MODEL-BASED 3-D FACE ANIMATION SYSTEM (LIP-SYNCHRONIZED) DESIGN FROM A VIDEO SOURCE

A Thesis Submitted
to the College of Graduate Studies and Research
in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in the Department of Electrical Engineering
University of Saskatchewan

by
Kyung-Yung Choi

Saskatoon, Saskatchewan, Canada
Fall 2002

© Copyright Kyung-Yung Choi, 2002. All rights reserved.
PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Postgraduate degree from the University of Saskatchewan, it is agreed that the Libraries of this University may make it freely available for inspection. Permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professors who supervised this thesis work or, in their absence, by the Head of the Department of Electrical Engineering or the Dean of the College of Graduate Studies and Research at the University of Saskatchewan. Any copying, publication, or use of this thesis, or parts thereof, for financial gain without the written permission of the author is strictly prohibited. Proper recognition shall be given to the author and to the University of Saskatchewan in any scholarly use which may be made of any material in this thesis.

Request for permission to copy or to make any other use of material in this thesis in whole or in part should be addressed to:

Head of the Department of Electrical Engineering
57 Campus Drive
University of Saskatchewan
Saskatoon, Saskatchewan, Canada
S7N 5A9
ACKNOWLEDGMENTS

I am deeply grateful to my supervisor, Professor Kunio Takaya, for his invaluable support and guidance.

I would like to express thanks to my former classmate, Young-Suk Yoo, for his helpful comments on JAVA.

I would like to give my special thanks to my parents and my wife Jong-Min whose patient love enabled me to complete this work.

I would like to thank all those who have cared me in Saskatoon.

This project was financially supported by Telecommunications Research Laboratories (TRLabs).
ABSTRACT

This thesis proposes a model-based 3-D talking head animation system and then constructs a simple 3-D face model and its animation by using Virtual Reality Modeling Language (VRML) 2.0 in conjunction with a VRML's Application Programming Interface (API), JAVA. The system extracts facial feature information from a digital video source. The face detection and facial feature extraction are prerequisite stages to track the key facial features throughout the video sequence. Face detection is done by using relevant facial information contained in the normalized $YCbCr$ color space. Independent Component Analysis (ICA) approach is applied to the localized facial images to identify major facial components of a face. Then, an image processing approach is deployed to extract and track the key facial features precisely. Streams of the extracted and determined facial feature parameters are transferred to the animation control points of the designed VRML 3-D facial model. Since the face model is defined in the 3-D space while a given video source is a 2-D presentation, some heuristic rules are embedded to estimate the coordinates of unmeasurable points for
the visually acceptable 3-D talking head model and animation. A standard video data set Miss America, QCIF formatted 30Hz, video is used as a test sequence, and the capability of the proposed system is verified and demonstrated.
# Table of Contents

PERMISSION TO USE ................................. i

ACKNOWLEDGMENTS ............................... ii

ABSTRACT ........................................ iii

TABLE OF CONTENTS ............................. v

LIST OF FIGURES .................................... viii

ABBREVIATIONS ................................ xi

1 INTRODUCTION ................................... 1

1.1 Overview of 3-D Face Animation ......................... 1

1.2 Statement of the Problem ............................ 3

1.3 Background ...................................... 4

1.3.1 Face Detection and Recognition ......................... 4

1.3.2 Methods of Face Detection .......................... 5

1.3.3 Methods of Face Recognition ......................... 7

1.3.4 Face Modeling and Animation ......................... 7

1.4 Approach to the Problem ............................ 9

1.5 Structure of Thesis ................................ 10

2 FACE LOCALIZATION ............................. 11

2.1 Color Discrimination ................................ 11

2.1.1 YCbCr Color Space ............................ 12
2.1.2 Image Subtraction and Color Analysis ........................................... 12

2.2 Other Processing ........................................................................ 15
  2.2.1 Connection and Opening ....................................................... 15
  2.2.2 Projection and Filling .............................................................. 15
  2.2.3 Shape Filtering ...................................................................... 16

3 FACIAL FEATURE EXTRACTION ....................................................... 19
  3.1 Principal Component Analysis (PCA) ........................................... 19
  3.2 Independent Component Analysis (ICA) ...................................... 22
    3.2.1 Statistical Independence .................................................... 23
    3.2.2 ICA Concept .................................................................... 24
  3.3 ICA Representation of a Face .................................................... 28
    3.3.1 Finding Region of Interest with ICA ................................... 31
  3.4 Mouth Corner Points Extraction ............................................... 32
    3.4.1 Prewitt Filtering using Empirical Approaches ....................... 33

4 FACE ANIMATION WITH VRML ....................................................... 37
  4.1 An Example of a VRML Scene .................................................. 37
  4.2 Overview of Face Animation System ....................................... 40
  4.3 Generating a Facial Model ....................................................... 41
    4.3.1 Modeling Facial Components ............................................. 41
    4.3.2 Modeling Skin ................................................................. 43
    4.3.3 Heuristic Rules .................................................................. 43
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>FAPs in MPEG-4 [27].</td>
<td>3</td>
</tr>
<tr>
<td>2.1</td>
<td>Skin color distribution in $nYCbCr$ color space. Both axis are scaled in $[0, 1]$.</td>
<td>13</td>
</tr>
<tr>
<td>2.2</td>
<td>The two sequential frames with an interval of 2, from which a subtracted image is calculated.</td>
<td>13</td>
</tr>
<tr>
<td>2.3</td>
<td>Color discrimination process.</td>
<td>14</td>
</tr>
<tr>
<td>2.4</td>
<td>The skin color density map (a), threshold holding (b), and size filter with considering value of pixel (c).</td>
<td>15</td>
</tr>
<tr>
<td>2.5</td>
<td>Connection (a) and opening (b).</td>
<td>16</td>
</tr>
<tr>
<td>2.6</td>
<td>Projection and row filling.</td>
<td>16</td>
</tr>
<tr>
<td>2.7</td>
<td>Ellipse model.</td>
<td>17</td>
</tr>
<tr>
<td>2.8</td>
<td>Shape filtering.</td>
<td>18</td>
</tr>
<tr>
<td>3.1</td>
<td>2-D data set and a PCA basis: The major direction of given data is found on the solid line by PCA and minor in the dotted line.</td>
<td>20</td>
</tr>
<tr>
<td>3.2</td>
<td>Whitening process: (a) independent source, (b) observed, and (c) whitened data.</td>
<td>26</td>
</tr>
<tr>
<td>3.3</td>
<td>First 21 ICA bases of facial images in the test video sequence are shown</td>
<td>30</td>
</tr>
<tr>
<td>3.4</td>
<td>Chosen mouth like ICA bases.</td>
<td>31</td>
</tr>
<tr>
<td>3.5</td>
<td>Extracted approximate mouth area of the test images.</td>
<td>32</td>
</tr>
</tbody>
</table>
3.6 Luminance image (a), Prewitt filtered image (b), intersections for finding corners (c). ................................................. 34
3.7 Corrected luminance image (a), Prewitt filtered image (b), intersections for finding corners (c). ............................... 35
3.8 Corrected luminance image (a), Prewitt filtered image (b), intersections for finding corners (c). ............................... 35
3.9 Test images (top row) and the results of exact mouth (bottom). ... 36
4.1 An example of a VRML scene. ......................................... 40
4.2 Frontal and profile view of bottom half of a simple wire frame talking head model, constructed by IndexedFaceSet and Extrusion node for skin surface and lip model, respectively, is shown. ... 43
4.3 The outer line (solid line) of upper and lower lip is shown in X-Y coordinates along the paths of the extruded shape (dotted line). ... 44
4.4 Jaw rotation according to the bottom lip angle point. .......... 46
4.5 Data loading with JAVA API of VRML. .............................. 48
4.6 Interaction among various nodes used in the animation of a talking head model and animation with VRML. ................. 48
5.1 Test video sequence, Miss America: Images are ordered from left to right and top to bottom ................................. 51
5.2 Skin colored pixels are shown after applying skin color density map. ... 52
5.3 Images after connect operation followed by opening process. .... 53
5.4 Projection and filling are performed on Figure 5.3. ............... 54
5.5 Normalized final face area after applying the shape filtering. .... 55
5.6 ICA bases. 3, 6, 14, 15, 16, 18, 23, 30, and 35-th ICA bases are selected as mouth-like features. ICA bases are ordered from left to right and from top to bottom. ........................................... 57

5.7 Approximate mouth areas of Figure 5.5 and results of mouth corner extraction are shown. ................................................................. 58

5.8 Finding horizontal corners with Prewitt filter and a heuristic approach. 59

5.9 Finding vertical corners with Prewitt filter and a heuristic approach. 60

5.10 VRML face model and its components. ........................................ 62

5.11 Comparison between the bottom half of the odd numbered facial images of test sequence and corresponding VRML animation frames. .. 63
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-D</td>
<td>Three Dimensional</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>BIFS</td>
<td>Binary Format for Scenes</td>
</tr>
<tr>
<td>DCT</td>
<td>Discrete Cosine Transform</td>
</tr>
<tr>
<td>FAP</td>
<td>Face Animation Parameter</td>
</tr>
<tr>
<td>Hz</td>
<td>Hertz (1/second)</td>
</tr>
<tr>
<td>ICA</td>
<td>Independent Component Analysis</td>
</tr>
<tr>
<td>JPEG</td>
<td>Joint Photographic Experts Group</td>
</tr>
<tr>
<td>Kbps</td>
<td>Kilo bits per second</td>
</tr>
<tr>
<td>MPEG</td>
<td>Motion Picture Expert Group</td>
</tr>
<tr>
<td>PCA</td>
<td>Principal Component Analysis</td>
</tr>
<tr>
<td>QCIF</td>
<td>Quarter Common Interchange Format</td>
</tr>
<tr>
<td>RGB</td>
<td>Red Green Blue</td>
</tr>
<tr>
<td>TTS</td>
<td>Text To Speech</td>
</tr>
<tr>
<td>VR</td>
<td>Virtual Reality</td>
</tr>
<tr>
<td>VRML</td>
<td>Virtual Reality Modeling Language</td>
</tr>
</tbody>
</table>
1. INTRODUCTION

1.1 Overview of 3-D Face Animation

Along with an increasing need of human-machine interface in multimedia applications, the interactive communication with an avatar has drawn considerable attention. This is an attractive approach to bridge the gap between human and machine, and becoming an important research subject.

An avatar is a replica of a person who wants to be immersed in the virtual reality (VR) environment, and audio/visual information associated with avatars is accessible to the remote users when the Internet application is combined. A typical application which utilizes this concept is a multi-lingual TV broadcaster avatar. When an avatar is integrated with a text-to-speech (TTS) synthesizer for a number of languages, the avatar generates mouth and jaw motions of the corresponding phonemes to each language. Characters of the avatars can be shared through this means.

