 56, 129, 130] from a video sequence with minimal user interaction. Generally, the first step to be performed is to obtain the human faces from the first frame of a video sequence. The face regions should be searched in the video sequence, which may have a simple or a complex background. This is a challenging task because the human face is highly variable. The detection performance can be affected by the presence of glasses, different skin color, gender, facial hair, facial expressions, etc.

After detecting a face, its facial features, such as the eye corners, nose tip, mouth corners, etc., have to be detected. The location of these facial features can help improving the construction quality of the final face model. The active shape model (ASM) [50] can be used to search for the face contour and to adapt the facial features accurately. Once the facial features have been adapted, a face mesh, such as the

CANDIDE model [5], can be employed to model the human face. Based on the two base images and detected facial features, the system computes the face mesh geometry and the head pose. Corner detection, matching, motion estimation, 3D reconstruction and model fitting are involved in this stage.

Since human face is a 3D non-rigid object, there is little or no prior knowledge about the face in the scene and its movements. After the facial features have been detected, the head pose is tracked in the subsequent frames. In this stage, facial features are tracked to estimate the head pose movement. The CANDIDE model is then adapted to multiple 2D human faces at different perspective to estimate the human face depth information. The face model will be refined with curve fitting when more detailed depth information about the human face is available. The curve-fitting process adjusts the face mesh geometry to estimate the original curve of a human face from a video sequence. This curve-fitting process is a computational and time-consuming process, as extensive searching in the image space is required. Finally, a facial texture map is generated by blending all the images from the video sequence. The tracked face in the previous process is registered with facial features and corner matching. Therefore, a textured 3D human face model can be approximated.

1.3 Our Methods for Human Face Modeling

The objectives of this research are to investigate and develop effective techniques for efficient human face modeling. The human face is the most important object for many applications, such as model-based video coding, film making and computer games. Our human face modeling system focus on three major parts: human face detection, face tracking and 3D face construction. In our face detection

algorithm, skin colors ranging from poor lighting to extremely strong lighting are modeled by means of a mixture-of-Gaussian model. In the detection process, the mean-shift algorithm [17] is first used to segment an image into different color regions. Maximum-Likelihood is then applied to determine whether the regions are of a skin color or not. With the skin-color regions, possible human eyes are detected based on the color information and the eyes' characteristics. A possible face region is formed by pairing two possible eye candidates. These possible face candidates are then verified as a face or not by means of a two-stage procedure with an eigenmask. Finally, the face boundary region of each face candidate is further verified using a probabilistic approach to reduce the false alarms. After a face has been detected, it can be tracked in the subsequent video sequence. In our face-tracking algorithm, facial features including the left eye, right eye and mouth are represented by Gabor wavelet, and are tracked in the first stage. For efficient tracking, a triangular structure formed by the three facial features is considered and a greedy algorithm is employed in the searching process. Then, an adaptive template matching method, which can adapt to the changing appearance of the tracked face, is devised for tracking a region. Finally, a 3D face-construction method based on multiple 2D images is proposed. In our 3D face-construction algorithm, the depths of those important feature points in the 3D face model are estimated based on a multiple of 2D human face images. One advantage of our method is that prior camera calibration is not required. The CANDIDE model [5] is used as our 3D mesh to adapt to a human face. To construct the 3D model of a human face, an iteration procedure is proposed to minimize the depths so that the CANDIDE model can best represent the faces in views from different perspectives.

1.4 Organization of the Thesis

The rest of this thesis will give an overview of existing techniques for face detection, face tracking and 3D reconstruction, as well as the respective techniques devised and developed in this thesis. Chapter 2 will present the state-of-the-art technology for human face detection, face tracking and 3D reconstruction, as well as the problems associated with this technology. Chapter 3 outlines our efficient approach for human face detection. The techniques used include skin-color segmentation, possible eye candidate detection, a two-step eigenmask face verification, and face boundary verification. In skin-color segmentation, a mixture-of-Gaussian model is used to describe the skin color under different lighting conditions. The mixture parameters are determined by using the expectation-maximization algorithm. The mean-shift algorithm [17] is used to segment an input image into a number of color regions. Then, a region-based approach is proposed to segment skin color from a background under various illuminations. Possible eye candidates are detected by means of valley detection and mean-shift algorithm in color image. Possible face regions are then formed by pairing two possible eye candidates. A face verification method using an eigenmask is proposed to verify a face in two regions: the upper face region and the lower face region. Finally, the contour of a selected face is further verified by a probabilistic function. In Chapter 4, an effective face-tracking method will be described. An adaptive template-matching approach is proposed for face tracking. Facial features are tracked based on the facial features represented by the Gabor wavelet in the first stage. Finally, the adaptive template matching is used to verify the tracked face region. This adaptive template, which is based on a key-frame representation technique for video analysis, can adapt to the slowly changing appearance of a tracked region. This can make the tracking capable

of following the recent appearance of a tracked region, as well as the previous appearances of the region. In Chapter 5, a novel 3D face-construction algorithm is proposed. The depth estimation of a human face from multiple images without requiring prior camera calibration is proposed. The CANDIDE model [5] is used as a 3D mesh to adapt to the human faces. An iteration procedure is proposed to obtain an optimal structure which can represent the multiple training face images well. Finally, a summary of the major contributions and a conclusion to this thesis are provided in Chapter 6.

Chapter 2
Literature Review

2.1 Introduction

In this chapter, a brief survey of the current literature in face detection, face tracking and constructing of 3D face model is given. The human face is the first to be detected in our human face modeling system. Different human face detection methods are reviewed in Section 2.2. After a human face is detected, the motion of the face is tracked in the subsequent frames. The state-of-the-art tracking methodologies are discussed in Section 2.3. Since human face is a non-rigid 3D object, the appearance of a human face changes with poses, and expressions, etc. Finally, a 3D human face model is constructed by using information obtained from tracking. Recent methods for constructing 3D face models are presented in Section 2.4.

2.2 Detecting Faces in a Still Image

Visual analysis and object recognition are the most challenging tasks in computer vision. The computational models of automatic face recognition system are to develop the process of how the people recognize each other's face. A first step of any face processing system is detecting the faces location in images, if present. Face detection from a single image is a challenging task because the face is variability in scale, location, orientation, and pose. Facial expression, occlusion, and lighting conditions also change the overall appearance of faces.

Face Detection [125] is defined as: Given an arbitrary image, the goal of face detection is to determine whether or not there are any faces in the image and, if present, return the image location and extent of each face. A robust face detection algorithm has to combat with the following factors:

- **Pose.** The face in an image can vary due to the relative camera-face pose, such as frontal, 45 degree, profile upside down. Because of the pose change, some facial features such as an eye or the nose may be partially or wholly occluded.
- **Presence or absence of structural components.** Facial features such as beards, mustaches, and glasses may or may not be presented. In addition, these facial features have a great deal of variability such as shape, color and size.
- **Facial expression.** The appearances of face can be affected by with different facial expressions, such as smile, sad, disgusted, etc.
- **Occlusion.** Faces may be partially or wholly occluded by other objects, e.g. faces may be occluded by other faces in a picture with a group of people.
- **Image orientation.** Faces in an image can have different rotation about the camera's optical axis.
- **Imaging conditions.** The appearances of face can be affected by different lighting conditions, camera characteristic and quality of lens.

There are many related researches to face. Face localization [48], which is a simplified detection problem, searches the position of a single with the assumption that an input image contains one face only. Facial feature detection [19, 30] is to detect the presence and location of facial feature, such as eyes, nose, nostrils, eyebrow, mouth lips, ears, etc., with the assumption that there is a single face in an

image. Face recognition [13] searches for the most similar faces by comparing the input image and faces in a face database. Face authentication [108] is to verify the identity of an individual in the input image. Face tracking [21, 26] is to estimate the next location and orientation of a face in a video sequence. Facial expression [25] recognition is to identify the affective states of humans, such as happy, sad, disgusted, etc. Obviously, face detection is the first step in any face processing system.

Face detection methods based on learning algorithms have attracted much attention recently and have demonstrated excellent results. These methods are data driven which relies heavily on the training sets. Another issue is how to evaluate the performance of the proposed detection methods. Many recent face detection papers compare the performance of several methods in terms of detection and false alarm rates. Besides, many metrics have been adopted to evaluate algorithms, such as learning time, execution time, the number of training samples, as well as the ratio between detection rates and false alarms. Detection rate is defined as the number of faces correctly detected by the proposed algorithm. An image region is declared as a face by a detector if the image region covers more than a certain percentage of a face in the image. In general, there are two types of errors: *false negatives* in which faces are missed resulting in low detection rates and *false positives* in which a wrong image region is declared as a face. The detection rate can be improved or degraded by turning one's method parameters, thus, a fair evaluation should be considered during comparison between different methods.

There are more than 150 reported approaches for face detection. The research in face detection has broaden implications in computer vision research on object recognition. Most of the model-based and appearance-based approaches to 3D object recognition have been limited to rigid objects. These methods attempt to robustly

perform identification over a broad range of camera locations and illumination conditions. Face detection can be considered as a two-class, “face” or “non-face,” recognition problem. Consequently, face detection is to recognize a class of object from images with a great deal of within-class variability. This variability is captured by using large training sets of images, thus, a much broader class of recognition problem can be achieved.

The existing methods for face detection from a gray scale or color image are classified into four categories:

1. **Knowledge-based methods.** These are rule-based methods, which encapsulate human knowledge about a typical face, are highly related to facial features. These methods are designed mainly for face localization.
2. **Feature invariant approaches.** These algorithms aim to locate faces feature and its structure under different pose, viewpoint, or different lighting conditions. These methods are designed mainly for face localization.
3. **Template matching methods.** The face is described by several standard patterns in this matching method. The correlation between an input image and template are computed for detection. These methods can be used for both face localization and detection.
4. **Appearance-based methods.** Models or templates are learned from a set of training images. The model captures the representative variability of facial appearance and the used for detection. These methods are designed mainly for face detection.

Table 2.1 summarizes these four categories algorithms and their representative works for face detection in a single image. In this chapter, we would focus on the

motivation and general approach of feature invariant approach and appearance-based approach.

Approach	Representative Works
Knowledge-based	
	Rule-based multi-resolution method face detection for face detection[121].
Feature invariant	
• Facial Features	Edges grouping for face detection [51, 126].
• Texture	Space Gray-Level Dependence matrix (SGLD) for face detection [24].
• Skin Color	Mixture-of-Gaussian for skin color modeling[64, 123].
• Multiple Features	Skin color, size and shape integration for skin color detection [45].
Template matching	
• Predefined face templates	Shape template for facial feature detection [19].
• Deformable Templates	Active Shape Model (ASM) for face identification [50].
Appearance-based method	
• Eigenface	Eigenvector decomposition and clustering for face recognition [112].
• Neural Network	Ensemble of neural networks and arbitration schemes for face detection [85].
• Support Vector Machine (SVM)	SVM with polynomial kernel for face detection [72].
• Naive Bayes Classifier	Joint statistics of local appearance and position for object recognition [91].
• Hidden Markov Model (HMM)	Higher order statistics HMM for face detection[82]

Table 2.1. Face detection methods under different approach.

2.2.1 Knowledge-Based Top-Down Methods

The development ground of face detection methods are the rules that based on researcher's knowledge to a human faces. The features of a face can be described by simple rules. Basically, a face is composed with two eyes, a nose, and a mouth in an image. The relationships between features can be represented by their relative distances and positions. Therefore, the first step is to extract facial features in an input image, and follow by a face candidates verification based on the coded rules.

The difficulty of this approach is to translate human knowledge into well-defined rules. If the rules are too general, many false positives will be generated. If the rules are too strict, face detection will fail without passing through all the rules. Moreover, this approach has its limitation in detection under different poses and

various lighting conditions. It is difficult to include all possible cases to detect a face since those rules can violate each other under different conditions.

2.2.2 Feature Invariant Approaches

In contrast to the knowledge-based top-down approach, invariant features approach is another approach for face detection. The underlying assumption is to detect face in different poses and under different lighting conditions effortlessly as human beings. Thus, it requires some properties or features which are invariant over these variabilities. Numerous methods have been proposed to detect facial features and to verify the existence of a face. Edge detectors are commonly used to extract facial features such as eyebrows, eyes, nose, mouth, and hair-line. A statistical model is then built to describe their relationships and to verify the existence of a face. The problem of these feature-based algorithms is that the image features can be severely corrupted due to illumination, noise, and occlusion. The feature boundaries can also be weakened by the existence of shadows.

Facial Feature

Leung *et al.* [51] developed a probabilistic method to locate a face in a complex background based on local feature detectors and random graph matching. In their approach, the goal is to locate a face which is formulated by a certain arrangement of facial features that is more likely to be a face. A typical face is described by five features, two eyes, two nostrils, and nose/lip junction. Relative distance is computed between any pair of facial features of the same type, such as, left-eye, right-eye pair. The relative distances are then modeled by a Gaussian distribution. A facial template is defined by averaging the responses to a set of multi-orientation, multi-scale

Gaussian derivative filters over a number of faces in a data set. Given an input image, possible facial features candidate in the input image are first identified by matching the filter response at each pixel against a template vector of responses. The top two strongest responded possible features are selected to search for the other facial features. The facial features cannot appear in an arbitrary arrangements, thus, a statistical model of mutual distances can be used to estimate the expected locations of the other features. Moreover, because of the covariance of the estimated candidates, the expected feature locations can be estimated with high probability. Constellations are then formed only from candidates that lie inside the appropriate locations. The most face-like constellation is then determined. Random graph matches the best constellation in which the nodes of the graph are correspondent to face features. The arcs represent the distances between different features. Furthermore, the constellation is ranked by a probability density function where the probability measures the constellation corresponds to a face versus the probability it was generated by an alternative mechanism, i.e., non-face. In their experiments, 150 images were used and in which a face was declared to be correctly detected if three or more features were correctly located on the faces. This system is able to achieve a correct localization rate of 86 percent.

Skin Color

Human skin color is an effective tool in many applications from face detection to hand tracking. Although the human skin color has great variation between different races, several studies have shown that the major difference largely lies between their intensity rather than their chrominance [30, 31, 123]. Several color spaces have been utilized to label pixels as skin including RGB [38, 39, 89], normalized RGB [20, 21,

43, 66, 71, 80, 103, 105, 122, 123], HSV (or HSI) [45, 90, 101, 102], YCrCb [12, 113],YIQ [23, 24], YES [86], CIE XYZ [14], and CIE LUV [124].

Color information is an efficient tool to identify facial areas and specific facial features if the skin color can be modeled properly to adapt different lighting conditions. However, such skin color models are effective conditionally that the result can be affected significantly by the spectrum of the light source variation. In other words, color appearance is often unstable due to changes in both background and foreground lighting. McKenna *et al.* [65] proposed an adaptive color mixture model to track faces under varying illumination. A stochastic model estimates an object's color distribution online and adapts to accommodate changes under varying lighting conditions. Experimental results show that their system can track faces under varying illumination, however, this method cannot be applied to detect faces in a single image.

Skin color cannot be used alone to detect or track faces, thus, several modular systems [31, 65, 102, 123, 124] have been developed to combine shape analysis, color segmentation, and motion information for locating or tracking heads and faces in an image sequence.

Multiple Feature

Recently, numerous face detection approaches that combine several features have been proposed. Features like skin color, size, and shape are utilized to find possible face candidates in the first step. These candidates are then further verified by using local, detailed features such as eye brows, nose, and hair. Connected component analysis or clustering algorithms are used to group skin-like pixels

together. If the shape of a connected region appears as an elliptic or oval shape, it becomes a possible face candidate. Finally, local features are used for verification.

Shape and color are used for face localization and facial feature extraction in [102]. Initially, skin color segmentation is performed in HSV color space. At a coarse resolution, connected components are then determined by region growing. The best fit ellipse is computed using geometric moments for each connected component. Face candidates are selected if connected components are well approximated by an ellipse. Subsequently, facial features would be searched inside the connected components for further verification.

A Gaussian skin color model is used to classify skin color pixels in [109, 110]. A neural network is trained to detect face with the extracted geometric moments. Their experiments show a detection rate of 85 percent over 100 images test set.

2.2.3 Template Matching

In template matching, a standard face pattern, which is usually frontal face template, is manually predefined by a function. Given an input image, the correlation between the standard face patterns is computed for the face contour, eyes, nose, and mouth independently. The existence of a face is determined based on the correlation value. The advantage of this approach is its simple implementation, however, the face detection is not completely effective with variation in scale, pose, and shape. In order to achieve scale and shape invariance, multi-resolution, multi-scale, sub-templates, and deformable templates have been proposed.

Predefined Templates

Sakai *et al.* [87] proposed to detect frontal faces in an image in 1969. Each feature, such as eyes, nose, mouth, and face contour, was modeled by a sub-template. Each sub-template is defined in terms of line segments. Lines in the input image are extracted based on greatest gradient change and then matched against with the sub-templates. The correlations between sub-images and contour templates are computed to determine the location of possible face candidates. Then, the matching with the other sub-templates can be performed at the position of possible face candidates.

Deformable Templates

Yuille *et al.* [127] used deformable templates to model facial features that fit a prior elastic model, such as eyes, mouths, etc. In this approach, parameterized templates are used to describe facial features. An energy function is defined to measures edges, peaks, and valleys in the input image which corresponds to the parameters in the template. By minimizing the energy function parameters, the best fit of the elastic model is found. Their experimental results demonstrate good performance in tracking non-rigid features, however, the major drawback of this approach is that the deformable template must be initialized near to the object of interest.

2.2.4 Appearance-Based Methods

This approach is similar to template matching. The “templates” or models in appearance-based methods are learned from a set of training images. In general, appearance-based methods rely on techniques from statistical analysis and machine learning to determine the relevant characteristics of face and non-face images. The learned characteristics are embedded in the form of distribution models or

discriminant functions that are used for face detection. Due to the high-dimensional representation of the model, dimensionality reduction is usually carried out for computation efficiency and detection efficacy.

Most of the appearance-based methods are presented in probabilistic framework. An image region or feature vector derived from an image is represented as a random variable x. This random variable is characterized to "faces" and "non-faces" by the class-conditional density functions $p(x|face)$ and $p(x|non\text{-}face)$. Face or non-face can be classified by using Bayesian classification or maximum likelihood. Another approach is to determine face and non-face class with discriminant function, i.e., decision surface, separating hyper-plane, threshold function. Conventionally, image patterns are first projected to a lower dimensional space. Then, a discriminant function is formed for classification [67, 112], or a nonlinear decision surface can be formed using multilayer neural networks [85]. Recently, support vector machines and other kernel methods have been proposed. These methods implicitly project patterns to a higher dimensional space and then form a decision surface between the projected face and non-face patterns [72].

Eigenfaces

An early attempt of using eigenvectors in face recognition was proposed by Kohonen [46]. A simple neural network is demonstrated to perform face recognition for aligned and normalized face images. The neural network computes a face description by approximating the eigenvectors of the image's autocorrelation matrix. These eigenvectors are later known as the Eigenfaces.

Face recognition and detection using principal component analysis are proposed by Turk and Pentland [112]. Similar to [44], principal component analysis on a set of

training images is performed to generate the Eigenfaces which span a subspace of the image space. Images of faces are projected onto the subspace and clustered. Similarly, non-face training images are projected onto the same subspace and clustered as well. Those faces images do not change radically when they are projected onto the face space, however, the projection of non-face images appear quite differently. For face detection over an image, the distance between an image region and the face space is computed for all locations in the image. The distance from face space is used as a measure of "faceness." The result of calculating the distance from face space is regarded as "face map." The minimum value in the face map is considered as the detected face. The idea of eigenvector decomposition and clustering is widely available in many works on face detection, recognition and feature extractions.

Neural Networks

Neural networks [85] have been widely adopted in many pattern recognition problems, such as optical character recognition, object recognition, and autonomous robot driving. Various neural network architectures have been proposed for face detection. It considers face detection as a two class pattern recognition problem. The complex class conditional density can be train by using neural networks. The major drawback of the neural network architecture is that the parameters are extensively tuned, such as number of layers, number of nodes, learning rates, etc., to get exceptional performance.

Support Vector Machines

Support Vector Machines (SVMs) were first applied to face detection by Osuna *et al.* [72]. Polynomial function, neural networks, or radial basis function (RBF) can be trained by using SVMs. Most classifiers, such as Bayesian, neural networks, and RBF, are trained by minimizing the training error, i.e., *empirical risk.* SVMs operate on another induction principle, called *structural risk minimization*, which aims to minimize an upper bound on the expected generalization error. An SVM classifier is a linear classifier. A hyper-plane is chosen to minimize the expected classification error of the unseen test patterns. This optimal hyper-plane is defined by a small subset of weighted training vectors, called support vectors. Estimating the optimal hyper-plane is equivalent to solving a linearly constrained quadratic programming problem. However, computation is memory intensive and time consuming. To solve the problem, a more efficient method is developed by Osuna *et al.* [72] to train an SVM for large scale problems, and is applied to face detection. Experiment carries out with two test sets of 10,000,000 test patterns of 19x19 pixels. Their system has slightly lower error rates and runs approximately 30 times faster than the system developed by Sung and Poggio [106].

Naïve Bayes Classifier

Apart from the SVM method in [72], which has modeled the global appearance of a face, Schneiderman and Kanade [92] described a naive Bayes classifier to estimate the joint probability of local appearance and location of face patterns at multiple resolutions [91]. Local appearance is emphasized because some local patterns of an object are more unique than others. The intensity patterns around the eyes are much more distinctive than the pattern found around the cheeks. There are two reasons for using a naive Bayes classifier. First, a better estimation of the

conditional density functions of these sub-regions is provided. Second, a naive Bayes classifier provides a functional form of the posterior probability to capture the joint statistics of local appearance and position on the object. The face image is decomposed into four rectangular sub-regions at each scale. These sub-regions are then projected to a lower dimensional space using PCA and quantized into a finite set of patterns. The statistics of each projected sub-region are estimated from the projected samples to encode local appearance. Their method decides the existence of a face when the likelihood ratio is larger than the ratio of prior probabilities. With an error rate of 93.0 percent on data set 1, the proposed Bayesian approach shows comparable performance to [92]. This method is able to detect some rotated and profile faces. Lately, Schneiderman and Kanade [85] further extend this method with wavelet representations to detect profile faces and cars.

Hidden Markov Model

In Hidden Markov Model (HMM) [82], the underlying assumption is that patterns are characterized as a parametric random process. The parameters of this process can be estimated in a precise and well-defined manner. In order to develop the HMM for a pattern recognition problem, at first, a number of hidden states need to be decided to form a model. Secondly, the HMM can be trained to learn the transitional probability between states from the examples. Each example is represented as a sequence of observations. Finally, the HMM would maximize the probability of the training data by adjusting the parameters in the HMM model with the standard Viterbi segmentation method and Baum-Welch algorithms [81]. After the HMM has been trained, the output probability of an observation determines the class to which it belongs.

A face can be divided into several regions such as the forehead, eyes, nose, mouth, and chin. An appropriate order, from top to bottom and from left to right, can be used to observe these face regions. This approach aims to associate facial regions with the states of a continuous density Hidden Markov Model instead of relying on accurate alignment as in template matching or appearance based methods where facial features such as eyes and noses need to be aligned with respect to a reference point. HMM-based methods usually consider a face pattern to be a sequence of observation vectors where each vector is a strip of pixels. During training and testing, an image is scanned in some order, usually from top to bottom. The observation is taken as a block of pixels. For face patterns, the boundaries between strips of pixels are represented by probabilistic transitions between states. The image data within a region is modeled by a multivariate Gaussian distribution. An observation sequence consists of all intensity values from each block. The output states correspond to the classes to which the observations belong to. After the HMM has been trained, the output probability of an observation determines the class to which it belongs to. HMMs have been applied to both face recognition and localization. For face localization, the HMM is trained for a generic model of human faces from a large collection of face images. The image location is declared to be a face if the probability obtained in each rectangular pattern in the image block is larger than a certain threshold.

2.2.5 Face Image Database

A training data set of face images are always required for most face detection methods. Therefore, the databases originally developed for face recognition experiments can be used as training sets for face detection. Since these databases

were constructed to empirically evaluate recognition algorithms in certain domains, the characteristics of these databases and their applicability to face detection are reviewed. Although numerous face detection algorithms have been developed, most of them have not been tested on data sets with a large number of images. Table 2.2 summarizes the characteristics of the mentioned face image databases.

FERET Database

The FERET database was constructed to develop automatic face recognition capabilities that can be employed to assist security, intelligence and law enforcement personnel [75]. It has 14,051 eight-bit grayscale images of human faces. It includes face images of various poses, including profiles of alternative expressions and of different illuminations. For some people, it includes face images with eye glasses worn, with different hair length, and both. Pose variations of the face images were captured systematically. Because of its large amount of facial images, it is one of the best-known face databases.

Yale Database

The Yale face database was made by the Center for Computational Vision and Control, at Yale University [9]. It contains gray face images of 15 people, where there are images of 11 variations for each person. Images for each person are normal images, images with or without glasses, images with light variations (such as center-light, left-light and right-light), and images with expression variations (such as happy, sad, sleepy, surprised and winking).

Purdue AR Face Database

The Purdue AR database contains over 3,276 color images of 126 people (70 males and 56 females) in a frontal view [62]. This database is designed for face recognition experiments under several mixing factors, such as facial expressions, illumination conditions and occlusions. All the faces appear with different facial expression (neutral, smile, anger and scream), illumination (left light source, right light source and sources from both sides), and occlusion (wearing sunglasses or scarf). The images were taken during two sessions separated by two weeks. All the images were taken by the same camera setup under tightly controlled conditions of illumination and pose.

MIT face database

The MIT face database was made by the MIT Media Laboratory [112]. It contains the face images of 16 male people. It includes images of 3 pose variations (upright, right, left), 3 light variations (head-on, 45 deg, 90 deg) and 3 scale (camera zoom) variations (full, medium, small). It includes 6 levels of Gaussian pyramidals (480×512, 240×256, 120×128, 60×64, 30×32, 15×16).

AT&T Database

The face database from AT&T Cambridge Laboratories (formerly known as the Olivetti database) consists of 10 different images for forty distinct subjects [88]. The images were taken at different times, varying the lighting, facial expressions, and facial details (glasses).

HHI Database

The HHI Database [2] from Heinrich-Hertz-Institut is designed for face image databse in MPEG-7. The database contains 12 people with different view, frontal, near frontal, 45-degree and profile. Each person in database contains different lighting condition ranging from dark and shadowed face to strong overhead projected face. There are totally 206 color images at 640×480.

CMU Pose, Illumination, and Expression Database

The CMU Pose, Illumination, Expression (PIE) Database [99] collected 41,368 images of 68 people between October and December 2000. Each person was captured under 13 different poses, 43 different illumination conditions, and with 4 different expressions at 640×488.

Face Image Database	**URL**	**Description**
FERET Database [75]	http://www.nist.gov/humanid/feret	Male and female faces are collected. Each image contains a single face with certain expression.
Yale Database [9]	http://cvc.yale.edu	The database contains image with different expressions, glasses under different illumination conditions.
Purdue AR Database [62]	http://rvl1.ecn.purdue.edu/~aleixaleix_face_DB.html	Total 3,276 face images, including male and female, are collected with different facial expressions and occlusions under different illuminations.
MIT Database [112]	ftp://whitechapel.media.mit.edu/pub/images/	Total 16 people participated in the database. Each person contains 27 images under various illumination conditions, scale and head orientation.
AT&T Database [88]	http://www.uk.research.att.com	Database contains 40 subjects, 10 images per subject.
HHI Database [2]	http://www.darmstadt.gmd.de/mobile/hm/projects/MPEG7/Documents/N2466.html	MPEG-7 face database contains 206 images with different perspective and illumination conditions.
CMU Pose, Illumination, and Expression Database [99]	http://www.ri.cmu.edu/projects/project_418.html	Total 41,368 images of 68 people were captured under 13 poses, 43 different illumination conditions and 4different expressions.

Table 2.2. Face image database.

2.3 Face Tracking in Video Sequence

Automatic detection and tracking of human body parts (e.g. face, arms) is a challenging research topic with applications in many domains such as face recognition, human join audio and video localization, security guard surveillance in office, building and communities, traffic surveillance, and detection of military targets, etc. These applications attempt to detect, recognize and track certain objects from image sequence, and more generally, to understand and describe object behaviors. An intelligent face tracking algorithm can monitor the human face location and perspective change in video sequence in order to estimate the pose for human face modeling. The prerequisites for effective automatic tracking using a single camera include the following stages [36]: modeling of environments, motion detection, classification of moving objects, and tracking.

2.3.1 Motion Detection

Motion detection is widely available in most of the visual tracking system. The aim of motion detection is to segment the moving objects from the rest of an image. Subsequent processes, such as tracking and behavior recognition, are greatly dependent on it. The process of motion detection usually involves environment modeling, motion segmentation and object classification, which intersect each other during processing.

Environment Modeling

The continuous updating and construction of environmental models are indispensable from visual tracking. The models can be classified into two-dimensional (2D) models in the image plane and three-dimensional (3D) models in

real world coordinates. More applications are available in 2D models due to their simplicity.

1) **Fixed cameras.** The major drawback is the continuous update of the background images from a dynamic sequence automatically. The background images updating process can be affected by unfavorable factors, such as illumination variance, shadows and shaking branches.
2) **Pure translation (PT) cameras.** A holistic background image [98] can be acquired by patching up panorama graph to form an environment model. The transformation relationship between different images can be described by homography matrices.
3) **Mobile cameras.** Temporary background images [111] is constructed by using motion compensation.

The current work of 3D environmental models [86] is still limited to indoor scenes because of the difficulty of 3D reconstructions of outdoor scenes. This is because unexpected environment changes happen frequently at outdoors, such as lighting conditions at different hour, weather change, pedestrian in the background, etc.

Motion Segmentation

The aim of motion segmentation is to detect moving regions such as vehicles and humans, in a video sequence. Later processes, such as tracking and behavior analysis, consider focus of attention of the detected moving regions for further processing. At present, temporal or spatial information are used in most segmentation methods in the video sequence. Several conventional approaches for motion segmentation are listed in the following:

1) **Background subtraction.** This method is a popular method for motion segmentation, especially with a relatively static background. Moving regions are detected in an image by subtracting the current image and the reference background image in a pixel-by-pixel fashion. It is simple, but extremely sensitive to changes in dynamic scenes which can be affected by lighting, occluded objects, etc.
2) **Temporal differencing.** This method extracts moving regions by using the pixel difference between two or more consecutive frames in a video sequence. Temporal differencing is sensitive to dynamic environments, and thus, unable to extract all relevant pixels, e.g., many holes would be left inside the moving regions.
3) **Optical flow.** In this method, the characteristics of flow vectors of moving objects are used to detect moving regions in a video sequence. Independent moving objects can be detected with the presence of camera movement. However, most optical flow methods are computational and sensitive to noise, and thus, it is not suitable to apply to real-time video streams without specialized hardware.

Object Classification

In a natural scene, different moving regions may contain different moving targets. For example, a road traffic video sequence captured by surveillance cameras can probably include humans, vehicles and other moving objects such as flying birds and moving clouds, etc. The moving object has to be classified correctly for further tracking and behavior analyzing. Object classification is a standard pattern

recognition issue. Two main categories of object classification are presented in the following:

1) **Shape-based classification.** The description of different shape and its motion regions such as points, boxes, silhouettes and blobs are available for classifying moving objects. Lipton *et al.* [52] use the dispersedness and area of image blobs as classification metrics to classify all moving-object blobs into humans, vehicles and clutter. Precise classification results are made by temporal consistency constraints.
2) **Motion-based classification**. In general, a periodic property is always found in non-rigid articulated human motion. This periodic property has been used as a strong cue for classification of moving objects. Cutler *et al.* [22] detects and analyzes periodic motion by using a similarity-based technique. The moving object is tracked by computing the self-similarity between current and previous frame as it evolves over time. The underlying assumption is that self-similarity measure is also periodic for periodic motion. Therefore, the periodic motion is detected and is characterized by using time-frequency analysis. Tracking and classification of moving objects can be implemented using periodicity. Based on this useful cue, human motion can be distinguished from other moving other objects, such as vehicles, fly birds, etc.

2.3.2 Model-Based Tracking

Model-based tracking algorithms track moving objects by matching projected object models, which were produced with prior knowledge, to image data. The models are usually constructed off-line with manual measurement, CAD tools or

computer vision techniques. As human face is non-rigid object, thus, model-based human face tracking is reviewed here.

Analysis-by-synthesis is the general approach for model-based human face tracking. This approach is also known as a predict-match-update style. First of all, based on prior knowledge and tracking history, face model for the next frame is predicted. Then, the synthesized predicted model is projected into the image plane to compare with the image data. The similarity between the projected face model and the image data can be measured by a correlation function, distance measurement, or other specific evaluation function. Different searching strategies can be adopted, either recursive searching or using sampling techniques. Once the correct face is found, the tracked face is then updated to the model. Estimation of the face in the first frame requires special handling, such as face detection or manual registration. Generally, model-based human face tracking involves three main issues:

- construction of human face models [4];
- representation of prior knowledge of motion models and constraints [40];
- prediction and searching strategies [18].

There are some disadvantages of model-based tracking algorithms, such as the necessity of model construction, high computational cost, etc.

2.3.3 Active Contour-Based Tracking

Active contour-based tracking algorithms track objects by using the contour boundary. The contours are updated dynamically in successive frames [8, 68]. Paragios *et al.* [74] proposed to use geodesic active contour objective function and level set formulation scheme to detect and to track multiple moving objects in a video sequences. Malik *et al.* [47, 59] applied active contour-based methods to track

vehicle successfully. However, there are some problems in active contour-based tracking. First, the tracking precision is limited at the contour level. Second, the 3D pose of an object, which is recovered from its contour in image data, is a demanding problem. Finally, tracking algorithm towards automation is a difficult task since the active contour-based algorithms are highly sensitive to the initialization step at early stage.

2.3.4 Feature-Based Tracking

Feature-based tracking algorithms extract elements from image data. These elements are then clustered into higher-level features. Features recognition and tracking are performed from image data. According to the selected features, feature-based tracking algorithms can further classify into three categories: global feature-based algorithms, local feature-based algorithms, and dependence-graph-based algorithms.

- **Global feature-based algorithms.** Centroids, perimeters, areas, some orders of quadratures and colors, etc. are used in these algorithms. A good example of global feature-based tracking is proposed by Polana *et al.* [78]. The tracking algorithm use a rectangular box to bound a person is bounded. The centroid of this rectangular is selected as the feature for tracking. By using velocity of the centroids, the tracking algorithm can still distinguish effectively even there is occlusion between two people.
- **Local feature-based algorithms.** Line segments, curve segments, corner vertices, etc., are used in these algorithm.
- **Dependence-graph-based algorithms.** A variety of distances and geometric relationships between features are used in these algorithms.

2.4 3D Face Reconstruction.

The heart of all computer vision research is to analyze and interpret visual information. One of the most interesting and difficult problems is to generate a 3D face models with realistic looking. Three-Dimensional (3D) reconstruction is the task to recover 3D geometry from two-dimensional (2D) views. In the human visual system, three-Dimensional (3D) world is perceived as retinal images in our eyes through a projection process. Human visual system has the ability to perceive in-depth information by using physiological and psychological cues [58, 60], including, binocular parallax, monocular movement parallax, accommodation, convergence, linear perspective, shades and shadows, etc. The most important depth cue in our visual system is binocular parallax. The depth of an object is provided by two slightly different images at slightly different location perceived by the left and right eyes. These two images are then combined in our visual system to reconstruct a 3D model. Similar to binocular parallax, monocular movement parallax fuses two slightly different images at slightly different location in our visual system. These two cues are physiologically different. However, in computational perspective, binocular can be considered as a special case of monocular movement parallax. The ability to compute 3D properties of the world from two or more two-dimensional (2D) images is an important step toward emulating the human visual system.

Comparing with human visual system, camera is an eye of computer. For any 3D reconstruction task, an accurate camera calibration is required. The calibration algorithm can be classified into two categories, pre-calibrated and online calibrated. Pre-calibration algorithm assumes a prior calibration of the camera parameters; online calibration algorithm calibrates the camera parameters during run time. In the case of

pre-calibrated reconstruction algorithm, a 3D model can be reconstructed accurately by using a full calibrated camera. On the other hand, an online calibrated reconstruction algorithm can be used to reconstruct a scene from some previously recorded video sequence. In the following section, the current projective geometry and the methodology in both pre-calibrated and online calibrated reconstruction algorithm will be reviewed.

2.4.1 Projective Geometry

To describe a 3D entity, Euclidean geometry is a common place to be used. The Euclidean geometry is used to quantify the notion of parallelism, angle between lines, ratio of lengths, etc. However, Euclidean geometry is inappropriate in imaging process. After an object has been captured by a camera, the Euclidean geometry in 3D world is no longer able to maintain in image world. The entities mentioned before cannot be preserved. This is because the perspective project of the camera is different from our 3D world. Therefore, a more general geometry form is required to describe the image formation process.

The group of transformations associated with projective geometry, including Euclidean geometry as a special case and perspective projection, which can be completely depicted with rotation and translation. Perspective projection is particular important to projective geometry. The projective framework provides a much simpler formulas. It also reduces the need to handle many special cases. Most importantly, it creates a natural concept of duality. Here is an example of projective duality. A 2D point with coordinate $(a/b, b/c)$ and a line $ax + by + c = 0$ can be represented as a 3D vector (a, b, c). Therefore, for any theorems proven for points, there is a

corresponding dual proof for lines. The action of perspective camera becomes a linear operation under projective geometry.

Pinhole Camera Model

A camera is a mapping between the 3D world and a 2D image plane. As shown in Fig 2.1, let the *centre of projection* be the origin of Euclidean coordinate system, and consider the plane Z=*f*, where *f* is the *focal length* of the camera. This plane is called *image plane* or *focal plane*. The pinhole camera model consider the mapping between a point in space with coordinates $\mathbf{X}=(X, Y, Z)^T$ and the point on the image plane. The line joining of the point **X** to the centre of projection, **C**, meets the image plane. By using similar triangles, the mapping between the point $(X, Y, Z)^T$ and the 2D point $(f X/Z, f Y/Z, f)^T$ on the image plane can be computed quickly. The line from the camera center perpendicular to the image plane is called the *principal ray*. The principal ray intersects the image plane at the *principal point*, **p**.

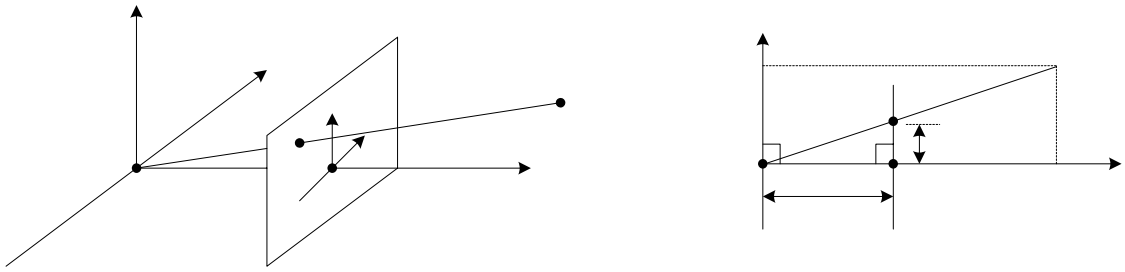


Fig. 2.1. Pinhole camera geometry with camera centre, **C**, and principal point, **p**. The camera centre is placed at the coordinate origin. The image plane is placed in front of the camera centre.

If homogenous vectors are used to represent the world and image points, then central projection can be expressed in matrix form between linear mapping and their homogeneous coordinates as:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ 1 \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \qquad (2.1)$$

The above equation can be written in algebraic form with 3×4 *camera projection matrix*, *P*, as

$$\mathbf{x} = P\mathbf{X} \qquad (2.2)$$

where **X** is the world point represented by the homogeneous 4-vector $(X, Y, Z, 1)^T$, and **x** is the image point represented by a homogeneous 3-vector.

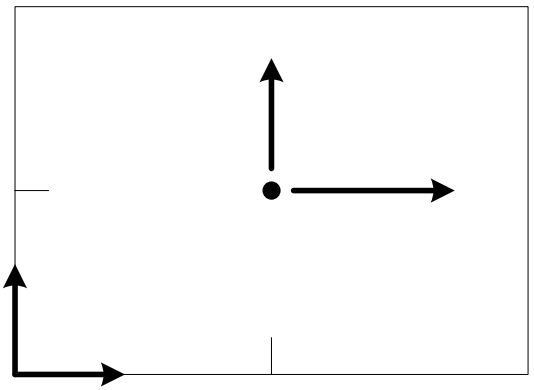


Fig. 2.2. Image (x, y) and camera (x_{cam}, y_{cam}) coordinate system.

Generally, the principal point will be located at (p_x, p_y), as shown in Fig. 2.2. The mapping between a 3D location $(X, Y, Z)^T$ and the 2D image plane is

$$(X, Y, Z)^T = (f\frac{X}{Z} + p_x, f\frac{Y}{Z} + p_y)^T \qquad (2.3)$$

This equation can then be further expressed in homogeneous coordinates as

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \qquad (2.4)$$

Let *K* be the matrix

$$K = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}. \qquad (2.5)$$

The projection equation can be rewritten as

$$\mathbf{x} = K[I \mid 0]\mathbf{X}_{\text{cam}}. \qquad (2.6)$$

The matrix K is called the *camera calibration matrix*. In (2.6), $(X, Y, Z, 1)^T$ is written as $\mathbf{X}_{\text{cam}}$ to emphasize the camera location at the origin of a Euclidean coordinate system. The principal axis of the camera points straight down the Z-axis and the point $\mathbf{X}_{\text{cam}}$ is expressed in the Euclidean coordinate system. This coordinate system is called *camera coordinate frame*.

Generally, points in space can be expressed in *world coordinate system*, which is a different form of Euclidean coordinate system. The two coordinate systems are related through a rotation and a translation as shown in Fig. 2.3.

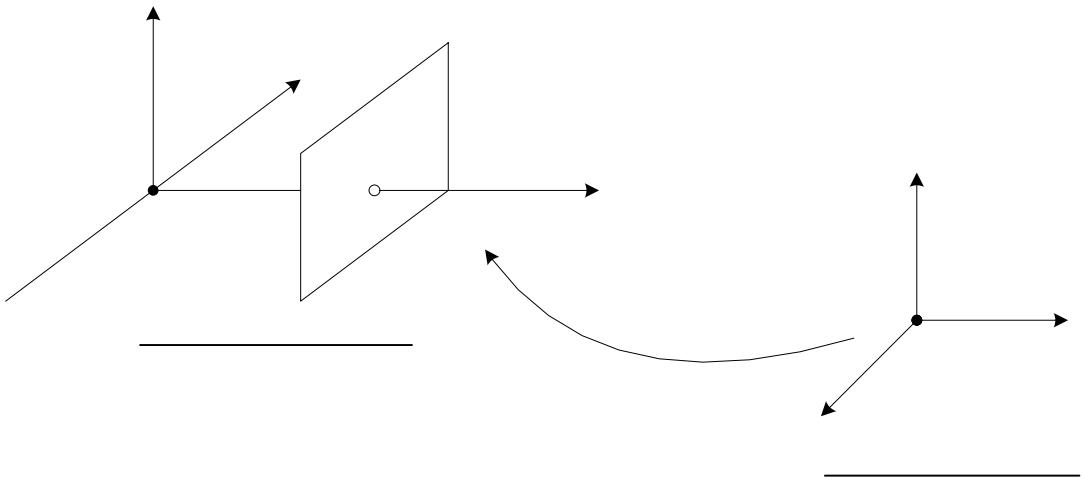


Fig. 2.3. The Euclidean transformation between the world and camera coordinate frames.

If there is a point of inhomogeneous 3-vector $\tilde{\mathbf{X}}$ in the world coordinate system and the same point $\tilde{\mathbf{X}}_{\text{cam}}$ is in the camera coordinate system, then the mapping can be written as $\tilde{\mathbf{X}}_{\text{cam}} = R(\tilde{\mathbf{X}} - \tilde{\mathbf{C}})$, where $\tilde{\mathbf{C}}$ is the camera center location in the world

coordinate system, and R is a 3×3 rotation matrix representing the orientation of the camera coordinate system. The homogeneous coordinates can be written as:

$$\mathbf{X}_{\text{cam}} = \begin{bmatrix} R & -R\tilde{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} R & -R\tilde{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \mathbf{X}. \qquad (2.7)$$

By writing everything algebraically leads to the equation

$$\mathbf{x} = KR[I \mid -\tilde{\mathbf{C}}]\mathbf{X}, \qquad (2.8)$$

where $\mathbf{X}$ is in a world coordinate system now. The parameters in K are called the *intrinsic* camera parameters which contains the internal orientation information. The parameters of R and $\tilde{\mathbf{C}}$ which relate the camera orientation and position to a world coordinate system are called the *extrinsic* camera parameters which contains the external orientation information.

For additional generality, the general form of the camera calibration matrix can be written as

$$K = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}, \qquad (2.9)$$

where the added parameter s is referred to as the *skew* parameter. For normal camera, the skew parameter is normally 0. However, it can be non-zero values under certain unusual circumstance. For non-zero skew value, it implies that the x and y axes of the image plane are not perpendicular. In practice, this situation is unlikely to be happened. The non-zero skew might arise when taking an image, the picture is re-photographed, or the negative is enlarged. For CCD cameras, there is a possibility of having non-square pixels if image coordinates are measured in pixels. It introduces an extra effect of unequal scale factors in each direction. Therefore, α_x and α_y in the

camera calibration matrix are $\alpha_x = fm_x$ and $\alpha_y = fm_y$, where m_x and m_y are the pixel dimensions in the *x* and *y* directions, respectively. In terms of pixels dimensions, the principle point x_0 and y_0 are $x_0 = m_x p_x$ and $y_0 = m_y p_y$, respectively.

Epipolar Geometry

Epipolar Geometry can completely describe the projective geometry between two views of a scene, as shown in Fig. 2.4. In other words, the intrinsic geometry of two views can be described by using epipolar geometry. Stereo matching is an important application as the epipolar geometry limits the correspondence searching into a one-dimensional search space.

Suppose a point **X** in 3D that is imaged in two views with point **x** in the first view, and point **x'** in the second view. A typical stereo matching problem is to find the correspondence between **x** in one image and **x'** in another image. Both camera centers **C** and **C'**, the points **x**, **x'**, and **X** are coplanar. This plane is called *epipolar plane* π. The line that connects the two camera centers is called the *baseline*. The points **e** and **e'** where the baseline intersects the two views are called *epipoles*. The lines connecting **x**, **e** and **x'**, **e'** are the *epipolar lines*. From the definition of perspective projection, points **C**, **x**, and **X** are collinear and any point on this line between **x** and **X** projects as **x** in the first image. Therefore, the correspondence of **x** must lie on the projection of the line from **x** to **X** in the second image.

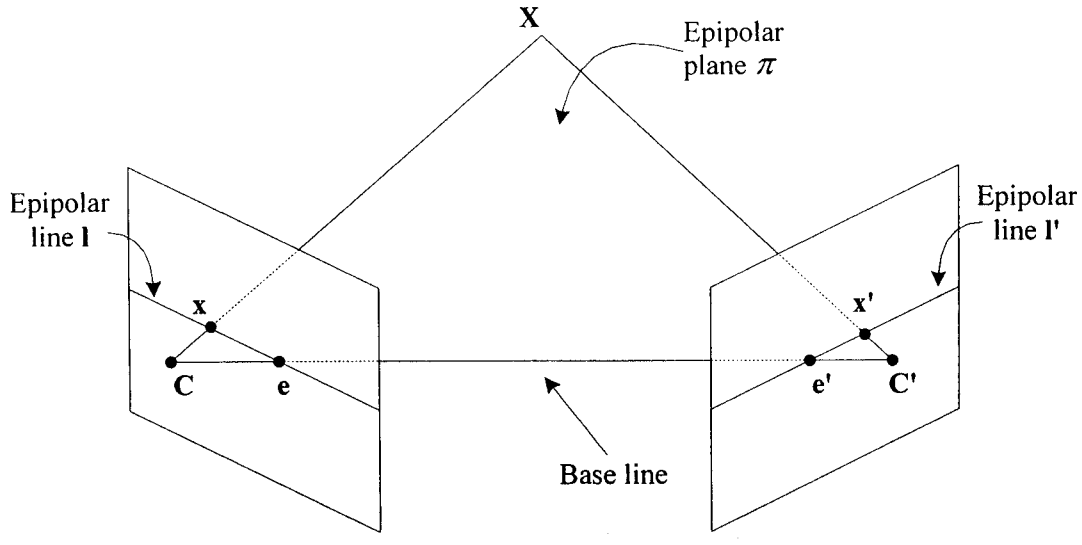


Fig. 2.4. The epipolar geometry.

Fundamental Matrix

The epipolar geometry is the intrinsic projective geometry between two views. It is independent of scene structure. The only dependent is the internal camera parameters and relative pose. Intrinsic geometry is encapsulated in the fundamental matrix, **F**. The fundamental matrix is a 3×3 matrix of rank 2. A mapping is formed between a point **X** is imaged as **x** in the first image, and a correspondence **x'** in the second image, then the image points satisfy the *epipolar constraint*, $\mathbf{x'}^T\mathbf{Fx} = 0$.

The fundamental matrix is independent of scene structure. However, the correspondences of imaged scene points can be computed alone without knowing about the cameras' internal parameters or relative pose. The properties of the fundamental matrix in [34] are briefly summarized in the following:

- If **F** is the fundamental matrix of the pair of cameras (**C**, **C'**) then $\mathbf{F}^T$ is the fundamental matrix for (**C'**, **C**).
- For the point **x** in the first image, its epipolar line is given by $\mathbf{l'}=\mathbf{Fx}$. Similarly, for **x'** in the second image, its epipolar line is $\mathbf{l}= \mathbf{F}^T \mathbf{x'}$.

- The epipolar line contains the epipole **e'**, where an epipole is the projection of the other camera centre onto the current image plane.
- **F** satisfies the constraint *det*(**F**)=0, which means that **F** is not of full rank.
- **F** is a projective map that takes a point to a line. If **l** and **l'** are corresponding epipolar lines, then any point on l maps to the same line **l'**. Thus, there is no inverse mapping.

2.4.2 Pre-calibrated Reconstruction

An accurate prior calibration of the cameras is required for pre-calibrated reconstruction algorithms. Thus, both of the camera's intrinsic and extrinsic parameters are computed in advance. Recently, a practical method, which requires the camera to capture a planar pattern in a at least two different orientations, is proposed by Zhang [128]. Since the metric information about the scene is provided by the camera calibration, the reconstructed result is called a metric reconstruction. Because of the unknown translation, rotation and scale of the world coordinate system, a metric reconstruction only estimates the shape up to a similarity transform. These inherent indeterminacies are called *gauge freedoms*. Pre-calibration reconstruction is further classified into 3 sub-categories, including image-based algorithms, voxel-based algorithms and object-based algorithms, and described in the following section.

Image-Based Reconstruction

Generally, image-based reconstruction perform 3D reconstruction task by using sparse image features [1, 70] and dense pixel matchings [131]. Feature-based algorithms make use of sparse image features to accurately reconstruct 3D task based on feature detection and feature correspondence. Dense reconstruction algorithms

rely heavily on the accuracy of the densely matched pixels. The pixel displacement between consecutive frames can be approximated by optical flow when the sampling along the time axis is also dense.

Both reconstruction algorithms follow a very similar path but different in the amount of data processed. Triangulation is usually used to determine 3D point from pairs of matched image pixels in both methods. If noise is absence, triangulation is trivial. However, with the presence of noise, the triangulation problem becomes more complicated. The back projected rays from the two images are not able to meet in 3D space. A suitable point of "intersection" is needed. The reconstruction becomes metric as the camera calibration is already known. Therefore, the concept of distance and perpendicularity is defined clearly. The required 3D location can be estimated by using the midpoint of the common perpendicular between the two back projected rays. Furthermore, with an assumed Gaussian noise model, a provably optimal triangulation method is available. From a pair of point correspondences $\mathbf{x}$ and $\mathbf{x}'$, this algorithm seeks an alternate pair $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$. Therefore, the sum of squared distances of the original points pair is minimized subject to the epipolar constraint. Thus, the optimal points, which are the closest to the original point correspondence, are then lie on a pair of corresponding epipolar lines. The minimized distance can be found between the pair of epipolar lines, $\mathbf{l}$ and $\mathbf{l}'$, and the original point correspondence. Furthermore, the pencil of epipolar lines is parameterized in the first image with a suitable parameter t. Therefore, the minimization problem [33] can be reduced to find the real roots of a 6 degree polynomial.

Voxel-Based Reconstruction

Volumetric representation [61] of scene structure becomes practical due to of the rapid growth of computational storage and power. Various approaches have been proposed to recover volumetric scene structure from a sequence of images. Visual hull is an early attempt to solve this approximation problem. A visual hull is defined by a set of camera locations, the cameras' intrinsic parameters, and silhouettes from each view. Generally, it is the maximal volume that creates all possible silhouettes of an object. The visual hull is known to include the object, and to be included in the convex hull of the object. Methods which approximate the visual hull are referred to as *volume intersection* or *silhouette intersection*.

Similar to the convex hull, the visual hull is an approximation method to estimate the object actual shape. However, the size of the visual hull decreases monotonically with the number of 2D images [100]. The basic approach of visual hull is to segment the foreground and background object in each 2D image. The segmented 2D silhouettes are then back projected and intersected to yield a volume segment representation for further surface description processing.

Comparing with the image-based methods, which require solving the difficult correspondence problem, the voxel-based methods have the advantage allowing geometric reasoning to be performed in 3D. Thus, explicit correspondence is not required. In addition, the occlusion problem can be handled probably. These are the advantages of the voxel-based algorithms. Memory consumption is one of the main design considerations of voxel-based algorithm. For a moderate scene having 100 voxels in each dimension, a total of 10^6 voxels are required to represent the entire scene. Therefore, coarse reconstruction would be applied due to limited memory. Moreover, the ordinal visibility constraint imposes a very tight constraint on the camera locations. In particular, cameras are usually not allowed to surround the scene.

Object-Based Reconstruction

Object-based reconstruction algorithms aim at recovering a surface description of the objects in the scene. On the other hand, voxel-based reconstruction algorithms fill the scene with voxels and determine visibility of each voxel. Faugeras *et al.* [27] proposed the level-set reconstruction method by using object centered 3D reconstruction technique from image sequences. Various principles are applied to their previous work for dense depth recovery [83, 84], the reconstruction problem is reformulated to solve the surface evolution problem by using the level-set technique. Perfect Lambertian surfaces are assumed in their the level-set approach. In particular, stereo matching problems caused by specularities present in the scene are not addressed. Jin *et al.* [41] proposed an improved level-set approach that handles specular surfaces. In [3], an improved approach is proposed to handle specular, as well as translucent surfaces.

2.4.3 Online Calibrated Reconstruction

Under various circumstances, calibration in advance may not be available. It is a very realistic scenario for a number of applications. For example, in a video indexing application, the final video data is given without even knowing the type of video camera, and prior calibration objection in the video sequence. Furthermore, different focusing and zooming in the video sequence can cause the change of the camera intrinsic parameters. Therefore, online calibrated reconstruction methods can be performed for 3D reconstruction under these scenarios.

The key difference in various online calibrated reconstruction methods is the approach that camera intrinsic and extrinsic parameters are to be estimated. This is an

on-the-fly process and is often referred to as camera self calibration or auto-calibration. Auto-calibration methods can be sub-divided into two classes: scene constraints and geometric constraints. Because of the scene constraints, the calibration parameters can be determined without the use of fundamental matrices and initial project reconstruction.

Projective Reconstruction

Projective Reconstruction is an online calibration reconstruction technique to reconstruct 3D scene by using the estimated fundamental matrix. The projective reconstruction problem for all feature correspondences and all views can be solved efficiently when the image sequence contains more than two frames. A projective reconstruction consists of a set of 3D points $\{\mathbf{X}_i\}$ and a set of camera projection matrices $\{\mathbf{P}_i\}$. However, projective reconstruction can only reconstruct up to projective transform. Thus, for any projective transformation H, $\{\mathbf{P}_i H^{-1}\}$ and $\{H\mathbf{X}_i\}$ yield an equally valid reconstruction. The disadvantage of this algorithm is quite sensitive to correspondence errors. Therefore, outlier matchings should be rejected by using the epipolar constraint before invoking this algorithm.

Calibration Using Scene Constraints

Auto-calibration can be simplified if images are taken within constrained environments, such as architectural constrained environment or man-made scene. A large number of parallel lines can be found within this environment. These parallel lines intersect at a point on the plane at infinity. These intersection points, called *vanishing points*, will be used for projection. By knowing the vanishing points location in three dominant directions in an image, the determination of the camera

intrinsic parameters is greatly simplified. In fact, closed form solutions can be obtained for these parameters as a function of the vanishing points [11].

Before obtaining camera calibration parameter, vanishing points have to be computed first in order to take the benefit of parallel structure.

1. **Computing the Vanishing Points.** The estimation of vanishing points from detected line segments can be computed in two steps, *accumulation step* and the *search step.* Accumulation step detects line segments to vote for locations in the accumulation space. The same vanishing point could potentially share the same accumulation space. Search step searches the vanishing points by possessing a large number of votes in the accumulation space.
2. **Computing the Camera Calibration.** The camera intrinsic parameters can now be estimated after the vanishing points have been detected in an image. When a point **X** is perspectively projected to the point **x** on the image plane, the usual projection equation $\mathbf{x} = \mathbf{PX}$, where $\mathbf{P} = K\left[R|-t\right]$ is the projection matrix, is obtained. The matrix *R* and the vector *t* represent the rotation and translation of the camera relative to the world coordinate system respectively, while the calibration matrix *K* contains the intrinsic parameters.

Calibration Using Geometric Constraints

This is a more flexible class of algorithms to perform auto-calibration using only geometric constraints from a sequence of image. Two different methods are described in the following:

1. **The Dual Image of the Absolute Quadric Method (DIAC).** A projective entity, which conveys the calibration information from frame to frame, is required to perform auto-calibration. It is equivalent to apply similarity

transforms to the entire scene and taking snapshots of the transformed scene with a video camera if all objects in the scene are rigid. This calibration approach estimates both calibration parameters and transformed position of the plane at infinity at the same time. The advantage of this approach is that projective reconstruction can be directly upgraded to metric reconstruction.

2. **The Kruppa Equations.** The Kruppa equations [28, 63] use the dual image of the absolute conic and assume the equality of the DIACs in both views. Comparing with the methods that estimate the absolute dual quadric, the advantage of using Kruppa equations is the calibration process does not require a prior projective reconstruction. In practice, when multiple views are applied, the performance of this approach is inferior compared to the absolute dual quadric formulation. This is because the Kruppa equations do not explicitly enforce the degeneracy of the dual quadric. Also, there is a common supporting plane for the absolute conic over multiple views. In addition, the Kruppa equations reduce the complexity when the camera motion is purely translational. The ambiguities of calibration by Kruppa equations are discussed in [104].

2.5 Conclusion

In this chapter, we have reviewed four different approaches for face detection and seven face image databases, three different approaches for face tracking in video sequence, and two different three-dimensional (3D) reconstruction algorithms. Different face detection methods have been surveyed. Face detection methods can be categorized into four categories: knowledge-based top-down method, feature invariant approaches, template matching, and appearance-based methods. Although significant progress has been made in the last two decades, a robust detection system should be

effective under full variation in: lighting conditions, orientation, pose, and partial occlusion, facial expression, presence of glasses, facial hair, and a variety of hair styles.

For face image database, the HHI database [2], the AR Face database [62], and the CMU Pose, Illumination, and Expression database [99] are recommended. These databases contain color image with varying lighting condition. Others are gray scale images only. The rapid development of technology lower the price of color input device, such as digital video, digital camera, web camera, etc. Skin color information should be considered during face detection.

Face tracking in video sequence is an active and important research. The existing state-of-the-art methods have been described with the focus on detection and tracking. For detection of moving objects, environmental modeling, motion segmentation and object classification are involved. For tracking of moving objects, active-contour based, feature based and model based have been studied.

For 3D face reconstruction algorithms, two large categories, pre-calibrated reconstruction and online calibrated reconstruction, have been reviewed. There are still a lot of open issues for 3D reconstruction to be improved. On the pre-calibrated side, feature correspondence and dense stereo matching are still open research topics for image-based methods. A lot of effort is put into solving the global optimization problems. For voxel and level-set-based approaches, research interests are focused on efficient data representation and ability to handle non-Lambertian surfaces. On the online calibrated side, a lot of research effort is devoted to estimate camera parameters.

Chapter 3
An Efficient Color Face Detection Algorithm Under Different Lighting Conditions

3.1 Introduction

Face detection is the first step in any face processing system. Yang *et al.* [125] gave a survey of the current face detection technology and provided a definition of face detection as follows: Given an arbitrary image, the goal of face detection is to determine whether or not there are any faces in the image and, if present, to return the image location and extent of each face. This is a challenging task because the human face is highly variable. The detection performance may be affected by the presence of glasses, different races, genders, facial hair, facial expressions, lighting conditions, etc. Furthermore, human face is a three-dimensional object, and has different perspective and uneven illumination. As a result, a true face may not be well detected.

Numerous approaches have been proposed for the detection of human faces in gray-level images. These include the probabilistic approach [53, 67, 73], component-based approach [120], neural networks [85], example-based approach [68], and more often, a combination of all of these.

These approaches usually search over a range of scales and locations for possible human faces and verify the patterns with a pattern classifier. Moghaddam and Pentland [67] applied the eigenface decomposition technique to reduce face image from a high-dimensional image space to a lower dimensional; the images are then trained with the expectation-maximization algorithm to optimize the mixture

parameters. Multiscale saliency maps based on maximum likelihood are then computed for single face detection. Liu [53] purposed using Bayesian Discriminating Features to detect multiple faces in an image. Face and non-face classes are trained to discriminate input images during detection. 4,500 non-face class patterns, which lie close to the face class, are generated from nine natural images that do not contain any human faces. Xie *et al.* [120] presented a statistical fusion framework for component-based face detection. The training of the component detectors, including the left eye, right eye and lower face, employs AdaBoosting. Papageorgiou and Poggio proposed an example-based learning approach by using a large set of positive and negative examples to implicitly derive an object model class to be classified with the support vector machine. Mohan *et al.* [68] presented an example-based framework for detecting objects in static images by components in two steps. The example-based component detectors are first trained to find components separately, including the human body, head, legs, left arm and right arm. Then, a second example-based classifier combines the results of the component detectors to classify a pattern as either a “person” or a “non-person”. Rowley *et al.* [85] developed a neural network-based face detection system, which examines an image via small windows and decides whether the windows contain a face or not.

Recently, face segmentation based on the color-based approach [12, 32, 35, 36, 114] has received mass attention since color provides more information than gray-scale intensity, especially when the skin is under different illumination conditions. The basic idea of these approaches is that skin colors for people of different races and illumination are distributed more or less in the same region in a color space. Consequently, the search for faces in an image can be restricted within the skin-color regions. For example, Chai and Ngai [12] proposed a skin-color segmentation

method under a range of luminance and chrominance. Hsu *et al.* [36] proposed a face detection algorithm for color images under varying lighting conditions based on a lighting compensation technique and a nonlinear color transformation to detect skin regions and to generate face candidates based on a spatial arrangement of the skin patches. Eyes, mouth, and boundary maps based on luminance and chrominance are constructed to verify a face candidate. Wong *et al.* [119] proposed to perform skin-color segmentation according to different illumination conditions, using the genetic algorithm [117] to search for possible face candidates. Greenspan *et al.* [32] presented a mixture-of-Gaussians distribution to model the color distribution of shadowed face images.

In this chapter, we found that the red component of skin color becomes saturated under strong illumination. Therefore, to further improve the performance of skin-color detection, a color compensation scheme [118] is proposed to extend the range of red component to its saturated level. Then, the skin color is modeled with the mixture-of-Gaussians model to segment skin-color regions according to different illumination conditions. The performance of our algorithm is tested with the HHI MPEG-7 face database [2], the AR face database [62] and the CMU Pose, Illumination and Expression (PIE) database [99], which contain face images under a wide range of lighting conditions, including poor conditions, under shadow, different scales and with glasses. Experimental results show that our proposed algorithm is very fast and can achieve a high detection rate. The details of our approach for face detection will be described in the following sections.

3.2 Human Face Detection

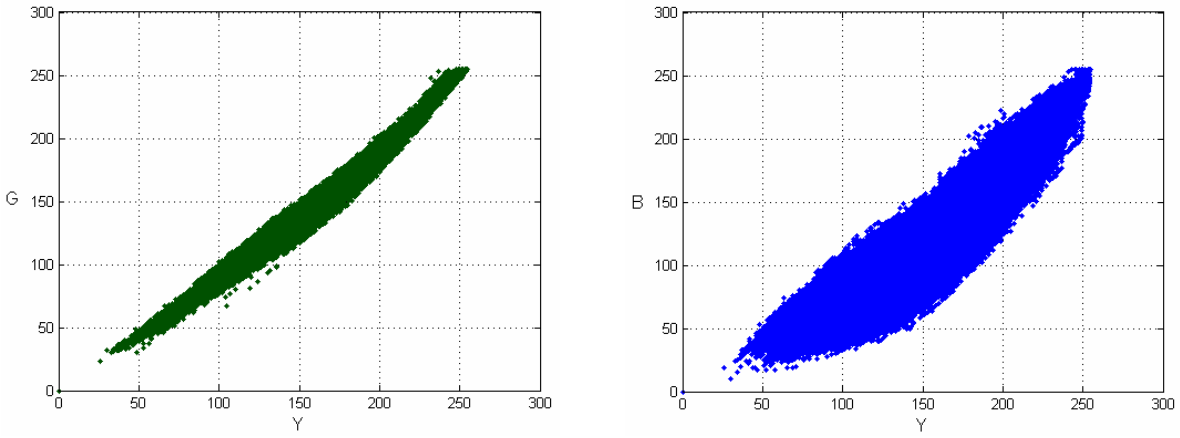
Our approach to detecting face regions in a color image consists of three steps. The first step is to segment the face color by using the mean-shift algorithm [17]. The segmented regions are then processed by a color compensation scheme, and the skin-color distributions under different illuminations are modeled by means of the maximum-likelihood method. In the second step, our algorithm focuses on searching possible eye candidates within the segmented skin-color regions. Possible face candidates are formed by grouping pairs of eye candidates. Finally, a two-step procedure based on an eigenmask for face verification is performed. Once the face has been verified and short listed, the face contour is further verified with a Gaussian function in order to further improve the reliability and accuracy of our face detection algorithm.

3.3 Face-color Segmentation

Color information has been a commonly used technique for segmenting human face regions from a complex background. In [114], skin-like regions are extracted by using both the normalized RGB color model and the HSV color model. In [12], the chrominance information in the YCbCr color space is used for the segmentation of skin-like regions. However, these methods can achieve good results if the face images are captured under good lighting conditions, but some skin pixels will not be located under poor lighting. In order to overcome these problems, we therefore propose a robust color compensation method with the use of the mixture-of-Gaussian model [32] to represent skin color under various illuminations.

3.3.1 Skin Color Compensation

Characteristics of Skin Color under Different Lighting Conditions



(a) (b)

Fig. 3.1. Skin-color distribution: (a) green color component against Y, and (b) blue color component against Y.

We investigate the distributions of skin color based on 220,000 face skin pixels. It is found that the skin-color distributions of the green and blue color components are linear to the luminance intensity Y, as shown in Fig. 3.1. However, the intensity of the red color component is saturated when the luminance component Y is greater than a value of about 175. Fig. 3.2 shows the distribution of the red color component under different luminance intensities. Since the skin tones reflect more red light than green or blue, so the intensity level of red color detected by the capturing device will be saturated. The skin-color distributions in the CbCr plane under normal lighting conditions (i.e. $60<Y\leq175$) and strong lighting conditions (i.e. $Y>175$) are shown in Fig. 3.3. It is obvious that the two color spaces are different. Therefore, the performance of skin-color segmentation will be degraded if the effect on the red color component under strong lighting conditions is not taken into account in the detection process.

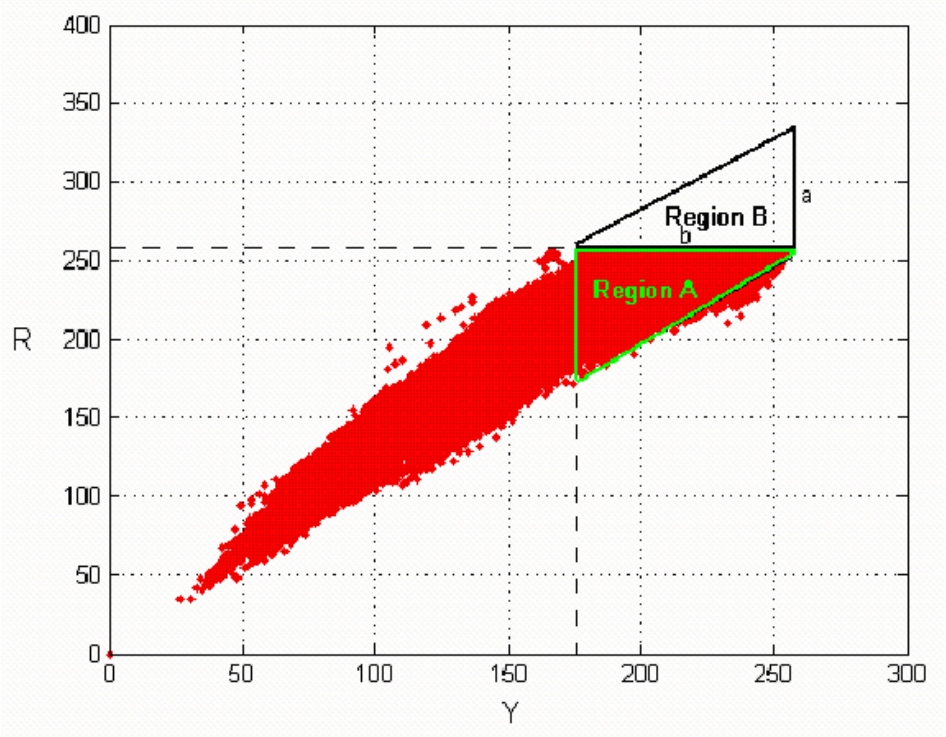
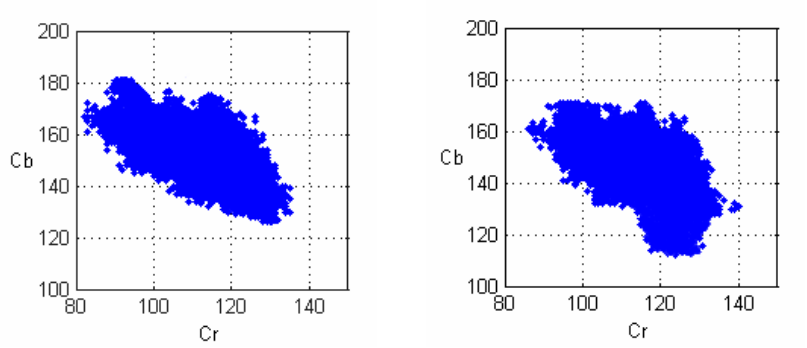


Fig. 3.2. The distribution of the red color component of skin pixels under different luminance intensities.



(a) (b)

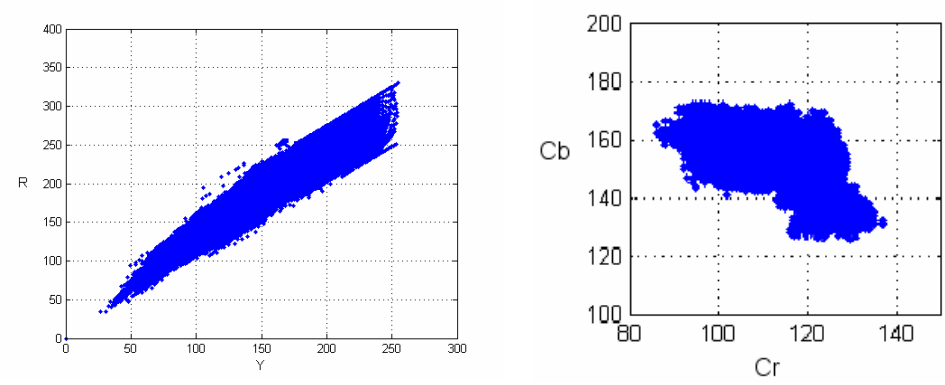
Fig. 3.3. Skin-color distribution under (a) normal lighting conditions (60<Y≤175), and (b) strong lighting conditions (Y>175).

Color Compensation for Skin-Color Segmentation

The difference in the color spaces for skin color in the CbCr plane under normal and strong lighting conditions is due to the saturation of the red component as detected by the capturing device. In order to compensate for this effect on skin-color segmentation, "Region A" in Fig. 3.2 is mapped into the region covering both "Region A" and "Region B." The equation of this mapping process is defined as follows:

$$\tilde{R} = \frac{a}{h}(h - (255 - R)) + 255 - h, \qquad (3.1)$$

where $h = \frac{a}{b}(255 - Y)$, R is the intensity level of the red component, a is the height of "Region B", b is the distance between the saturated point (Y=175) and the maximum intensity of the Y component, and h is the estimated height of "Region A" at a particular value of Y. Based on the red color distribution as obtained in our experiment, the value of a and b are 75 and 80, respectively. The mapped results in the R-Y plane and the CbCr plane are illustrated in Fig. 3.4(a) and 3.4(b), respectively. Fig. 3.4(a) shows that the distribution of the red component under strong lighting conditions is similar to that found under normal lighting conditions after applying the compensation process. This means that the skin color can be segmented by using the same thresholds or the same skin color map under different lighting conditions.

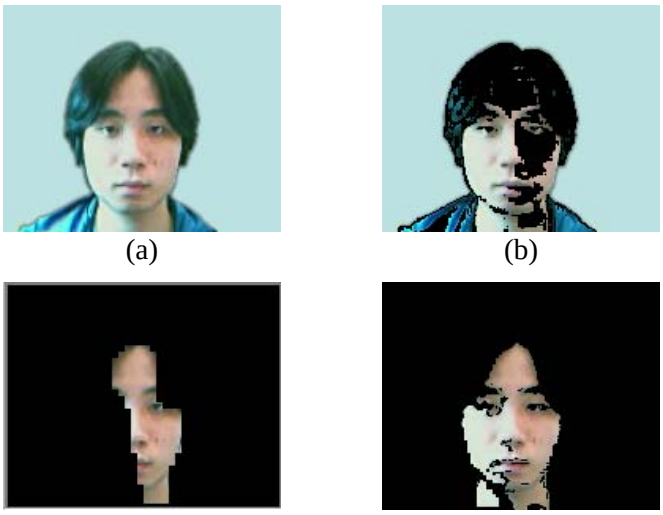


(a) (b)

Fig. 3.4. Results after the compensation process: (a) compensated results in the red color component, and (b) the compensated results in the CbCr plane under strong lighting conditions.