Currently, avatars can be designed mainly in two ways; hardware-based and model-based approach. The former can generate a very realistic avatar using special 3-D equipment such as a laser-scanner. The appearance of this type of avatar is a more clone-like image and people feel that it is more trustworthy. However, due to the complexity and large volume of the model driven by this method, it is a difficult task to implement in real time applications. On the other hand, the latter generates a synthesized speaker-independent model with given rules and/or extracted features from image sources. The movements of avatars can be generated efficiently if the condensed and meaningful information is obtained from the image sources. When the channel capacity is considered, this attribute is very valuable.

1In the Hindu religion, an avatar is an incarnation of a deity. Here, we see it as an interactive representation of a human in a virtual reality environment.
MPEG-4 (Motion Picture Expert Group) is the first international multimedia standard. The object-based audio/visual representation capability for synthetic/natural contents mainly distinguishes MPEG-4 from MPEG-1 and MPEG-2, previous versions of audio/visual standards. An object-based scene can be composed by synthetic/natural independent objects. As previously mentioned, enhanced broadcasting with an avatar in a scene can be an example. The synthetic background, the natural audio object, voices, and a synthetic visual object, an avatar, are coded independently and construct the object-based scene. With the MPEG-4 object-based technique, new and possibly more rich applications can be developed for enhanced broadcasting, more efficient visual presentation on mobile systems, a customized character in computer games, etc.

MPEG-4 defines facial feature points on the neutral state of a face model for the face animation. The values of feature points of a face model indicate how much each face action unit deforms from the neutral state of the face. Extracted or generated facial feature points from an image source with rules represent changes in facial expression. A series of values of feature points at each instant of time generates the animation sequence.

Figure 1.1 provides the spatial reference of 84 Face Animation Parameter (FAP) points in the neutral state of a face model defined by MPEG-4. Feature points are grouped to each unit composing of a face including parts, such as a nose, eyes, a mouth, and so forth. These separated units should be properly coordinated on a face model by taking into consideration the spatial relationship of one another.

MPEG-4 standardizes a limited number of FAP in order to perform the face animation efficiently through the low bit-rate channels where the bandwidth is highly limited or the broad bandwidth is not guaranteed. From a communication aspect, the encoder can use MPEG-4 BInary Format for Scenes\(^2\) (BIFS) to define the face model, and it is transmitted to the decoder. The extracted animation parameters are encoded

\(^2\)BIFS is the compressed format in which scenes are defined and modified in MPEG-4. This has been designed as an extension of the VRML 2.0 specification in a binary form.
using an arithmetic encoder and a Discrete Cosine Transform (DCT). The decoder receives the face model and the animation parameters separately. The face model is reconstructed on the receiver’s side. Then, animation parameters are mapped to the corresponding points. This approach requires only a very small amount of transfer time without sacrificing the resolution of the images. Many research results related to face modeling and animation have been reported with different techniques [5] [10] [12] [24] [28] [32] [36].

![Figure 1.1 FAPs in MPEG-4](image)

**1.2 Statement of the Problem**

The objective of this research is to design a prototype of the model-based 3-D face animation system by extracted facial information from a digital video source. The system should integrate face detection, facial feature extraction, 3-D face modeling, and animation stages. A video source containing images of a speaker can provide
us with natural face expressions when a face and its facial features are localized and extracted properly using the image analysis techniques. The stream of the extracted facial feature points must then be transformed to the animation control parameters that define a 3-D facial model.

The objective of the research presented is to generate a 3-D face animation on the Internet by means of a model-based scheme, which is efficient in transfer rate and can be accomplished at a low cost. The model-based face animation is especially a valuable asset of MPEG-4 face animation, which allows us to have convincing facial expressions through a very low channel capacity by only transmitting FAPs instead of sending whole image pixels. Theoretically, it is enough to have a 2-4 Kilo Byte/Second (KB/S) transfer rate to transmit compressed FAPs and to present a smooth face animation. Also the Internet compatible 3-D graphic platform should be inexpensive when such a model-based scheme as MPEG-4 is supported.

From the obvious reason that a facial image in a video source is 2-D whereas the target object is a 3-D face model, any missing information on the Z-coordinate should be generated in a certain way. The heuristic rules are embedded in the presented system to construct 3-D wire frame face model from the 2-D plane by assigning the Z-coordinate of unmeasurable points based on the X-Y coordinates of relevant and linked feature points.

1.3 Background

This section provides a review of some existing techniques capable of composing a face animation. In general, a face animation system consists of: detection, facial feature extraction, face modeling, and animation stages; these are hierarchically linked. The following subsections describe each of the previous stages studied.

1.3.1 Face Detection and Recognition

The human-machine interface technique by recognizing facial images is getting more important in various fields such as: advanced surveillance, desk-top user log-in,
database of portrait photographs, and authentication by face in security systems. Although a number of researches have been intensely conducted during the last couple of decades, face detection is still a challenge for engineers as well as psychophysiologists, who try to understand how the brain reacts on facial expressions. In order to have automatic face detection and recognition, we need to overcome the following apparent problems.

1. The face is a complex model to describe completely.

2. A certain exaggerated expression changes the shape of the face dramatically.

3. There are chances of occlusions by objects such as a hand, a book, wearing (sun)glasses or a hat, mustache or beard, and different hair length.

4. Unknown 2-D orientation and 3-D rotation of a face has to be considered.

5. The shape of a face may be appeared differently due to varying lighting conditions.

Due to these problems, we define face detection as a task to identify faces in the scene, and face recognition as a task to extract authentic facial features representing an individual from a given image source. Face detection can be simplified to the face localization problem when only one face is given in the image. There are several ways to detect and recognize faces. The following subsections review those methods.

1.3.2 Methods of Face Detection

Various approaches have been reported and mainly categorized into several parts based on knowledge [20], template [14] [18] [21] [44], features [6] [7] [26] [38] [41], and appearance [11] [22] [33] of still image or a video sequence. Often face detection, regarded as a prerequisite stage for face recognition, is integrated as part of facial feature extraction techniques. A comprehensive survey for the face detection is presented by Yang et al [43].
In the knowledge-based detection approaches, a rule is designed based on our intuition of detecting a face. The spatial relation between facial features usually makes a logic such that nose is placed between eyes and mouth. After finding all the features of a given image, the logic is applied to verify whether the face or faces are in the scene. Rule-based approach is quite an efficient method only when a rule is properly designed for the restricted conditions. However, in the real world solution, there are chances a logic omits certain events, and sometimes it is not an easy task to translate our intuition for designing an effective set of rules.

The most widely used method in the appearance-based approach would be a neural network. This approach uses a moving window that scans over the entire image, then determines whether the image in the moving window belongs to a face or non-face class. Classification is achieved by numerous numbers of training face and non-face image sets. One problem with this method is the difficulty of defining non-face image sets. To avoid this problem, the bootstrap system is applied. It is, initially, exposed to a small number of manually chosen non-face training image sets to train the system partially. During the training, the randomly chosen non-face images from a certain (some part similar to a face) scene are used. If the system falsely detects some of those images, we add those images to the non-face class and train the system again. This method has been widely used in the neural network system. In general, such a neural network based detection system consists of several levels, the coarse-to-fine decision layers are processed hierarchically. Although this method is computationally expensive compared to other approaches, a very high detection rate is reported especially for up-right frontal faces.

In fact, the combined information from the various facial features tends to provide us with more reliable decisions than using only one specific feature. Methods combining the properties of the various facial features has become more popular [6] [7] [19] [21] [26] [33]. Usually this approach takes advantage of the color information. Color is a useful feature allowing human skin to be easily captured by a video camera. Although a number of color subspaces have been developed to distinguish skin color
from others as well as to find better representation of human skin by showing skin color falling to small concentrated portion of the color space, it is generally impossible to detect the exact face region using the color feature alone. Therefore, it is logical to combine the color feature with other features such as shape, motion information in video scene, and hair. This approach can screen the face candidates detected by other features by verifying if the face color falls into selected skin color ranges.

1.3.3 Methods of Face Recognition

Face recognition can be defined as a task of identifying one or more persons in a given scene using a stored database of faces. Generally, face recognition schemes extract the information relevant to a face from training facial image sets. The information of each training facial image is compared to the test image.

Among the many face recognition techniques, the eigenface approach is most widely used by many researchers [4] [8] [19] [23] [25] [39]. The eigenface can be obtained by the Principal Component Analysis (PCA), which calculates the eigen values and eigen vectors from the covariance matrix of training facial images. Then, the eigenface corresponds only to larger eigen values, which are significant among the eigen values. All training facial images can be projected onto a subspace composed of major eigen vectors. The linear combination of those eigen vectors and the distance vector of a test facial image from that of an average face of training images generates a coefficient vector. Also, the coefficient vector of a new face can be compared with those from the test images to determine how closely the test face matches the face in the training images. This approach has a computational advantage because of its reduced dimensionality.

1.3.4 Face Modeling and Animation

Several approaches for synthetic face modeling were considered when face animation was built into MPEG-4. Thalmann and coworkers [10] presented a method of using two facial photographs, a frontal and a profile view. Features extracted from the
pictures are fit into the well-defined generic face model by edge filtering and matching algorithms. This approach performs well enough to construct from cartoon characters to near picture quality cloned models.

Another popular approach is using 3-D equipment such as a laser-scanner and multiple cameras. In the former approach, the output of the laser-scanned data combined with the color information of scanned surface generates an accurate and natural head model. In the stereo image approach using multiple cameras, the measured distances from the object to the cameras are used to determine the 3-D surface structure. Then, based on the locations of the feature points, the previously designed generic face model is deformed to match the 3-D surface transformed by the stereo images. The depth of an image and the intuition of a viewer from a distance are well presented in these techniques. However, these approaches are inappropriate for an ordinary video image taken by a single camera.

The use of professional computer graphic software such as 3D Studio MAX, Maya, Softimage, or Lightwave 3D is the other main stream. Spline modeling and box molding methods are widely used and can provide us with a sophisticated model. Similar to the 3-D hardware method, it has drawbacks in the cost of the expensive software package and the vast amount of time it consumes even for a trained designer. The amount of file size of a complex head model as well as animation parameters is not suitable where channel capacity is limited, although high compression techniques are being developed.

Model-based synthetic face modeling and animation is an effective approach especially in a video conference or generating avatars for the agents in the web scenes. MPEG-4 uses this model-based approach and specifies the face animation parameters (FAPs). One possible platform to implement the efficient yet inexpensive 3-D environment can be Virtual Reality Modeling Language (VRML), which is one of most widely used platforms on the Internet to describe 3-D objects. More about the properties of VRML and suggested VRML face modeling and animation will be discussed in chapter 4.
1.4 Approach to the Problem

In this thesis, we address the system capable of generating the model-based 3-D talking head animation from a digital video source, as well as performing the animation on the Internet. The presented system consists of three stages including face localization, facial feature extraction, and VRML face modeling and animation.

For the face localization stage, color information is mainly used. We search the portion of human skin color in a normalized $YCbCr$ color space in conjunction with the motion information from a subtracted image under the constrained conditions. Other relevant information such as size, shape, and geometrical features are combined to finalize a face candidate.