The performance of our method of skin-color segmentation is compared with the methods proposed in [114] and [12]. The black color in Fig. 3.5 represents the non-skin regions, while other colors are the segmented possible face regions. The process of skin-like region detection using the HSV color space works well over the strong light regions but fails to detect those skin regions under shadow. Furthermore, the use of YCbCr color space does not work properly for those face regions under strong

lighting conditions. In contrast, our approach can detect most of the face regions by using a simple thresholding technique irrespective of the lighting conditions.



(a) (b)

(c) (d)

Fig. 3.5. Skin-color segmentation results: (a) original image, (b) result based on normalized RGB and HSV color spaces [114], (c) result based on YCbCr color space [12], and (d) result based on our proposed method.

3.3.2 Color Image Segmentation with Mean-Shift Algorithm

Mean-shift algorithm [17] is a kernel-based density estimation technique which has been used in many applications including data clustering, image segmentation [16], object tracking [18], etc. In particular, it is a nonparametric and robust technique to analyze feature spaces.

In color image segmentation, the feature space, which is the spatial domain of color images, can be considered as an empirical probability density function (p.d.f.). Dense regions in the feature space are considered as local maxima of the p.d.f., that is, the *modes* of the unknown density. Once the location of a mode is determined, the cluster associated with the mode is delineated based on the local structure of the feature space.

Given n training color vectors x_i, i=1, …, n, in the d-dimensional space $\Re^d$. The feature space can then be modeled by an unknown kernel density function K

$$K_{h_s h_r}(x) = \frac{C}{h_s^2 h_r^3} \sum_{i=1}^{n} k\left(\left\|\frac{x^s}{h_s}\right\|\right) k\left(\left\|\frac{x^r}{h_r}\right\|\right), \qquad (3.2)$$

where x^s is the spatial part, x^r is the color part of a feature vector, $k(x)$ is the common profile in both domains, h_s and h_r are the corresponding kernel bandwidths, and C is a constant for normalization.

The sample mean at x is defined as:

$$\mathrm{m}(x) = \frac{\sum_{i=1}^{n} x_i K(x - x_i)}{\sum_{i=1}^{n} K(x - x_i)}. \qquad (3.3)$$

The vector, x, is updated in the form of iteration such that $x \leftarrow \mathrm{m}(x)$ with $\mathrm{m}(x) = \{\mathrm{m}(x); x \in \Re^d\}$. The difference $\mathrm{m}(x) - x$ is called the *mean shift*, and the repeated movement of data points to the sample mean is called the mean-shift algorithm. The mean-shift algorithm iterates until converged with zero gradient, i.e. $\mathrm{m}(x) - x = 0$. The convergence is guaranteed at a nearby point. Once the mean-shift algorithm is converged, the local mean is shifted toward the region where the majority of the points reside, that is, the local maxima or the mode of the region.

Many segmentation algorithms treat the *mean* as an optimized measure for partitioning the feature space. For example, the *K*-Means clustering algorithm [93] is aimed at minimizing the within-group sum of squared errors. The initial cluster centers are randomly or strategically chosen, and there is no guarantee that any execution of the algorithm will reach the global minimum. With the algorithm, we can only say that a local minimum has been reached, and the optimization goal becomes elusive. With the mean-shift algorithm, the clustering process can be viewed as the result of some natural process since the convergence to the local maxima is guaranteed.

The kernel used in the experiment is the Epanechnikov kernel [15]. This kernel provides a similar performance to the Gaussian kernel but with a much simpler structure. The Epanechikov kernel profile is defined as follows:

$$k_E(x) = \begin{cases} 1 - \|x\|^2 & 0 \le x \le 1, \\ 0 & x > 1. \end{cases} \tag{3.4}$$

Our color image segmentation algorithm consists of two steps.

1. Mean-shift filtering which smoothes the input image by running the mean-shift algorithm.
2. Mean-shift segmentation which delineates the smoothed image and purges the small regions.

3.3.3 Skin-color Modeling

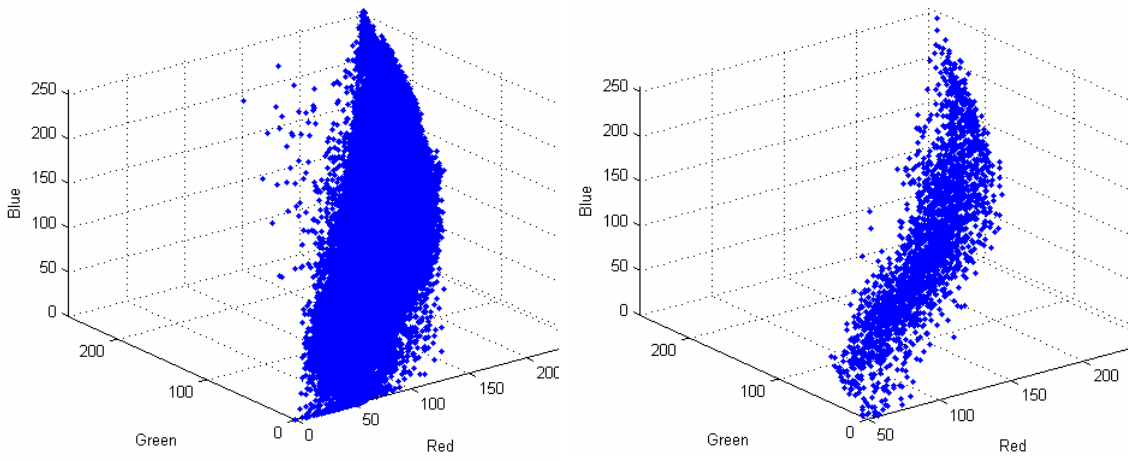
A skin-color model can be trained by learning from a large set of sample skin-color pixels. The training set is generated in a pre-processing stage. Since we consider face color under varying illumination, the skin-color distribution is no longer unimodal. In fact, the face color is distributed from shadowed faces to strong overhead light-projected faces, as well as faces under normal lighting condition. One way to tackle this varying distribution is to use a Gaussian mixture model.

In our approach, the training skin-color samples are obtained from face regions after segmentation. Each segmented face region is represented by a skin color, which is then used for training. This can ensure that there is no outlier skin color in the training set. The training face images are extracted from the HHI MPEG-7 face database, and each face image is segmented with the mean-shift algorithm, as described in the previous section. After segmenting a face image into a number of regions, the modes of the skin-color pixels are left in the respective facial regions. In other words, the mode is used to represent the facial skin-color pixels in a segmented

face region. Fig. 3.6 shows some face images and their corresponding segmented results under various lighting conditions using the mean-shift algorithm [17].

Fig. 3.6. Face images under various lighting conditions: (a) normal lighting, (b) segmented faces in (a), (c) dark and side light, (d) segmented faces in (c), (e) strong overhead light, and (f) segmented faces in (e).

A total of 366,431 skin-color pixels were extracted from 206 faces in the HHI MPEG-7 face database. After color image segmentation with the mean-shift algorithm, only 2,425 skin-color pixels were left. The average number of regions in each segmented face is 11.7, i.e. 11.7 pixels per segmented face on average. Figures 3.7(a) and 3.7(b) show the color distributions of the original faces and segmented faces. In Fig. 3.7(a), there are pixels other than the skin-color distribution. Obviously, this is due to the colors other than the skin color, such as eyeball, eyebrow, mustache, lip, etc. In Fig. 3.7(b), there are only a few pixels apart from the skin-color distribution. After segmentation, the color differences among facial regions are largely removed, and the colors of the skin pixels are fused into the skin-color domain. This is obvious in the segmentation results shown in Fig. 3.6; the eyes, mouth and eyebrows are removed and segmented to become skin-like colors. By observing the change of skin-color distribution in Fig. 3.7(a), the mean-shift process sharpens the skin-color pixel distribution in Fig. 3.7(b) to converge towards the mean of skin colors. This leads the skin colors in uneven lighting to converge to an optimal pixel value, thus less variance will appear under extreme illumination conditions.



(a) (b)

Fig. 3.7. RGB distributions of the face color: (a) RGB distribution of original faces, and (b) RGB distribution of segmented faces.

K-Means Face-color Clustering

Before applying the Gaussian mixture model to describe the distributions of the segmented skin colors in Fig. 3.7(b), we have to cluster the skin colors to obtain the initial parameters and the number of clusters for the Gaussian mixture model. Each of the clusters should clearly represent the face color under different lighting conditions. The *K*-Means algorithm is applied to cluster the skin-color pixels for $k \geq 1$. As the RGB color space does not contain any information about lighting, the face color is first translated to the compensated YCbCr color space as described in Section 3.3.1. The face color is then divided into $k+1$ decision regions; *k* face-color regions and the complementary non-face region. In our approach, we set $k = 5$, which can empirically segment the skin and non-skin color well throughout our experiments. Fig. 3.8 shows the result after the *K*-Means algorithm.

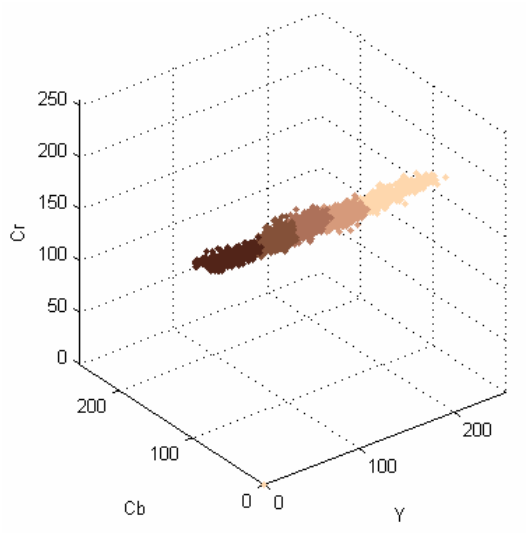


Fig. 3.8. Skin-color clustering with the K-Means algorithm, where each cluster is displayed with its *mean* color.

Face-color Modeling via Gaussian Mixture Model

With the results from the *K*-Means algorithm, the skin-color distribution can be modeled using the Gaussian mixture model, which is a semi-parametric model. In order to optimize the mixture parameters, maximum-likelihood can be used to seek the best parameters based upon the results from the *K*-Means algorithm. Then, the Expectation-Maximization (EM) algorithm [32, 69] is used to estimate the many-to-one mapping from the Gaussian mixture model.

The modeling starts with the general estimation model. The distribution of an input color vector $x \in \Re^d$ is a mixture of k Gaussian probability density functions:

$$f(x \mid \theta) = \sum_{j=1}^{k} \alpha_j \frac{1}{\sqrt{(2\pi)^d \left|\sum_j\right|}} \exp\left\{-\frac{1}{2}(x-\mu_j)^{\mathrm{T}} \sum_j^{-1} (x-\mu_j)\right\} \qquad (3.5)$$

where the parameter set $\theta = \left\{\alpha_j, \mu_j, \sum_j\right\}_{j=1}^{k}$ consists of the probability $\alpha_j > 0, \sum_{j=1}^{k} \alpha_j = 1$, the mean $\mu_j \in \Re^d$, and the covariance matrix $\sum_j$ which is a $d \times d$ a positive definite matrix. Given a set of color vectors $x_1, \ldots, x_n$, the mixture parameters can be estimated using the ML principle

$$\theta_{\mathrm{ML}} = \arg\max_{\theta} f(x_1,...,x_n \mid \theta) \qquad (3.6)$$

The estimation problem is best solved using the Expectation-Maximization (EM) algorithm [69], which consists of the following two-step iterative procedure:

- Expectation step:

$$\tau_{tj}^{i} = \frac{\alpha_j^i f(x_t \mid \mu_j^i, \Sigma_j^i)}{\sum_{c=0}^{k} \alpha_c^i f(x_t \mid \mu_c^i, \Sigma_c^i)}, \quad j = 1,...,k;\ t = 1,...,n; \qquad (3.7)$$

- Maximization step:

$$\alpha_j^{i+1} = \frac{1}{n}\sum_{t=1}^{n} \tau_{tj}^{i},$$

$$\mu_j^{i+1} = \frac{\sum_{t=1}^{n} \tau_{tj}^{i} x_t}{\sum_{t=1}^{n} \tau_{tj}^{i}},$$

$$\Sigma_j^{i+1} = \frac{\sum_{t=1}^{n} \tau_{tj}^{i} (x_t - \mu_j^{i+1})(x_t - \mu_j^{i+1})^{\mathrm{T}}}{\sum_{t=1}^{n} \tau_{tj}^{i}}. \qquad (8)$$

The expectation step computes the a posteriori probabilities τ_{tj}^{i} of the point x_t belonging to the j^{th} cluster with i=1, 2, … until converged. The maximization step uses the data from the expectation step as if they were the actual measured data to determine an ML estimate of the parameters. The updating process performs the expectation step and the maximization step iteratively until converged. The convergence can be determined by examining the change of the parameter set θ in two successive iterations less than a predefined threshold, i.e. $\left\|\theta^{i+1} - \theta^{i}\right\| < \varepsilon$, for some ε and some appropriate distance measure $\|\cdot\|$. In our approach, the iteration process will be stopped when the change of the probabilities is less than a threshold of 1%.

From modeling the face color, a new parameter set $\theta = \{\alpha_j, \mu_j, \Sigma_j\}_{j=1}^{k}$ is estimated. Based on the skin-color model, a pixel in a color image is determined to be of skin color or not by the following steps:

- Each pixel in the RGB color format is first translated to the compensated YCbCr color space.
- With the input color vector x=(y, cb, cr), the Gaussian p.d.f., f_j, j=1,…,k, are computed to find the one with the highest probability, i.e. $\arg\max_j f(x \mid \alpha_j, \mu_j, \Sigma_j)$.
- Suppose that $f(x \mid \alpha_j, \mu_j, \Sigma_j)$ results in the highest probability. The following discriminant function is computed to obtain the face mask, $P_{face}(x)$,

$$P_{face}(x) = \begin{cases} 1 & \text{if } [(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j) < threshold]; \\ 0 & \text{otherwise;} \end{cases} \qquad (3.9)$$

  where 1 represents face color and 0 is non-face color.

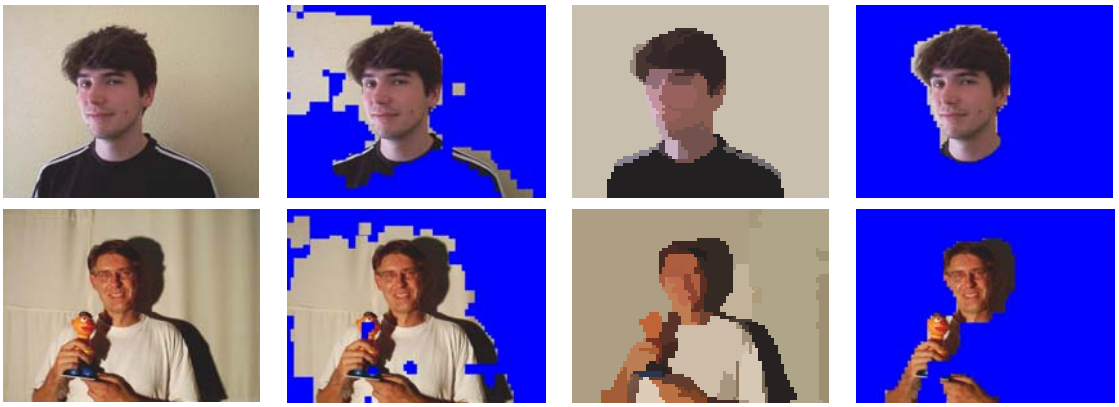
For a probability that is less than a pre-defined threshold, the color vector x will be declared a non-skin color; otherwise, the color vector x is skin color. The threshold is determined empirically, and is set at the standard deviation of the Gaussian distribution concerned, σ, throughout our experiment. With this threshold, the skin-color region and background can be segmented reliably.

Region-based Skin-color Segmentation

Traditional skin-color segmentation is performed based on a pixel-by-pixel approach. Each pixel in an image is checked to determine whether it is of skin color or not. After this skin-color segmentation process, some small holes will be introduced at the eye, nose and mouth regions. These holes can be removed by using the morphological open and close operations. Although the morphological operations

have been used to fill and to remove small holes, some big holes in the background cannot be filled, as shown in Fig. 3.9(b). The result will become worse when the image concerned has uneven lighting condition, or is under a strong illumination effect.

Because of the drawback in the pixel-by-pixel approach, we propose performing skin-color segmentation by using a region approach. In our algorithm, an image is scaled to 80×60 to reduce the computational complexity. The color image is then segmented by the mean-shift algorithm, and the results are shown in Fig. 3.9(c). As a result, the image is segmented into many regions. The details of the eyes, nose and mouth are merged with the skin color. In each region, this is represented by the mode of the color pixels. Maximum likelihood by means of mixture-of-Gaussians with the optimized mixture parameters obtained by the EM algorithm is then applied. If the probability of a mode is larger than a certain threshold, the corresponding color will be classified to be a skin color, and thus, the whole region will be declared a skin-color region. The average number of regions in an image is 25 after segmentation based on the HHI MPEG-7 face database. Comparing to the pixel-by-pixel method, the region-based approach can achieve a better performance level under uneven lighting conditions, as shown in Fig. 3.9(d).



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

Fig. 3.9. Skin-color segmentation under uneven light conditions: (a) original image, (b) skin-color segmentation with pixel-by-pixel approach, (c) color image segmentation with mean-shift algorithm, and (d) skin-color segmentation with region-based approach.

3.4 Possible Eye Candidate Detection

An efficient way to detect an eye region is by means of valley detection [49], as the gray-level intensities in an eye region are relatively lower than in its neighborhood. However, under poor lighting conditions, the eye regions are usually shadowed; this causes the valley detection to fail. In this section, we will present an improved method which can detect eye regions more reliably under various lighting conditions.

Since the eyes are surrounded by skin, there is a significant color difference between the eye region and the skin color. Under the YCbCr color space, we can observe that the iris has a lower gray-level intensity, a higher Cb value, and a lower Cr value than the surrounding skin color. These kinds of properties will be used to determine whether a pixel in a segmented skin-color region is a possible eye candidate or not.

In our approach, an image is first segmented with the mean-shift algorithm at resolution 80×60, denoted as $I_{MS}(c)$, where $c=(y, cb, cr)$. Fig. 3.10(b) shows the mean-shift segmented image with the eyes under overhead lighting. In the segmented

image, the details of the eye are removed and replaced by skin-like color. The skin color is still preserved. Throughout our experiment, the original image, $I(c)$, and the segmented image, $I_{MS}(c)$, are scaled to 200×150. At this resolution, the computational complexity is reduced and the details of the eye regions are retained.

Suppose that the Y, Cb, and Cr components of a pixel in an image are denoted as $I(y)$, $I(cb)$, and $I(cr)$, respectively, while as $I_{MS}(y)$, $I_{MS}(cb)$, and $I_{MS}(cr)$ for the corresponding mean-shifted or segmented image. Based on the observed difference in color between the iris and the skin, $I(y)$ should be less than a certain threshold; the ratio between $I(cb)$ and $I_{MS}(cb)$ should be greater than 1; and the ratio between $I(cr)$ and $I_{MS}(cr)$ should be less than 1 for an eye candidate. For the skin color, the above two ratios should all be very close to 1. The eye candidates in an image can therefore be determined as follows:

$$P_y(y) = \begin{cases} 1 & I(y) < t_y \\ 0 & \text{otherwise} \end{cases},$$

$$P_{cb}(cb) = \begin{cases} 1 & \dfrac{I(cb)}{I_{MS}(cb)} > t_{cb} \\ 0 & \text{otherwise} \end{cases},$$

$$P_{cr}(cr) = \begin{cases} 1 & \dfrac{I(cr)}{I_{MS}(cr)} < t_{cr} \\ 0 & \text{otherwise} \end{cases}, \qquad (3.10)$$

where $P_y(y)$, $P_{cb}(cb)$ and $P_{cr}(cr)$ represent a possible eye candidate when the Y, Cb and Cr components are used, respectively, and the thresholds t_y, t_{cb} and t_{cr} are the corresponding thresholds for the Y, Cb and Cr components. Therefore, possible eye candidates, $Eye(x)$, can be determined as follows:

$$Eye\left(x = \{y, cb, cr\}\right) = P_y(y) \cap P_{cb}(cb) \cap P_{cr}(cr) \cap P_{face}(x). \qquad (3.11)$$

In the above equation, we confine the detection within the segmented skin-color regions, $P_{face}(x)$, from (3.9). The possible eye candidates, $Eye(x)$, are illustrated in Fig. 10(c).

(a) (b) (c)

Fig. 3.10. Possible eye candidate detection: (a) original image, (b) color image segmentation with mean-shift algorithm, and (c) possible eye candidate.

From Figure 3.10(c), it can be observed that the possible eye candidates are located around the face contour and hair, and some in the face region. Some of these eye candidates should be removed to reduce the number of possible face candidates, so that the computational complexity of searching the possible face candidates can be reduced. We observe that the eye has a strong horizontal edge, while the face contour has a strong vertical edge when the face orientation is between –45˚ and +45˚. In addition, the skin color is yellowish, and the eye is white and dark. Therefore, the difference in red component should be large. These properties are used to reduce the number of possible eye candidates.

The edge map of a face image is generated using the Sobel edge detector, but only with its luminance component. The horizontal edge is denoted as $S_H(y)$ and the vertical edge as $S_V(y)$, where y is the luminance component. The set of selected possible eye candidates, $Eye'(x)$, is formed as follows:

$$Eye'(x) = (S_H(y) < T_{SH}) \cap (\overline{S_V(y) < T_{SV}}) \cap Eye(x), \qquad (3.12)$$

where T_{SH} and T_{SV} are the thresholds for the horizontal and vertical edge intensities. In (3.12), whenever a possible eye candidate has a strong horizontal edge intensity and a weak vertical edge intensity, it will not be removed. As some possible eye

candidates may fall into the hair and skin regions, these candidates can be removed by measuring the difference in the red component in their neighborhood. A 3×3 window is located at the possible eye candidates, as shown in Fig. 3.11.

R_1	R_2	R_3
R_4		R_5
R_6	R_7	R_8

Fig. 3.11. A 3×3 window for red component.

A possible eye candidate will be removed if

$$\max(R) - \min(R) < T_{red}\,, \tag{3.13}$$

where $R=R_1,\ldots, R_8$ and T_{red} is the threshold for the difference between the maximum and minimum red components within the window. The possible eye candidates are removed using (3.12) and (3.13), and the results are shown in Fig. 3.12(b). Although the number of possible eye candidates has been greatly reduced, some of the remaining possible eye candidates are connected to each other. Further elimination can be done without removing the local information. A 3×3 searching window is located at each possible eye candidate to group the surrounding candidates into one. The results are shown in Fig. 3.12(c).

(a) (b) (c)

Fig. 3.12. Possible eye candidate reduction: (a) possible eye candidates, (b) reduction of possible eye candidates by Sobel and red color distance, and (c) possible eye candidates in (b) grouped by 3×3 window.

Since the size of a human face is proportional to the distance between its two eyes, a possible face region containing the eyebrows, eyes, nose and mouth can be formed based on this relationship. In our approach, a square block is used to represent

possible face candidates. The size of the square block is determined by means of the head model in [117]. Based on the head model and the locations of the possible eye pairs, a population of possible face regions with different locations, sizes, and orientations are generated by pairing the possible eye candidates. A possible face is formed if the following criteria are satisfied:

$$T_{\theta 1} < \theta < T_{\theta 2}\,,\; T_{size1} < F_{size} < T_{size2}\,,$$

$$\left|F_{left\_eye} - F_{right\_eye}\right| < T_{eye}\,,\text{and } Total_{seg} > T_{seg}\,, \tag{3.14}$$

where θ represents the orientation of the face, as shown in Fig. 3.13, F_{size} is the face length, F_{left_eye} and F_{right_eye} are the average gray-level intensities at the left and right eye regions, and $Total_{seg}$ is the area of segmented skin-color in the skin-color segmentation with mixture-of-Gaussian model.

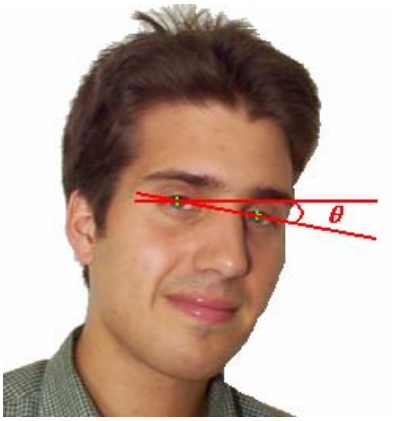


Fig. 3.13. The orientation of the face.

The range of the orientation angle of a human head is assumed to be between $T_{\theta 1}$ and $T_{\theta 2}$, while the face size is assumed to vary from T_{size1} to T_{size2}. The similarity between the left eye and right eye is measured by calculating the absolute difference in their gray-level intensities, and is assumed to be less than a threshold T_{eye}. The threshold T_{seg} is to ensure that most of the face area is covered by skin-colored pixels. Fig. 3.14 shows those possible face regions satisfying (3.14).

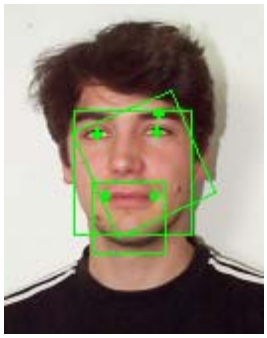


Fig. 3.14. Selection of possible face regions.

Once a face candidate is selected, its orientation and size are normalized. Then, histogram normalization [76] is applied to the Y component of the selected face candidate to compensate for non-uniform lighting; this can help improve detection reliability and accuracy. In this normalization process, the histogram of a possible face region is transformed into the histogram of a reference face image, as shown in Fig. 3.15. The method is effective since all faces have the same structure, similar shape and illumination properties. Thus, the transformed face can have approximately the same illumination as the reference face. Finally, the image will be passed to the final stage for further verification.

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

Fig. 3.15. (a) Reference face. (b) Possible face candidate. (c) Gray-scale image of the face candidate in (b). (d) Histogram normalization of the face candidate in (b).

3.5 A Two-step Face Verification using Eigenmask

In order to determine whether the normalized face candidate is a face or not, the similarity between the face candidate and a face template is measured. In our approach, a two-step face verification procedure is performed. Instead of using a

single face template, a face region is separated into two parts: the upper part contains the eyes, while the lower part contains the nose and mouth. This can make the similarity measure more localized. A true face will be declared only if a face region has both its upper and lower parts similar to the corresponding two face templates. In our algorithm, the nose and mouth form the lower part of our face template, while the eyes form the upper part. All the training images used to construct the face templates are normalized to a specific size.

Both the upper and lower templates are gray-scale images, and are obtained by calculating the average of a set of pre-processed training face images. Let $F_{template}$ be the face template, which is divided into the upper and lower parts of our face template. Suppose a face region is normalized to a size of $M \times N$. $F_{template}$ is calculated as follows:

$$F_{template}(y) = \frac{1}{n}\sum_{i=0}^{n} P_i(y), \qquad (3.15)$$

where n and $P_i(y)$ are the number of training face images and the i^{th} training face image, respectively. The training set contains face images of different races, ages, with and without glasses and a moustache. The training images and the corresponding face template are shown in Figures 16 and 18(a) for the upper faces, and Figures 3.17 and 3.18(b) for the lower faces.

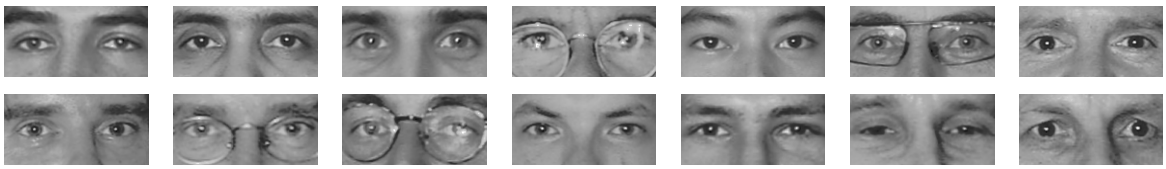


Fig. 3.16. The training set for upper face.

Fig. 3.17. The training set for lower face.

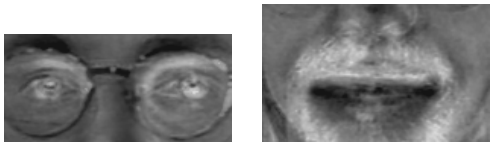
(a) (b)

Fig. 3.18. Face templates: (a) upper face template, (b) lower face template.

The distance between a possible face region and the corresponding face template can be measured by means of the Euclidean distance with a certain weighting function based on the importance of the human facial features. The weighting function can be obtained from the eigenmask in [112], denoted as E_{mask}. The eigenmask is generated from the first eigenface of the training face images. Each of the training face images is considered as a vector, Γ, of dimension $1\times MN$, and $F_{template}$ in (3.15) is represented as a vector, Ψ, of dimension $1\times MN$. Then, the covariance matrix A of dimension $MN\times MN$ is computed based on the training face images as follows:

$$A = \Phi \cdot \Phi^{\mathrm{T}}, \tag{3.16}$$

for $\Phi = (a_0, a_1, ..., a_n)$ and $a_i = \Gamma_i - \Psi$, and Ψ is the average of Γ_i, where n is the number of training face images. The eigenvectors of the matrix A represent the principal components of the training face images, which have larger magnitudes at important locations such as the eyes, nose, and mouth. The eigenmask used in the distance measure is generated based on the absolute value of the first eigenvector. Figures 3.19 shows the eigenmasks generated whose values are normalized between 0~255 and are used as a weighting function in the distance measurement. As shown in Figures 19, the eye, nose and mouth regions have higher magnitudes than other facial regions and with glasses.



(a) (b)

Fig. 3.19. The eigenmasks: (a) eigenmask for upper face, (b) eigenmask for lower face.

The use of the eigenmasks can reduce false alarms and increase the distance when comparing a non-face region to the upper and lower face templates. The distance measure between an input candidate and the template is given as follows:

$$\varepsilon = \frac{1}{MN}\sum_{y=0}^{M}\sum_{x=0}^{N} E_{mask}(x,y)\cdot[I(x,y) - F_{template}(x,y)]^2 , \qquad (12)$$

where I is the luminance component of the possible face candidate. The face candidate will be declared a true face if the values, ε_{upper} and ε_{lower}, for the upper and lower face parts are both smaller than the thresholds T_{upper} and T_{lower}, respectively. In our algorithm, as the upper region is more important, we compute the upper part first. The calculation of the lower face will be performed if the upper face has been verified. Therefore, the computation required can be reduced. For overlapping regions, the one with the lowest value of

$$\varepsilon_{total} = \frac{\varepsilon_{upper} + \varepsilon_{lower}}{2}, \qquad (3.17)$$

will be chosen as the true face region.

3.6 Face Boundary Verification

Once the face has been verified and selected in the two-step eigenmask verification process, the selected faces are further verified based on the appearance of their face boundaries in order to reduce the false alarm. Some face boundary regions, as shown in Fig. 20, are first extracted, and feature vectors, x, are formed. The black rectangles in Fig. 3.20(b) represent face regions, which have been verified by the two eigenmasks. The feature vector of a face boundary does not include the face region.

Fig. 3.20. Examples of face boundary regions: (a) original face, (b) face boundary regions extracted from the original faces.

Since the number of training face boundaries is much smaller than the dimensionality of the corresponding feature vectors, the covariance matrix is singular in practice. Thus, the face boundary vector, x, is first projected onto a lower-dimensional subspace by means of principal component analysis [67]. The low-dimensional feature vector $y = \mathbf{E}_M^T \tilde{x}$, where $\tilde{x} = x - \bar{x}$ and $\mathbf{E}_M$ is a matrix containing M column vectors, which are the eigenvectors of the corresponding covariance matrix with the largest eigenvalues. The face boundary region can be modeled by a Gaussian density function:

$$P(y) = \frac{\exp\left[-\frac{1}{2}(y-\mu)^T \Sigma^{-1}(y-\mu)\right]}{(2\pi)^{M/2}|\Sigma|^{1/2}}, \qquad (3.18)$$

where $P(y)$ is an M-dimensional Gaussian density function with mean vector μ and covariance Σ of the low-dimensional training samples. In our algorithm, the width of the face boundary is 5 pixels at each of the four borders. As the boundary regions mainly contain the face contours only, so a Gaussian density function is sufficient to represent the appearance of the face boundary, rather than using a mixture-of-Gaussians model. With a face candidate verified by the two eigenmasks, its boundary region is projected onto the eigenspace to form a low-dimensional vector. Then the likelihood, $P(y)$, of the region being a face boundary is measured by using (3.18). If $P(y)$ is larger than a certain threshold, the face candidate will be considered a true face.

3.7 Experimental Results

Experiments were carried out to evaluate the detection performance of our algorithm based on using the HHI MPEG-7 face database, the AR face database and the CMU PIE face database. The aim of our proposed method is to detect the frontal or near-frontal view of faces under varying lighting conditions and facial expressions. In our experiments, a face is considered to be correctly detected if the detected position of the two eyes is exactly matched. A face is said to be missed if the face region can be located but the two eye positions are mismatched. A detection is said to be a false alarm if the selected face candidate does not cover the true face region.

The HHI MPEG-7 face database [2] contains 206 images, each of 640×480 pixels in size. The database contains people of different races, and under varying lighting conditions ranging from dark and shadow to strong overhead-projected, and from frontal view to profile file. Thus, 151 images with varying lighting conditions were selected from the HHI face database, and those with profile views were ignored in the experiment. The detection performance is tabulated in Table 3.1 and results are shown in Fig. 3.21. Using the traditional pixel-by-pixel skin-color segmentation, the detection rate is 85.4% without face boundary verification. By using our proposed region-based skin-color segmentation, the overall detection rates are 92.7% and 91.3% with and without the use of face contour verification, respectively. To investigate the performance of our algorithm under different lighting conditions, we have classified the face images according to the lighting conditions; these include overhead lights, dark or side lights, strong overhead lights, and strong side lights. The corresponding detection rates are 100%, 93.5%, 86.4%, and 80.8%, respectively.

The AR face database [62] contains 126 faces, with 70 men and 56 women each of 768×576 pixels in size. The face images in this database are of frontal view, but have different facial expressions, illumination conditions, and occlusions. We selected 920 images with varying illumination and facial expressions from this database. The images with occlusion are ignored. The detection performance is tabulated in Table 3.2 and results are shown in Fig. 3.22. Since the images in the AR face database do not have background, only the region-based skin-color segmentation is employed. The overall detection rates are 97.6% and 96.0% with and without using face contour verification, respectively. We have classified face images according to gender and lighting conditions (overhead lights and strong side lights). The corresponding detection rates for male face images are 99.7% and 94.6%, and 100% and 94.6% for females, respectively.

The CMU Pose, Illumination, and Expression (PIE) database [99] contains 41,368 images of 68 people, and each image is 640×486 pixels in size. There are 13 different poses, 43 different illumination conditions, and 4 different expressions for each person. We selected 3,264 images that contain frontal and near frontal faces with different genders, different races, and under varying lighting conditions ranging from dark lights, side lights and strong lights. With the traditional pixel-by-pixel skin-color segmentation, the detection rate is 76.0% without using face boundary verification. Using our proposed region-based skin-color segmentation, the overall detection rates are 87.6% and 84.2% with and without the use of face boundary verification, respectively. The detection performance is tabulated in Table 3.3 and results are shown in Fig. 3.23. We have classified the face images according to different lighting conditions, including dark lights, strong side lights, and strong

overhead lights. The corresponding detection rates are 90.9%, 86.0% and 88.1%, respectively.

Lighting condition	Overhead lights	Dark or side lights	Strong overhead lights	Strong side lights	Total
No. of images	54	49	22	26	151
Detection method	Pixel-by-pixel skin-color segmentation and no face boundary verification				
No. of correctly detected faces	53	43	16	17	129
No. of missed face	1	5	3	7	16
No. of false alarms	0	1	3	2	6
Detection Rate (%)	98.1%	87.7%	72.7%	65.4%	84.8%
Detection method	Region-based skin-color segmentation and no face boundary verification				
No. of correctly detected faces	53	46	18	21	138
No. of missed face	1	2	2	3	8
No. of false alarms	0	1	2	2	5
Detection Rate (%)	98.1%	93.5%	81.8%	80.8%	91.3%
Detection method	Region-based skin-color segmentation and with face boundary verification				
No. of correctly detected faces	54	46	19	21	140
No. of missed face	0	2	2	3	7
No. of false alarms	0	1	1	2	4
Detection Rate (%)	100%	93.5%	86.4%	80.8%	92.7%

Table 3.1. Detection performance based on the HHI MPEG-7 face database.

Lighting condition	Overhead lights	Strong side lights	Overhead lights	Strong side lights	
Gender	Male	Male	Female	Female	Total
No. of images	298	223	231	168	920
Detection method	Region-based skin-color segmentation and no face boundary verification				
No. of correctly detected faces	297	205	231	150	883
No. of missed face	1	18	0	18	37
No. of false alarms	0	0	0	0	0
Detection Rate (%)	99.7%	91.9%	100%	89.0%	96.0%
Detection method	Region-based skin-color segmentation and with face boundary verification				
No. of correctly detected faces	297	211	231	159	898
No. of missed face	1	12	0	9	22
No. of false alarms	0	0	0	0	0
Detection Rate (%)	99.7%	94.6%	100%	94.6%	97.6%

Table 3.2. Detection performance based on the AR face database.

Lighting conditions	Dark	Strong side lights	Strong overhead lights	Total
No. of images	408	1360	1496	3264
Detection method	Pixel-by-pixel skin-color segmentation and no face boundary verification			
No. of correctly detected faces	353	965	1163	2481
No. of missed face	41	385	316	742
No. of false alarms	14	10	17	41
Detection Rate (%)	86.5%	70.9%	77.7%	76.0%
Detection method	Region-based skin-color segmentation and no face boundary verification			
No. of correctly detected faces	362	1099	1287	2748
No. of missed face	35	255	196	486
No. of false alarms	11	6	13	30
Detection Rate (%)	88.7%	80.8%	86.0%	84.2%
Detection method	Region-based skin-color segmentation and with face boundary verification			
No. of correctly detected faces	371	1169	1318	2858
No. of missed face	30	187	167	384
No. of false alarms	7	4	11	22
Detection Rate (%)	90.9%	86.0%	88.1%	87.6%

Table 3.3. Detection performance based on the CMU PIE database.

The experiments were conducted on a Pentium IV 1.7GHz computer. As the resolution of the images is high, each image is scaled to a size of 80×60 when we perform the mean-shift algorithm and skin-color segmentation, and then the images are scaled to 200×150 for possible eye candidate detection and face verification. The resolutions used can reduce the computational complexity of our algorithm, and so speed up the detection process without losing the details of faces and eyes. The average processing time for locating faces in a picture based on the HHI face database, the AR face database and the CMU PIE database ranges from 0.3s to 1.5s. The experiments show that our method can achieve a high detection rate irrespective of lighting conditions and facial expressions. When the face boundary is also considered, the detection rate is improved and false alarms are reduced. For the HHI database, our algorithm can achieve a detection rate of 92.1% as compared to 88.89% in [119] and 90.07% for frontal and near-frontal faces in [36]. In the case of missed faces, we have found that the failure of our algorithm is mainly due to severely poor lighting conditions at the eye regions, or strong yellow lighting projected on the

clothes and hair. The yellow light makes the clothes and hair assume a skin-like color. Some false alarms and cases of missing faces occur when the HHI face database, the AR face database and the CMU PIE database are used, as shown in Figures 3.24 and 3.25, respectively. As our algorithm can also detect multiple faces in an image, some examples of the detection are shown in Fig. 3.26. In conclusion, our method can achieve a fast and high face detection rate under varying lighting conditions and facial expressions.

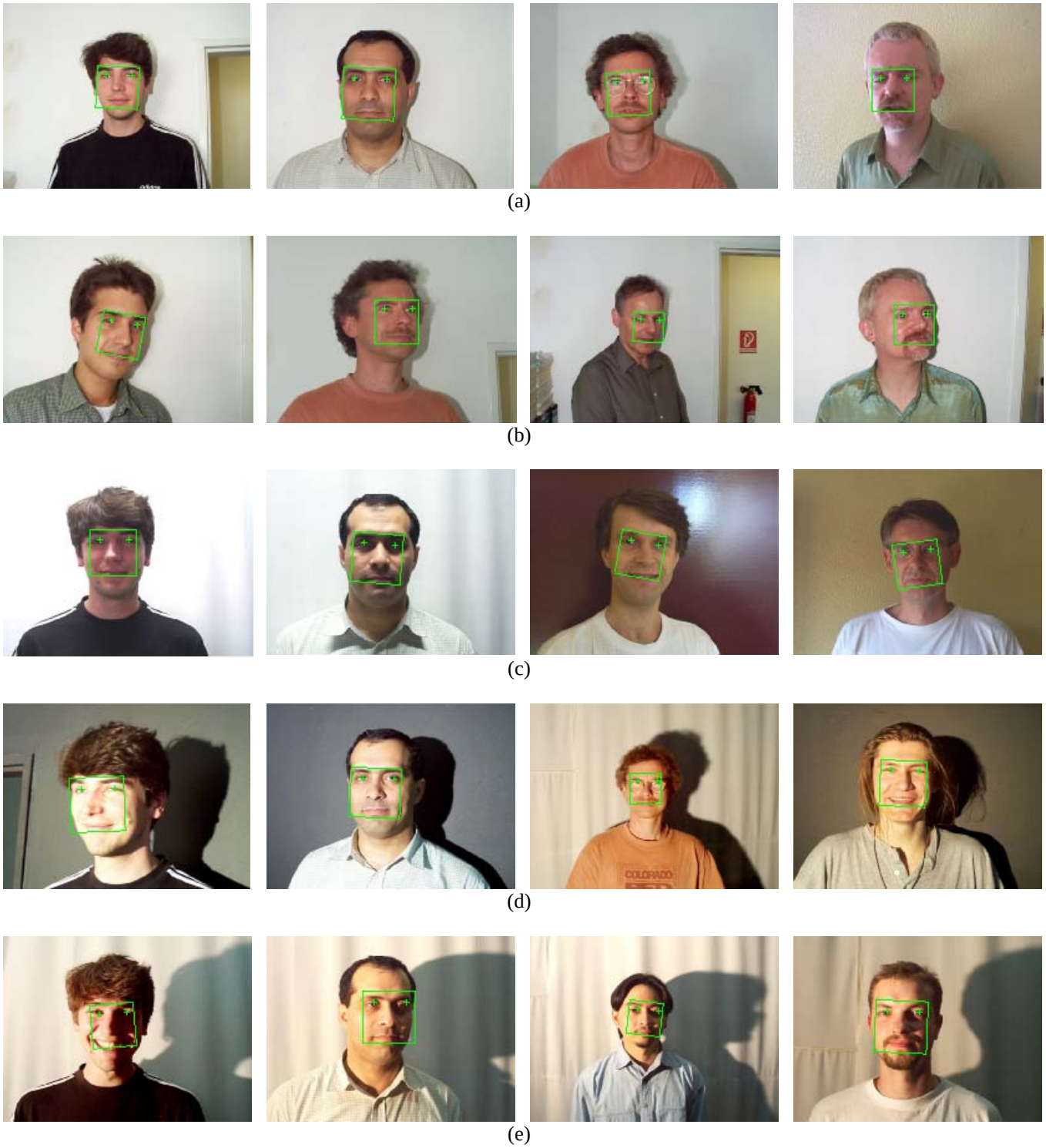


Fig. 3.21. Face detection results with the HHI MPEG-7 face database: (a) frontal view faces under overhead lights, (b) near frontal view faces under overhead lights, (c) faces under dark or side lights, (d) faces under strong overhead lights, and (e) faces under strong overhead side lights.

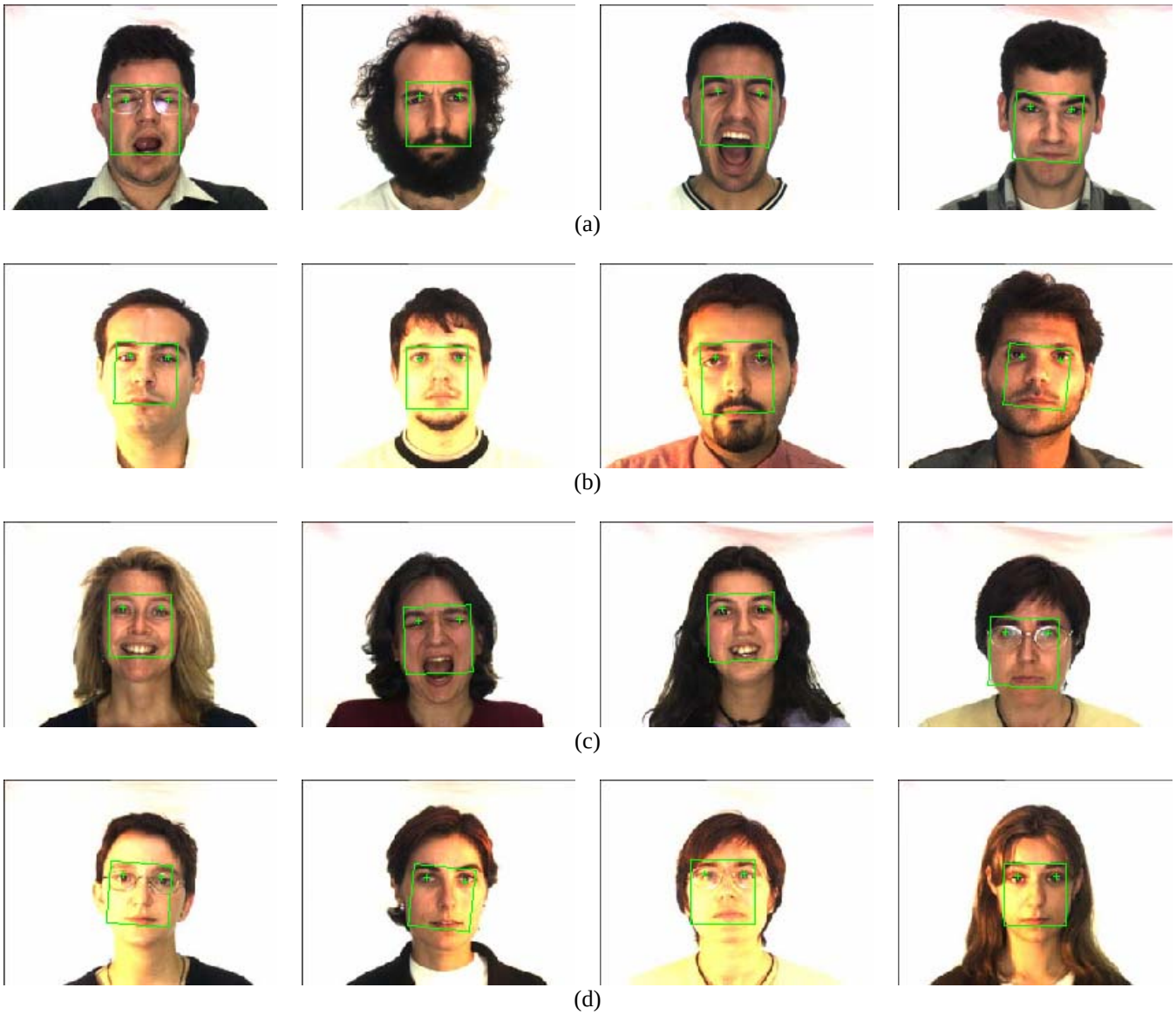


Fig. 3.22. Face detection results with the AR face database: (a) frontal view male faces with different facial expressions under overhead lights, (b) frontal view male faces under side lights, (c) frontal view female faces with different expressions under overhead lights, and (d) frontal view female faces under side lights.

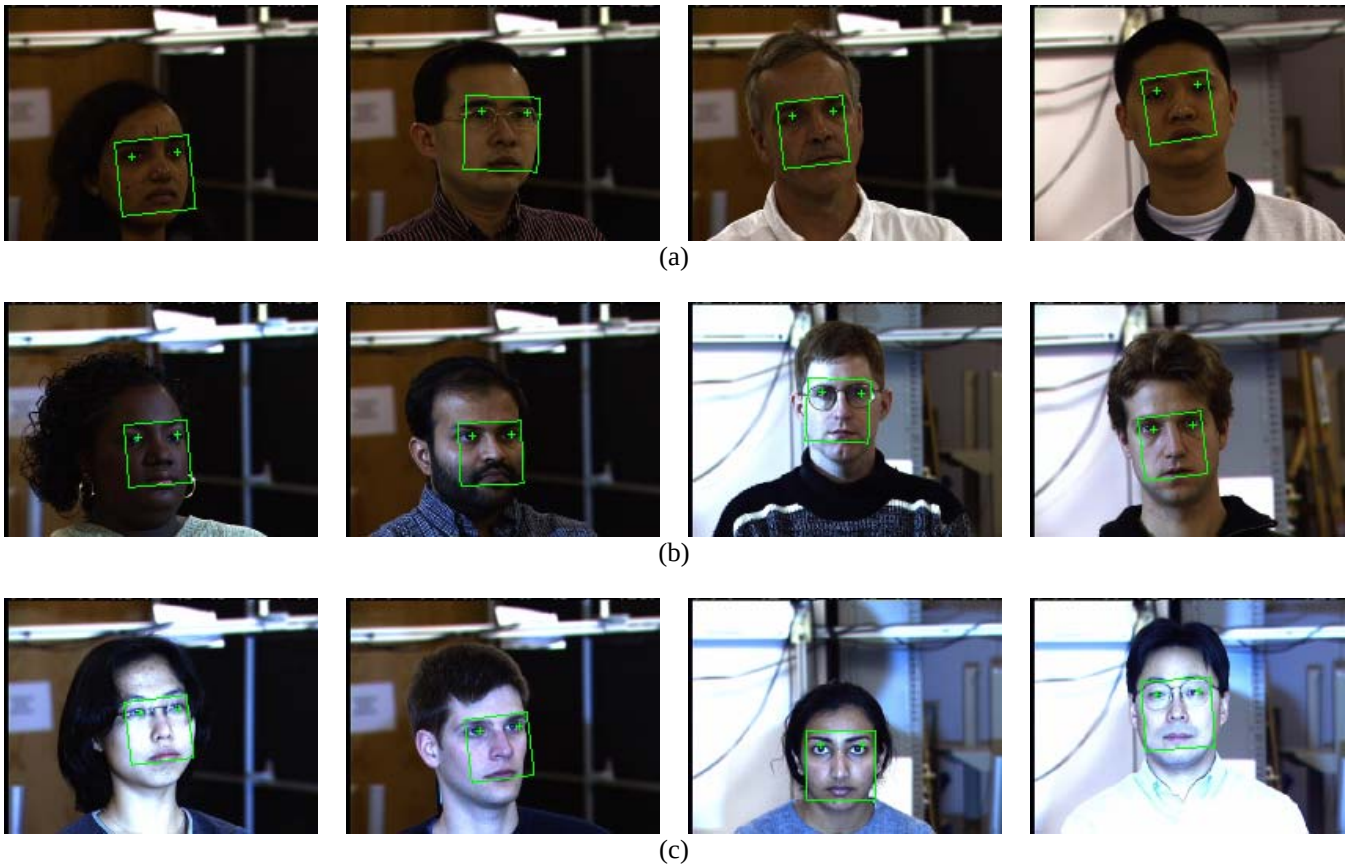


Fig. 3.23. Face detection results for frontal and near frontal view with the CMU PIE database: (a) dark lights, (b) side lights, and (c) strong overhead lights.

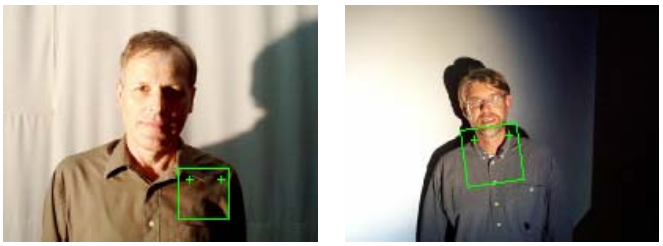


Fig. 3.24. Examples of false alarms.

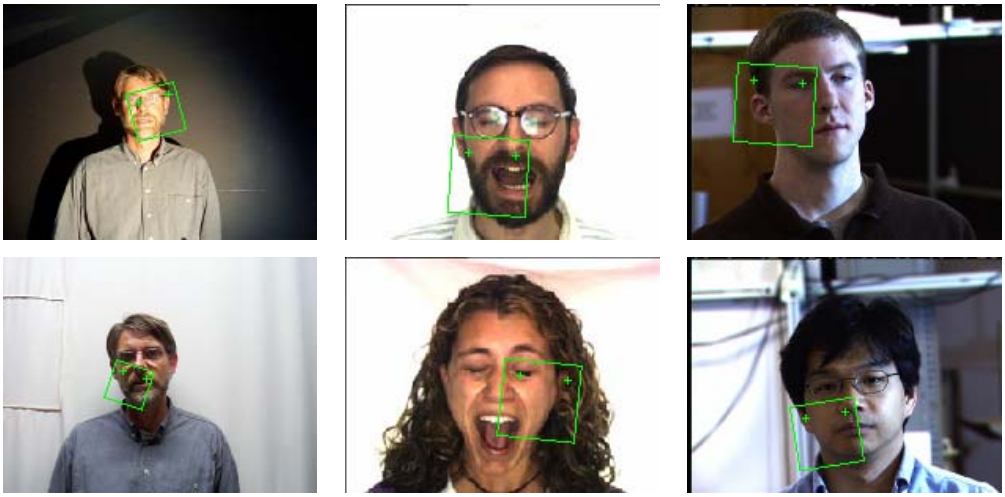


Fig. 3.25. Examples of missed faces.

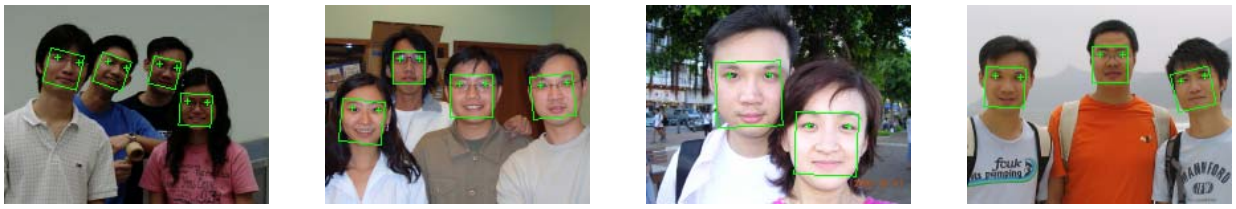


Fig. 3.26. Examples of multiple-face detection.

3.8 Conclusion

In this chapter, we have proposed a more reliable face detection approach under varying lighting conditions. In our algorithm, a color compensation scheme is adopted to alleviate the effect of strong lighting conditions on the skin colors. Then, we consider the distributions of the skin color of segmented regions under different lighting. Based on the color information, possible eye candidates are detected within the face-like regions, and possible face candidates are then formed by pairing two possible eye candidates in a face-like region. A two-step eigenmask verification process is proposed, with a weighting function used to measure the distance between a face candidate and the face template. Finally, boundary regions of the face candidates are verified with a Gaussian density function to reduce false alarms. Experimental results show that our algorithm can achieve a higher detection rate and reduce the

number of false alarms as compared to [36, 119]. Furthermore, our method can detect faces of different sizes and orientations under varying lighting conditions and different facial expressions.

Once the face has been detected in the first frame of a video sequence, the information about the face region will be used in the next stage for serching facial feature locations. After the required facial features have been located, the face in the subsequent video frames will be tracked. The face tracking algorithm will be described in the next chapter.

Chapter 4
Adaptive Template Matching for Face Tracking in Video Sequence

4.1 Introduction

For many vision-driven interactive user applications, face tracking is an essential processing step. Pose determination and gestures recognition can be obtained from face position and orientation. A limitation of most face-tracking algorithms is the lack of an effective appearance models. This is especially the case for template-matching methods, wherein the template is usually unable to adapt to an appearance changed over a long duration. When a face moves fast in a video sequence, the tracking will drift away from the target.

Visual tracking has been extensively studied in recent years. Most of the works has embodied some representation of image appearances. Jepson *et al.* [40] proposed to model the appearance of a face using the phase information based on the Gabor wavelet. The tracking algorithm is formulated by three components, namely, a stable component, a two-frame transient component, and an outlier process. The stable component adapts to the slowly varying properties of image appearance throughout the tracking process. The two-frame transient component provides additional information when the appearance model is being initialized or when the appearance is changing quickly. The outlier process accounts for outliers, which are expected to arise due to failures in tracking, occlusion, or noise. These components are updated and optimized with an online expectation-minimization algorithm. Approximately half of the time was spent computing the wavelet transform, which degrades the tracking efficiency. Loutas *et al.* [57] proposed a probabilistic model based on feature

point sets generated automatically and entropy measurement of motion parameters as the product of a prior probability and a likelihood function for face tracking and head orientation estimation. Comaniciu *et al.* [18] proposed a histogram-based target representation regularized by spatial masking with an isotropic kernel. The similarity between the target in a current frame and that in the previous frame is measured by means of the Bhattacharyya coefficient. The basin of attraction of the local maxima is optimized by the mean shift procedure.

This chapter proposes a robust, adaptive template matching method for face tracking. Facial features, including left eye, right eye and mouth, are being tracked based on the Gabor wavelet representations in the first stage. The greedy algorithm is used for tracking a face region efficiently. In the searching process, a triangular structure formed by the three important facial features is employed. In addition, an adaptive template matching algorithm can adapt to the slowly changing appearance of a face region, which is based on a key frame representation method for video analysis, namely Temporally Maximum Occurrence Frame [107].

4.2 The Gabor Feature

The Gabor wavelet [54, 79] has been used widely in texture analysis to provide features for texture classification and segmentation. It has been shown that Gabor wavelet representation optimally minimizes the uncertainty in the space and the frequency domain. The underlying texture information can be characterized by the orientation and the scale factor of the Gabor wavelets.

A Gabor wavelet is a complex exponential modulated by a Gaussian function in the spatial domain, as shown below,

$$h(x, y, w_0, \theta) = \frac{1}{2\pi\sigma^2} \exp\left[\frac{(x\cos\theta + y\sin\theta)^2 + (-x\sin\theta + y\cos\theta)^2}{2\sigma^2} \right] \times \left\{ \exp\left[iw_0 (x\cos\theta + y\sin\theta) \right] - \exp\left(\frac{-w_0^2\sigma^2}{2} \right) \right\}, \quad (4.1)$$

where the pixel position is defined by the x, y coordinates in the spatial domain, w_0 is the radial center frequency, θ is the orientation of the Gabor wavelet, and σ is the standard deviation of the Gaussian function along the x and y axes. The second term of the Gabor wavelet compensates for the DC value, because the DC response of the cosine component is a non-zero mean, while the sine component is zero mean. The Gabor representation can then be computed from the convolution of the image, $I(x,y)$, and the Gabor wavelets, $h(x,y)$, i.e.

$$C(x, y, w_0, \theta) = h(x, y) * I(x, y). \quad (4.2)$$

where * denotes the convolution operator. The convolution result, $C(x, y, w_0, \theta)$, corresponds to the Gabor wavelet at the radial center frequency w_0 and orientation θ. A multi-hierarchical Gabor representation of the facial image, $I(x,y)$, can be computed with a set of w_0 and θ. To reduce the computation complexity, the convolution is computed using the fast Fourier transform (FFT), point-by-point multiplications, and then the inverse fast Fourier transform (IFFT) on both the facial image and the Gabor wavelet. The magnitudes of the responses can be used as a measure of the local properties of an image because magnitude is slow varying to position, while the phases are very sensitive to location. The similarity between two Gabor wavelets, C and C' can be measured by the following similarity function [116],

$$S(C, C') = \frac{\sum_{i=1}^{N} a_i a'_j}{\sqrt{\sum_{i=1}^{N} a_i^2 \sum_{i=1}^{N} a_i'^2}}, \quad (4.3)$$

where a and a' represent the magnitudes of the respective Gabor wavelets representations, and N is the total number of Gabor representations. The number of Gabor representations is equal to the number of scales multiplied by the number of orientations. $S(C, C')$ is a correlation function with local optima forming large attractor basins. This leads to rapid and reliable convergence with simple search methods such as gradient descent or diffusion.

4.3 Temporally Maximum Occurrence Frame

Temporally Maximum Occurrence Frame (TMOF) [107] is a key frame representation scheme for video analysis and retrieval. This representation is constructed based on the most significant visual content in a video shot, and the idea is illustrated with six frames in a video sequence, as shown in Fig. 4.1. In the first three frames, a bus stops at the bottom left corner and a helicopter is landing on a house. In the last three frames, the bus drives away and the helicopter remains on the roof. Therefore, the key frame of this video sequence should contain the house, the bus, and the helicopter, as shown in Fig. in Fig. 4.1(g).

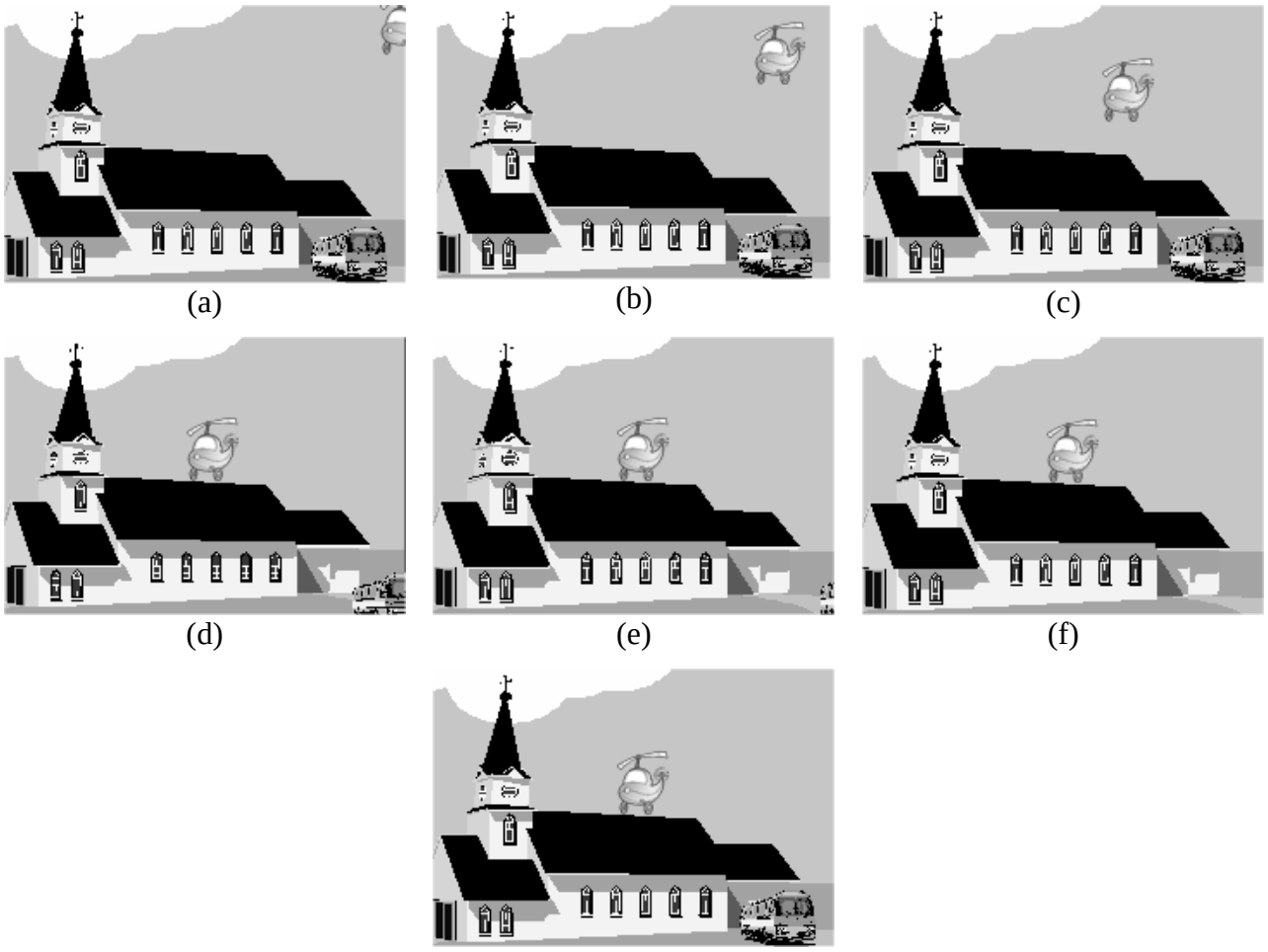


Fig. 4.1. An ideal representation frame for a sequence with six frames. (a) Frame 1. (b) Frame 2. (c) Frame 3. (d) Frame 4. (e) Frame 5. (f) Frame 6. (g) The Ideal Representation Frame. (Original image courtesy of K. W. Sze, K. M. Lam and G. Qiu)

The construction of a TMOF representation is based on the probabilities of occurrence of the pixel values at each pixel location for all the frames in the video sequence. The idea of constructing a key representation for a tracked region using the TMOF scheme is illustrated in Fig. 4.2.

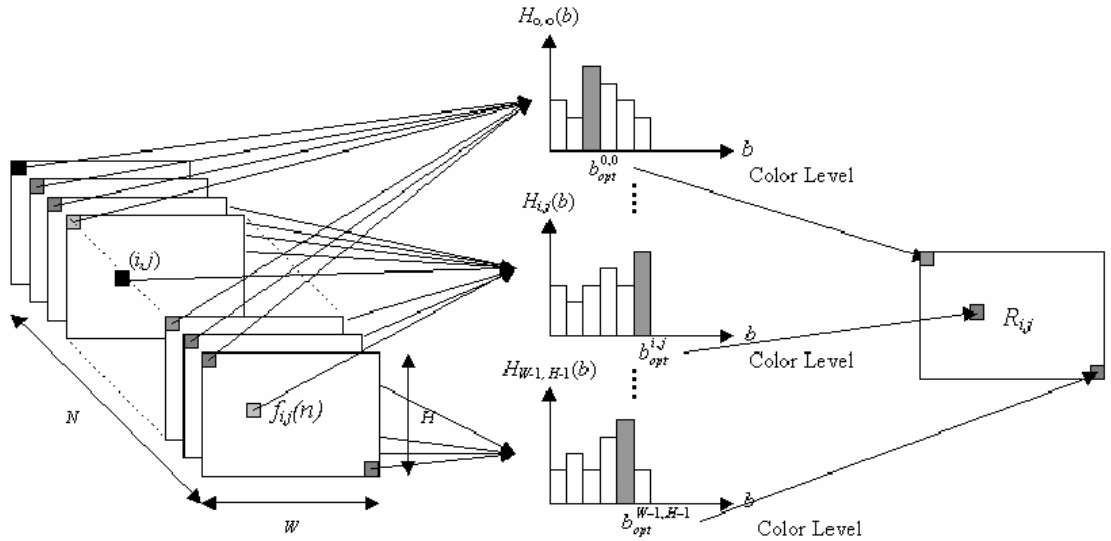


Fig. 4.2. The construction of the TMOF for a video sequence. (Original image courtesy of K. W. Sze, K. M. Lam and G. Qiu)

At each pixel location, a histogram is formed based on the respective pixel values. The number of histograms is equal to the image size. Each histogram accumulates the pixel values of the corresponding location in a video sequence. The histogram is then smoothed using a Gaussian function. At each pixel location of TMOF, the values to be selected are those with the highest probability of occurrence in the smoothed histogram. Therefore, the TMOF is computed as follows:

$$TMOF(i, j) = b_{opt}, \text{ for } 0 \le i \le W'\text{-}1 \text{ and } 0 \le j \le H'\text{-}1 \tag{4.4}$$

where $W' \times H'$ is the TMOF size, and b_{opt} is chosen as

$$b_{opt} = \arg\max_{b} \{H'_{i,j}(b)\}, \text{ for } 0 \le b \le B. \tag{4.5}$$

$H'_{i,j}$ is the smoothed histogram, which can be computed by convolving the histogram, $H_{i,j}$ with a Gaussian filter as follows:

$$H'_{i,j}(b) = H_{i,j}(b) * G(\sigma, b), \tag{4.6}$$

where $G(\sigma, b)$ and σ are a Gaussian function and its variance, respectively. Histogram, $H_{i,j}(b)$, is computed as follows:

$$H_{i,j}(b) = \sum_{n=0}^{N-1} \delta\left[f_n(i,j) - b\right], \text{ for } 0 \le b \le B, \tag{4.7}$$

and $\delta(m-n) = \begin{cases} 1 & \text{for } m = n, \\ 0 & \text{for } m \ne n. \end{cases}$, where histogram, $H_{i,j}$, is constructed by the corresponding pixels at each pixel location (i,j), $f_n(i,j)$ is the pixel value at location (i,j) in frame n, N is the number of frames in the video sequence, and B is the number of bins in the histogram. The number of bins in a histogram is the same as the number of intensity levels for a pixel. At the beginning of the tracking, the histogram cannot be distributed evenly; thus, the TMOF may bias towards a certain pixel value and may not be representative. In this case, the number of bins in a histogram needs to be reduced in order to obtain a better-estimated TMOF representation, and the tracking should rely less on the TMOF.

4.4 Face Tracking

For each image in a video sequence, the important facial features, i.e. the left eye I_{left_eye}, right eye I_{right_eye} and mouth I_{mouth}, are extracted. These facial features are tracked based on two schemes. The first scheme is to search for the possible position of these facial features based on feature windows of size 16×16, as shown in Fig. 4.3. The other scheme is based on the Gabor wavelet representations of the facial features.

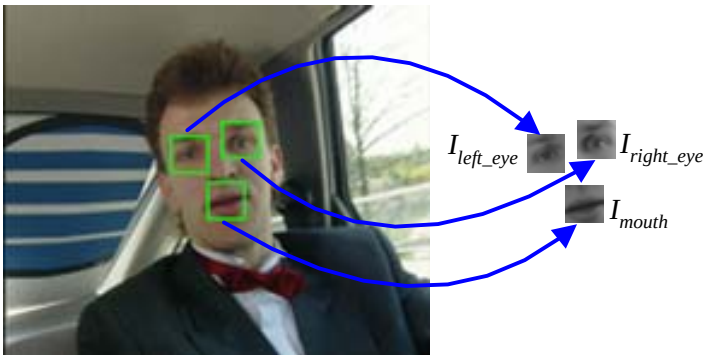


Fig. 4.3. Facial features extraction.

For the second scheme, the Gabor wavelet representations for the left eye, right eye, and mouth are computed and denoted as C_{left_eye}, C_{right_eye} and C_{mouth}, respectively. The location of the facial features in the first frame is manually selected in our experiment. Facial feature detection is assumed to have been done during the phase of face detection. Searching for the facial features in a current frame with reference to the locations in its previous frame is performed with the use of the greedy algorithm.

The face in the first frame is extracted and normalized, according to the locations of the facial features to form the first face template, $F_1(x,y)$. This first face template is used as the model for tracking in the first stage and is considered as the "long-term memory" in the following tracking process. During tracking, a TMOF template, $TMOF(x,y)$, is constructed with the number of N tracked face regions from the previous frames. Empirically, the maximum number of nearest tracked face regions used is $N = 30$ which can reflect the recent face appearance. In the first few frames, there is not enough information to create the pixel intensity histogram used for constructing the TMOF template. This updating process starts at frame 4. Since the TMOF template is updated according to the tracked face regions in a video sequence, it is considered as the "short-term memory."

The fast greedy algorithm in [48] is modified to compute the similarity of the respective facial features between a current and its previous frame. It is a fast iteration method to maximize the facial feature similarity. The computations required in this algorithm can be divided into the following main processes:

i. The best position of each of the three feature points is searched by considering the 3×3 neighborhoods of the feature points in the previous frame. This searching process is illustrated in Fig. 4.4, where $(x_{i,k}, y_{i,k})$ represents the

position of the *i*th facial feature (i = 1, 2, and 3) and k is the current frame number. With the three feature points, a triangular structure is formed.

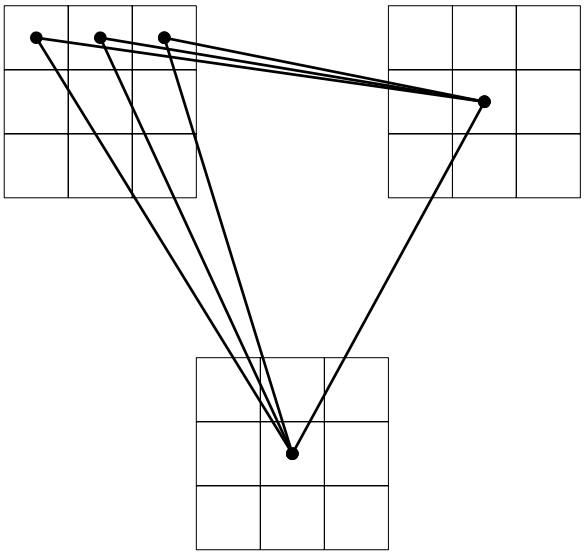


Fig. 4.4. The fast greedy algorithm searches the position of the three important facial features.

ii. For each possible location of the feature points, the similarity function (4.3) based on the Gabor representation is computed. The size of the search window is 3×3.

The triangular structure formed by the three feature points, as shown in Fig. 4.5, is used for tracking in the subsequent frames. The triangular structure should have limited changes between the successive frames. In our algorithm, the following three changes are limited between the triangular structures in two successive frames:

- percentage change of area,
- percentage change of angle change, and
- percentage change of the length of each side of the triangles.

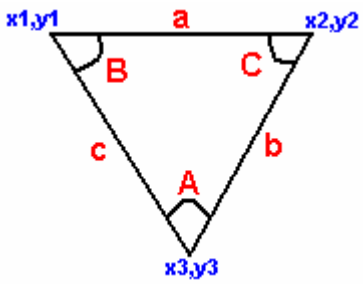


Fig. 4.5. A triangular structure.

The position of the feature points that results in the largest similarity based on the Gabor representation will be selected as the best location and used as the initial position for the next iteration. The iteration process continues until either the position of the facial feature points does not change in an iteration or the maximum number of iteration is reached.