After locating faces in the video source, we need to extract the major facial features. Independent Component Analysis (ICA) processing is involved for this task. From the series of located facial images, ICA separates statistically independent signals that can be regarded as the major components of a face in this specific application. Thus, we find the approximate location of the major facial components with the chosen ICA bases set and their associated coefficients. Prewitt filtered images and an empirical concept are addressed to have a reliable and robust performance for extracting exact feature points from the area given by the ICA approach.

The parameters of these extracted feature points are then mapped to the control points of the animation unit of the VRML face model. The animation sequence will be generated without any special 3-D software packages or equipment. JAVA programming language with help of VRML's Application Programming Interfaces (APIs) enables a VRML file to access animation parameter data files located outside the VRML. Heuristic rules assigning unmeasurable points based on the coordinates of feature points are embedded to supply any missing information for the visually acceptable 3-D face wire frame from the 2-D video sequence.
1.5 Structure of Thesis

This thesis is organized into six chapters.

Chapter 1 provides background information on face animation systems and a literature review of the current state of the techniques. The objective of this thesis and a brief description of the approaches to the problem are also presented.

Chapter 2 provides methods for face localization. Color discrimination technique in conjunction with motion information is presented. A number of image processing techniques and operators for finalizing a face candidate are addressed.

Chapter 3 describes methods for facial feature extraction. This chapter observes the PCA processing, followed by the ICA concept. The properties of ICA is examined and how ICA contributes for extracting major facial components is shown. Other image analysis techniques involved are presented here to place the key facial feature points from the results of ICA approach.

Chapter 4 introduces the 3-D graphic platform VRML. Detailed descriptions of VRML face modeling and heuristic rules for missing information are presented. The method of VRML animation with JAVA API is shown.

Chapter 5 provides the experimental configurations and results of the test video source for each stage of the methods described. A discussion of the results is given.

Chapter 6 concludes this research, and future work is suggested to enhance system performance.
2. FACE LOCALIZATION

Face localization or detection is as important as the facial feature extraction stage. The former is the prerequisite stage for the feature extraction, and the well designed face detection system requires the less complicated methods for face recognition. Eventually, the accurate facial information can be obtained.

The method used here for face localization has the following constrained conditions: There is one up-right frontal face shown with fixed background throughout a color video sequence. Although this condition excludes many video clips, the cases of face telecasting or storytelling satisfy the condition. Under this constrained condition, the presented face localization system is mainly divided into two stages, human skin color discrimination and shape filtering. In the first stage, human skin color filtering is applied to the subtracted image showing motion. The extracted skin color region is, then, evaluated to find the most dominant image cluster, which is to be a face. In the second stage, the elliptical shape filtering is applied to find the exact face region from the dominant image cluster. The following sections below describe in detail each of the stages.

2.1 Color Discrimination

Color discrimination has some drawbacks in accommodating varying luminance condition, background color close to human skin, and different facial colors. However, it is still a favorable method because of its computational efficiency and robustness against the orientation of a face. Due to the dense distribution in a small area of the chrominance space, human skin color has been proved to be an effective feature for the face detection especially when it is combined with the motion information in a video sequence. It has also been adopted in most of the face detection systems at the
front end in conjunction with other attributes of a face. A number of color spaces have been evaluated including HSV [9] [38], YES [34], YCbCr [9] [26], YIQ [41], CIE XYZ [42] and other color spaces [35].

2.1.1 YCbCr Color Space

YCbCr color space was developed for digital video standards and is used by most image compression standards (JPEG, H.261, MPEG). RGB color space is separated into a luminance part, Y, and two chrominance parts, Cb and Cr, corresponding to blue and red components respectively. YCbCr color space takes advantage over RGB in both storage size and bandwidth. The human eye is, in fact, more sensitive to brightness changes than that of color. Therefore, the transmission of the chrominance parts is not important as the luminance part. The YCbCr color space is taking advantage of the compression ratio by the factor of 2 over RGB color space. The Eq. (2.1) shows the transformation from RGB to YCbCr color space.

\[
\begin{align*}
Y &= 0.257 \cdot R + 0.504 \cdot G + 0.098 \cdot B + 16 \\
Cb &= -0.148 \cdot R - 0.291 \cdot G + 0.439 \cdot B + 128 \\
Cr &= 0.439 \cdot R - 0.368 \cdot G - 0.071 \cdot B + 128
\end{align*}
\] (2.1)

2.1.2 Image Subtraction and Color Analysis

The image subtraction avoids the false detection from the skin like background color and reduces the search space as the skin color filtering can be applied only to the moved pixels in the normalized YCbCr color space. The YCbCr color space has been chosen because it is not only standard digital video coding but has a compact skin color distribution in a chrominance plane. In YCbCr color space, the luminance component alone does not provide useful information for finding skin color regions. However, it helps us to extract the head-only region in some cases. For example, the shade area right below the chin can be removed with the luminance information combined with chrominance components, see Figure 2.3 (c). This combination of luminance and chrominance, in other words the normalized YCbCr color space, finds
the head position to cut out a face alone image. In Figure 2.1, skin color patches are shown for those manually collected from both indoor and outdoor video scenes. Most of the skin color pixels are fallen into a certain region of the normalized YCbCr color space.

\[
\begin{align*}
 nCb &= \frac{Cb}{Y + Cb + Cr} \quad \text{(2.2)} \\
 nCr &= \frac{Cr}{Y + Cb + Cr} \quad \text{(2.3)}
\end{align*}
\]

**Figure 2.1** Skin color distribution in nYCbCr color space. Both axis are scaled in [0, 1].

**Figure 2.2** The two sequential frames with an interval of 2, from which a subtracted image is calculated.

The image subtraction process is carried out with respect to the luminance component for two different images taken two frames apart. An example is shown in
Figure 2.3 (a). After normalizing a subtracted image and thresholding, the binary image is obtained by assigning a 1 to moved pixel while a 0 is otherwise assigned as shown in Figure 2.3 (b).

The moved pixels that fall in the facial skin color defined within small portion of $nYCbCr$ color space are extracted as shown in Figure 2.3 (c). Although, the facial region represented by facial skin colored pixels is detected, some unnecessary parts, such as the hair portion, are also detected. These are removed by a skin color density map of skin color pixels detected in Figure 2.3 (c).

![Figure 2.3 Color discrimination process.](image)

The skin color density map is shown in Figure 2.4 (a). In this map, the concentrated portion found in the middle is regarded as the facial skin color. For extracting the concentrated middle portion, a threshold value is applied to the color density map, Figure 2.4 (b). A size filter is, then, applied in order to reject the isolated clusters. Then, if the surrounding area of a given pixel is less than a threshold, this pixel is discarded. After these processes, a finalized skin color map is obtained from Figure 2.4 (c). This skin color map is applied to moved pixels in Figure 2.3 (c). All frames are examined to see whether moved pixels belong to the facial skin color map.
Figure 2.4 The skin color density map (a), threshold holding (b), and size filter with considering value of pixel (c).

The result of applying the skin color map to Figure 2.4 (c) is shown in Figure 2.3 (d).

2.2 Other Processing

2.2.1 Connection and Opening

In Figure 2.3 (d), there are many small holes in the skin color clusters which are considered as a skin part. Most of these holes are not detected as moved pixels in the subtraction process. These holes need to be filled for the subsequent shape filtering process, which is described later. A connection operator is used to fill such holes. This operator is to connect the pixels if a hole is smaller than a 5 x 5 pixel size. The result is shown in Figure 2.5. The opening operator [31], the erosion followed by the dilation, is performed on the connected image. This removes thin lines and small objects. In the face localization, it is found useful to discard possible small skin-like color clusters which may occur due to the light-reflected hair or clothes. We can, thus, obtain the localized face based on the skin color.

2.2.2 Projection and Filling

After finalizing the opening process, the projection is applied to eliminate other unwanted skin areas. By summing up the pixels of a face candidate column and row
wise, the distribution of clusters can be found, as shown in Figure 2.6 (a). The index of columns or rows of the longest cluster are chosen at each step. A finalized face area is shown in Figure 2.6 (b). Small holes in the finalized face area need to be closed for the next process. Filling the holes provides a better fit ellipse model, which will be explained later. Each row of the final face, obtained by projection process, is examined. If a hole, the zero valued pixel(s) in the skin color part, is found and has relatively small length compared to that of the skin color part, the hole is filled. A result is shown in Figure 2.6 (c).

2.2.3 Shape Filtering

Under the assumption that the frontal facial images are of oval shape, we extract the face only area within the finalized skin color area where other skin parts may still exist. The upper vertex of an ellipse is placed to the top and middle of the
skin colored region. The horizontal length of an ellipse is the same as that of a face candidate. Initially, the ellipse is set as a circle. Keeping this center of the ellipse fixed, this major axis of this ellipse is gradually enlarged downward to the bottom of skin color region. The equation of an ellipse is given by

$$\frac{(x-x_c)^2}{a^2} + \frac{(y-y_c)^2}{b^2} = 1$$

Figure 2.7 Ellipse model.

where $x_c$ and $y_c$ are the centers of the minor and major axes of the ellipse with $a \leq b$. The area of the ellipse is $A_E = \pi a c$ where $c$ is incrementally increased by $c = a, a + 1, \ldots, b$. During the enlarging process, the ratio of the area of the skin colored pixel (binary valued) below the minor axis to that of the bottom half of an ellipse is calculated.

For shape filtering, the area of the bottom half of an ellipse is chosen, because the upper half is often occluded by different hairstyles; the upper half of an ellipse does not properly fit the face candidate. Ideally, the largest ratio gives the best fit of an ellipse to the skin color region. However, in some cases, the ratio tends to have the
largest value when the ellipse is a circle placed inside of a face. As a result, we may have a face only region that excludes the chin or mouth area. Proper facial feature extraction will not result in this case. Therefore, the algorithm is adjusted in such a way that we choose the maximum ratio when a ratio does not reach a certain value, $T_A$. Otherwise, we choose the ratio whose value is the closest to $T_A$. The facial image is, then, normalized (resized) to a pre-assigned size using the linear interpolation for the subsequent steps of ICA analysis. Figure 2.8 shows an example of the shape filtering process. A circle is gradually enlarged and it becomes an ellipse that touches the end of the bottom row of skin color pixel. As shown in Figure 2.8, shape filtering is useful when the neck lines remain thick enough after the opening process.

Figure 2.8  Shape filtering.
3. FACIAL FEATURE EXTRACTION

This chapter describes two main concepts applied to the facial feature extraction stage: the Principal Component Analysis (PCA) and the Independent Component Analysis (ICA). And the key feature point extraction method is also described based on Prewitt filtering in conjunction with an empirical approach which is found later in this chapter. In the first section, a short review of the PCA is presented including a basic concept of data presentation with reduced dimensionality followed by ICA section describing the fundamentals and algorithm. In the third section, we show how to employ the ICA technique for the facial images located in the video sequence. In the final section, within the approximate region of the facial component found by the ICA, the positions of mouth corners are extracted by the empirical analysis of the Prewitt filtered images. Some preliminary results are shown.