During the search for the best facial feature points, the area formed by the facial feature points is extracted and normalized. This area forms a possible face, and is denoted as $pf(x,y)$. The template similarity distance used for tracking based on the first face template $F_1(x,y)$ and the TMOF template $TMOF(x,y)$ is given as follows:

$$T_d = \left| pf(x,y) - \lambda \cdot F_1 - (1-\lambda) \cdot TMOF(x,y) \right|, \tag{4.8}$$

where $|\cdot|$ represents the sum of squares in a template, and λ is a weighting factor used to control the relative importance of the first face template and the TMOF. For the first several frames, there is an insufficient number of frames to construct the TMOF template. Thus, only the first face template is considered, i.e. $\alpha = 1$ and $T_d = |pf(x,y) - F_1(x,y)|$. Finally, a tracked region is measured using an energy function, which combines the correlation based on the Gabor representations of feature points and the template distance measure as follows:

$$\begin{aligned} Energy = & -\alpha \cdot \left[S(C_{left\_eye}, C'_{left\_eye}) + S(C_{right\_eye}, C'_{right\_eye}) + S(C_{mouth}, C'_{mouth}) \right] / 3 \\ & + \beta \cdot T_d / T_{d,\max} + \gamma \cdot d / d_{\max} \end{aligned}, \tag{4.9}$$

where α, β and γ are weighting factors, where $\alpha+\beta+\gamma=1$, $T_{d,\max}$ is the maximum similarity distance in (4.8) used to normalize the second term in the range of 0~1, d is the distance of the estimated mouth point that drift away from the point perpendicular to the mid-point of both estimated eye locations, and $d_{\max}$ is the maximum value of d for normalization in the range of 0~1. The last term is used to force the position of the mouth's center to be close to the line that is perpendicular to and passes through the middle of the line formed by joining the two eye locations. The back-tracked region is the one that has the lowest energy value in the iterations. Once a face is tracked successfully, the tracked face region is also extracted and normalized so as to update the *TMOF* template.

4.5 Experimental Result

Our experiments were conducted based on the Carphone, Foreman and Salesman video sequence of the MPEG-1 content set. The number of frames in the three videos is 96, 300 and 200, respectively. For facial feature tracking, Gabor wavelet representation or intensity representation of each facial feature is used. Different combinations of the weighting factor in (4.9) are used to test the effectiveness of each component. For face template matching, two different methods either TMOF plus first frame template or first frame only are used. Five tests were performed for each video sequence. Both the Carphone and Foreman videos are very challenging, as the head moves much more vigorously than in the Salesman video. The speed of our tracking algorithm is approximately 4 frames per second on a Pentium 4 1.7GHz computer. Most of the time was spent on computing the Gabor representations for the possible face regions.

Experiment	Representation in facial feature	α	β	γ	Face template matching method
1	Gabor wavelet	0.0	1.0	0.0	TMOF + First Frame
2	Gabor wavelet	0.0	1.0	0.0	First Frame only
3	Gabor wavelet	0.2	0.6	0.2	TMOF + First Frame
4	Gabor wavelet	0.2	0.6	0.2	First Frame only
5	Intensity	0.2	0.6	0.2	TMOF + First Frame

Table 4.1. List of parameters in each face tracking experiment.

In the Carphone video sequence, a moving head with blinking eyes and a talking mouth causes sudden changes in orientation for the frames 23-28, as illustrated in Figs. 4.6(c)-(d), and for frames 28-38, as illustrated in Figs. 4.6(d)-(e). The head moves back and forth from frames 53 to 62, as shown in Fig 4.6(e)-(i). The lighting conditions on the face change between frames 53 and 70, as shown in Figs. 4.6(g)-(h). In addition, the size of the eyes and mouth changes frequently in the video sequence, as shown in Figs. 4.6(d)-(i).

In Fig. 4.6, each of the figures also contains one or two faces at the upper left corner. The upper one is the normalized face being tracked. The lower one is the face template generated using TMOF. As discussed before, the TMOF template (the lower one) used in the next frame is updated by using the tracked face (the upper one) in the previous frame. In experiment 1, 3 and 5, the TMOF was created after face had been tracked successfully in frame 3. Thus, at frame 4, a face is tracked using a combination of the first face template and the TMOF template. All the templates and the tracked faces are normalized to a size of 16×16 pixels. It can be observed that the TMOF template in Fig. 4.6(b) has different face appearance as compared to Fig. 4.6(i). This is because the TMOF representation is based on the peak pixel values over the recent 30 frames, which reflects the recent appearance change. This is especially obvious in Foreman sequence as the perspective change is reflected in the TMOF template in Fig 4.16.

In experiment 1, both TMOF and the first frame are considered for face template matching with an average error of 3.38 pixels, and in experiment 2, only the first frame is considered with average an error of 3.67 pixels. By using different face template matching methods, both experiments 1 and 2 show that the mouth point drifts away from the mouth center frequently, as shown in Fig. 4.7 and Fig. 4.9, and the average errors of the mouth point are 4.66 pixels and 3.56 pixels, respectively.

In experiment 3, the performance is improved the average error is reduced to 2.35 pixels compared to both experiments 1 and 2. In particular, the average error of the mouth is reduced by half to 1.78 pixels. The errors of these three facial features, as shown in Fig. 4.11, are varied less when compared to those in experiments 1 and 2. Compared to experiment 4, the average error is 3.00 pixels. In this experiment, the same set of weighting factor is used but the first frame is used in the face template matching. We can observe that experiment 3 shows a better performance level.

Experiment 5 results in the highest average error, which is 4.36 pixels. In this testing case, only intensity is considered in the measurement of the facial feature similarity distance between the previous and current frames. The left eye is seriously drifted away from the actual location, as shown in Fig. 4.15, and the average error is 7.28 pixels.

In each of the experiments, a table is also given to show the error distance between the estimated and actual facial feature location. Table 4.2 tabulates the average error and standard deviation of the experiments.

	Experiment	Left Eye	Right Eye	Mouth	Average Error
Average Error	1	3.11	2.38	4.66	3.38
Standard Deviation		2.02	1.69	3.22	
Average Error	2	2.26	2.19	3.56	2.67
Standard Deviation		1.37	1.32	2.47	
Average Error	3	2.46	2.83	1.78	2.35
Standard Deviation		1.65	1.50	1.19	
Average Error	4	3.27	3.68	2.04	3.00
Standard Deviation		2.08	2.26	1.66	
Average Error	5	7.28	2.49	3.31	4.36
Standard Deviation		4.13	1.41	2.46	

Table 4.2. Experiment results of Carphone video.

Fig. 4.6. Tracking results using the Carphone video in experiment 1. (a) Frame 0. (b) Frame 12. (c) Frame 23. (d) Frame 28. (e) Frame 38. (f) Frame 40. (g) Frame 45. (h) Frame 62. (i) Frame 70.

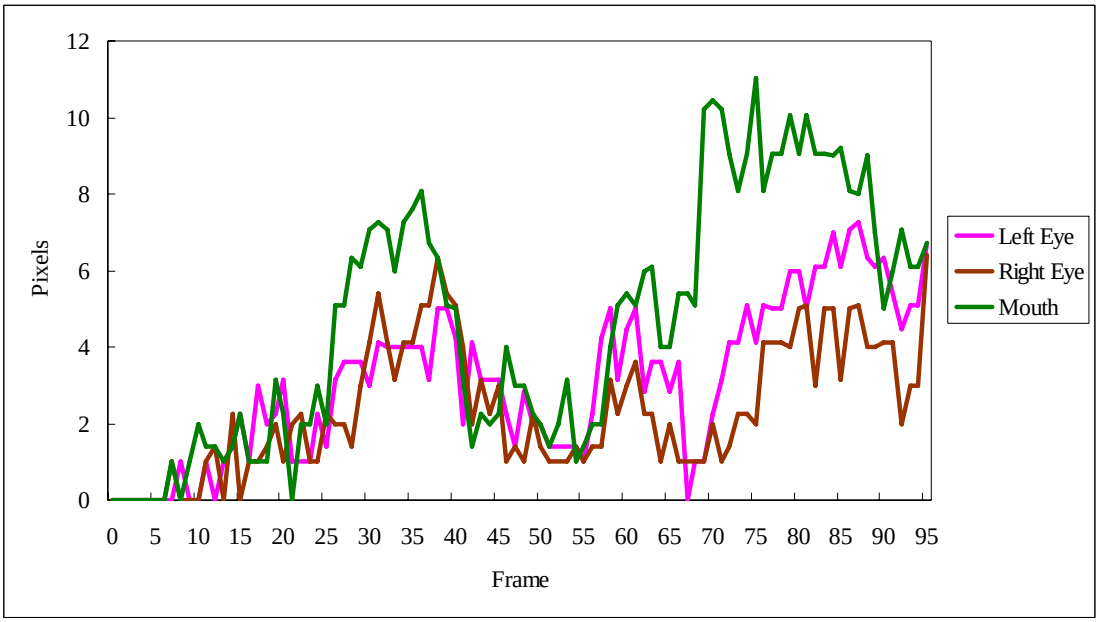


Fig. 4.7. Error distance between estimated and actual position of the facial features in Carphone video in experiment 1.

Fig. 4.8. Tracking results using the Carphone video in experiment 2. (a) Frame 0. (b) Frame 12. (c) Frame 23. (d) Frame 28. (e) Frame 38. (f) Frame 40. (g) Frame 45. (h) Frame 62. (i) Frame 70.

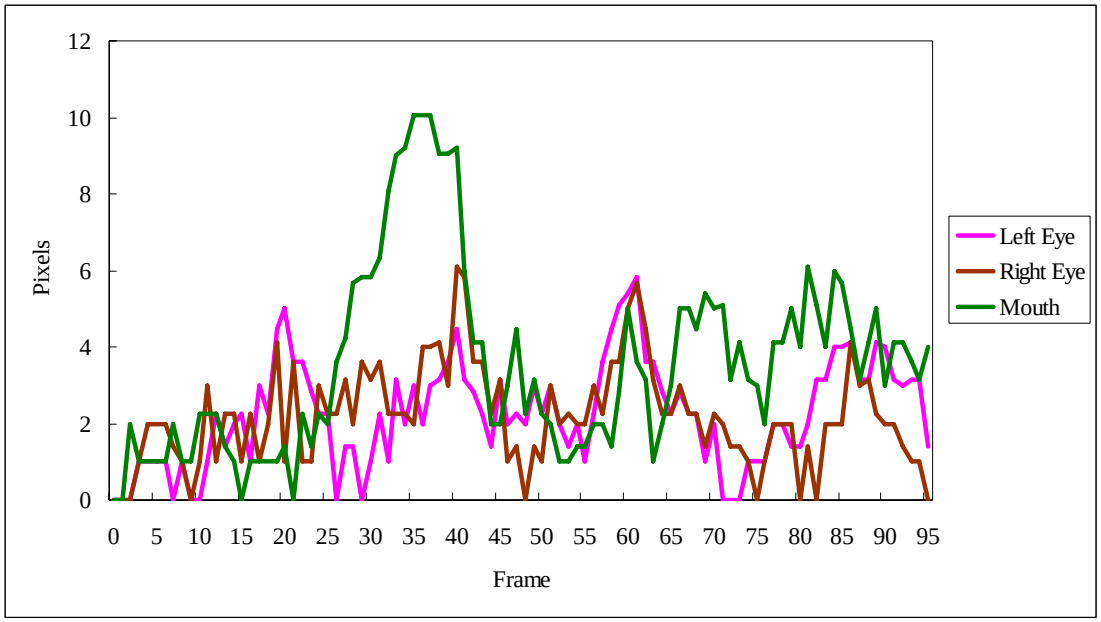


Fig. 4.9. Error distance between estimated and actual position of the facial features in Carphone video in experiment 2.

Fig. 4.10. Tracking results using the Carphone video in experiment 3. (a) Frame 0. (b) Frame 12. (c) Frame 23. (d) Frame 28. (e) Frame 38. (f) Frame 40. (g) Frame 45. (h) Frame 62. (i) Frame 70.

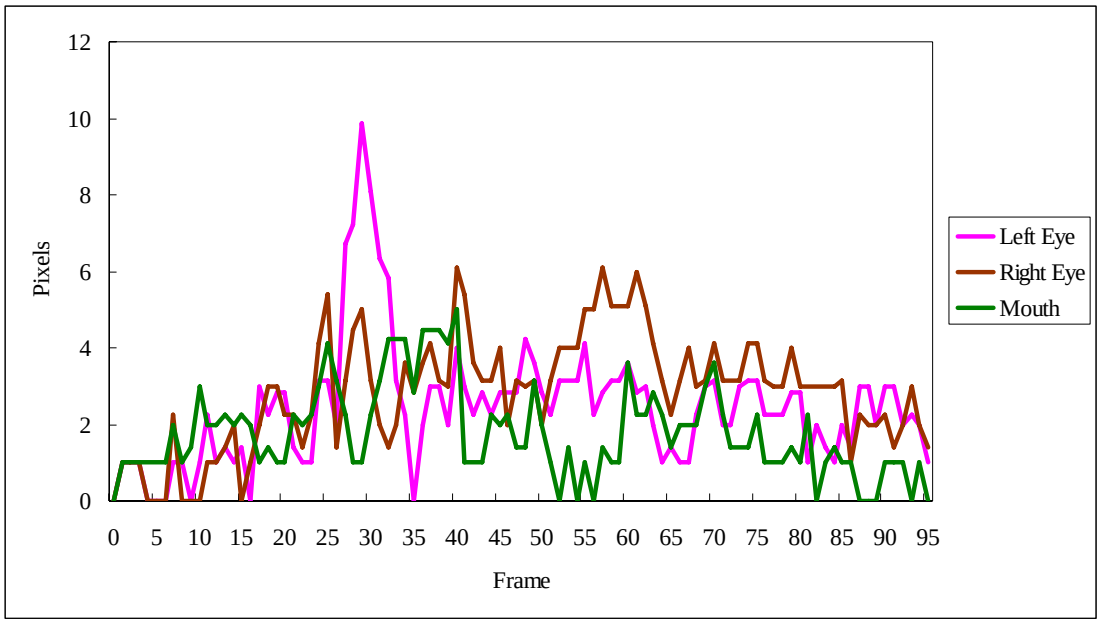


Fig. 4.11. Error distance between estimated and actual position of the facial features in Carphone video in experiment 3.

Fig. 4.12. Tracking results using the Carphone video in experiment 4. (a) Frame 0. (b) Frame 12. (c) Frame 23. (d) Frame 28. (e) Frame 38. (f) Frame 40. (g) Frame 45. (h) Frame 62. (i) Frame 70.

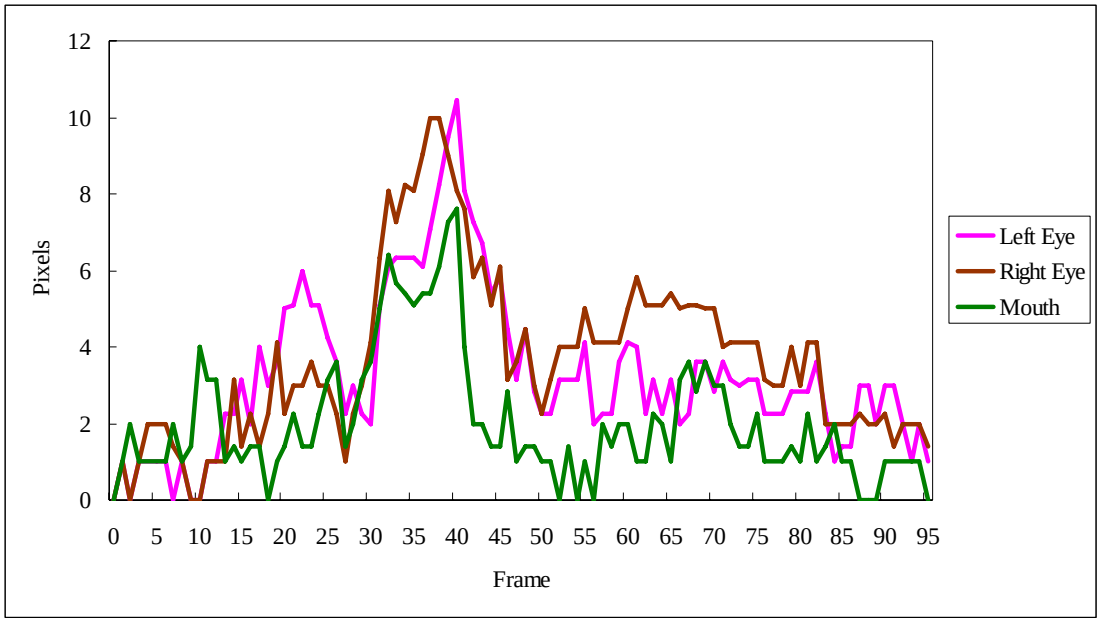


Fig. 4.13. Error distance between estimated and actual position of the facial features in Carphone video in experiment 4.

Fig. 4.14. Tracking results using the Carphone video in experiment 5. (a) Frame 0. (b) Frame 12. (c) Frame 23. (d) Frame 28. (e) Frame 38. (f) Frame 40. (g) Frame 45. (h) Frame 62. (i) Frame 70.

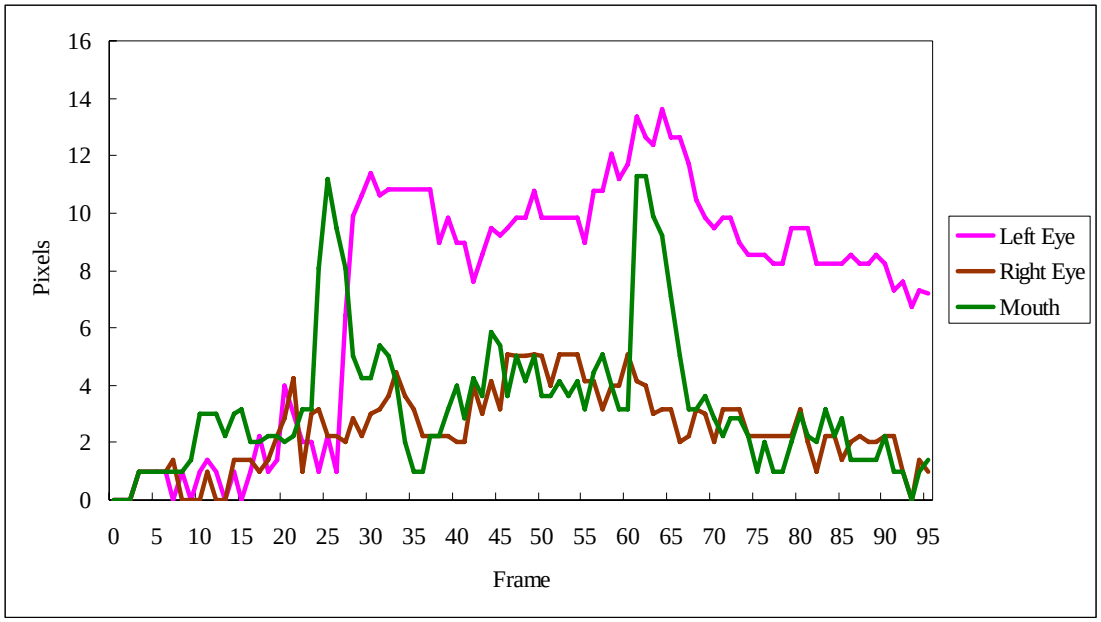


Fig. 4.15. Error distance between estimated and actual position of the facial features in Carphone video in experiment 5.

In Foreman video sequence, there are 185 out of 300 frames contain human face. The moving head in this video sequence contains changes in orientation, as shown in Fig. 4.16(g)-(h), loss of right eye for the frames 156-169, as shown in Figs 4.16 (i), and changes in perspective view in the whole sequence, as shown in Figs. 4.16 (a)-(i). The rapid change in perspective view from Figs. 4.16(d)-(e) causes the facial features to drift away from the facial features. However the position of the facial feature can be relocated to the feature center, as illustrated in Fig. 4.16(f), with the help of the face templates. Because of fast head motion in frames 130-155, the tracking of mouth was failed. For the frames 156-169, because of a loss of information at the right eye, as shown in Fig. 4.16(i), the tracking cue failed to follow the right-eye position in the subsequent frames. Starting from frame 170, the camera pans to right and head quickly moves out from the scene. This action introduces a large error between estimated and actual facial features position as shown in Fig. 4.17.

Table 4.3 tabulates the average error and standard deviation of the experiments. Starting from frame 156, the information about the right eye is lost due to fast camera motion. Thus, this video is divided into two result sets; one is for frames 0-184 and the other one is for frames 0-155. The performance of our algorithm is greatly improved without taking into account of the information lost and camera effect.

In experiment 1 with an average error of 4.14 pixels, both TMOF and the first frame are considered for face template matching, and in experiment 2, with average error of 4.97 pixels, only the first frame is considered. By using different face template matching methods, both experiments 1 and 2 show that the mouth point drifts away from the mouth center seriously when fast motion occurred, as shown in

Fig. 4.16(h) and Fig. 4.18(h). The corresponding average errors of the mouth point are 5.05 and 5.91, respectively.

In experiment 4, the experimental result is improved with an average error of 4.14 pixels. Compared to both experiments 1 and 2, the average error of the mouth point is reduced to 3.85 pixels. The errors in all the three facial features, as shown in Fig 4.23, vary less when compared to experiment 1. Compared to experiment 3, the average error is 4.15 pixels, which uses the same set of weighting factor but both TMOF and the first frame are employed in the face template matching. The results are very close to each other. The result of experiment 4 is slightly outperform those in experiment 3.

Experiment 5 has the highest average error of 7.01 pixels, in all the experiments. In this testing case, only intensity is considered in the measurement of the facial feature similarity distance between the previous and current frames. The right eye and mouth point are seriously drifted away from the actual locations, as shown in Fig. 4.25, and the average errors are 8.11 pixels and 9.58 pixels, respectively.

	Experiment	Left Eye	Right Eye	Mouth	Average Error
Average Error	1	8.05	6.37	8.27	7.56
Standard Deviation		13.66	8.28	9.16	
Average Error (frame 0-155)		3.41	3.95	5.05	4.14
Standard Deviation (frame 0-155)		3.48	2.59	3.72	
Average Error	2	9.65	6.27	8.54	8.15
Standard Deviation		14.78	7.08	8.09	
Average Error (frame 0-155)		4.79	4.22	5.91	4.97
Standard Deviation (frame 0-155)		5.42	2.47	4.02	
Average Error	3	5.86	6.66	5.03	5.85
Standard Deviation		8.30	6.59	3.90	
Average Error (frame 0-155)		3.59	4.96	3.90	4.15
Standard Deviation (frame 0-155)		2.32	2.77	2.72	
Average Error	4	5.78	6.54	5.07	5.79
Standard Deviation		8.40	6.83	4.39	
Average Error (frame 0-155)		3.70	4.88	3.85	4.14
Standard Deviation (frame 0-155)		3.05	3.39	2.92	
Average Error	5	4.97	8.87	10.43	8.09
Standard Deviation		5.18	6.50	7.59	
Average Error (frame 0-155)		3.33	8.11	9.58	7.01
Standard Deviation (frame 0-155)		2.70	5.93	7.86	

Table 4.3. Experiment results of Foreman video.

Fig. 4.16. Tracking results using the Foreman video in experiment 1. (a) Frame 0. (b) Frame 8. (c) Frame 20. (d) Frame 60. (e) Frame 90. (f) Frame 120. (g) Frame 131. (h) Frame 151. (i) Frame 158.

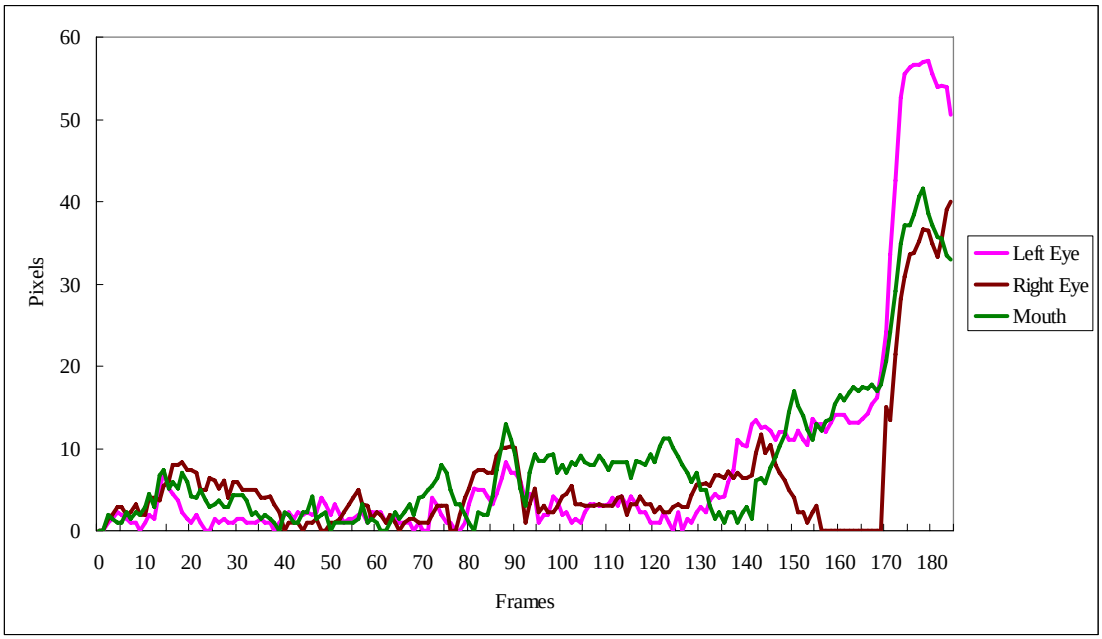


Fig. 4.17. Error distance between estimated and actual position of the facial features in Foreman video sequence in experiment 1.

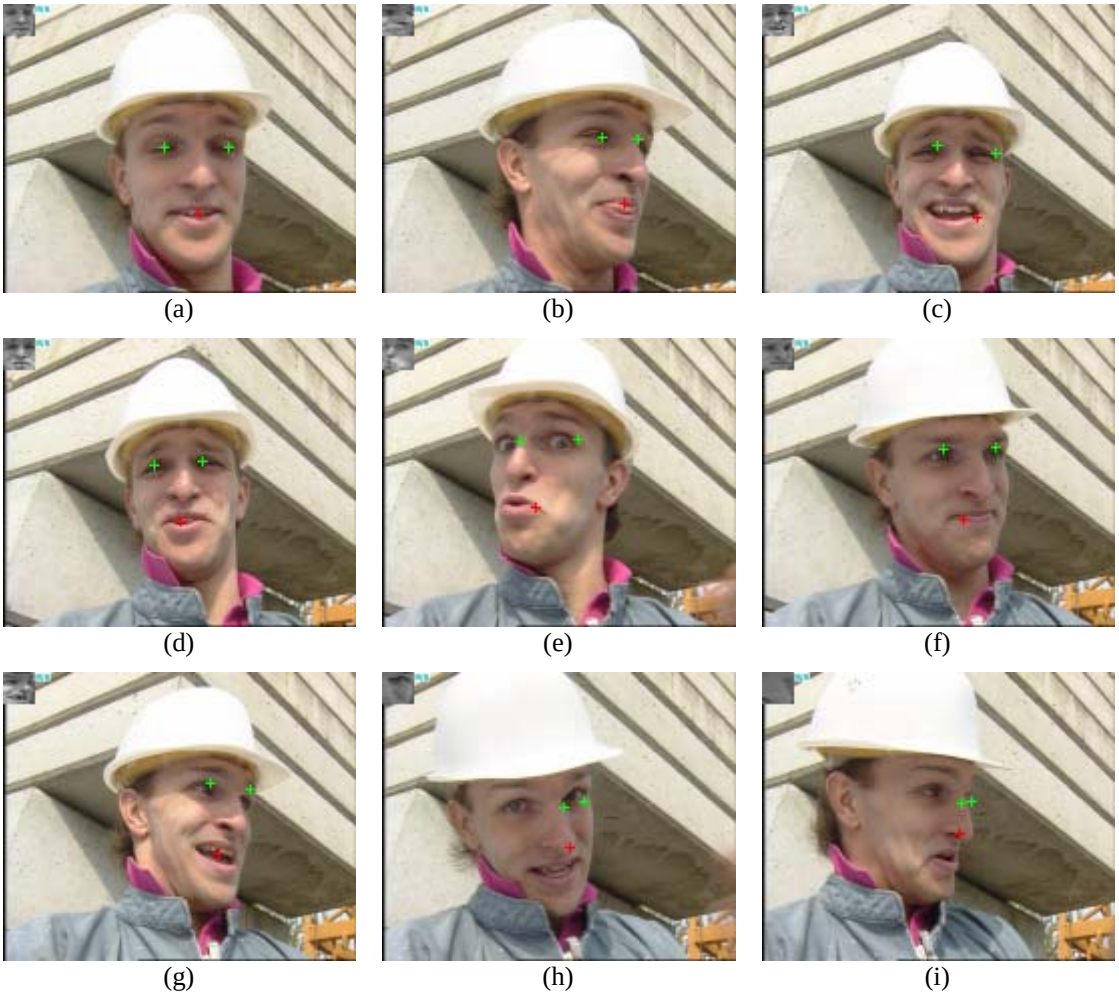


Fig. 4.18. Tracking results using the Foreman video in experiment 2. (a) Frame 0. (b) Frame 8. (c) Frame 20. (d) Frame 60. (e) Frame 90. (f) Frame 120. (g) Frame 131. (h) Frame 151. (i) Frame 158.

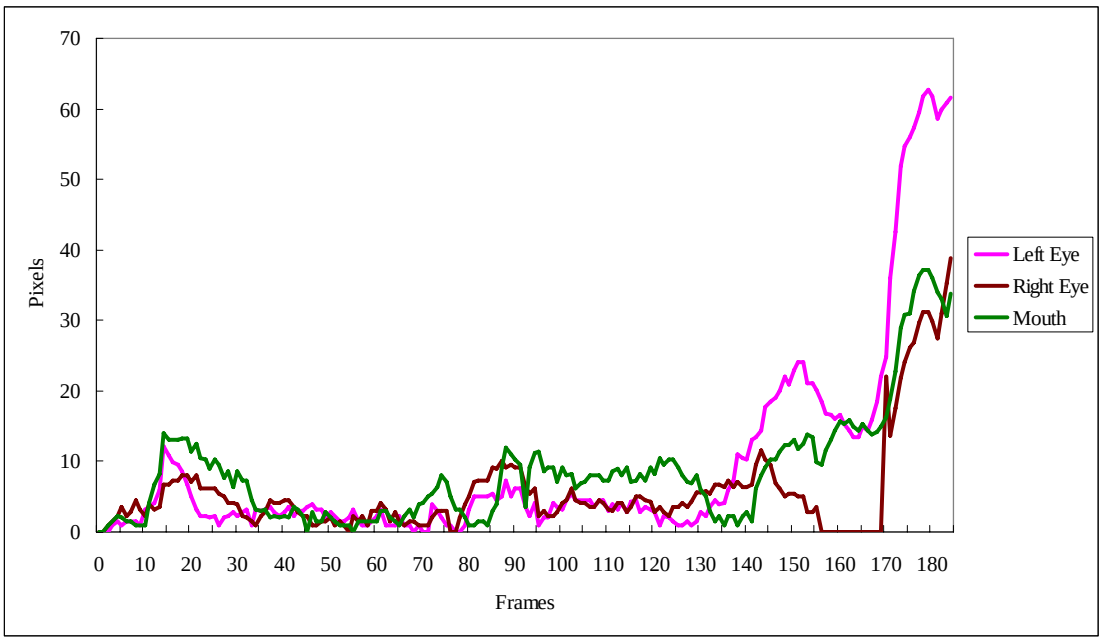


Fig. 4.19. Error distance between estimated and actual position of the facial features in Foreman video sequence in experiment 2.

Fig. 4.20. Tracking results using the Foreman video in experiment 3. (a) Frame 0. (b) Frame 8. (c) Frame 20. (d) Frame 60. (e) Frame 90. (f) Frame 120. (g) Frame 131. (h) Frame 151. (i) Frame 158.

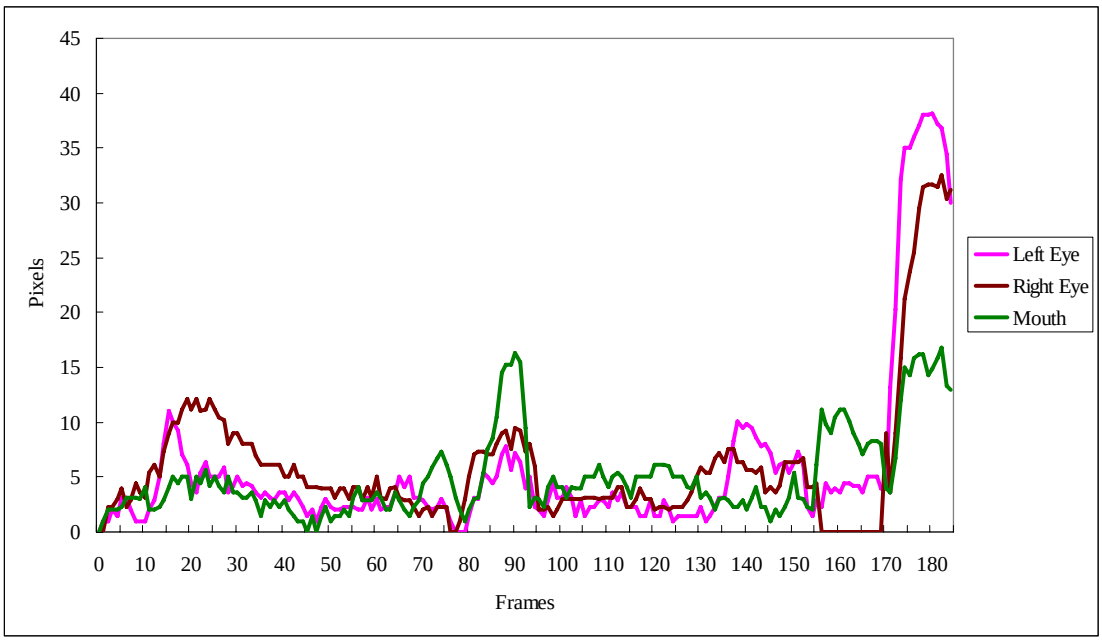


Fig. 4.21. Error distance between estimated and actual position of the facial features in Foreman video sequence in experiment 3.

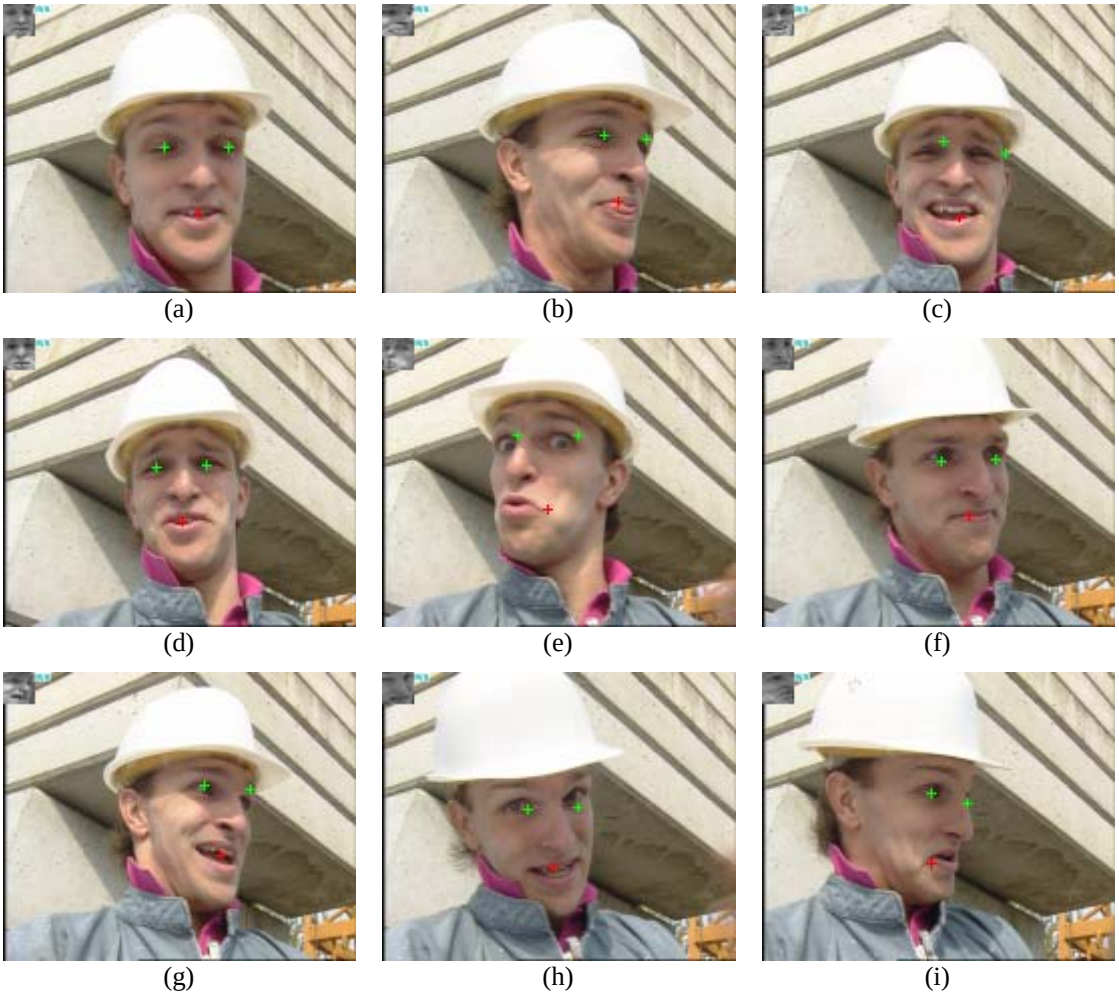


Fig. 4.22. Tracking results using the Foreman video in experiment 4. (a) Frame 0. (b) Frame 8. (c) Frame 20. (d) Frame 60. (e) Frame 90. (f) Frame 120. (g) Frame 131. (h) Frame 151. (i) Frame 158.

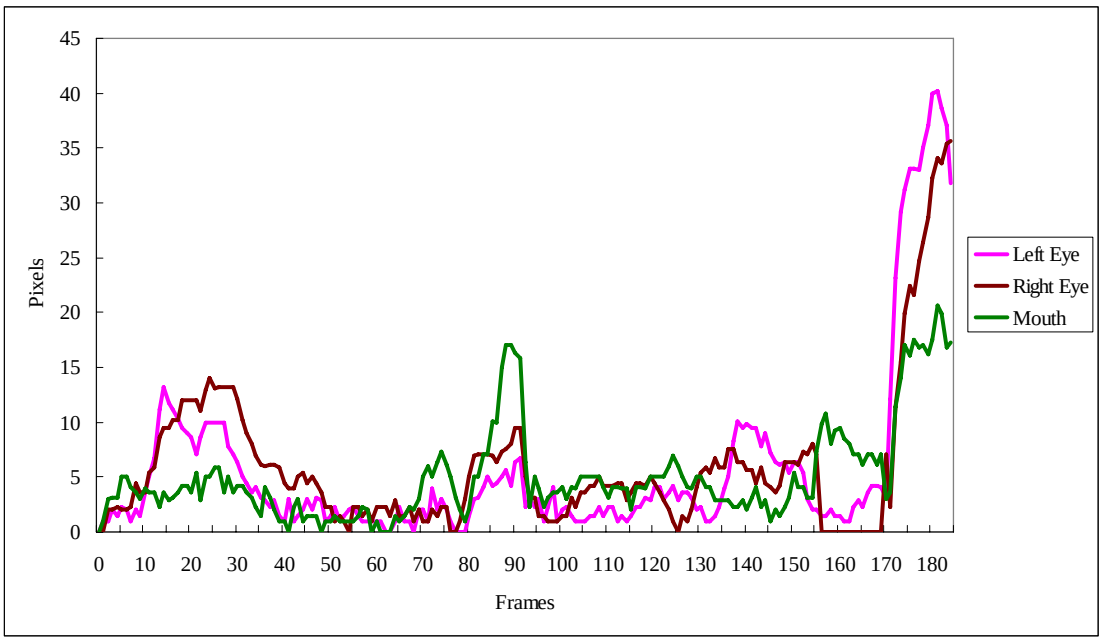


Fig. 4.23. Error distance between estimated and actual position of the facial features in Foreman video sequence in experiment 4.

Fig. 4.24. Tracking results using the Foreman video in experiment 5. (a) Frame 0. (b) Frame 8. (c) Frame 20. (d) Frame 60. (e) Frame 90. (f) Frame 120. (g) Frame 131. (h) Frame 151. (i) Frame 158.

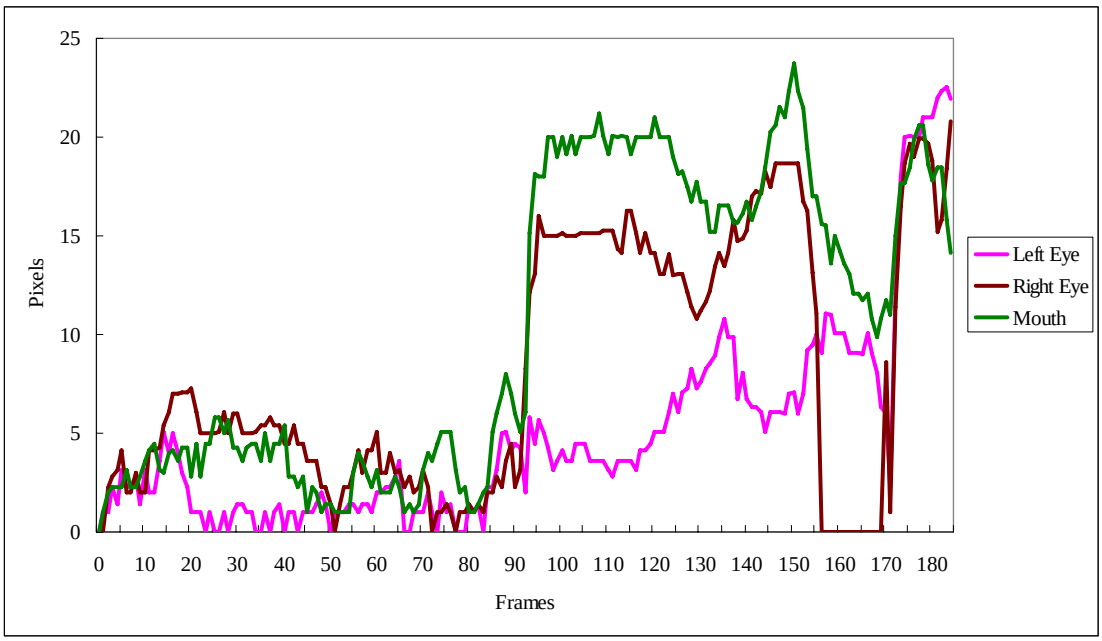


Fig. 4.25. Error distance between estimated and actual position of the facial features in Foreman video sequence in experiment 5.

When compared to the Carphone and Foreman video sequences, head movement in Salesman video sequence is less dynamic and more stable. The tracking cues track the facial features successfully in all test cases.

Experiment 2, which uses the first face as a face template for matching during tracking, has the lowest average error of 1.03 pixels, and the averages for left eye, right eye and mouth are 0.83 pixels, 1.28 pixels and 1.06 pixels, respectively. Experiment 1, which uses TMOF and the first face in face template matching during tracking, achieve a satisfactory performance with an average error of 1.04 pixels when compared to experiment 2.

As this video sequence is more static compared to the "Carphone" and "Foreman" video sequences, experiments 3 and 4 have a satisfactory performance level with average errors of 1.11 pixels and 1.17 pixels, respectively, but with relatively higher average errors when compared to experiments 1 and 2.

Experiment 5 has the highest average error in all the experiments. Since the video sequence is more static than the previous two sequences, the average error does not have a large variation when compared to experiments 1, 2, 3 and 4. It has a satisfactory result with an average error of 1.33 pixels.

Experimental results are tabulated in Table 4.4 and charts are shown in Fig. 4.26-35.

	Experiment	**Left Eye**	**Right Eye**	**Mouth**	**Average Error**
Average Error	1	0.88	1.39	0.84	1.04
Standard Deviation		0.67	0.69	0.66	
Average Error	2	0.83	1.28	1.06	1.03
Standard Deviation		0.66	0.67	0.85	
Average Error	3	0.65	1.61	1.07	1.11
Standard Deviation		0.54	0.83	0.79	
Average Error	4	0.77	1.56	1.20	1.17
Standard Deviation		0.65	0.76	0.87	
Average Error	5	1.08	1.27	1.64	1.33
Standard Deviation		0.76	0.74	0.87	

Table 4.4. Experiment results of Salesman video.

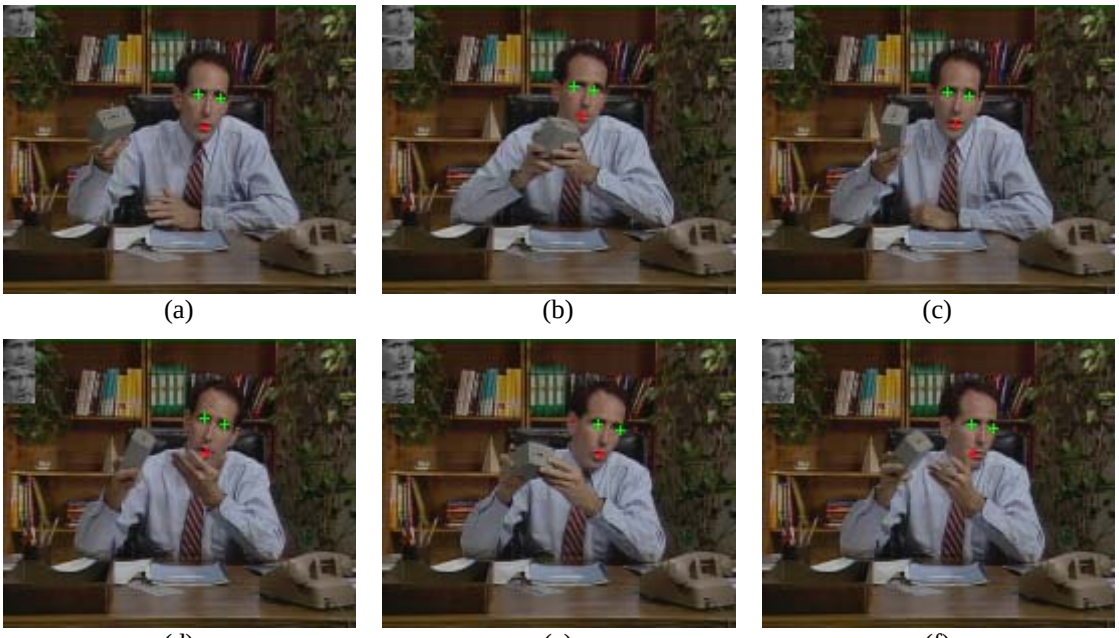


Fig. 4.26. Tracking results using the Salesman video in experiment 1. (a) Frame 0. (b) Frame 50. (c) Frame 75. (d) Frame 120. (e) Frame 140. (f) Frame 199.

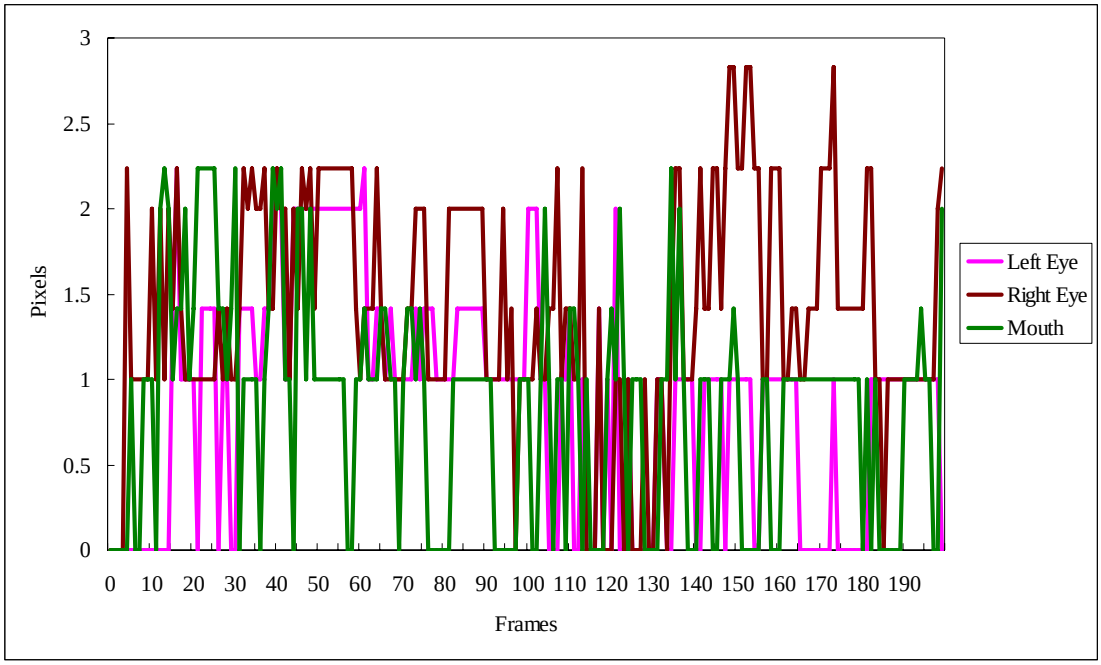


Fig. 4.27. Error distance between estimated and actual position of the facial features in Salesman video in experiment 1.

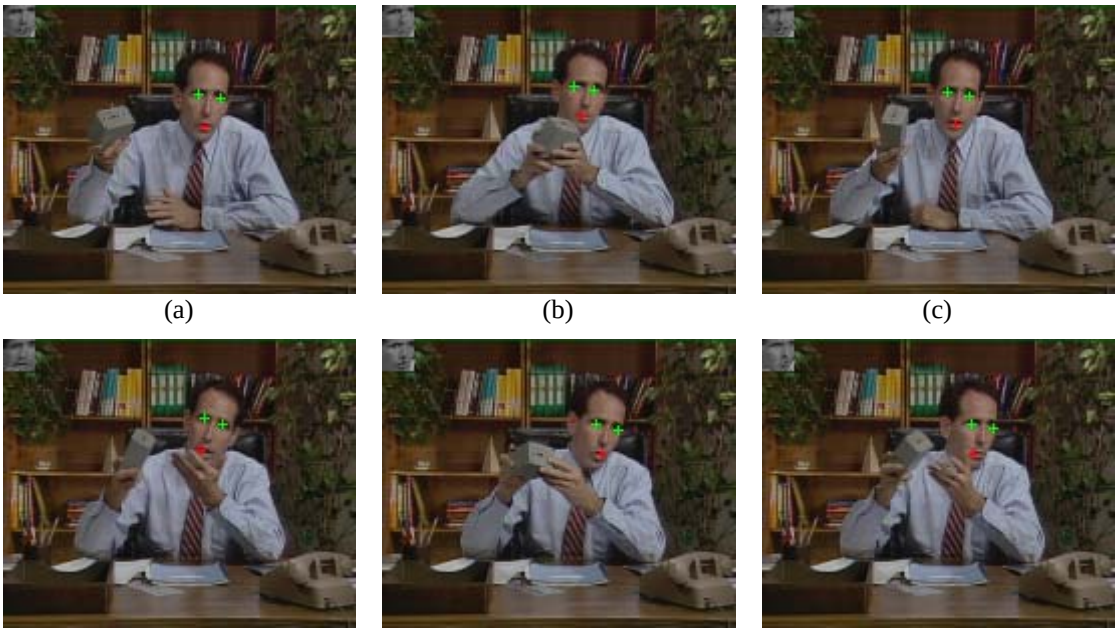


Fig. 4.28. Tracking results using the Salesman video in experiment 2. (a) Frame 0. (b) Frame 50. (c) Frame 75. (d) Frame 120. (e) Frame 140. (f) Frame 199.

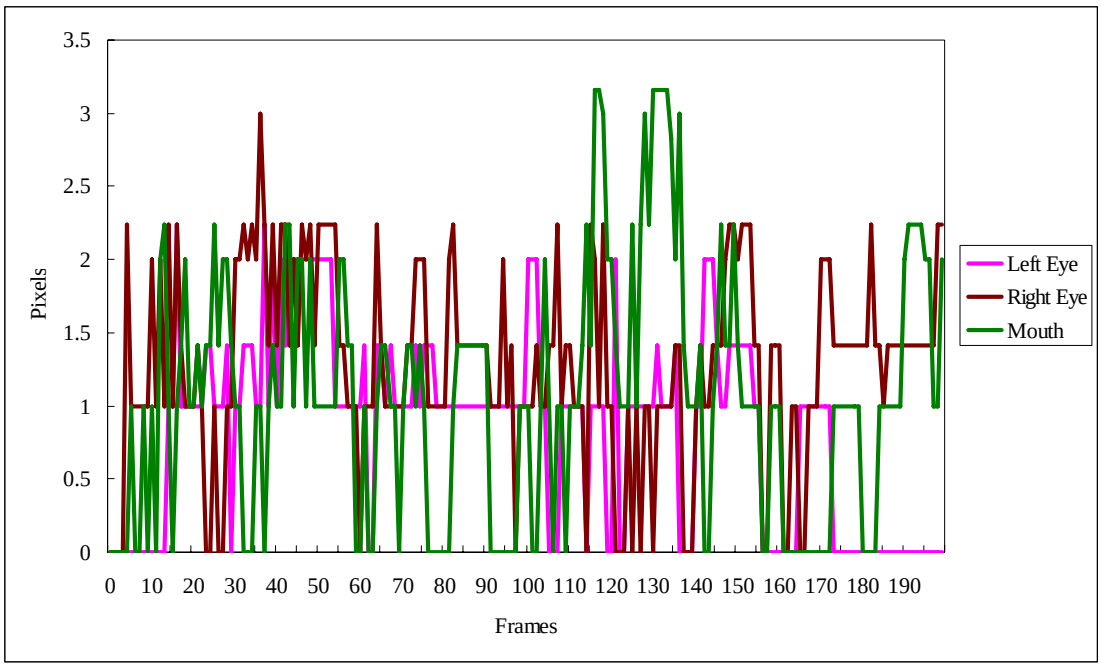


Fig. 4.29. Error distance between estimated and actual position of the facial features in Salesman video in experiment 2.

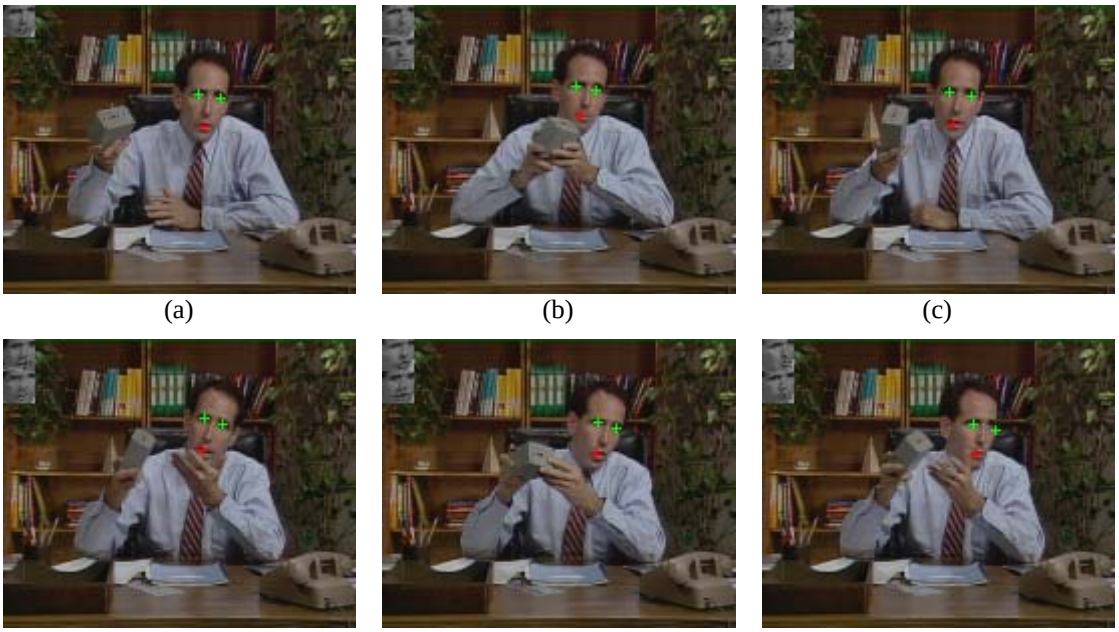


Fig. 4.30. Tracking results using the Salesman video in experiment 3. (a) Frame 0. (b) Frame 50. (c) Frame 75. (d) Frame 120. (e) Frame 140. (f) Frame 199.

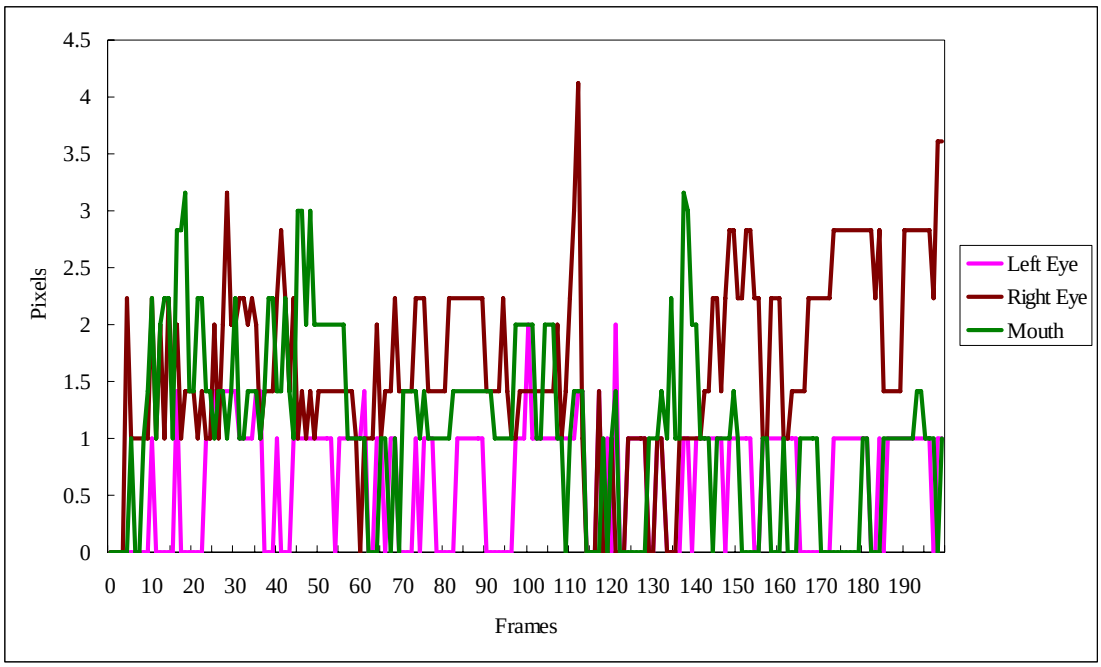


Fig. 4.31. Error distance between estimated and actual position of the facial features in Salesman video in experiment 3.

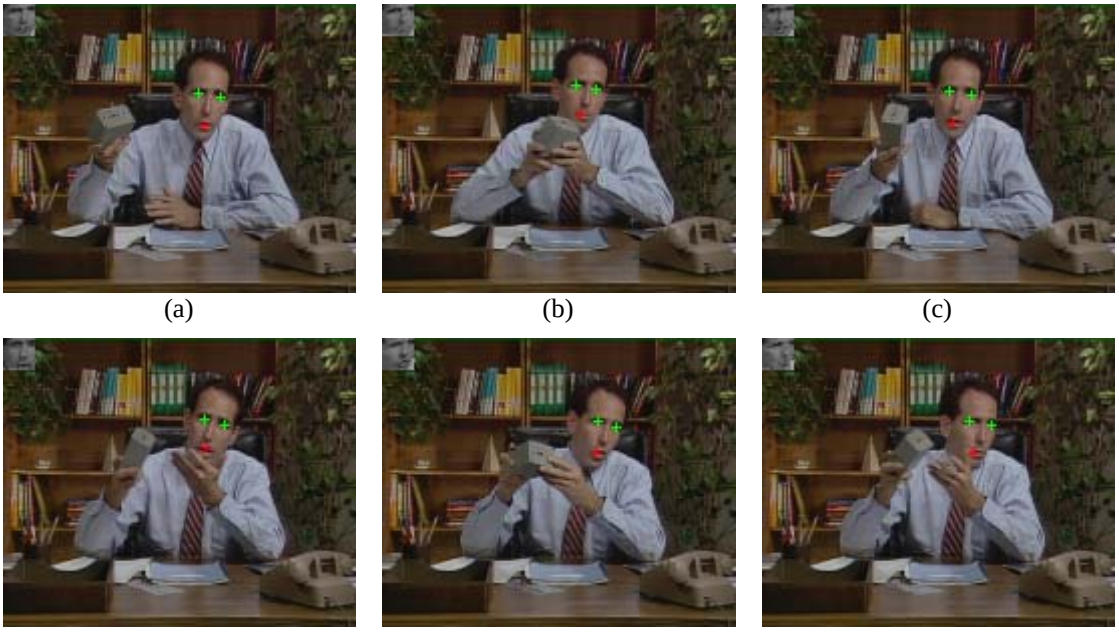


Fig. 4.32. Tracking results using the Salesman video in experiment 4. (a) Frame 0. (b) Frame 50. (c) Frame 75. (d) Frame 120. (e) Frame 140. (f) Frame 199.

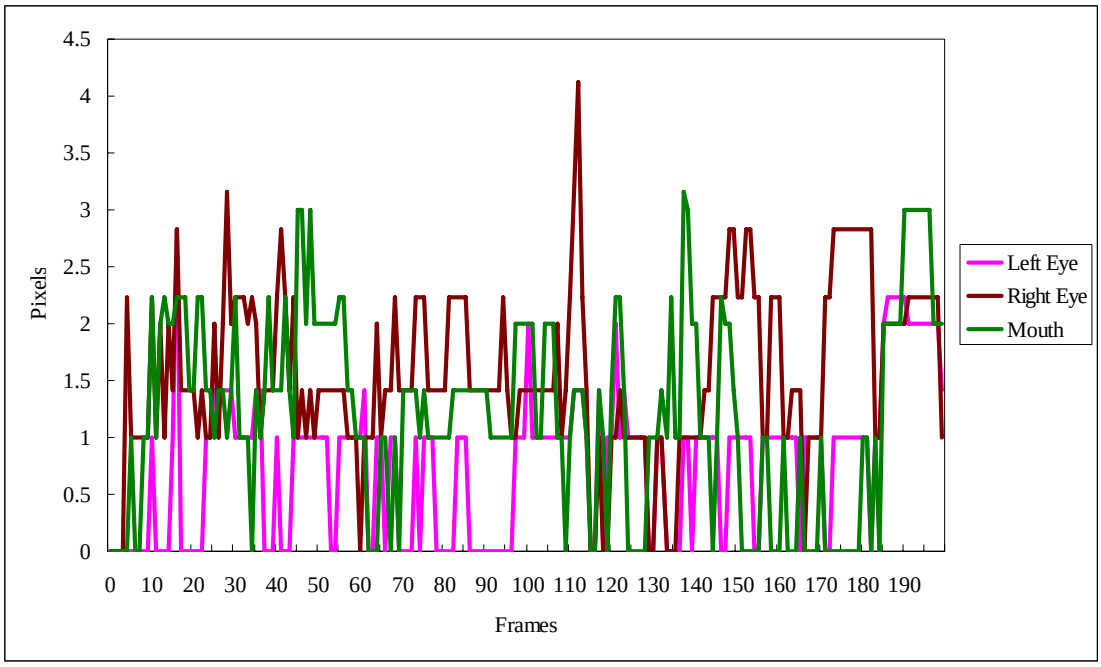


Fig. 4.33. Error distance between estimated and actual position of the facial features in Salesman video in experiment 4.

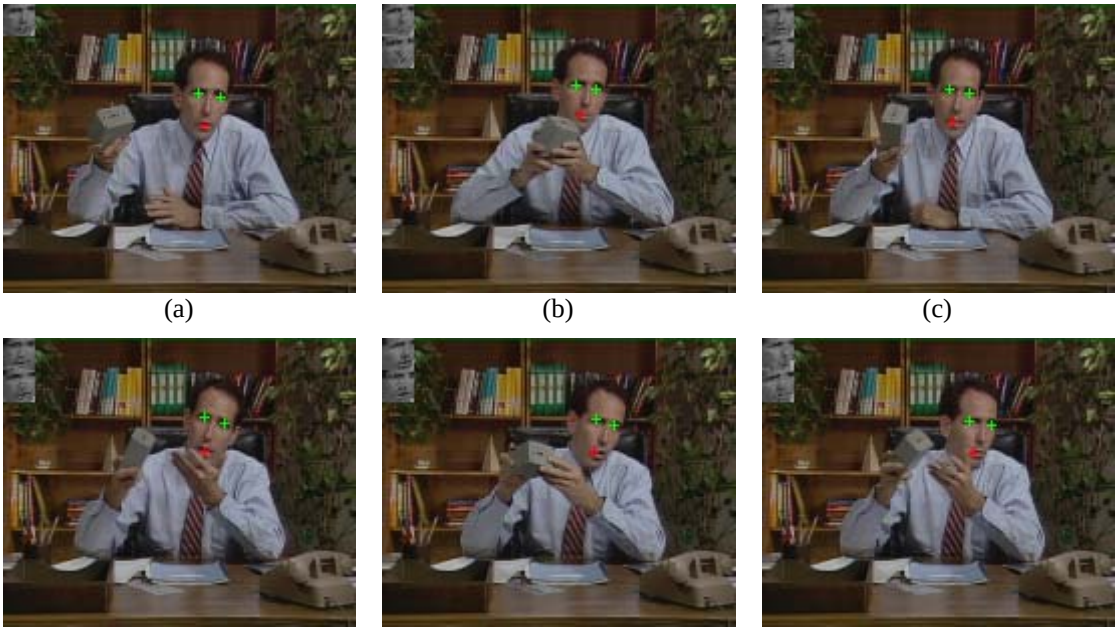


Fig. 4.34. Tracking results using the Salesman video in experiment 5. (a) Frame 0. (b) Frame 50. (c) Frame 75. (d) Frame 120. (e) Frame 140. (f) Frame 199.

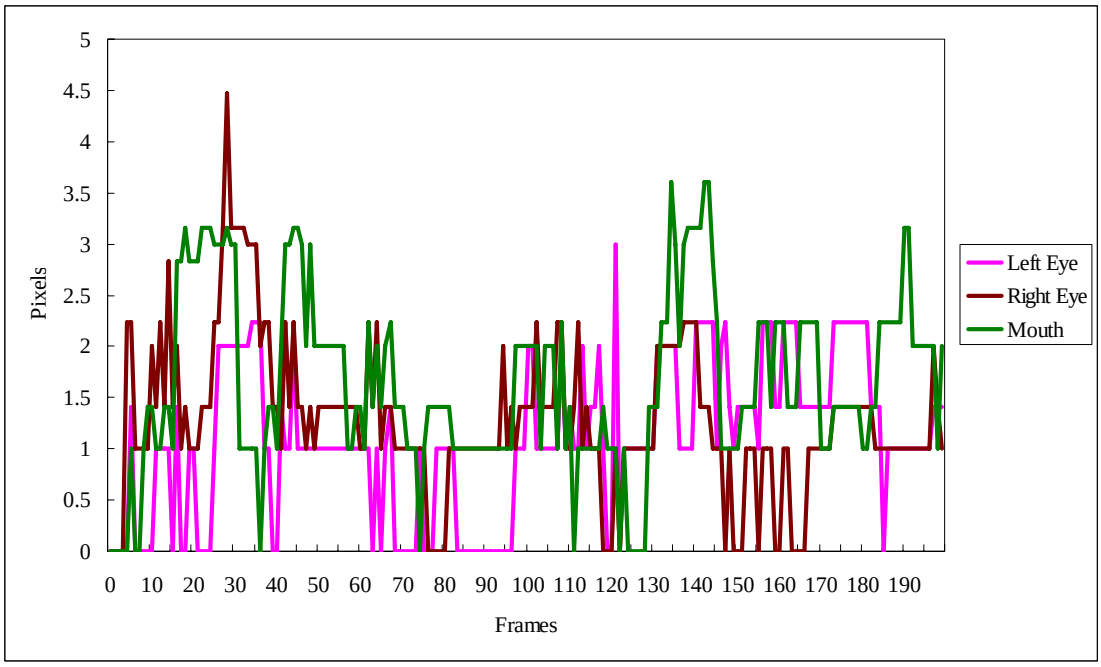


Fig. 4.35. Error distance between estimated and actual position of the facial features in Salesman video in experiment 5.

4.5 Conclusion

In this chapter, we have purposed a face tracking method based on a combination of the Gabor wavelet representations of the important facial features and the TMOF face template. The TMOF face template adapts to changes in face appearance in a video sequence. A modified greedy algorithm is proposed to search for the facial feature locations that are first verified based on the properties of the triangular structure and the Gabor representation similarity. Finally, the possible face regions are further verified using the TMOF face template and first face template. Using the parameter sets 3 and 4 listed in Table 4.1, experimental results show that our algorithm is robust to the rapid head movement, orientation change and illumination change. However, in the video sequence "Salesman", the parameter sets 1 and 2 can achieve better performance level in less dynamic video sequences. An adaptive method should be used to measure the activity in a video sequence.

During face tracking, the pose estimation of human face with ASM [50] should be performed concurrently. The CANDIDE model [5] can also adapt to the human face structure for 3D reconstruction which described in next chapter.

Chapter 5
Construction of 3D Face Structure

5.1 Introduction

The generation of a realistic-looking and animated human face model is a challenging and difficult problem, and has been a computer vision research topic for over 25 years. The construction of face model is necessary for computer games, virtual presence, video conferencing, online chat, model-based video coding etc. Nowadays, laser scanners are the most popular tools to acquire 3D structures. However, these scanners are very expensive. In addition, the data are usually noisy and require post-processing, such as hand-touch-up and manual registration before animating the 3D model. Computers and digital cameras are relative powerful and inexpensive now; thus, the construction of a 3D face model from multiple 2D images attracts a lot of interest. The techniques available nowadays are still manually intensive and computationally expensive.

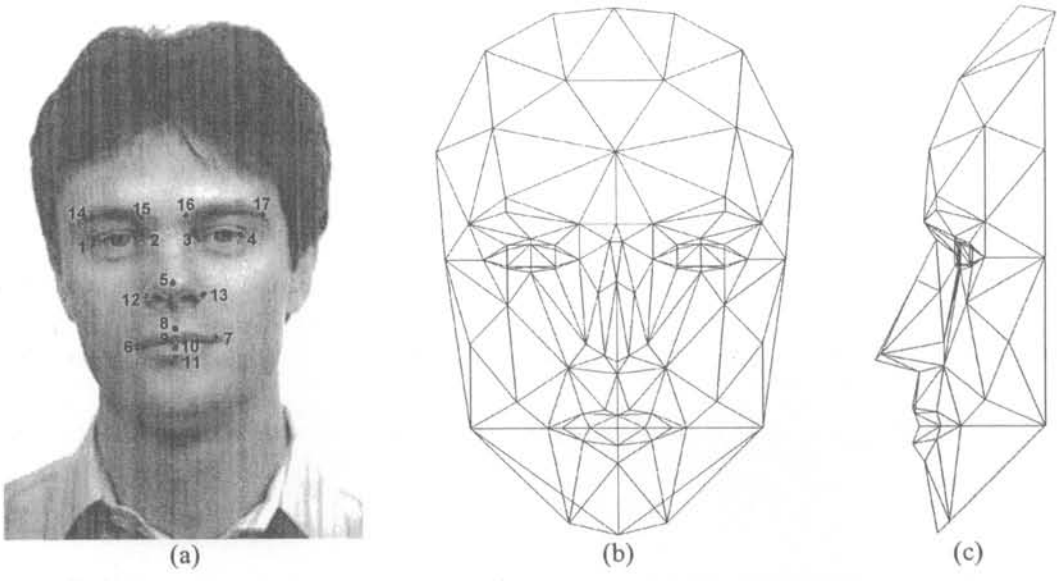
Lots of work is required to construct a 3D model from multiple images. Sengupta *et al.* [94, 95] proposed a method to derive the 3D face model from a few monocular images by using affine epipolar geometry, described in [97], and spline-fitting techniques to estimate actual shape of a human face. Shan and Liu [55, 96] proposed a new model-based bundle adjustment algorithm to recover the 3D model of an object from a sequence of images with unknown motions. The model-based bundle adjustment method can optimally reconstruct the shape of the human face. Ikeda [37] combines segmentation with photometric stereo to reconstruct shapes for multi-colored objects by using three images. Pighin *et al.* [77] developed a system to reconstruct 3D model by fitting a 3D mesh model across multiple images. The

correspondences among the images are specified manually by user. Kang *et al.* [42] used the linear spaces of geometrical models to construct 3D face models from multiple images. This approach required manually alignment of a generic mesh to one of the images, which is a tedious task for general user.

In this chapter, we propose an efficient algorithm to estimate the depth of a 3D face model for a human face from multiple images without prior camera calibration. In this method, we use the CANDIDE model [5] as our 3D mesh to adapt to human faces. To estimate the depth of a human face, an iteration procedure is proposed to minimize the similarity distance between the CANDIDE model and faces viewed at different perspective.

5.2 Similarity Measure by Matching 2D Point Sets

A set of 17 feature points, $\left\{\left(x_{\mathbf{P}_i}, y_{\mathbf{P}_i}\right)\right\}_{i=0}^{17}$, as shown in Fig. 5.1 (a), is located in an image that may be at different perspective angles. Similarly, a set of 17 feature points, denoted as $\left\{\left(x_{\mathbf{Q}_i}, y_{\mathbf{Q}_i}\right)\right\}_{i=0}^{17}$, were also identified in the CANDIDE model, as shown in Fig.5.1(b).



(a) (b) (c)

Fig. 5.1. (a) The 17 feature points in a human face image. (b) Frontal view of the CANDIDE model. (c) Profile view of the CANDIDE model.