3.1 Principal Component Analysis (PCA)

The PCA is a classical approach that takes dominant information of data and reconstructs the original data with reduced dimensionality in terms of minimizing reconstruction error or maintaining major properties. In the feature extraction the dimension of the original data, \( n \), will be reduced to an effective number of features, \( m \ll n \) especially when \( n \) is very large. Many sampled data in 2-D space \( (n = 2) \) are shown in Figure 3.1 and this data can be considered to have one major direction \( (m = 1) \) and one minor direction. The direction of the maximum deviation, presented by the solid line, can be obtained by the PCA approach. In this subsection, we observe the background of the PCA algorithm and the concept of data reduction.

Let us consider a vector \( \mathbf{x} = [x_1, x_2, \ldots, x_n]^T, \mathbf{x} \in \mathbb{R}^n \), representing the data set with zero mean, \( E\{\mathbf{x}\} = 0 \). We shall find the information that presents the major
properties of the given data with a reduced order, $m \ll n$. It is shown in [13] that the elements of $x$ are observed most independently when the data set is projected to the eigen vectors of the covariance matrix $C \in \mathbb{R}^{n \times n}$ of the given data set.

$$C = E\{xx^T\}$$  \hspace{1cm} (3.1)

$$Cv_i = \lambda_i v_i, \quad i = 1, 2, \ldots, n$$  \hspace{1cm} (3.2)

And $n$ eigen vectors, $v_1, v_2, \ldots, v_n$, are corresponding to their eigen values, $\lambda_1, \lambda_2, \ldots, \lambda_n$. Eigen values and eigen vectors are found by Eq. (3.2) and they can be written in matrix form,

$$CV = VD$$  \hspace{1cm} (3.3)

and

$$C = VDV^T$$  \hspace{1cm} (3.4)

where $D = diag(\lambda_1, \lambda_2, \ldots, \lambda_n)$ and $V = [v_1, v_2, \ldots, v_n]$. Before discussing the data dimension reduction, the data presentation needs to be considered. It is the projection of data $x$ onto the eigen vector, $v_i$, which is a scalar.

$$z_i = v_i^T x = x^T v_i, \quad i = 1, 2, \ldots, n$$  \hspace{1cm} (3.5)
And Eq. (3.5) can be presented in matrix form,

$$z = V^T x$$

(3.6)

where $z = [z_1, z_2, \cdots, z_n]^T$. Since the eigen vector matrix $V$ of real symmetric matrix $C$ is orthogonal, the original data set $x$ can be reconstructed as following.

$$x = Vz = \sum_{i=1}^{n} z_i v_i$$

(3.7)

As we mentioned earlier, the PCA provides a good estimation of the data with low dimensionality or equivalently, we have effective data representation with less number of features. This data dimensionality reduction can be made by omitting small variance components in Eq. (3.7). By proceeding in this way, we discard the directions in which the data set has less significant amount of energy. First, we arrange the eigen values in descending order.

$$\lambda_1 > \lambda_2 > \cdots > \lambda_n$$

(3.8)

Eigen vector matrix is arranged by its corresponding eigen values. Let us represent the approximate data,

$$\hat{x} = \sum_{j=1}^{m} z_j v_j, \quad m < n$$

(3.9)

then the approximate error vector is

$$e = x - \hat{x} = \sum_{k=m+1}^{n} z_k v_k$$

(3.10)

Here, from the face that $v_i^T v_j = 0$ where $i \neq j$, the inner product of the approximate error and approximate data is zero, or $\hat{x}$ and $e$ are orthogonal.

$$e^T \hat{x} = \sum_{k=m+1}^{n} z_k v_k^T \sum_{j=1}^{m} z_j v_j = \sum_{k=m+1}^{n} \sum_{j=1}^{m} z_k z_j v_k^T v_j = 0$$

(3.11)

From Eq. (3.11), we conclude that when the approximation error vector, $e$, has a smaller variance or magnitude, the reconstructed data with less dimensionality, $\hat{x}$, approaches the original data vector, $x$. Total variance can be obtained from Eq. (3.3)
and the followings.

\[
\sigma^2 = E\{zz^T\} \\
= E\{(V^Tx)(x^TV)\} \\
= V^TE\{xx^T\}V \\
= V^TCV \\
= V^TVDV^TV \\
= D
\]

(3.12)

where, \(\sigma = \text{diag}(\sigma_1, \sigma_2, \cdots, \sigma_n)\). Therefore the total variance is found by

\[
\sum_{i=1}^{n} \sigma_i^2 = \sum_{i=1}^{n} \lambda_i
\]

(3.13)

The dimensionality reduction, with a sense of reconstruction error minimizing, can be achieved by discarding zero or small valued eigen values and its corresponding eigen vectors. The PCA approach has been widely applied in various fields such as: image compression, noise canceling, and data visualization. In this thesis, \(V\) is used at the whitening stage of the ICA when applied to face images in the video sequence. \(x\) is the concatenation of the column vectors in each frame and 60 frame are taken, \(n = 60\), which is reduced to \(m = 45\) for further use as explained in section 3.3.

### 3.2 Independent Component Analysis (ICA)

Similar to the PCA approach, the objective of the ICA [3] [14] [15] [16] [17] is to find a suitable representation of the original data. It is a generalization of the PCA approach in terms of using the higher order statistics in addition to the second order used in the PCA. More specifically, given random variable data \(x\), ICA finds a good approximation matrix \(A\), that relate the unobserved source signal, \(s\) to \(x\) with a linear combination,

\[
x = As
\]

(3.14)

under the assumptions that the unobserved source signal, \(s\), is statistically independent and non-normal. Before we proceed with the ICA concept, the statistical independence needs to be addressed.
### 3.2.1 Statistical Independence

Statistical independence, a major concept in the ICA algorithm used in facial feature extraction part, is reviewed by two abstracted events $A$ and $B$. If the probability of occurrence of one event $A$ is not affected by the occurrence of the other event, $B$, then the two events are said to be statistically independent and it can be written in

$$ P(A \mid B) = P(A) \quad (3.15) $$

From Eq. (3.15) and the definition of the conditional probability in Eq. (3.16),

$$ P(A \mid B) = \frac{P(A \cap B)}{P(B)} \quad (3.16) $$

we have following,

$$ P(A \cap B) = P(A)P(B) \quad (3.17) $$

This concludes that the probability of the joint occurrence of two events is the product of each other’s probability when they are statistically independent. Now we apply this concept to two independent random variables $x$ and $y$, $A = \{x \leq x\}$ and $B = \{y \leq y\}$ are function of $x$ and $y \in \mathbb{R}$.

$$ P(x \leq x, y \leq y) = P(x \leq x)P(y \leq y) \quad (3.18) $$

In terms of the probability distribution functions Eq. (3.18) can be rewritten as

$$ F_{x,y}(x, y) = F_x(x)F_y(y) \quad (3.19) $$

Then we have probability density function by differentiating Eq. (3.19).

$$ f_{x,y}(x, y) = f_x(x)f_y(y) \quad (3.20) $$

From the conditional distribution function with the event $B$ and by substituting Eq. (3.19) into Eq. (3.21), we have the conditional distribution function which is to be the marginal distribution function in Eq. (3.22) and Eq. (3.23) with similar procedures.

$$ F_x(x \mid B) = \frac{P(x \leq x, y \leq y)}{P(y \leq y)} = \frac{F_{x,y}(x, y)}{F_y(y)} \quad (3.21) $$
By differentiating Eq. (3.22) and Eq. (3.23), we also have the conditional density functions.

\[
\begin{align*}
    f_x(x | B) &= f_x(x) \\
    f_y(y | A) &= f_y(y)
\end{align*}
\] (3.24) (3.25)

These confirm the generally accepted property of statistical independence. The occurrence of an independent random variable does not depend on any other variables. Independent variables are not affected by others. If we could transform the observed data \( x \) into a set of independent variables in the vector \( s \), then we can visualize or analyze the property of the independent data separately.

### 3.2.2 ICA Concept

**Mixed Signal and Independent Component**

This subsection covers the background and fundamental understandings of the ICA algorithm. The development of the ICA in here is mainly adopted from Hyvärinen and Oja [16]. Let us consider the following ICA model,

\[
x = As = \sum_{i=1}^{n} s_i a_i
\] (3.26)

where, \( x = [x_1, x_2, \cdots, x_n]^T \), \( a_i = [a_{1,i}, a_{2,i}, \cdots, a_{n,i}]^T \) and \( A = [a_1, a_2, \cdots, a_n] \in \mathbb{R}^{n \times n} \). The zero-mean random vector, \( x \), is a linear combination of the independent random variable, \( s_i \), having the weight, \( a_i \). Unless either \( s \) or \( A \) is known, we cannot determine the variance of an independent variable, \( s_i \). Hence, we need to introduce an assumption that the magnitude of \( s_i \) is a unit variance.

\[
E\{s_i^2\} = 1
\] (3.27)

Under the assumption of Eq. (3.27), the sign of \( s_i \) can be either positive or negative.

If we can find the unknown coefficient matrix, \( A \), the independent random vector can
be found by solving the inverse matrix of $A$.  

$$s = A^{-1}x$$

(3.28)

Defining $W^T = A^{-1}$, we have  

$$s = W^T x$$

(3.29)

However, we can not find the exact solution of the weight matrix, $W$, since $A$ is unknown. We are going to estimate $W$ by using the following process. Let $y$ be a scalar variable,  

$$y = w^T x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

(3.30)

where $w$ is an arbitrary column vector of $W$. $y$ in Eq. (3.30) should be independent if $A$ is properly determined. Introduce a new variable, $k$,  

$$k = A^T w$$

(3.31)

Then Eq. (3.30) can be written by  

$$y = w^T x = w^T A s = k^T s$$

(3.32)

with Eq. (3.26) and Eq. (3.30). From Eq. (3.32), $y$ is the linear combination of independent components, $s_i$, with a coefficient vector, $k$. Referring to the Central Limit Theorem,

Given $n$ independent random variables, the distribution of the sum of $n$ independent random variables approaches a normal distribution as $n$ increases.

$k^T s$ has a more normal distribution than the one independent component, $s_i$, has, and equivalently, $k^T s$, with only one non-zero elements in $k$, has the least normal distribution when it approaches $s_i$. In this sense, the problem lays on finding $w$ that maximizes the non-normality of the linear combination of $w^T x$ for $s_i$. From Eq. (3.32),  

$$w^T x = k^T s,$$

(3.33)
the linear combination of a random vector $x$ and a weight vector $w$ is the one of the independent components. We now need a measure of non-normality to use the concept above, and kurtosis is a measure to calculate the non-normality.

\[
kurt(y) = E\{y^4\} - 3\{y^2\}^2
\]  

$kurt = 0$ means a normal distribution. The typical examples of super-normal, $kurt > 0$, and sub-normal, $kurt < 0$, distribution function are Laplacian and uniform distribution, respectively.

**Whitening Process**

Before we directly use the data $x$ to the input of the ICA algorithm, data $x$ needs to be transformed in a certain way, through a process called whitening. From Eq. (3.26), we know that $s_i$ is also zero-mean as is $x$. Then covariance between two different variables $s_i$ and $s_j$ is 0, where $i \neq j$ and is 1 for $i = j$. Therefore, covariance matrix of $s$ is an identity matrix, $I$, and $s$ are uncorrelated. This does not necessarily mean that they are independent. Uncorrelateness is included in independence. However, if we can manipulate the data $x$ so that it becomes uncorrelated, then we may approach our final objective of finding the independent components more easily.