These two point sets, which are denoted as **P** and **Q**, are each represented by a 2×17 matrix. The *i*th column of the matrix **P** represents the coordinates of the *i*th feature point in the corresponding image. In order to compare differences between the image and the CANDIDE mode, the difference between the two point sets, **P** and **Q**, are measured. The difference between two matrices can usually be computed using the Frobenius norm. The Frobenius norm is the same as the Euclidean distance between respective points in the images. In our algorithm, a face is represented by 17 feature points which have different degrees of significance and reliability. Thus, a weighted Euclidean distance, D_w, is used to compute the distance between **P** and **Q**.

$$\begin{aligned} D_w &= \sum_{i=0}^{17} \kappa_i \left\| \mathbf{P}_i - \mathbf{Q}_i \right\|^2 \\ &= \sum_{i=0}^{17} \kappa_i \left\{ \left(x_{\mathbf{P}_i} - x_{\mathbf{Q}_i} \right) + \left(y_{\mathbf{P}_i} - y_{\mathbf{Q}_i} \right) \right\}^2 \end{aligned} \tag{5.1}$$

where κ_i is the weighting factor for $\mathbf{P}_i$ and $\mathbf{Q}_i$.

The similarity measure, which is proposed by Werman *et al.* [115], can normalize and align the two point sets before performing the comparison. First of all,

each point set is translated to its centroid, so that its first moment is zero. Before computing the norm between the two point sets, there are two operations, namely *normalization* and *point alignment*, to remove differences between the two point sets due to irrelevant effects. Normalization is to normalize the size of the two point sets. Point alignment, which includes rotation, translation and scale, is applied to the image to obtain optimal alignment between the two point sets. In our approach, scale normalization is first applied to the point sets as follows:

$$\mathbf{P}' = \frac{\mathbf{P}}{\|\mathbf{P}\|}, \qquad \mathbf{Q}' = \frac{\mathbf{Q}}{\|\mathbf{Q}\|}. \tag{5.2}$$

The point set **P'** and **Q'** are aligned with a scaled rotation as follows:

$$\mathbf{P}'' = sR \cdot \mathbf{P}' \tag{5.3}$$

where s is a scalar and R is a 2×2 rotation matrix,

$$R = \begin{pmatrix} \cos\mu & \sin\mu \\ -\sin\mu & \cos\mu \end{pmatrix} \tag{5.4}$$

where μ is the angle of rotation for alignment. The respective values of μ and s can be obtained by minimizing the following expression:

$$\|sR \cdot \mathbf{P}' - \mathbf{Q}'\| = \min_{s',R'} \|s'R' \cdot \mathbf{P}' - \mathbf{Q}'\|^2. \tag{5.5}$$

The optimal alignment can be obtained by differentiating the distance expression, $tr\left[(sR \cdot \mathbf{P}' - \mathbf{Q}') \cdot (sR \cdot \mathbf{P}' - \mathbf{Q}')^T\right]$, with respect to μ and s, and equating the partial derivatives to 0. The equations are solved as below:

$$s = \sqrt{rt^2[\mathbf{P}'(\mathbf{Q}')^T] + tr^2[\mathbf{P}'(\mathbf{Q}')^T]}, \text{ and}$$

$$\tan\mu = \frac{rt[\mathbf{P}'(\mathbf{Q}')^T]}{tr[\mathbf{P}'(\mathbf{Q}')^T]}. \tag{5.6}$$

where *tr*[] of matrix is the sum of its diagonal elements and *rt*[] is the difference between its off-diagonal elements. The 2D alignment transformation can therefore be given as follows:

$$sR = \begin{pmatrix} tr[\mathbf{P}'(\mathbf{Q}')^T] & rt[\mathbf{P}'(\mathbf{Q}')^T] \\ -rt[\mathbf{P}'(\mathbf{Q}')^T] & tr[\mathbf{P}'(\mathbf{Q}')^T] \end{pmatrix}. \qquad (5.7)$$

Based on the above procedures, the point set of an input face is normalized, aligned and compared with the normalized point sets of the CANDIDE model.

5.3 Depth Estimation

In our approach, the CANDIDE model is first adapt to faces at 0° by local adaptation, as shown in Fig. 5.2(b). The feature points of faces at different views have been manually selected. The estimation of the depths at the different feature points is performed as follows:

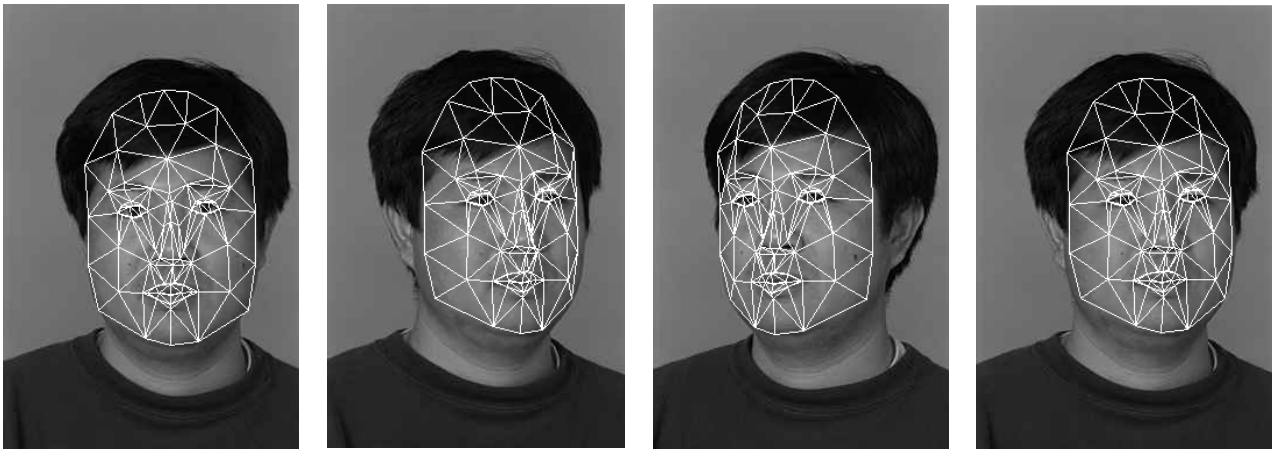
1. Adjust the *z*-coordinates of one of the feature points in the 0° CANDIDE model.
2. Rotate the 0° CANDIDE model to the corresponding perspective view angle of face image.
3. Project the rotated 0° CANDIDE model to the 2D plane.
4. Calculate the similarity distance using (5.1) between the point sets of the projected model and of the 2D image.
5. Go back to step 1 to adjust the *z*-coordinates until minimum similarity distance is found.

Each feature point iterates through the above procedure until the overall similarity distance between the different perspective view and the 0° CANDIDE model is minimized or until a maximum number of iterations has been reached. Figures 5.2(a)-(c) shows the adaptation between the feature points and the CANDIDE model. After

adaptation, the depth of the feature points of the CANDIDE model should be a good estimation of the corresponding depth of the human face in the 2D face images used in the iteration process. The trained CANDIDE is then used to adapt to a testing image of the same person at a different perspective, as shown in Fig. 5.2(d); the similarity distance is also computed, as shown in Table 5.1(b). This similarity distance can be used as a measure of the accuracy of the 3D CANDIDE model used to represent the 3D structure of the human face concerned.

5.4 Experimental Result

Our experiments were conducted based on the FERET database [75]. This database includes face images of various poses, including profiles of different expressions and different illuminations. Pose variations of the face images were captured systematically. Since the database contains many face images of many different people, we selected 10 people with 3 different poses for training in order to estimate the depth of the feature points, while another face in a different pose was used for testing. The training images and their adaptation are shown in Fig. 5.2-5.11(a)-(c). The adaptation to testing images is shown in Fig. 5.2-5.11(d). Table 5.1-5.10(a) show the changes of the z-coordinate of each facial feature, and the distance between the CANDIDE model and the facial feature points before and after the depth estimation at different perspectives and in different testing images. Table 5.1-5.10(b) show the similarity distance between the CANDIDE model and the facial feature points before and after the depth estimation.



(a) (b) © (d)

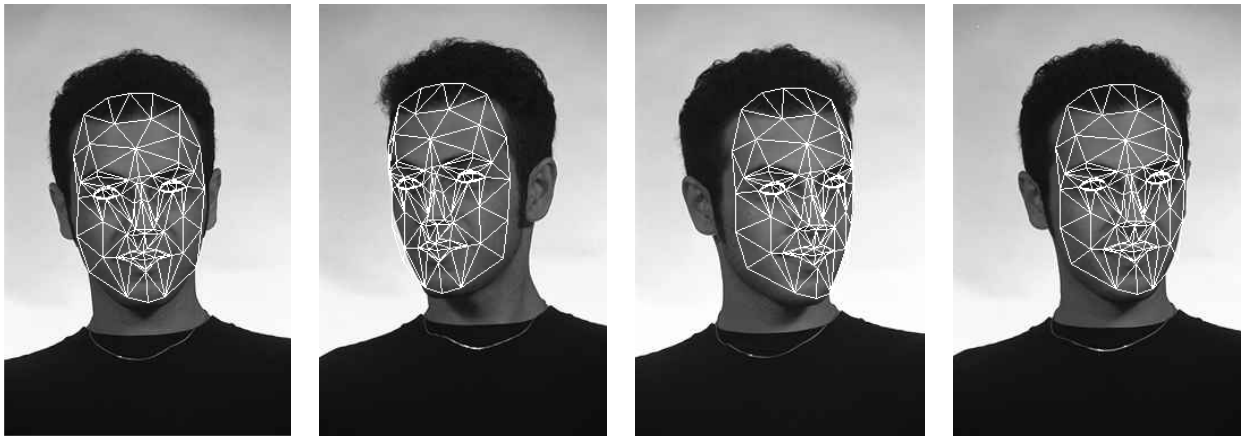
Fig. 5.2. Face adaptation with CANDIDE model. (a) Training face at 0° perspective. (b) Training face at 25°perspective. (c) Training face at -25° perspective. (d) Testing face at 15° perspective.

Facial Features	z-coordinate	The distance between feature points and CANDIDE model before/after depth estimation.					
		before at 25°	after at 25°	before at -25°	after at -25°	before at 15°	after at 15°
1	8.48	0.005251	0.013118	0.026768	0.007399	0.002956	0.008355
2	-3.59	0.008993	0.005035	0.010700	0.002236	0.008482	0.004843
3	-1.78	0.005173	0.008116	0.007321	0.002021	0.003896	0.001256
4	6.34	0.036902	0.024232	0.007702	0.019615	0.024838	0.017708
5	-2.47	0.010330	0.005608	0.003840	0.004110	0.006202	0.002497
6	2.33	0.011828	0.015734	0.015781	0.010935	0.011355	0.013582
7	2.14	0.006247	0.003934	0.013116	0.012491	0.003885	0.002868
8	-1.77	0.010279	0.009403	0.017043	0.016344	0.008914	0.010244
9	-2.43	0.010532	0.009187	0.012435	0.010073	0.013104	0.013663
10	-2.80	0.015386	0.014567	0.016530	0.015137	0.018333	0.018811
11	-0.90	0.015007	0.016048	0.010501	0.007518	0.003070	0.004366
12	2.24	0.004680	0.008945	0.016948	0.014376	0.007744	0.010102
13	1.10	0.017409	0.016027	0.012376	0.013716	0.031348	0.030462
14	2.97	0.020833	0.026689	0.027335	0.021070	0.024632	0.027709
15	-5.86	0.021218	0.008096	0.005528	0.008720	0.011830	0.004496
16	-8.30	0.021298	0.015513	0.026861	0.025090	0.020382	0.014143
17	9.99	0.038444	0.020324	0.014449	0.006691	0.014987	0.003785

Table 5.1(a). Face adaptation for Fig. 5.2.

Angle	Training image 25°	Training image -25°	Testing image 15°
2D similarity distance before depth estimation	0.008875	0.005626	0.005119
2D similarity distance after depth estimation	0.004643	0.002704	0.003794

Table 5.1(b). The similarity distance before and after iteration for Fig. 5.2.



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

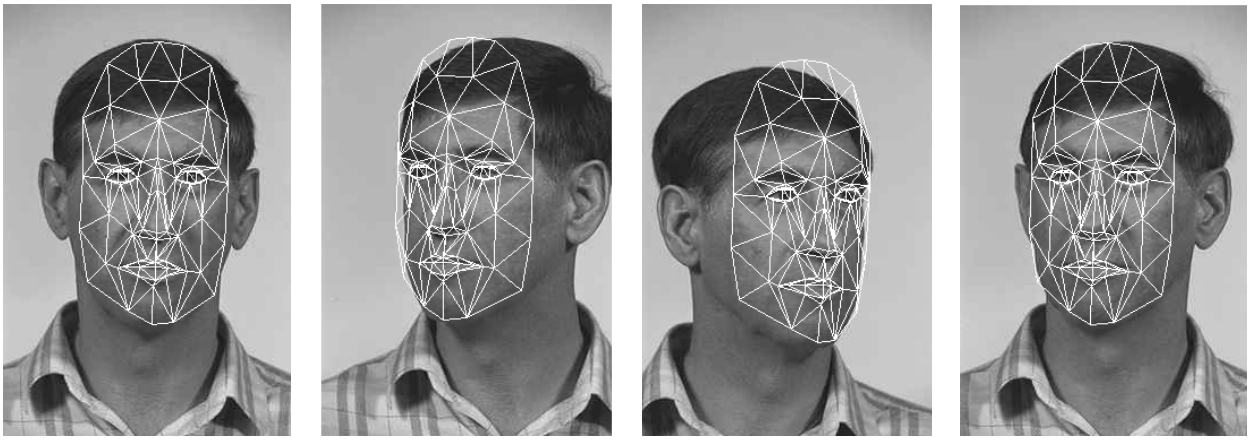
Fig. 5.3. Face adaptation with CANDIDE model. (a) Training face at 0° perspective. (b) Training face at -22°perspective. (c) Training face at 22° perspective. (d) Testing face at 10° perspective.

Facial Features	z-coordinate	The distance between feature points and CANDIDE model before/after depth estimation.					
		before at -22°	after at -22°	before at 22°	after at 22°	before at 10°	after at 10°
1	-0.12	0.012453	0.015838	0.013178	0.011643	0.018180	0.017242
2	-6.96	0.034672	0.022485	0.011415	0.021169	0.011171	0.015866
3	-3.22	0.007343	0.010532	0.016178	0.013311	0.009562	0.006995
4	-2.14	0.013938	0.014665	0.009353	0.013160	0.018882	0.020184
5	9.99	0.050655	0.027949	0.012625	0.014951	0.006861	0.006192
6	-5.92	0.020595	0.010427	0.013878	0.004438	0.016592	0.012490
7	-3.90	0.003544	0.004199	0.014171	0.006271	0.014644	0.010990
8	-3.50	0.016674	0.012513	0.006845	0.006382	0.002086	0.004308
9	-1.95	0.001920	0.001983	0.008087	0.008073	0.008427	0.009082
10	-2.22	0.001920	0.002048	0.008087	0.007700	0.008427	0.009008
11	-8.57	0.018335	0.010336	0.020672	0.006163	0.020629	0.016062
12	4.16	0.018970	0.007477	0.005956	0.007235	0.015360	0.015164
13	9.99	0.028627	0.003031	0.024164	0.000611	0.002726	0.011578
14	-7.00	0.040742	0.035495	0.008846	0.013707	0.017439	0.021199
15	-5.65	0.005225	0.012164	0.020764	0.017811	0.011244	0.009329
16	7.21	0.023787	0.005101	0.020140	0.007472	0.006715	0.005380
17	-0.58	0.001452	0.003090	0.023127	0.020677	0.006376	0.005933

Table 5.2(a). Face adaptation for Fig. 5.3.

Angle	Training image -22°	Training image 22°	Testing image 10°
2D similarity distance before depth estimation	0.012509	0.003892	0.001927
2D similarity distance after depth estimation	0.003207	0.001495	0.001935

Table 5.2(b). The similarity distance before and after iteration for Fig. 5.3.



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

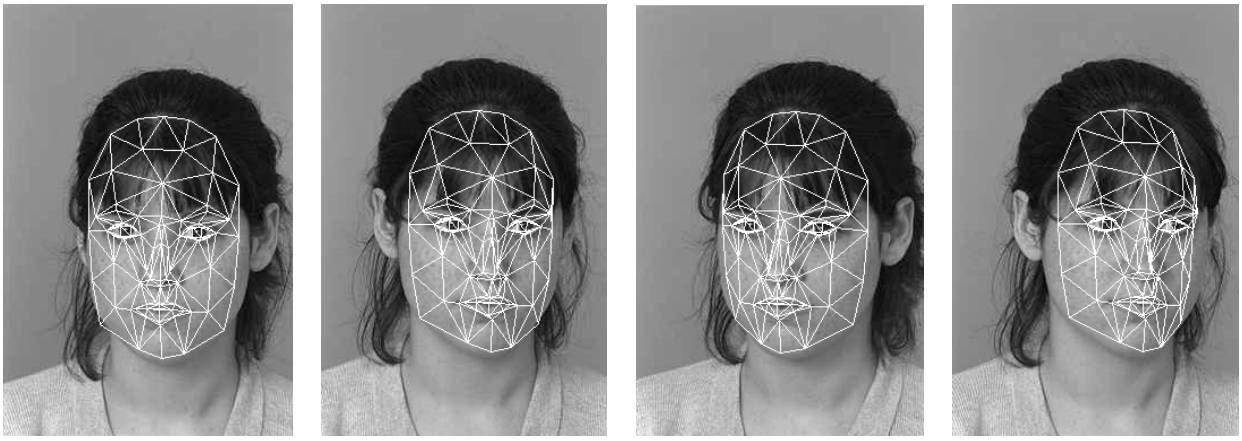
Fig. 5.4. Face adaptation with CANDIDE model. (a) Training face at 0° perspective. (b) Training face at -22°perspective. (c) Training face at 22° perspective. (d) Testing face at -10° perspective.

Facial Features	z-coordinate	The distance between feature points and CANDIDE model before/after depth estimation.					
		before at -22°	after at -22°	before at 22°	after at 22°	before at 10°	after at 10°
1	-1.45	0.003689	0.005724	0.008518	0.011476	0.003490	0.004622
2	-8.54	0.036409	0.025573	0.005666	0.013072	0.007836	0.002942
3	-3.60	0.006409	0.006684	0.014946	0.012162	0.007108	0.006006
4	2.53	0.002291	0.006868	0.017611	0.015108	0.014478	0.018376
5	9.99	0.060163	0.037567	0.038263	0.017184	0.024133	0.013499
6	-9.57	0.032645	0.019384	0.012077	0.023309	0.021198	0.015561
7	-8.83	0.014954	0.004972	0.005464	0.010016	0.009469	0.005592
8	-1.72	0.006554	0.005494	0.004507	0.004563	0.007996	0.007344
9	0.09	0.013300	0.010825	0.008065	0.005251	0.009565	0.007625
10	0.12	0.013300	0.010811	0.008065	0.005221	0.009565	0.007605
11	-4.18	0.002578	0.005264	0.011026	0.010785	0.004272	0.005075
12	-3.96	0.003256	0.003834	0.010579	0.009597	0.004332	0.005525
13	4.89	0.009732	0.009709	0.022401	0.011610	0.001483	0.007144
14	-0.87	0.008631	0.011079	0.016669	0.019067	0.021733	0.022671
15	-5.60	0.011828	0.008799	0.018868	0.015597	0.016255	0.018940
16	-7.70	0.002920	0.009421	0.027724	0.017410	0.009737	0.009410
17	-2.13	0.030435	0.030043	0.027088	0.025808	0.014893	0.014643

Table 5.3(a). Face adaptation for Fig. 5.4.

Angle	Training image -22°	Training image 22°	Testing image -10°
2D similarity distance before depth estimation	0.012832	0.006001	0.003998
2D similarity distance after depth estimation	0.005515	0.002716	0.003039

Table 5.3(b). The similarity distance before and after iteration for Fig. 5.4.



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

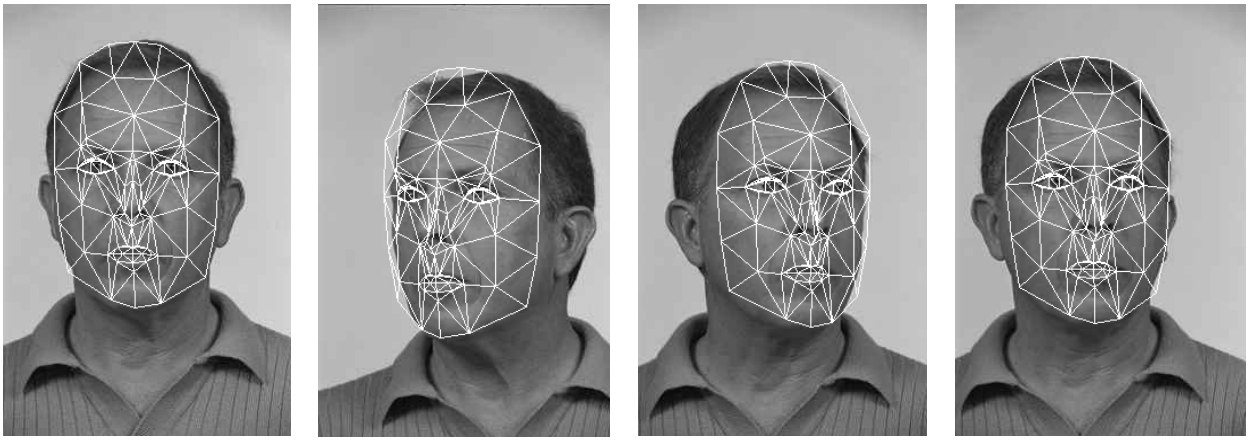
Fig. 5.5. Face adaptation with CANDIDE model. (a) Training face at 0° perspective. (b) Training face at 15°perspective. (c) Training face at -15° perspective. (d) Testing face at 25° perspective.

Facial Features	z-coordinate	The distance between feature points and CANDIDE model before/after depth estimation.					
		before at 15°	after at 15°	before at -15°	after at -15°	before at 25°	after at 25°
1	5.96	0.007063	0.001973	0.011531	0.004938	0.014645	0.007976
2	-1.71	0.005273	0.007743	0.011665	0.010184	0.006596	0.003492
3	2.63	0.012209	0.009372	0.006006	0.007268	0.006298	0.003642
4	9.99	0.026424	0.013555	0.011632	0.006818	0.038729	0.017864
5	0.86	0.006272	0.005920	0.008900	0.009298	0.004323	0.003558
6	0.45	0.016289	0.016198	0.012568	0.012907	0.022001	0.021883
7	1.07	0.014175	0.014666	0.009688	0.009684	0.020691	0.021209
8	-4.60	0.016503	0.009646	0.002888	0.004213	0.020428	0.009039
9	-2.78	0.013363	0.008810	0.005293	0.006454	0.015483	0.007812
10	-3.11	0.013357	0.008344	0.005283	0.006755	0.015482	0.007037
11	-6.69	0.013093	0.005439	0.014405	0.011568	0.028409	0.018453
12	1.63	0.003003	0.004219	0.007404	0.006188	0.000961	0.002106
13	2.76	0.012098	0.009232	0.012474	0.013986	0.027295	0.022174
14	0.37	0.012946	0.012908	0.019608	0.018869	0.023634	0.023472
15	5.46	0.032731	0.026391	0.015837	0.023014	0.021529	0.015926
16	-6.27	0.009406	0.002468	0.006366	0.003554	0.006835	0.011620
17	2.77	0.010064	0.008838	0.011714	0.011978	0.031565	0.026666

Table 5.4(a). Face adaptation for Fig. 5.5.

Angle	Training image 15°	Training image -15°	Testing image 25°
2D similarity distance before depth estimation	0.004715	0.002228	0.009430
2D similarity distance after depth estimation	0.002519	0.001910	0.004895

Table 5.4(b). The similarity distance before and after iteration for Fig. 5.5.



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

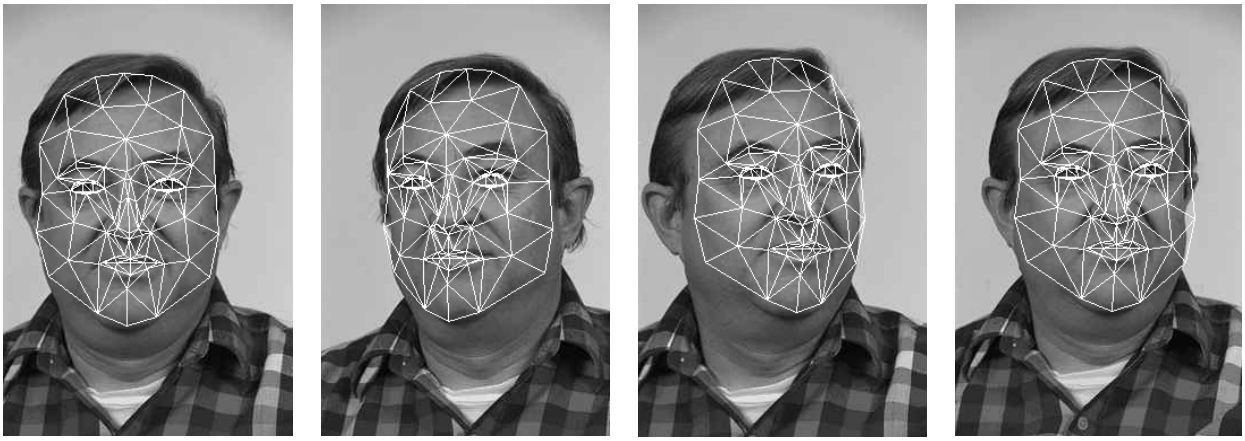
Fig. 5.6. Face adaptation with CANDIDE model. (a) Training face at 0° perspective. (b) Training face at -22°perspective. (c) Training face at 22° perspective. (d) Testing face at 10° perspective.

Facial Features	z-coordinate	The distance between feature points and CANDIDE model before/after depth estimation.					
		before at -22°	after at -22°	before at 22°	after at 22°	before at 10°	after at 10°
1	-6.99	0.022382	0.021495	0.020119	0.009960	0.018564	0.013475
2	-7.54	0.012103	0.012504	0.020239	0.006980	0.014596	0.009461
3	-8.91	0.018855	0.005404	0.011931	0.005284	0.006581	0.004158
4	2.31	0.011953	0.010755	0.010576	0.007133	0.003910	0.003638
5	9.99	0.041384	0.021194	0.032785	0.023837	0.019258	0.014730
6	-1.06	0.007439	0.006114	0.005129	0.006419	0.002688	0.002627
7	1.21	0.009062	0.011892	0.016421	0.013504	0.015153	0.013907
8	-2.93	0.006539	0.006282	0.004197	0.001474	0.000774	0.001629
9	-1.07	0.005293	0.004308	0.006629	0.008413	0.001119	0.001770
10	-1.24	0.005293	0.003973	0.006629	0.008690	0.001119	0.001912
11	-6.74	0.016960	0.006862	0.001851	0.012192	0.001591	0.004086
12	3.64	0.019166	0.011738	0.003810	0.006288	0.007312	0.006247
13	7.77	0.012447	0.004625	0.028221	0.018844	0.021769	0.014684
14	6.30	0.018238	0.009530	0.003940	0.011398	0.010887	0.008581
15	-3.85	0.019134	0.016657	0.007895	0.005941	0.012407	0.014996
16	-10.00	0.035532	0.020547	0.038872	0.020755	0.023388	0.015039
17	9.99	0.034691	0.017305	0.012598	0.019516	0.015512	0.023412

Table 5.5(a). Face adaptation for Fig. 5.6.

Angle	Training image -22°	Training image 22°	Testing image 10°
2D similarity distance before depth estimation	0.010930	0.006106	0.004098
2D similarity distance after depth estimation	0.002313	0.001412	0.002630

Table 5.5(b). The similarity distance before and after iteration for Fig. 5.6.



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

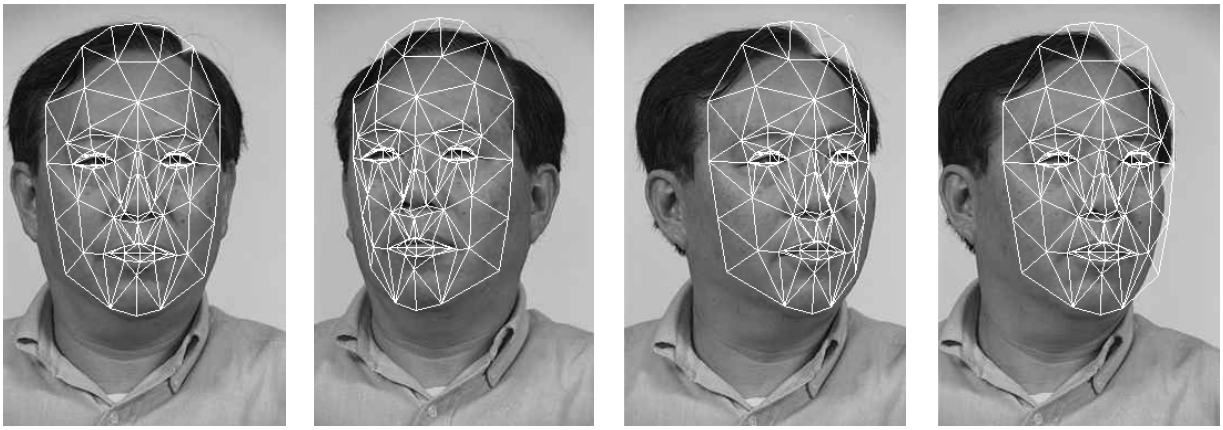
Fig. 5.7. Face adaptation with CANDIDE model. (a) Training face at 0° perspective. (b) Training face at -22°perspective. (c) Training face at 22° perspective. (d) Testing face at 10° perspective.

Facial Features	z-coordinate	The distance between feature points and CANDIDE model before/after depth estimation.					
		before at -22°	after at -22°	before at 22°	after at 22°	before at 10°	after at 10°
1	9.99	0.046167	0.033913	0.010427	0.005146	0.017962	0.024412
2	-2.66	0.012910	0.018074	0.011303	0.005590	0.009358	0.013048
3	-2.95	0.006919	0.014787	0.017281	0.015803	0.008933	0.012330
4	8.89	0.036154	0.025128	0.013708	0.012586	0.003342	0.010709
5	7.13	0.004660	0.012408	0.021810	0.009968	0.016445	0.011540
6	2.64	0.011470	0.012956	0.006856	0.008989	0.003060	0.002031
7	5.10	0.005735	0.012242	0.006533	0.011489	0.021847	0.025688
8	-1.87	0.020574	0.014837	0.014525	0.012978	0.009479	0.009471
9	-1.07	0.021716	0.017181	0.006397	0.002650	0.002094	0.001257
10	-1.20	0.021716	0.016963	0.006397	0.002608	0.002094	0.001327
11	-2.15	0.029326	0.023005	0.006014	0.004038	0.009264	0.009147
12	-0.40	0.018252	0.017291	0.002655	0.001619	0.011984	0.012659
13	1.84	0.018173	0.019597	0.014460	0.012684	0.014758	0.013965
14	8.17	0.026594	0.017794	0.019697	0.014442	0.004372	0.001464
15	-1.27	0.020623	0.025777	0.020872	0.015214	0.014598	0.012054
16	-7.84	0.013637	0.014070	0.008446	0.009682	0.009954	0.016238
17	-0.60	0.001105	0.003333	0.013072	0.015728	0.004111	0.004259

Table 5.6(a). Face adaptation for Fig. 5.7.

Angle	Training image -22°	Training image 22°	Testing image 10°
2D similarity distance before depth estimation	0.006062	0.004011	0.003219
2D similarity distance after depth estimation	0.002143	0.001935	0.004492

Table 5.6(b). The similarity distance before and after iteration for Fig. 5.7.



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

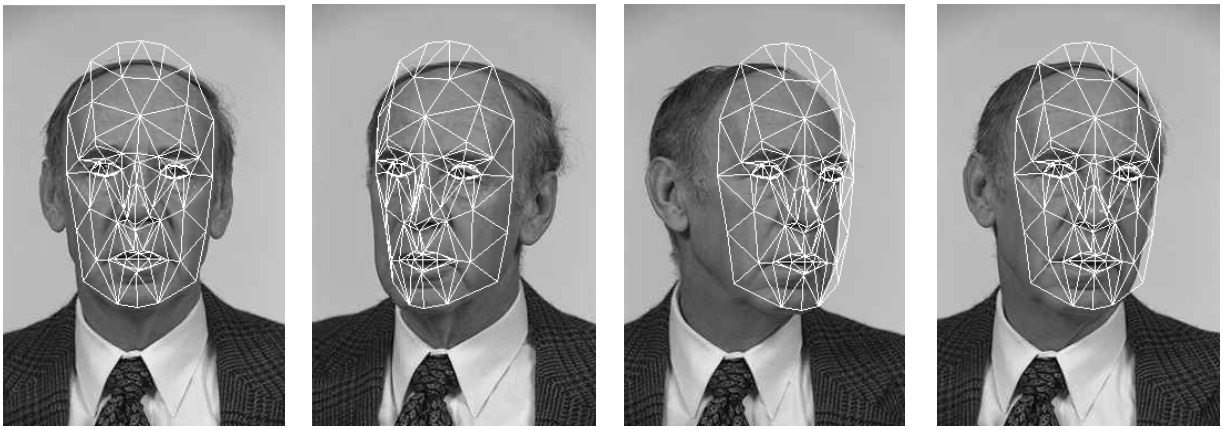
Fig. 5.8. Face adaptation with CANDIDE model. (a) Training face at 0° perspective. (b) Training face at -22°perspective. (c) Training face at 22° perspective. (d) Testing face at 10° perspective.

Facial Features	z-coordinate	The distance between feature points and CANDIDE model before/after depth estimation.					
		before at -22°	after at -22°	before at 22°	after at 22°	before at 10°	after at 10°
1	9.99	0.011007	0.007511	0.015212	0.007971	0.013102	0.012511
2	0.66	0.012738	0.011598	0.007865	0.008515	0.003244	0.003902
3	1.55	0.001707	0.003004	0.010025	0.010629	0.008980	0.009918
4	2.99	0.014433	0.017801	0.009191	0.006037	0.012484	0.014459
5	-2.03	0.023625	0.023820	0.011927	0.010279	0.023570	0.025380
6	3.67	0.007745	0.005963	0.022772	0.018556	0.013566	0.011700
7	-1.49	0.011955	0.015372	0.011862	0.014019	0.028175	0.026644
8	-1.29	0.010909	0.010652	0.006997	0.010305	0.002773	0.004477
9	0.01	0.009932	0.010674	0.006900	0.008886	0.005302	0.005286
10	0.02	0.009932	0.010658	0.006900	0.008871	0.005302	0.005279
11	-3.16	0.005599	0.011671	0.009034	0.013120	0.003195	0.001347
12	9.41	0.023355	0.009391	0.010749	0.003364	0.009441	0.004723
13	-3.86	0.015617	0.008312	0.005116	0.012443	0.005480	0.007102
14	7.39	0.008444	0.007316	0.022763	0.029662	0.014646	0.013673
15	-8.15	0.019077	0.007006	0.035500	0.020381	0.012727	0.006239
16	-7.85	0.027959	0.013895	0.017945	0.003558	0.003156	0.009594
17	6.57	0.001872	0.009811	0.013467	0.013945	0.010589	0.014979

Table 5.7(a). Face adaptation for Fig. 5.8.

Angle	Training image -22°	Training image 22°	Testing image 10°
2D similarity distance before depth estimation	0.003684	0.003430	0.003297
2D similarity distance after depth estimation	0.001020	0.001237	0.003459

Table 5.7(b). The similarity distance before and after iteration for Fig. 5.8.



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

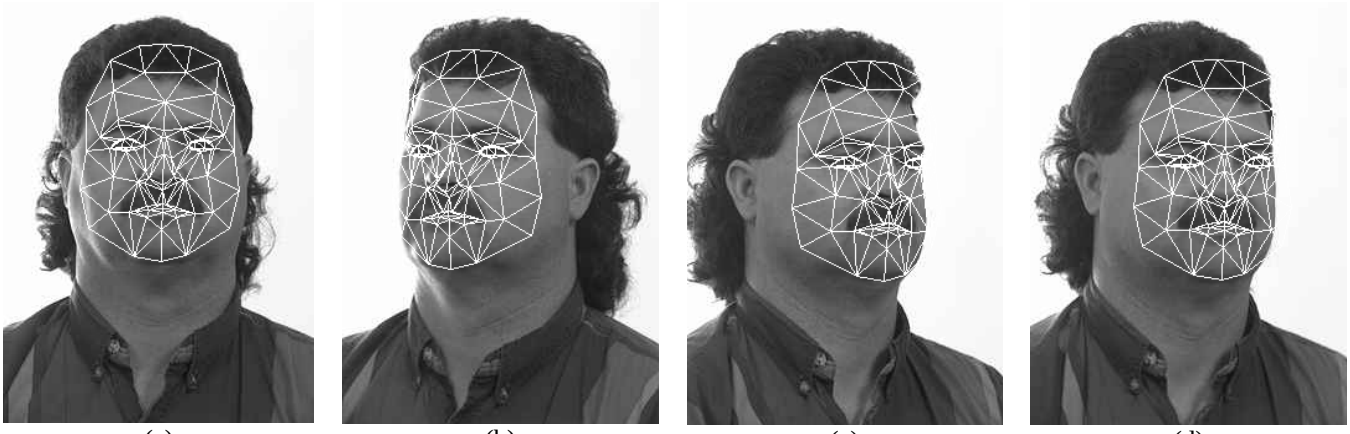
Fig. 5.9. Face adaptation with CANDIDE model. (a) Training face at 0° perspective. (b) Training face at -22°perspective. (c) Training face at 22° perspective. (d) Testing face at 10° perspective.

Facial Features	z-coordinate	The distance between feature points and CANDIDE model before/after depth estimation.					
		before at -22°	after at -22°	before at 22°	after at 22°	before at 10°	after at 10°
1	2.54	0.015358	0.016012	0.017009	0.015954	0.010837	0.011707
2	-7.31	0.024986	0.011828	0.012523	0.014158	0.010257	0.014010
3	0.38	0.009710	0.007435	0.010444	0.012521	0.006531	0.007387
4	0.46	0.020843	0.019214	0.015240	0.017330	0.016916	0.016936
5	9.99	0.007246	0.013594	0.058285	0.038159	0.036651	0.029586
6	0.67	0.017132	0.015119	0.012381	0.014383	0.032064	0.032986
7	-2.29	0.020422	0.023278	0.024878	0.021952	0.009012	0.007649
8	-3.19	0.010204	0.012860	0.013687	0.008541	0.018022	0.015682
9	-1.46	0.012880	0.013415	0.009867	0.007931	0.021664	0.021008
10	-1.69	0.012880	0.013781	0.009867	0.007492	0.021664	0.020813
11	-5.93	0.011249	0.018921	0.015536	0.006620	0.026064	0.021457
12	6.14	0.027052	0.016968	0.004743	0.011884	0.025009	0.022062
13	2.00	0.015453	0.019167	0.017531	0.012214	0.008865	0.006408
14	4.28	0.030457	0.034569	0.022185	0.017253	0.015936	0.012886
15	-4.02	0.043025	0.035588	0.022369	0.027943	0.027074	0.029057
16	-2.86	0.013494	0.010528	0.008021	0.005804	0.007795	0.009132
17	-6.14	0.022288	0.019444	0.041118	0.040530	0.044101	0.046376

Table 5.8(a). Face adaptation for Fig. 5.9.

Angle	Training image -22°	Training image 22°	Testing image 10°
2D similarity distance before depth estimation	0.005163	0.010375	0.004708
2D similarity distance after depth estimation	0.003248	0.005966	0.003486

Table 5.8(b). The similarity distance before and after iteration for Fig. 5.9.



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

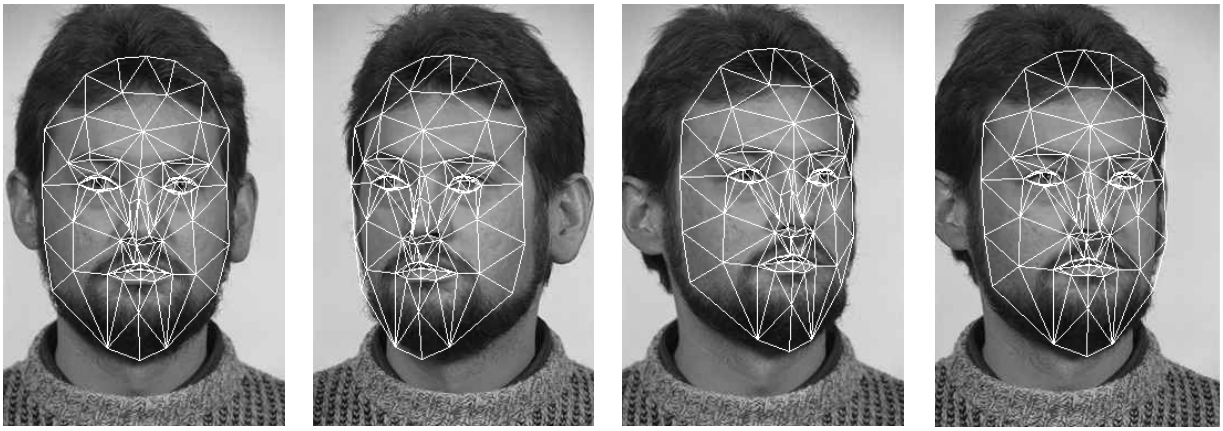
Fig. 5.10. Face adaptation with CANDIDE model. (a) Training face at 0° perspective. (b) Training face at -22°perspective. (c) Training face at 22° perspective. (d) Testing face at 10° perspective.

Facial Features	z-coordinate	The distance between feature points and CANDIDE model before/after depth estimation.					
		before at -22°	after at -22°	before at 22°	after at 22°	before at 10°	after at 10°
1	0.58	0.023896	0.022780	0.012934	0.011834	0.018663	0.018295
2	-1.47	0.025989	0.021611	0.011616	0.009611	0.004302	0.003944
3	-4.12	0.012092	0.008239	0.038543	0.029835	0.040730	0.036896
4	-6.95	0.034687	0.023484	0.020188	0.006400	0.018980	0.012491
5	9.99	0.048251	0.025439	0.043671	0.017328	0.047388	0.035895
6	6.36	0.024231	0.011404	0.040899	0.024898	0.026931	0.027569
7	-1.77	0.024813	0.025597	0.007602	0.009418	0.009433	0.009369
8	0.14	0.017448	0.017450	0.014848	0.014436	0.013548	0.012973
9	1.01	0.021194	0.017891	0.015838	0.012122	0.016616	0.014866
10	1.14	0.021194	0.017569	0.015838	0.011807	0.016616	0.014729
11	-2.34	0.015219	0.021094	0.014771	0.017512	0.023356	0.024015
12	6.83	0.010140	0.017820	0.037490	0.020565	0.034392	0.026677
13	9.36	0.030847	0.005673	0.024446	0.015519	0.025524	0.024356
14	-10.00	0.043436	0.021193	0.045383	0.020699	0.039136	0.028502
15	-0.56	0.031019	0.028505	0.015320	0.013941	0.004999	0.004811
16	-4.65	0.017501	0.008422	0.040071	0.030513	0.022060	0.021981
17	-10.00	0.027050	0.012083	0.050128	0.027844	0.030747	0.020899

Table 5.9(a). Face adaptation for Fig. 5.10.

Angle	Training image -22°	Training image 22°	Testing image 10°
2D similarity distance before depth estimation	0.010082	0.017239	0.014247
2D similarity distance after depth estimation	0.004298	0.005636	0.009347

Table 5.9(b). The similarity distance before and after iteration for Fig. 5.10.



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

Fig. 5.11. Face adaptation with CANDIDE model. (a) Training face at 0° perspective. (b) Training face at -22°perspective. (c) Training face at 22° perspective. (d) Testing face at 10° perspective.

Facial Features	z-coordinate	The distance between feature points and CANDIDE model before/after depth estimation.					
		before at -22°	after at -22°	before at 22°	after at 22°	before at 10°	after at 10°
1	7.52	0.010645	0.009718	0.010615	0.012063	0.002994	0.003932
2	-8.41	0.004777	0.010852	0.034551	0.019273	0.014182	0.007455
3	-3.05	0.006951	0.003017	0.012346	0.007621	0.008244	0.007168
4	7.15	0.007565	0.010440	0.011014	0.005478	0.072851	0.078613
5	9.99	0.021970	0.013166	0.030173	0.016879	0.040903	0.033613
6	0.00	0.008386	0.008993	0.010576	0.010440	0.011888	0.011673
7	-3.30	0.013688	0.008240	0.015921	0.021734	0.026913	0.029566
8	-2.26	0.005826	0.007509	0.009524	0.011567	0.017846	0.019568
9	-2.16	0.001479	0.004439	0.007741	0.011421	0.017899	0.019602
10	-2.58	0.001542	0.005175	0.007737	0.012100	0.016928	0.019097
11	-7.82	0.005909	0.008771	0.004448	0.015446	0.018339	0.023765
12	9.28	0.020221	0.005678	0.019103	0.003046	0.006411	0.008174
13	7.81	0.010140	0.012985	0.029192	0.016860	0.025752	0.019511
14	0.13	0.004662	0.004283	0.028746	0.028602	0.030136	0.029942
15	-0.92	0.014181	0.013845	0.025319	0.023447	0.005781	0.004857
16	-8.23	0.025481	0.011193	0.022677	0.011304	0.010287	0.003792
17	-0.74	0.009105	0.008030	0.009919	0.008729	0.021898	0.021369

Table 5.10(a). Face adaptation for Fig. 5.11.

Angle	Training image -22°	Training image 22°	Testing image 10°
2D similarity distance before depth estimation	0.003579	0.006646	0.013569
2D similarity distance after depth estimation	0.001384	0.001520	0.014202

Table 5.10(b). The similarity distance before and after iteration for Fig. 5.11.

According to our experiment results, the distance between the feature points of the CANDIDE model and the human faces, and the similarity distance are minimized after depth estimation. The adaptation to the testing image is also minimized once the CANDIDE model is trained. For a better adaptation result, more images can be used as training image in depth estimation. The results show that the depth is successfully estimated, and the information can be used for 3D modeling and texture mapping.

5.5 Conclusion

In this chapter, a novel and efficient method is proposed to estimate the depth of the 3D face model of a human face from multiple images without requiring camera calibration. In this method, CANDIDE model is first adapted to the face image at frontal-view. In order to estimate the depth in each perspective, an iteration procedure is performed to minimize the similarity distance between the frontal-view CANDIDE model and the faces at different perspective view by adjusting the depth of the feature points in the CANDIDE model.

After the depth of a human face is determined, its texture information can be combined with the CANDIDE model to generate a realistic human face model. Texture mapping and curve-fitting techniques will be involved.

Chapter 6
Conclusion and Future Work

6.1 Conclusion

In this thesis, we have studied various technologies within the scope of human face modeling, including human face detection, face tracking and 3D face reconstruction. For human face detection, the knowledge-based method, the feature invariant approach, the template matching method and the appearance-based method have all been reviewed. The appearance-based method has been widely adopted; but a better performance in face detection can be achieved by using statistical analysis and the machine-learning method to discriminate between face and non-face images. Meanwhile, dimensionality reduction is usually carried out for the sake of computation efficiency and detection efficacy. Although significant progress has been made in the last two decades, existing techniques for face detection are still not robust enough, i.e. they should be effective under a full variation in lighting conditions, orientation, pose, partial occlusion, facial expression, presence of glasses, facial hair, and a variety of hair styles. For face tracking, the motion-detection, the model-based tracking, the active contour-based tracking and the feature-based tracking have all been reviewed. In human face tracking, the model-based approach has been widely adopted. However, the disadvantage with model-based tracking is its need for model construction, and its high computational cost for real time application. For 3D face reconstruction, basic projective geometry, the pre-calibrated reconstruction algorithm and the online calibrated reconstruction algorithm have all been reviewed. There are still many unresolved issues with performing 3D face reconstruction, such as the estimation of camera parameters in the online calibrated

reconstruction algorithm, the global optimization problem in the pre-calibrated reconstruction algorithm, etc.

In our research, we have proposed efficient methods for human face detection, face tracking and human face 3D reconstruction. Our approach to detecting face regions in a color image consists of three steps. The first step is to segment the face color by using the mean-shift algorithm. In this step, a color compensation scheme is proposed to extend the range of red component to its saturated level, as the red component of skin color saturates under strong illumination. According to different illumination conditions, the compensated skin color distribution is modeled using the mixture-of-Gaussians model to segment skin-color regions from the cluttered background. In the second step, possible eye candidates are searched within the segmented skin-color regions. By grouping pairs of eye candidates, possible face candidates are formed. Finally, a two-step procedure based on an eigenmask for face verification is performed. In order to further improve the reliability and accuracy, the face contour is further verified with a probability function after the face has been verified and short-listed. The performance of our human face detection algorithm is evaluated with the HHI MPEG-7 face database, the AR face database and the CMU Pose, Illumination and Expression (PIE) database, which contain face images under a wide range of lighting conditions, including poor conditions, under shadow, different scales and with glasses. Experimental results show that our human face detection algorithm is very fast and can achieve a high detection rate.

In face tracking, we have purposed a face tracking method that uses a combination of Gabor wavelet facial feature tracking and Temporally Maximum Occurrence Frame (TMOF) face template. A modified greedy algorithm is also purposed to search for facial feature locations by means of Gabor wavelet. The

TMOF face template can adapt to a changing face appearance in a video sequence. In our face-tracking algorithm, the location of facial features, including left eye, right eye and mouth, are first verified by the properties of triangular structure in the first stage. If the change of triangular structure within two frames is less than a certain threshold, Gabor representation is computed for similarity measurement. Finally, the possible faces are then verified by means of TMOF generated face template. The tracked face is then used to update the TMOF face template in order to adapt to the changing appearance in the video sequence. Experiment results show that our algorithm is robust to fast head movement and perspective change.

In human face 3D reconstruction, we have proposed to estimate depth of a human face from multiple images without prior camera calibration. A set 17 feature points was located in the input images at different perspective angles. The CANDIDE model is used as our 3D mesh to adapt to a human face. The rotation and translation between the model and the face are measured by a similarity transform. In order to estimate the depth of a human face, an iteration procedure is proposed to minimize the similarity distance between the CANDIDE model and faces at different perspective view.

6.2 Future Work

The human face-modeling system has made a lot of progress toward full automation. In our research, the system is divided into different parts and different steps. The integration of each step is required. However, the performance of each step is a concern. Face detection is the first step in the human face-modeling system. The detection performance can be affected by the presence of glasses, different skin color, gender, facial hair, facial expressions, etc. Once the face has been detected,

face tracking is performed to track the detected face in a video sequence. The face tracking can be affected by perspective variation, occlusion, fast motion, etc. Real-time performance is required for face tracking. Finally, construction of 3D face model is performed. Pose estimation is required for accurate depth estimation.

There are several possible directions for the future development of this project. Face detection under different lighting conditions can be improved. Also, the face-detection algorithm should be able to work for different perspective. The use of a multi-appearance model with a probabilistic approach should be able to improve the detection rate and to detect face at different poses. The performance of face tracking can be improved with an online appearance model-update algorithm, such as online PCA. A 3D face model can be used to adapt perspective change during tracking. Pose estimation can be applied to construct 3D face model during tracking. The pose can be estimated by using multi-model probability approach. A more accurate depth estimation of the 3D face model for a human face can then be performed using the similarity transform. The integration of these approaches can help to build a complete human face-modeling system. Finally, all of this work can be applied to face recognition and model-based video coding.

References

[1] "Visual Mapping by a Robot Rover," presented at Proceedings of International Joint Conference on Artifical Intelligence, 1977.

[2] "MPEG7 Content Set from Heinrich Hertz Institute," *http://www.darmstadt.gmd.de/mobile/hm/projects/MPEG7/Documents/N2466.html*, 1998.

[3] "Multi-View Stereo Beyond Lambert," presented at Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Madison, WI, 2002.

[4] J. K. Aggarwal, Q. Cai, W. Liao, and B. Sabata, "Non-Rigid Motion Analysis: Articulated & Elastic Motion," *Computer Vision and Image Understanding*, vol. 70, pp. 142-156, 1998.

[5] J. Ahlberg, "CANDIDE-3 - Un Updated Parameterised Face," Linkoping University, Lysator LiTH-ISY-R-2325, Jan. 2001.

[6] K. Aizawa and T. S. Huang, "Model-Based Image Coding Advanced Video Coding Techniques for Very Low Bit-Rate Applications," *Proceedings of the IEEE*, vol. 83, pp. 259-271, 1995.

[7] T. Akimoto, Y. Suenaga, and R. S. Wallace, "Automatic Creation of 3D Facial Models," *IEEE Trans. on Computer Graphics and Applications*, vol. 13, pp. 16-22, 1993.

[8] A. Baumberg and D. C. Hogg, "Learning Deformable Models for Tracking the Human Body," in *Motion-Based Recognition*, M. Shah and R. Jain, Eds. Norwell, MA: Kluwer, 1996, pp. 39-60.

[9] P. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces vs. Fisherfaces: Class Specific Linear Projection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 711-720, 1997.

[10] G. Bozdagi, A. M. Tekalp, and L. Onural, "3-D Motion Estimation and Wireframe Adaptation including Photometric Effects for Model-Based Coding of Facial Image Sequences," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 4, pp. 246-256, 1994.

[11] B. Caprile and V. Torre, "Using Vanishing Points for Camera Calibration," *Internation Journal of Computer Vision*, vol. 4, pp. 127-139, 1986.

[12] D. Chai and K. N. Ngan, "Face Segmentation Using Skin-Color Map in Videophone Application," *IEEE Trans. on Circuits and System for Video Technology*, vol. 9, pp. 551-564, 1999.

[13] R. Chellappa, C. L. Wilson, and S. Sirohey, "Human and Machine Recognition of Faces: A Survey," *Proceedings of the IEEE*, vol. 83, pp. 705-740, 1995.

[14] Q. Chen, H. Wu, and M. Yachida, "Face Detection by Fuzzy Matching," presented at Proceedings of Fifth IEEE International Conference on Computer Vision, 1995.

[15] Y. Cheng, "Mean Shift, Mode Seeking, and Clustering," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 790-799, 1995.

[16] D. Comaniciu and P. Meer, "Robust Analysis of Feature Spaces: Color Image Segmentation," presented at Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico, 1997.

[17] D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach Toward Feature Space Analysis," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 1-18, 2002.

[18] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 564-575, 2003.

[19] I. Craw, D. Tock, and A. Bennett, "Finding Face Features," presented at Proceedings of Second European Conference on Computer Vision, 1992.

[20] J. L. Crowley and J. M. Bedrune, "Integration and Control of Reactive Visual Processes," presented at Proceedings of Third European Conference on Computer Vision, 1994.

[21] J. L. Crowley and F. Berard, "Multi-Modal Tracking of Faces for Video Communications," presented at Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1997.

[22] R. Cutler and L. S. Davis, "Robust Real-Time Periodic Motion Detection, Analysis, and Applications," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 781-796, 2000.

[23] Y. Dai and Y. Nakano, "Extraction for Facial Images from Complex Background Using Color Information and SGLD Matrices," presented at Proceedings of First International Workshop on Automatic Face and Gesture Recognition, 1995.

[24] Y. Dai and Y. Nakano, "Face-Texture Model Based on SGLD and Its Application in Face Detection in a Color Scene," *Pattern Recognition*, vol. 29, pp. 1007-1017, 1996.

[25] G. Donato, M. S. Bartlett, J. C. Hager, P. Ekman, and T. J. Sejnowski, "Classifying Facial Actions," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 21, pp. 974-989, 1999.

[26] G. J. Edwards, C. J. Taylor, and T. F. Cootes, "Learning to Identify and Track Faces in Image Sequences," presented at Sixth International Conference on Computer Vision, 1998.

[27] O. Faugeras and R. Keriven, "Variational Principles, Surface Evolution, PDE's Level Set Methods, and the Stereo Problem," *IEEE Trans. on Image Processing*, vol. 7, pp. 336-344, 1998.

[28] O. D. Faugeras and Q. T. Luong, *Camera Self-Calibration: Theory and Experiments*, vol. 588: New York: Springer-Verlag, 1992.

[29] P. Fua and C. Miccio, "Animated Heads from Ordinary Images: A Least-Squares Approach," *Computer Vision and Image Understanding*, vol. 75, pp. 247-259, 1999.

[30] H. P. Graf, T. Chen, E. Petajan, and E. Cosatto, "Locating Faces and Facial Parts," presented at Proceedings of First International Workshop on Automatic Face and Gesture Recognition, 1995.

[31] H. P. Graf, E. Cosatto, D. Gibbon, M. Kocheisen, and E. Petajan, "Multimodal System for Locating Heads and Faces," presented at Proceedings of Second International Conference on Automatic Face and Gesture Recognition, 1996.

[32] H. Greenspan, J. Goldberger, and I. Eshet, "Mixture model for face-color modeling and segmentation," *Pattern Recognition Letters*, vol. 22, pp. 1525-1536, 2001.

[33] R. Hartley and P. Sturm, "Triangulation," *Computer Visual and Image Understanding*, vol. 68, pp. 146-157, 1997.

[34] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge University Press, 2000.

[35] I.-S. Hsieh, K.-C. Fan, and C. Lin, "A statistic approach to the detection of human faces in color nature scene," *Pattern Recognition*, vol. 35, pp. 1583-1596, 2002.

[36] R.-L. Hsu, M. Abdel-Mottaleb, and A. K. Jain, "Face detection in color images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 696-706, 2002.

[37] O. Ikeda, "Shape Reconstruction for Color Objects Using Segmentation and Photometric Stereo," presented at Proceedings of International Cconference on Image Processing, Singapore, 2004.

[38] T. S. Jebara and A. Pentland, "Parameterized Structure from Motion for 3D Adaptive Feedback Tracking of Faces," presented at Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 1997.

[39] T. S. Jebara, K. Russell, and A. Pentland, "Mixtures of Eigenfeatures for Real-Time Structure from Texture," presented at Proceedings of Sixth IEEE International Conference on Computer Vision, 1998.

[40] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi, "Robust Onine Appearance Models for Visual Tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1296-1311, 2003.

[41] H. Jin, S. Soatto, and A. J. Yezzi, "Variational Multiframe Stereo in the Presence of Specular Reflections," presented at Proceedings of 1st International Symposium 3D Data Processing Visualization and Transmission, Padova, Italy, 2002.

[42] S. B. Kang and M. Jones, "Appearance-Based Structure from Motion Using Linear Classes of 3-D Models," *Internation Journal of Computer Vision*, vol. 49, pp. 5-22, 2002.

[43] S. H. Kim, N. K. Kim, S. C. Ahn, and H. G. Kim, "Object Oriented Face Detection Using Range and Color Information," presented at Proceedings of Third International Conference on Automatic Face and Gesture Recognition, 1998.

[44] M. Kirby and L. Sirovich, "Application of the Karhunen-Loe've Procedure for the Characterization of Human Faces," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 103-108, 1990.

[45] R. Kjeldsen and J. Kender, "Finding Skin in Color Images," presented at Proceedings of Second International Conference on Automatic Face and Gesture Recognition, 1996.

[46] T. Kohonen, *Self-Organization and Associative Memory*: Springer, 1989.

[47] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao, and S. Russel, "Toward Robust Automatic Traffic Scene Analysis in Real-Time," presented

at Proceedings of International Conference on Pattern Recognition, Israel, 1994.

[48] K. Lam and H. Yan, "Fast Algorithm for Locating Head Boundaries," *Journal of Electronic Imageing*, vol. 3, pp. 351-359, 1994.

[49] K. M. Lam, "A Fast Approach for Detecting Human Faces in a Complex Background," presented at Proceedings fo the IEEE International Symposium on Circuits and Systems, 1998.

[50] A. Lanitis, C. J. Taylor, and T. F. Cootes, "An Automatic Face Identification System Using Flexible Appearane Models," *Image and Vision Computing*, vol. 13, pp. 393-401, 1995.

[51] T. K. Leung, M. C. Burl, and P. Perona, "Finding Faces in Cluttered Scenes Using Random Labeled Graph Matching," presented at Proceedings of Fifth IEEE International Conference on Computer Vision, 1995.

[52] A. J. Lipton, H. Fujiyoshi, and R. S. Patil, "Moving Target Classification and Tracking from Real-time Video," presented at Proceedings of IEEE Workshop on Applications of Computer Vision, 1998.

[53] C. Liu, "A Bayesian Discriminating Features Method for Face Detection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 725-740, 2003.

[54] D.-H. Liu, K.-M. Lam, and L.-S. Shen, "Optimal Sampling of Gabor Features for Face Recognition," *Pattern Recognition Letters*, vol. 25, pp. 267-276, 2004.

[55] Z. Liu, Z. Zhang, C. Jacobs, and M. Cohen, "Rapid Modeling of Animated Faces From Video," Microsoft Research, Redmond, WA MSR-TR-200-11, Feb. 28 2000.

[56] Z. Liu, Z. Zhang, C. Jacobs, and M. Cohen, "Rapid Modeling of Animated Faces From Video," Microsoft Corporation, Redmond, WA MSR-TR-2000-11, Feb. 28 2000.

[57] E. Loutas, I. Pitas, and C. Nikou, "Probabilistic Multiple Face Detection and Tracking Using Entropy Measures," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 14, pp. 128-135, 2004.

[58] Y. Lu, J. Z. Zhang, Q. M. J. Wu, and Z.-N. Li, "A Survey of Motion-Parallax-Based 3-D Reconstruction Algorithms," *IEEE Trans. on Systems, Man and Cybernetics, Part C: Applicaitons and Reviews*, vol. 34, pp. 532-548, 2004.

[59] J. Malik and S. Russel, "Traffic Surveillance and Detection Technology Development: New Traffic Sensor Technology," University of California, Berkeley, California PATH Research FInal Rep., UCB-ITS-PRR-97-6 1997.

[60] D. Marr, *Vision*: New Youk: W. H. Freeman, 1982.

[61] W. Martin and J. K. Aggarwal, "Volumetric Description of Objects from Multiple Views," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 5, pp. 150-158, 1983.

[62] A. M. Martinex and R. Benavente, "The AR face database," *CVC Technical Report #24*, 1998.

[63] S. J. Maybank and O. D. Faugeras, "A Theory of Self-Calibration of a Moving Camera," *Internation Journal of Computer Vision*, vol. 8, pp. 123-151, 1992.

[64] S. McKenna, S. Gong, and Y. Raja, "Modelling Facial Colour and Identity with Gaussian Mixtures," *Pattern Recognition*, vol. 31, pp. 1883-1892, 1998.

[65] S. McKenna, Y. Raja, and S. Gong, "Tracking Colour Objects Using Adaptive Mixture Models," *Image and Vision Computing*, vol. 17, pp. 223-229, 1998.

[66] Y. Miyake, H. Saitoh, H. Yaguchi, and N. Tsukada, "Facial Pattern Detection and Color Correction from Television Picture for Newspaper Printing," *Journal of Imaging Technology*, vol. 16, pp. 165-169, 1990.

[67] B. Moghaddam and A. Pentland, "Probabilistic Visual Learning for Object Representation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 696-710, 1997.

[68] A. Mohan, C. Papageorgiou, and T. Poggio, "Example-based object detection in images by components," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 349-361, 2001.

[69] T. K. Moon, "The Expectation-Maximization Algorithm," *IEEE Signal Processing Magazine*, pp. 47-60, 1996.

[70] H. P. Moravec, "Toward Automatic Visual Obstacle Avoidance," presented at Proceedings of International Joint Conference on Artifical Intelligence, 1977.

[71] N. Oliver, A. Pentland, and F. Berard, "LAFER: Lips and Face Real Time Tracker," presented at Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 1997.

[72] E. Osuna, R. Freund, and F. Girosi, "Training Support Vector Machines: An Application to Face Detection," presented at Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1997.

[73] C. Papageorgiou and T. Poggio, "A Trainable System for Object Detection," *International Journal of Computer Vision*, vol. 38, pp. 15-33, 2000.

[74] N. Paragios and R. Deriche, "Geodesic Active Contours and Level Sets for the Detection and Tracking of Moving Objects," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 266-280, 2000.

[75] P. J. Phillips, H. Moon, P. J. Rauss, and S. Rizvi, "The FERET Evaluation Methodology for Face Recognition Algorithms," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, 2000.

[76] P. J. Phillips and Y. Vardi, "Efficient Illumination Normalization of Facial Images," *Pattern Recognition Letters*, vol. 17, pp. 921-927, 1996.

[77] F. Pighin, J. Hecker, D. Lischinski, R. Szeliski, and D. H. Salesin, "Synthesizing Realistic Facial Expressions From Photographs," presented at Computer Graphics, Annual Conference Series, Siggraph, 1998.

[78] R. Polana and R. Nelson, "Low Level Recognition of Human Motion," presented at Proceedings of IEEE Workshop Motion of Non-Rigid and Articulated Objects, Austin, TX, 1994.

[79] R. Porter and N. Canagarajah, "Robust Rotation-Invariant Texture Classification," *IEE Proceedings on Vision, Image, and Signal Processing*, vol. 144, pp. 180-188, 1997.

[80] R. J. Qian, M. I. Sezan, and K. E. Matthews, "A Robust Real-Time Face Tracking Algorithm," presented at Proceedings of IEEE International Conference on Image Processing, 1998.

[81] L. R. Rabiner and B. H. Jung, *Fundamentals of Speech Recognition*: Prentice Hall, 1993.

[82] A. Rajagopalan, K. Kumar, J. Karlekar, R. Manivasakan, M. Patil, U. Desai, P. Poonacha, and S. Chaudhuri, "Finding Faces in Photographs," presented at Proceedings of Sixth IEEE International Conference on Computer Vision, 1998.

[83] L. Robert and R. Deriche, "Dense Depth Map Reconstruction: A Minimization and Regularization Approach which Perserves Discontinuities," presented at Proceedings of European Conference Computer Vision, Cambridge, U.K., 1996.

[84] L. Robert, R. Deriche, and O. Faugeras, "Dense Depth Recovery from Stereo Images," presented at Proceedings of European Conference on Artifical Intelligence, Vienna, Austria, 1992.

[85] H. A. Rowley, S. Baluja, and T. Kanade, "Neural Network-Based Face Detection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 22-38, 1998.

[86] E. Saber and A. M. Tekalp, "Frontal-View Face Detection and Facial Feature Extraction Using Color, Shape and Symmetry Based Cost Functions," *Pattern Recognition Letters*, vol. 17, pp. 669-680, 1998.

[87] T. Sakai, M. Nagao, and S. Fujibayashi, "Line Extraction and Pattern Detection in a Photograph," *Pattern Recognition*, vol. 1, pp. 233-248, 1969.

[88] F. S. Samaria, "Face Recognition Using Hidden Markov Models," *PhD thesis*, 1994.

[89] S. Satoh, Y. Nakamura, and T. Kanade, "Name-It: Naming and Detecting Faces in News Videos," *IEEE Multimedia*, vol. 6, pp. 22-35, 1999.

[90] D. Saxe and R. Foulds, "Toward Robust Skin Identification in Video Images," presented at Proceedings of Second International Conference on Automatic Face and Gesture Recognition, 1996.

[91] H. Schneiderman and T. Kanade, "Probabilistic Modeling of Local Appearance and Spatial Relationships for Object Recognition," presented at Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1998.

[92] H. Schneiderman and T. Kanade, "A Statistical Method for 3D Object Detection Applied to Faces and Cars," presented at Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 2000.

[93] S. Z. Selim and M. A. Ismail, "K-Means-Type Algorithms: A Generalized Convergence Theorem and Characterization of Local Optimality," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 81-86, 1984.

[94] K. Sengupta and P. Burman, "A Curve Fitting Problem and Its Application in Modeling Objects in Monocular Image Sequences," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 674-686, 2002.

[95] K. Sengupta and C. C. Ko, "Scanning Face Models With Desktop Cameras," *IEEE Trans. on Industrial Electronics*, vol. 48, pp. 904-912, 2001.

[96] Y. Shan, Z. Liu, and Z. Zhang, "Model-Based Bundle Adjustment with Application to Face Modeling," presented at Proceedings of International Conference on Computer Vision, 2001.

[97] L. S. Shapiro, A. Zisserman, and M. Brady, "3D Motion Recovery via Affine Epipolar Geometry," *Internation Journal of Computer Vision*, vol. 16, pp. 147-182, 1995.

[98] H. Y. Shum, M. Han, and R. Szeliski, "Interactive Construction of 3D Models from Panoramic Mosaics," presented at Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CA, 1998.

[99] T. Sim, S. Baker, and M. Bsat, "The CMU Pose, Illumination, and Expression Database," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1615-1618, 2003.

[100] G. Slabaugh, B. Culbertson, T. Malzbender, and R. Schafer, "A Survey of Methods for Volumetric Scene Reconstruction from Photographs," presented at International Workshop Volume Graphics, Stony Brook, NY, 2001.

[101] J. Sobottka and I. Pitas, "Segmentation and Tracking of Faces in Color Images," presented at Proceedings of Second International Conference on Automatic Face and Gesture Recognition, 1996.

[102] K. Sobottka and I. Pitas, "Face Localization and Feature Extraction Based on Shape and Color Information," Proceedings of IEEE International Conference on Image Processing, 1996.

[103] T. Starner and A. Pentland, "Real-Time ASL Recognition from Video Using HMM's," *Technical Report 375*, 1996.

[104] P. Sturm, "A Case Against Kruppa's Equations for Camera Self-Calibration," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1199-1204, 2000.

[105] Q. B. Sun, W. M. Huang, and J. K. Wu, "Face Detection Based on Color and Local Symmetry Information," presented at Proceedings of Third International Conference on Automatic Face and Gesture Recognition, 1998.

[106] K. K. Sung and T. Poggio, "Example-Based Learning for View-Based Human Face Detection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 39-51, 1998.

[107] K.-W. Sze, K.-M. Lam, and G. Qiu, "A New Key Frame Representation for Video Segment Retrieval," *accepted to appear in IEEE Transaction on Circuits and Systems for Video Technology*.

[108] A. Tefas, C. Kotropoulos, and I. Pitas, "Variants of dynamic link architecture based on mathematical morphology for frontal face authentication," presented at Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1998.

[109] J. C. Terrillon, M. David, and S. Akamatsu, "Automatic Detection of Human Faces in Natural Scene Images by Use of a Skin Color Model and Invariant Moments," presented at Proceedings of Third International Conference on Automatic Face and Gesture Recognition, 1998.

[110] J. C. Terrillon, M. David, and S. Akamatsu, "Detection of Human Faces in Complex Scene Images by Use of a Skin Color Model and Invariant Fourier-Mellin Moments," presented at Proceedings of International Conference on Pattern Recognition, 1998.

[111] T. Tian and C. Tomasi, "Comparision of Approaches to Egomotion Computation," presented at Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1996.

[112] M. Turk and A. Pentland, "Eigenfaces for Recognition," *Journal of Cognitive Neuroscience*, vol. 3, pp. 71-86, 1991.

[113] H. Wang and S. F. Chang, "A Highly Efficient System for Automatic Face Region Detection in MPEG Video," *IEEE Trans. on Circuits and System for Video Technology*, vol. 7, pp. 615-628, 1997.

[114] Y. Wang and B. Yuan, "A Novel Approach for Human Face Detection from Color Images Under Complex Background," *Pattern Recognition*, vol. 34, pp. 1983-1992, 2001.

[115] M. Werman and D. Weinshall, "Similarity and Affine Invariant Distances Between 2D Point Sets," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 810-814, 1995.

[116] L. Wiskott, J.-M. Fellous, N. Kruger, and C. v. d. Malsburg, "Face Recognition by Elastic Bunch Graph Matching," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 775-779, 1997.

[117] K.-W. Wong, K.-M. Lam, and W.-C. Siu, "An Efficient Algorithm for Human Face Detection and Facial Feature Extraction under Different Conditions," *Pattern Recognition*, vol. 34, pp. 1993-2004, 2001.

[118] K.-W. Wong, K.-M. Lam, and W.-C. Siu, "An Efficient Color Compensation Scheme for Skin Color Segmentation," presented at IEEE International Symposium on Circuits and systems (ISCAS2003), Bangkok, Thailand, 2003.

[119] K.-W. Wong, K.-M. Lam, and W.-C. Siu, "A Robust Scheme for Live Detection of Human Faces in Color Images," *Signal Processing: Image Communication*, vol. 18, pp. 103-114, 2003.

[120] B. Xie, D. Comaniciu, V. Ramesh, M. Simon, and T. Boult, "Component Fusion for Face Detection in the Presence of Heteroscedastic Noise," presented at Annual Conference of the German Society for Pattern Recognition (DAGM'03), Magdeburg, Germany, 2003.

[121] G. Yang and T. S. Huang, "Human Face Detection in Complex Background," *Pattern Recognition*, vol. 27, pp. 53-63, 1994.

[122] J. Yang, R. Stiefelhagen, U. Meier, and A. Waibel, "Visual Tracking for Multimodal Human Computer Interaction," *Proceedings of ACM Human Factors in Computer Systems Conference*, pp. 140-147, 1998.

[123] J. Yang and A. Waibel, "A Real-Time Face Tracker," *Proceedings of Third Workshop Applications of Computer Vision*, pp. 142-147, 1996.

[124] M.-H. Yang and N. Ahuja, "Detecting Human Face in Color Images," presented at Proceedings of IEEE International Conference on Image Processing, 1998.

[125] M.-H. Yang, D. J. Kriegman, and N. Ahuja, "Detection faces in images: a survey," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 34-58, 2002.

[126] K. C. Yow and R. Cipolla, "Feature-Based Human Face Detection," *Image and Vision Computing*, vol. 15, pp. 713-135, 1997.

[127] A. Yuille, P. Hallinan, and D. Cohen, "Feature Extraction from Faces Using Deformable Templates," *International Journal of Computer Vision*, vol. 8, pp. 99-111, 1992.

[128] Z. Zhang, "A Flexible New Technique for Camera Calibration," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1330-1334, 2000.

[129] Z. Zhang, Z. Liu, D. Adler, M. F. Cohen, E. Hanson, and Y. Shan, "Robust and Rapid Generation of Animated Faces From Video Images: A Model-Based Modeling Approach," Microsoft Corporation, Redmond, WA MSR-TR-2001-101, 2001.

[130] J. Y. Zheng, "Acquiring 3-D Models from Sequences of Contours," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 163-178, 1994.

[131] M. Zucchelli, "Optical Flow Based Structure from Motion," Ph.D. Thesis, Royal Institute of Technology, 2002.