![Figure 3.2 Whitening process: (a) independent source, (b) observed, and (c) whitened data.](image)

The observed signal in Figure 3.2 (b) is generated by multiplying the unknown arbitrary matrix $A$ to the source distribution in Figure 3.2 (a). Given the observed
data, all elements of \( A \) has to be found to estimate source data. This problem can be reduced to another distribution in Figure 3.2 (c) by the whitening process. After the whitening process, we only need to estimate the rotation angle rather than estimating all components of \( A \). This property will also simplify the problem of designing the objective function shown in the later section. To do this, we let the covariance matrix of transformed data \( \tilde{x} \), called the whitened data, be an identity matrix.

\[
E\{\tilde{x}\tilde{x}^T\} = I
\]  

(3.35)

This can be achieved by the PCA, the eigen space decomposition of the covariance matrix.

\[
E\{xx^T\} = VDV^T
\]  

(3.36)

Then the whitened data can be obtained by

\[
\tilde{x} = Mx
\]  

(3.37)

where the whitening matrix \( M = D^{-\frac{1}{2}}V^T \) and \( D^{-\frac{1}{2}} \) is the square root of the inverse of a diagonal eigen value matrix. Also we verify the unit variance of the whitened data.

\[
E\{\tilde{x}\tilde{x}^T\} = E\{D^{-\frac{1}{2}}V^Txx^TVD^{-\frac{1}{2}}\}
\]

\[
= D^{-\frac{1}{2}}V^TE\{xx^T\}VD^{-\frac{1}{2}}
\]

\[
= D^{-\frac{1}{2}}V^TVVDV^TVD^{-\frac{1}{2}}
\]

\[
= I
\]  

(3.38)

Finding Weight Matrix

From Eq. (3.38), we can induce a valuable property of a weight matrix, \( Q \) as follows. \( Q \) is introduced when we are dealing with the whitened data and it corresponds to \( W \) in Eq. (3.29). We rewrite the Eq. (3.37) with Eq. (3.26).

\[
\tilde{x} = Mx = Qs
\]  

(3.39)

where \( Q = MA, Q = [q_1, q_2, \cdots, q_n] \) and \( q_i = [q_{i,1}, q_{i,2}, \cdots, q_{i,n}]^T \). \( Q \) is a weight matrix to find \( s \) when the whitened data is given. Then the objective is relaxed
from finding the unknown matrix $A$ to the orthogonal matrix $Q$ with Eq. (3.38) and Eq. (3.27).

$$
E\{\hat{x}\hat{x}^T\} = E\{Qs\hat{s}Q^T\} \\
= QE\{s\hat{s}^T\}Q^T \\
= QQ^T \\
= I
$$

(3.40)

By recalling Eq. (3.34) and from Eq. (3.30) with a property of the whitened data $\hat{x}$, we find the weight vector $q_i$ which is the gradient of kurtosis function of $q_i^T\hat{x}$.

$$
kurt(q_i^T\hat{x}) = E\{(q_i^T\hat{x})^4\} - 3E\{(q_i^T\hat{x})^2\}^2 \\
= E\{(q_i^T\hat{x})^4\} - 3(q_i^TE\{\hat{x}\hat{x}^T\}q_i)^2 \\
= E\{(q_i^T\hat{x})^4\} - 3\|q_i\|^4
$$

(3.41)

The kurtosis function Eq. (3.41) approaches a maximum or a minimum away from zero if the distribution of $q_i^T\hat{x}$ is super-normal or sub-normal, respectively. After $Q$ is found, the independent components, $s$, can be calculated by

$$
s = Q^{-1}\hat{x} = Q^T\hat{x}
$$

(3.42)

The unknown matrix also can be estimated by $A = M^{-1}Q$.

There are a number of solutions [3] [14] [15] [16] [17] to find the weight vector with the concept of maximizing non-normality of Eq. (3.41).

### 3.3 ICA Representation of a Face

The PCA approach based on second order statistics has been successfully applied to face recognition problems [2] [4] [19] [25] [39]. However, researchers argue that the PCA method may lose important information contained in higher order statistics of facial images. On the other hand, a recently developed technique, the ICA has proven to be a powerful method in representing higher order statistical properties of signals and images. Eigenvectors of the PCA provide an overlap and disperse facial images from a given set of facial images, while the ICA bases provide non-overlapped spatially concentrated facial features of a face such as mouth, eyes, nose, etc [2].
With its authentic face recognition and presentation capability, the ICA has drawn considerable attention in the field of feature extraction.

The ICA method for finding the facial features or equivalently finding the independent basis image sets in the video sequence is conceptually based on the method developed by Bartlett et al [3]. Let us consider the following ICA model.

\[ S = W^T X \]  

(3.43)

where \( X = [x_1, x_2, \ldots, x_n]^T \), \( S = [s_1, s_2, \ldots, s_m]^T \), \( W = [w_1, w_2, \ldots, w_m] \), \( n \) and \( m \) (\( \leq n \)) are the number of frames in a video sequence and that of independent components respectively. The concatenation of the columns in a zero-mean and unit-variance facial image is a row vector \( x_i \) in \( X \in \mathbb{R}^{m \times N} \). \( X \) presents a collection of the localized facial images in a video sequence. A weight vector \( w_i \) in \( W \in \mathbb{R}^{m \times m} \) and the ICA basis vector \( s_i \) in \( S \in \mathbb{R}^{m \times N} \) are to be determined.

The fundamental concept in [3] is to represent image data with the independent component bases and their associated coefficients. The ICA performed on \( m \), \( m \leq n \), principal eigen vectors of the covariance matrix of input images. We reduce the dimension of independent bases set to an appropriate number, \( m \), by using the whitening matrix, \( M \in \mathbb{R}^{m \times n} \) when the PCA approach is applied to find the whitening matrix,

\[ M = D^{-\frac{1}{2}} V^T \]  

(3.44)

where \( D = diag(\lambda_1, \lambda_2, \ldots, \lambda_m) \) has the eigenvalues of covariance matrix of \( X \) in descending order. \( V = [v_1, v_2, \ldots, v_m] \) is the eigen vector matrix of \( X \) and \( v_i \) is the corresponding eigen vector to its eigen value \( \lambda_i \). The whitened input given by

\[ \bar{X} = MX \]  

(3.45)

is the input to be supplied to the ICA algorithm. The computationally efficient Hyvärinen's fast fixed algorithm [15] described by Eq. (3.46) is adopted. It finds the weight matrix \( Q \) with the whitened data, where \( Q = [q_1, q_2, \ldots, q_m] \) and \( q_i = [q_{i,1}, q_{i,2}, \ldots, q_{m,1}]^T \). When the kurtosis function converges to the maximum or minimum,

\[ q_i^+ = E\{\bar{X}(\bar{X}^T q_i)^3\} - 3q_i \]  

(3.46)
yields the best estimate of $q_i$. The updated weight vector, $q_i^+$, thus, represents a facial feature. The main concept of this iterative unsupervised ICA algorithm is that the extremum of a kurtosis function of $q_i$ can be found by solving Lagrange multiplier with the constraint $\|q_i\|^2 = 1$. With Eq. (3.41) and the constraint, we define a function

$$F(q_i, \delta) = f(q_i) + \delta^T g(q_i)$$

(3.47)

where $f(q_i) = E\{(q_i^T x)^4\} - 3 \|q_i\|^4$, the constraint function is $g(q_i) = \|q_i\|^2 - 1$, and $\delta$ is an undetermined Lagrange multiplier. By extremizing the function, $F(q_i, \delta)$, $q_i$ is found as a solution to satisfy

$$\frac{\partial F(q_i, \delta)}{\partial q_i} = 0$$

(3.48)

When Eq. (3.48) is met, Eq. (3.46) yields the solution with a scalar factor. The found weight vectors $q_i$ are transformed by the orthogonalization such as Gram-Schmidt method, and Eq. (3.40) is satisfied. The convergence proof of Eq. (3.46) is comprehensively demonstrated in [15].

After finding the weight matrix $Q$ from Eq. (3.46), the ICA bases are obtained by

$$S = Q^T \bar{X}$$

(3.49)

Figure 3.3 First 21 ICA bases of facial images in the test video sequence are shown
3.3.1 Finding Region of Interest with ICA

The Original input image sequence, $X$, can be approximately reconstructed by the ICA bases and their corresponding coefficients.

$$X \approx M^{-1}QS$$ (3.50)

The reconstruction, however, is not of our primary interest. The objective is to find the region of interest where some key components such as the eyes and mouth are located in the common facial images. In order to achieve this goal, only several ICA bases are needed to be selected among all ICA bases. Since some ICA bases have highly concentrated values representing specific parts of interest, major components are carefully studied. For instance, the first ICA basis in Figure 3.3 is considered to present a mouth. Although $m$ ICA bases, i.e. reduced dimensionality by the PCA, and their associated coefficient vectors are required to reconstruct the original images, only a few selected ICA bases are sufficient to reconstruct a localized image of a facial part such as around mouth area.

![Figure 3.4 Chosen mouth like ICA bases.](image)

The linear combination of highly localized ICA bases and their associated coefficients reflect only on the specific area in the images. By checking the normalized Euclidian norm $\gamma_i$ given by,

$$\gamma_i = \frac{\| \mathbf{s}_i \|_2}{\sum_{j=1}^{m} \| \mathbf{s}_j \|_2}, \quad i = 1, 2, \ldots, m$$ (3.51)

By rescaling $\gamma_i$, the highly localized ICA bases can be chosen.

$$\gamma'_i = \frac{\gamma_i}{\max\{\Gamma\}}$$ (3.52)
where $\|\cdot\|$ is the Euclidean norm and $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_m\}$. Here, we only choose the ICA bases corresponding to $\gamma_i$, which is greater than a certain threshold value, $T_\gamma$.

The major components of a face, such as the eyes or mouth-like images, were selected from the ICA basis by simple heuristic rules. For example, eyes-like basis can be found in the upper half and either on the left or right sides in an ICA basis image. The mouth-like basis can be found if highly concentrated values are located within the bottom half and in the middle portion of an ICA bases image. Once the ICA bases corresponding to each major component are found, we stack those square root of ICA bases up as shown in Figure 3.5 (a) and have the spatially concentrated facial component image. That portion is considered as the area where the major component is. In order to obtain the region of interest for the next process, a boundary box is set after a threshold value is applied. Some results of finding approximate mouth area using the ICA approach is shown in Figure 3.5 (b), (c), and (d).

![Figure 3.5](image)

Figure 3.5 Extracted approximate mouth area of the test images.

### 3.4 Mouth Corner Points Extraction

The ICA approach can successfully find the approximate locations of major components of a face. However, it is essential to have a robust method for finding precise coordinates of the feature points against inconsistent circumstances of a video source. In general cases, it is difficult to find feature points as many as the MPEG-4 specified for FAPs.
As an alternative, a few key feature points which take an important role in animation can be extracted. The coordinates of other points, that can not be extracted precisely, will be determined by the priori designed heuristic logic based on the information of extracted key feature points. We will discuss details about the determination of coordinates of those other points in the following chapter. In this section, the method applied for extracting the key feature points is described. We limit our scope to the bottom part of a face; the part linked to mouth, jaw, and chin has the most movement in a talking head, and the quality of animation of this part influences the overall performance of a face animation in terms of the realistic presentation.

3.4.1 Prewitt Filtering using Empirical Approaches

We concentrate on the extraction of mouth feature points. The horizontal and vertical mouth corners, the most dominant features, offer us valuable information. For example, we can estimate the height and width of a mouth from the vertical and horizontal corners, respectively, and those four points contribute to the design of a lip model as will be shown in the next chapter. When we observe the area around the mouth, the luminance component in a gray scale image provides information about the position of the mouth corners. Mouth corners appear with distinct luminance usually as dark spots. A threshold value applied to the luminance can detect the location of mouth corners. However, it is very sensitive to the lighting condition and image to image variations are excessively large. Prewitt filtered images can avoid the drawback of applying a fixed threshold value. Let us define \( L(i,j) \) as \( i,j \)th element of a luminance image \( L \). A Prewitt filtered image, \( P \), is obtained as following.

\[
P(i,j) = \sqrt{B^2}
\]

\[
B = \frac{1}{3} \cdot [(L(i-1,j+1) + L(i,j+1) + L(i+1,j+1))
\]
\[
-(L(i-1,j-1) + L(i,j-1) + L(i+1,j-1))]
\]

Here, only the vertical edge gradient is considered because of insignificant influence of the horizontal edge information from the region of interest. Firstly, \( 3 \times 3 \) Prewitt filter is performed twice on \( L \) in Figure 3.6 (a). And we have a filtered image, \( P \).
in Figure 3.6 (b). As we can see from those two images, the approximate outer boundaries of a mouth can be obtained from the low and high luminance values of Figure 3.6 (a) and (b) respectively. Then, we check the minimum and maximum values of each column in Figure 3.6 (a) and (b) and normalize the values.

\[ a_k = \min\{l_k\} \in [0, 1], \quad k = 1, 2, \ldots, n \]  \hspace{2cm} (3.55)

\[ b_k = \max\{p_k\} \in [0, 1], \quad k = 1, 2, \ldots, n \]  \hspace{2cm} (3.56)

where \( L^{m \times n} = [l_1, l_2, \ldots, l_n] \), \( P^{m \times n} = [p_1, p_2, \ldots, p_n] \), and \( l_k \) and \( p_k \) are \( k \)-th column of \( L \) and \( P \) respectively. We define \( a = [a_1, a_2, \ldots, a_n] \) and \( b = [b_1, b_2, \ldots, b_n] \), they are shown in Figure 3.6 (c).

![Figure 3.6](image)

**Figure 3.6** Luminance image (a), Prewitt filtered image (b), intersections for finding corners (c).

With the empirical knowledge, the approximate points of the left and right mouth corners are found by first and the last intersections of \( a \) and \( b \) with the row index of \( a \) and \( b \). The left corner is found at the first intersect of decreasing \( a \) and increasing \( b \). The right corner corner is found at the last intersect of increasing \( a \) and decreasing \( b \). However, in many cases, we found that intersections are placed rather on the insides of each mouth corner due to a less contrasted image. And the following may refine the image to have an accurate detection.

\[ L(i, j) = t, \quad \text{if} \quad L(i, j) < t, \quad i = 1, 2, \ldots, m, \quad j = 1, 2, \ldots, n \]  \hspace{2cm} (3.57)

where \( t \) is the average luminance value of the intersections of \( a \) and \( b \). As a result of this process, we have a less dimmed image in Figure 3.7 (a) and its Prewitt filtered
image in Figure 3.7 (b). After applying the same process for finding intersections with the images of Figure 3.7 (a) and (b), we have the updated intersections in Figure 3.7 (c) presenting locations of mouth corners, and those match our intuition.

![Figure 3.7 Corrected luminance image (a), Prewitt filtered image (b), intersections for finding corners (c).](image)

For the horizontal mouth corner extraction, as shown in Figure 3.8, we apply the same method to the 90 degree rotated image of the mid columns in Figure 3.7 (a).

![Figure 3.8 Corrected luminance image (a), Prewitt filtered image (b), intersections for finding corners (c).](image)

In the case that the mouth area found by the ICA process has a corner with a very little margin to the side of a image, we may have less than two intersections. In this case, we assign the Y-coordinate of the missed corner to the 1-st (or n-th) column and the X-coordinate to the average value of $a_1$ and $b_1$ (or $a_n$ and $b_n$). The concept contained in this method also could be extended to other major components of a face.
A number of different QCIF test images: Claire, Miss America, Foreman, and Carphone, are shown in Figure 3.9 starting from the left. This method has demonstrated to be reliable under normal luminance condition. The mouth area was correctly determined for each of test images.
4. FACE ANIMATION WITH VRML

VRML (Virtual Reality Modeling Language) is chosen as a 3-D face modeling and animation tool because of its ability of creating a dynamic virtual world on the Internet, and interactive features that allow a user to manipulate objects. Those abilities and features meet the demands arising in the application fields of the MPEG-4 face animation in many aspects\(^1\). Besides that, unlike other prohibitively expensive professional 3-D modeling software, we can easily download a VRML browser free of charge. VRML is a descriptive textual language that describes 3-D objects and controls their motion, which can be edited by any text editors. This chapter explains the methodology of face modeling and animation using VRML with extracted feature points. From the reason that the face model is defined in 3-D space while a given video source is a 2-D presentation, the heuristic rules are described to supply missing information such as depth for generating the visually acceptable 3-D facial images.

4.1 An Example of a VRML Scene

Before we get into the methodology of face modeling and animation, this section provides an example of a simple VRML file (*.wrl) to show how a VRML scene can be generated. The following rotation cube example illustrates the use of basic nodes, fields, and a simple animation sequence.

\begin{verbatim}
1    #VRML V2.0 utf8
2    Group {
3      children [ 
4        DEF Touch TouchSensor {}
5        DEF Clock TimeSensor {
\end{verbatim}

\(^1\)MPEG-4 does not define the 3-D animation engine in itself. It is left for the designer to define and produce a visual presentation of the 3-D animation.
cycleInterval 2.0
loop FALSE
}
DEF cube Transform {
  rotation 1 1 1 0.785
  children Shape {
    appearance Appearance {
      material Material {
        diffuseColor 1 1 1
      }
    }
  }
}
geometry IndexedFaceSet {
  coord DEF cubeCoordinate Coordinate {
    point [ -1 1 1 ,
          1 1 1 ,
          1 1 -1 ,
          -1 1 -1 ,
          -1 -1 1 ,
          1 -1 1 ,
          1 -1 -1 ,
          -1 -1 -1 ]
    coordIndex [ 0, 1, 2, 3, -1 ,
                 3, 7, 4, 0, -1 ,
                 0, 4, 5, 1, -1 ,
                 1, 5, 6, 2, -1 ,
                 2, 6, 7, 3, -1 ,
                 4, 5, 6, 7 ]
  }
}
DEF cubePath CoordinateInterpolator {
  key [ 0.0 0.5 1.0 ]
  keyValue [ -1 1 1 ,
             1 1 1 ,
             1 1 -1 ,
             -1 1 -1 ,
             -1 -1 1 ,
             1 -1 1 ,
             1 -1 -1 ,
             -1 -1 -1 ]
  0 1 1.414 ,
  1.414 1 0 ,
  0 1 -1.414 ,
  -1.414 1 0 ,
  0 -1 1.414 ,
  1.414 -1 0 ,
  0 -1 -1.414 ,
  -1.414 -1 0 ,
  1 1 1 ,
  1 1 -1 ,
  -1 1 -1 ,
  -1 -1 1 ,
  1 -1 1 ,
  1 -1 -1 ,
  -1 -1 -1 ]
}
ROUTE Touch.touchTime TO Clock.set_startTime
ROUTE Clock.fraction_changed TO cubePath.set_fraction
ROUTE cubePath.value_changed TO cubeCoordinate.set_point
The body of the VRML file consists of nodes and fields. A node, starting with an upper case, describes the shape and other attributes of an object by specifying types, fields, and field values. In VRML, a node accompanies fields, and a field can accompanies nodes.

The Group node, line 3, gathers various shapes to create a complex shape. There is only one shape, a cube, in this example. The children field, line 4, specifies those shapes to be included in the Group node. The Transform node, line 10, creates new coordinates common to such Shape nodes, line 11. The Shape node that generates an object, is included in the children field, line 12. The Shape node includes appearance and geometry fields, line 13 and 18. In the appearance field, Appearance node specifies material field, line 14. The Material node describes attributes, such as color here, of a material in the material field, line 15. In the geometry field of Shape node, IndexedFaceSet node is used, line 18. The IndexedFaceSet node creates a user-defined object on 3-D coordinates by assigning coordinates of points in the Coordinate node of the coord field, line 19-28. The coordIndex field in the Coordinate node defines each surface of an object by ordering indexes of individual points, line 29-34. CoordinateInterpolator node, line 38, specifies a path of each point described in the Coordinate node by assigning 3-D coordinates of each point in certain time. In this example, cycleInterval is assigned at 2.0, line 7, and key is listed at 0.0, 0.5, and 1.0, line 39. Therefore, if animation cycle took 2 seconds, the parameter values from line 40-47, 49-56, 58-65 correspond with the coordinates at 0.0, 1.0, 2.0 second. Line 70-72, show how the ROUTE syntax interacts with sensors and nodes to perform the animation. In order to use the ROUTE syntax, nodes need to have names previously defined by DEF syntax.

The animation parameters of each point are shown in line 40-65. The values of each column in keyValue present one of each coordinate. These parameters are only for 8 points in this example. However, when a complex object, such as a face, is

\(^2\)CoordinateInterpolator node interpolates the coordinates between the time 0.0 and 1.0 second and between 1.0 and 2.0 second to rotate the cube smoothly.
considered, numerous number of points are needed in a *Coordinate* node, and a listed number of *key* needs to be increased for smoother facial motion. As a result, the number of the *keyValue* will be the multiple of the number of *key* and *point*. When all the parameters are listed in one VRML file, the file size could become a problem.

Storing the external animation parameter data file can be a solution of this problem. When motion information is separately stored and delivered from the model information, the file size can be reduced which is to be transmitted at once. For example, a file describing a model can be transmitted first, then animation parameters can be sent later when a system needs new parameters to perform successive animation. This also corresponds with the concept of the model-based face animation. In this thesis, the animation parameter data files are generated by MATLAB and they are stored in binary format outside of a VRML file.

![An example of a VRML scene.](image)

**Figure 4.1** An example of a VRML scene.

### 4.2 Overview of Face Animation System

Face animation system presented consists of facial components modeling and the animation. In the modeling part, each of the major facial components is separately constructed, and detailed descriptions of those components are provided. The model-based facial animation allows us to generate a talking head without compromising the image quality, although only a small amount of motion information is transmitted. VRML with VRML’s API (Application Programming Interface) such as JAVA, play an
important roll in the model-based face animation task.

4.3 Generating a Facial Model

A VRML file presenting a face model is composed of several components commonly found in VRML files. VRML has several functional blocks, called nodes, which plays a specific role in the 3-D face modeling and animation. A Transform node is responsible for the rotation and translation. For instance, we can describe the nodding, shaking, and rotation of a head model by the Transform node with help of the time sequence 3-D coordinates parameters. An IndexedFaceSet constructs flexible human skin surface made of a number of surface segments of the flat VRML faces. Besides those nodes, we use an Extrusion node for presenting the extruded shapes in a face such as lips, teeth, and gums. By using the spine field in the Extrusion node, we can control the shape efficiently with a small number of parameters, therefore it is computationally efficient as well.

4.3.1 Modeling Facial Components

We start with constructing the lips of the neutral mouth, i.e., a closed mouth without expression. Since we deal with a video sequence of many frames, there is no difficulty in finding the neutral mouth shape. We define the neutral mouth with two measurable features. First, we measure the height and length of the mouth with the extracted vertical and horizontal mouth corners of each frame. The following rule is used for finding the frame where the neutral mouth is.

$$\min\{\text{height}_i + \frac{1}{\text{length}_i}\}, \quad i = 1, 2, \ldots, N$$

(4.1)

where $N$ is the number of frames in the video sequence. And the mouth in that frame is regarded as the neutral mouth. This is formulated based on the fact that the height and length of the neutral mouth are shorter and wider than that of others. Once we find the neutral mouth, $M_{\text{Neutral}}$, the thickness of each upper and lower lip is measured, and is maintained throughout the entire animation sequence. When a line connecting horizontal corners is drawn, then the thickness of each lip can be
measured. The vertical length from the vertical corners to the center of the line are
the thickness of each lip. Therefore, X-Y coordinates of 6 points, 4 mouth corners
and the center of the line for each lower and upper lip, are obtained from the neutral
mouth shape. However more points are desirable for making a lip model flexible.
When we design a smooth and flexible surface with 3-D software, a complex object
requires hundreds to thousands of polygons. The smoother surface demands the more
number of polygons, and faces computational penalties as a consequence. One reason
to use VRML is to avoid this problem with adequate nodes.

Each of upper and lower lip shapes can be simplified as a sliced half pipe whose
radius varies from point to point. Then, we can use the Extrusion node of VRML
to model a lip as a tilted half pipe. In the Extrusion node, there are several useful
fields that can be combined to make a lip-like extruded shape. The crossSection field
allows us to make a more flexible 2-D shape in the X-Z coordinates. An arch-like
shape presenting vertically round outer surface of the lips is designed by this field.
Then the spine field defines the path of vertical arch-like lines of various sizes with
a few points in 3-D coordinates. When vertical arch-like lines of the various sizes,
generated by the scale field, follows along the path, we have a 3-D extruded shape that
resembles the shape of the lip surface. Finally, the creaseAngle and orientation field
smooths the surface of lips and tilts the Y-coordinate of lip surface to an appropriate
angle. In this process, the thickness and length of the lip in the neutral mouth are
scaled to be mapped to the lip model.

The method of teeth and gums modeling is similar to that of lips except in using
rectangular shape in the crossSection field instead of an arch-like line. We also do not
tilt teeth and gums model, and do not smooth the surface of teeth. The Z-coordinate
of teeth and gums model are placed not to overlap the surfaces of lip or skin. The
default X-Y coordinates of the upper and lower teeth and gums are set accordingly
and the 3-D coordinates of the upper teeth and gums are fixed and kept throughout
the entire animation sequence because they are connected to the head bone.
4.3.2 Modeling Skin

Modeling skin part requires flexibility, smoothness, and the ability to construct irregular shapes. The IndexedFaceSet node fulfills these conditions and generates 3-D shapes of the skin area by assigning the 3-D coordinates of polygons in the coord field. The skin area is constructed as follows. First, we determined the number of polygons to construct the bottom half of a talking head shape, and assign the 3-D coordinates for each polygon. The values, in the coordIndex field, must be arranged in order that the adjacent surface segments is to be shared. As illustrated in the rough sketch of the bottom part of a face in Figure 4.2, we have 22 points define the shape of the skin area. The coordinates of points 0, 3, 4, and 7 in Figure 4.2 are remained same through the animation sequence. And points 8, 9, and 12 have the values related to the jaw angle.

![Figure 4.2](image)

**Figure 4.2** Frontal and profile view of bottom half of a simple wire frame talking head model, constructed by IndexedFaceSet and Extrusion node for skin surface and lip model, respectively, is shown.

4.3.3 Heuristic Rules

We were able to generate a basic lip model by using the extracted mouth corner points and several VRML nodes as explained in section 4.3.1. However, we need additional control points to make the lip shape more realistic to realize motions associated with the lips. A total of 10 points are chosen as shown in Figure 4.3.
Among 10 points, point 1, 2, 3, 4, 5, and 6 in Figure 4.3 have the same coordinate of the point 1, 2, 5, 6, 10, and 11 in Figure 4.2. Other X-Y coordinates are determined by following heuristic rules.

1. \( X_2 = 0.17 \cdot X_4 \)
2. \( X_{3,5} = 0.5 \cdot X_4 \)
3. \( Y_2 = Y_1 - 0.25 \cdot (Y_4 - Y_1) \)
4. \( Y_3 = Y_1 \)
5. \( Y_5 = Y_4 + 0.7 \cdot (Y_6 - Y_4) \)

Where, \( X_j \) and \( Y_j \) are X and Y-coordinate of point \( j \) in Figure 4.3, respectively. The rule determining Z-coordinate of all 10 points of the upper and lower lip is based on the X-coordinate of a corner point. The basic rule for determining the Z-coordinate of lip points is that if the length of a mouth is shorter than that of a neutral mouth, the Z-coordinate of lip points are going forward, and backward if longer.

Assume that we have \( Z_4 = 0 \), and \( Z_{1,6} = 1 \) in \( M_{\text{Neutral}} \). Then, we assign the value \( Z_{3,5} = 0.67 \), and \( Z_2 = 0.9 \). Where \( Z_j \) is the Z-coordinate value of point \( j \) in the mouth of the \( i \)-th frame, namely \( M_i \). We keep this ratio between points throughout.
the animation sequence. Now we have the rule when the length of $M_i, L_i$, is different from that of the neutral mouth, $L_{Neutral}$.

1. If $L_i \geq 1.25 \cdot L_{MN}$, then $Z_4 = -0.4$ and $Z_{1,6} = 0.9$
2. Else if $L_i \leq 0.75 \cdot L_{MN}$, then $Z_4 = 0.4$ and $Z_{1,6} = 1.1$
3. Else $Z_{1,6}$ and $Z_4$ are assigned linearly in $[0.9, 1.1]$ and $[-0.4, 0.4]$ respectively according to $L_i$
4. For $Z_{2,3,5}$, we keep the ratio between points that was set in the neutral mouth.

The animation of lips and a jaw delivers the major movements for the bottom part of a talking head. In order to do this, the rotated jaw angle should be extracted at each frame. The new origin, $(Z_7, Y_7)$, and the point $(Z_{11}, Y_{11})$ in the neutral state are chosen to calculate a lip angle, $\theta_{Neutral}$, between the Z-axis and the line connecting the origin to $(Z_{11}, Y_{11})$ in Figure 4.4.

\[
\theta_{Neutral} = \cos^{-1}\left(\frac{|z|}{\sqrt{z^2 + y^2}}\right) \cdot \frac{180}{\pi} \tag{4.2}
\]

\[
\theta_{Neutral} = -\theta_{Neutral}, \quad \text{if sign}(z) \neq \text{sign}(y) \tag{4.3}
\]

where $z = Z_{11} - Z_7$, $y = Y_{11} - Y_7$. The lip angle, $\theta$, of each frame is calculated by the same manner. Then the approximate rotated jaw angle, $\theta_i$, in the $i$-th frame is obtained.

\[
\theta_i = \theta - \theta_{Neutral} \tag{4.4}
\]

After finding $\theta_i$, the Y-Z coordinates of point 8, 9, and 12 in Figure 4.4 are determined by the rotation matrix, $R$.

\[
R = \begin{bmatrix}
\cos \theta_i & -\sin \theta_i \\
\sin \theta_i & \cos \theta_i
\end{bmatrix} \tag{4.5}
\]
Figure 4.4  Jaw rotation according to the bottom lip angle point.

The vertical and horizontal length of a face in Y and Z-coordinate are required to extract the jaw angle, $\theta_i$, for animating the accurate jaw movements. However, the horizontal length is unknown in a 2-D video plane, and measuring the vertical length does not give us accurate parameters in some cases such as nodding or low quality facial images. Alternatively, we choose the point, $Y_{11}$, and have a previously assigned horizontal length. Although comparing $Y_{11}$ of each frame to that of the neutral state can not provide us with an accurate rotation angle of a jaw, the approximate rotation angle of a jaw can be obtained in most cases.

4.4 Model-based Face Animation

The specification of VRML defines a JAVA program script and JavaScript interface, which enables the JAVA program to interact with the VRML browser. When VRML is combined with the JAVA program script or JavaScript, its capability of handling more complicated animation path can be dramatically improved, and VRML also becomes more flexible and powerful.

4.4.1 VRML Animation and JAVA API (Application Programming Interface)

For the animation purpose, the CoordinateInterpolator node and the ROUTE syntax are mainly used. The former specifies the 3-D coordinates of a model in the period of time, and precisely controls the internal time-dependant animation with the
output of a TimeSensor node. To connect the output of TimeSensor node to the input of the CoordinateInterpolator node, we need the ROUTE syntax. The diagram in Figure 4.6 shows the interaction among the various nodes to perform animation.

When we are dealing animation with VRML, it is necessary to manage a long time sequence by animation parameters embedded in a VRML file. Consequently, the file size becomes excessively large due to the numerous parameter lines specifying all animation sequence. This problem conflicts the advantage of model-based animation. As a solution, we use JAVA, one of VRML’s API, in Script node. This node has interface eventIn, eventOut, and field. Unlike other nodes, Script node cannot produce a certain shape or animation by itself alone. Instead, a provided program script placed in the Script node performs animation. This method extends VRML’s capability to the handling complex and customized animation tasks. The program script can be written in any language supported by VRML. JAVA and JavaScript are commonly used. As mentioned previously, embedding animation parameter data in a VRML file is inappropriate, and using JAVA program script to load the animation parameter data stored outside of VRML can be an alternative solution considering the file size. In order to do that, VRML’s specification of the node and field should be included in the JAVA program script in the Script node. The animation parameter data, called by JAVA program script, is converted to the type of a defined VRML eventOut. Then, this data is available to use as animation parameters in the Script node.

Figure 4.5 diagrammed this method. First, JAVA program loads the animation data files, and loaded data is converted to the defined type of VRML eventOut. After compiling this JAVA file, a CLASS file is generated, which contains the information we need to perform the animation. By calling this CLASS file in a Script node, the animation parameter data stored outside can be accessed.

The diagram in Figure 4.6 briefly shows how each node interacts to perform the animation. The sequence is ordered by numbers. First, the parameter data outside of VRML is accessed by Script nodes using VRML APIs, then each data is stored to
key and keyValue fields accordingly in the CoordinateInterpolator node. Then, the TimeSensor node sends the fraction time to the key field in the CoordinateInterpolator node. Finally, the CoordinateInterpolator node sends the output of each fraction time to the spine and point fields in Extrusion and Coordinate nodes, respectively. Each action is wired by the ROUTE syntax.

**Figure 4.5** Data loading with JAVA API of VRML.

**Figure 4.6** Interaction among various nodes used in the animation of a talking head model and animation with VRML.
5. EXPERIMENTAL RESULTS AND DISCUSSION

5.1 Face Localization

The presented face animation system is tested by the first 60 frames of the 30Hz 144 × 176 QCIF formatted Miss America color video sequence, Figure 5.1.

A subtracted image is obtained from the interval of two frames for the luminance component in the YCbCr color space. The subtracted image is normalized in [0, 1] and the threshold value 0.05 is applied to find moved pixels. The initial skin color threshold ranges for the normalized chrominance space are set to be 0.18 < nCb < 0.36 and 0.34 < nCr < 0.41. And, these ranges are applied to all moved pixels of the subtracted image. Then, the initial skin color region is detected.

In order to extract more reliable skin color region, a skin color density map is generated, which is obtained from the moved pixels fallen into the initial skin color threshold ranges of the normalized chrominance space. The values of skin color map are normalized in [0, 1], and a threshold value is set to be 0.3. The isolated clusters and insignificant valued pixels are discarded to have the finalized skin color density map. This map, obtained from the first subtracted image, is applied to all moved pixels in the subsequent subtracted images. Then, the finalized skin color region is obtained and shown in Figure 5.2.

The connect operation is performed to the finalized skin color region for the subsequent process, shape filtering. This operation closes small holes in the skin color region by checking the values of the neighbor pixels from a skin colored pixel, up to 5 pixel length to vertically and horizontally in both directions. During checking neighbors, if a skin color pixel is found after non-skin color pixel(s), then the gap
between two color pixels is closed. The opening process is followed to reject thin lines and small objects, and is shown in Figure 5.3. A $3 \times 3$ moving window is used for the erosion and dilation operation in opening process.

The projection process, described in section 2.2.2, is performed on Figure 5.3 to find the largest skin colored cluster where a face is placed. Holes in the largest skin colored cluster are filled and shown in Figure 5.4. In this filling process, the column index of the skin colored pixels in each row is obtained. If the number of skin colored pixels between the first and the last column index is larger than its counterpart, then the pixels between the first and the last column index are closed.

Then, the elliptical shape filtering is applied to the largest skin colored cluster assuming that the majority of the frontal face shape is oval, and has the threshold, $T_A = 0.9$. In case when a face is placed inside of the initial circle, the values in the vertical axis of the circle are checked from the bottom end. The final face region is reset when the skin colored pixel is found. The size of the localized face is normalized to a $60 \times 50$ image by the linear interpolation for the feature extraction stage, Figure 5.5.
Figure 5.1  Test video sequence, Miss America: Images are ordered from left to right and top to bottom
Figure 5.2  Skin colored pixels are shown after applying skin color density map.
Figure 5.3 Images after connect operation followed by opening process.
Figure 5.4 Projection and filling are performed on Figure 5.3.
Figure 5.5 Normalized final face area after applying the shape filtering.
5.2 Facial Feature Extraction

In order to locate major facial features, Hyvärinen's fast fixed-point algorithm is performed on the luminance component of 60, \( n = 60 \), facial images in Figure 5.5. The dimension of the facial feature, \( m \), is reduced to 45. The number of \( m \) is carefully chosen to extract the facial features which is similar to our perception of major components of a face. For example, \( n \) is very large and \( m \) approaches to \( n \), we have ICA bases spatially highly concentrated in a very small area. Or when \( m \rightarrow 1 \), we have spatially more spreaded ICA bases which is similar to that of PCA. Neither of the cases is appropriate for our task. With the chosen \( m \), ICA bases are shown in Figure 5.6.

Among all 45 ICA bases, some bases can be discarded because they have spatially spreaded values so that they, such as ICA bases in the last row, do not provide information where each facial component is located. In order to reject such ICA bases, \( T_\gamma = 0.3 \) in section 3.3.1 is used. The mouth-like ICA bases are selected by checking the location of concentrated values. We, then, find the approximate area of the mouth by stacking up mouth-like ICA bases. The approximate area of the mouth in each image of Figure 5.5 are shown in Figure 5.7.

The luminance component of the previously extracted mouth area and the information of its Prewitt filtered image are combined to extract 4 mouth corners of facial images without using threshold values. The method of finding the horizontal and vertical mouth corners is shown in Figure 5.8 and Figure 5.9, respectively. The intersections of two lines with the condition mention in section 3.4.1 present the location of the mouth corners. The \( x \)-axis presents the indexes of columns and rows in Figure 5.8 and Figure 5.9, respectively. The \( y \)-axis is scaled in \([0, 1]\) on both figures. The results of the method discussed in section 3.4.1 are shown in Figure 5.7 with asterisk marks on each corner.
ICA bases. 3, 6, 14, 15, 16, 18, 23, 30, and 35-th ICA bases are selected as mouth-like features. ICA bases are ordered from left to right and from top to bottom.

Figure 5.6
Figure 5.7 Approximate mouth areas of Figure 5.5 and results of mouth corner extraction are shown.
Figure 5.8 Finding horizontal corners with Prewitt filter and a heuristic approach.
Figure 5.9 Finding vertical corners with Prewitt filter and a heuristic approach.
5.3 Face Modeling and Animation with VRML

VRML 2.0 is used for the face modeling and animation in conjunction with JAVA program for the efficient animation parameter data access.

Since the CoordinateInterpolator node integrates the linear interpolation to compute intermediate values between the values in the keyValue field, we provide the parameters at the interval of two frames rather than providing that of all frames.

In order to estimate $\theta_i$ of each mouth, $M_N$ needs to be found by Eq. (4.1). And, the 9-th frame of the subsampled test images in Figure 5.11 is considered as $M_N$.

The lips have 12 spine points, 7 for the upper lip and 5 for the lower lip. They share two spine points, point 4 and 8 in Figure 4.3. Based on the Y-coordinate of point 6 in Figure 4.3, the estimated lip rotation angle, $\theta_i$, of every mouth image is estimated. Referring to the Figure 4.2, X-Y coordinates of 1, 6, and 11 are obtained by that of extracted corners in Figure 5.7. The Z-coordinate of 1, 6, and 11 and 3-D coordinates of point 2, 5, and 10 are estimated by the heuristic rules in section 4.3.3.

The 3-D lip model is shown in Figure 5.10 (a).

Each of upper teeth and upper gums has 17 spine points assigned. Their coordinates are fixed throughout the animation. Each of lower teeth and lower gums also has 17 spine points. During the animation, their coordinates are rotated according to $\theta_t$. The 3-D gums and teeth model are shown in Figure 5.10 (b).

Skin part in the bottom half of a face has 12 face set by assigned 22 points. Referring to the Figure 4.2, the coordinates of point 0, 3, 4, and 7 are fixed. The coordinates of point 6, 9, and 12 are rotated based on $\theta_i$. The bottom half of a facial model, composed by skin, teeth, gums, and lips, is shown in Figure 5.10 (c).

The animation parameters for each facial component are obtained from all subsampled test images and stored in files. In order to access these parameters stored outside of a VRML file, JAVA program is used as a VRML's API. Then, stored animation parameter files are mapped to the VRML animation parameters of the
previously constructed VRML facial model. The performance test of the presented system is demonstrated in Figure 5.11. The frontal views of the 3-D bottom half face model resemble the images in video frames in terms of shape and mouth opening motion.

Figure 5.10 VRML face model and its components.
Figure 5.11 Comparison between the bottom half of the odd numbered facial images of test sequence and corresponding VRML animation frames.
6. CONCLUSION

6.1 Conclusive Remarks

Virtual Reality (VR) environment has been applied to a number of areas. In the near future, we will experience this virtual reality not only in the scientific and industrial fields but also on our daily life. Considering the progress of VR in the last decade, we can imagine that we are going to meet the more humanized and intelligent virtual 3-D agents helping us on the Internet shops. There is no doubt that such an agent will interact with a user by means of the interactive animation of the 3-D model in VR. We will soon talk to each other through a cellular phone with our avatar displayed, sent from different geographical locations. One of the key tasks to realize such a human-machine interface is the image understanding technique. To understand images, image processing, vision analysis, artificial intelligence, and pattern recognition must work together. Researchers have devoted numerous efforts to interpret and fill the gap between the image and human perception of images. Limiting our scope to facial images, an approach to produce a 3-D talking face from a sequence of digital video images was demonstrated.

To realize authentic facial expressions from an individual contained in video source for MPEG-4 face animation, facial features were extracted, and feature points defined in the face model were determined. In the thesis, we developed and described all the stages involved in the model-based face animation system, which copied and animated the motion contained in a video sequence.

We have examined a face localization technique using a computationally efficient skin color filtering in the normalized $YCbCr$ color space with other information relevant to the face characteristics, and verified the concept of the facial feature extraction using an ICA approach. The ICA algorithm regards video frames as the independent
signals. Two significant advantages were noted. First, this approach finds the major components of a face and requires few notion of the face. The position of the major components of a face varies from individual to individual, and in a certain case, some parts do not appear at the expected locations. This may fail the algorithms, which heavily depends on the spatial relationship of the facial components. Therefore, only the basic spatial assumption of mouth and eyes is made in this approach. Good estimation of the approximate area of the major components is the other virtue of the ICA approach. In order to extract the area of the facial components, ICA requires less complicated methods compared with other existing methods, eventually, this made a subsequent analysis of individual facial components much easier. And, the feature point extraction technique avoided the critical threshold values to position the horizontal and vertical corner points of a mouth in most cases. Those points are matched to the human intuition.

Following the analysis of the facial components, we generated the VRML facial model and performed the face animation with extracted feature points, which are mapped to the animation parameter points in the 3-D face model. The face animation was viewed on the VRML browser, "Cortona". When a 3-D object is animated by VRML, the size of a VRML file containing the animation parameter data is inappropriately large. In this thesis, one of the VRML's API, JAVA program was used to overcome the file size problem and to meet the model-based animation scheme by accessing data files placed outside of a VRML file. This approach provides the face animation in very efficient way and meets the MPEG-4 coding concept. Experimental results showed that the suggested model-based face animation system integrating face localization, facial feature extraction, face modeling, and animation accomplished the goal of this project. Presented animation can perform under the data (uncompressed) transmission rate of 0.96 Kbps and 15.36 Kbps with and without the heuristic rules in the reciever's side respectively.

\(^1\) Cortona is an interactive 3-D object viewer developed by ParallelGraphics. It works as a VRML plug-in for popular Internet browsers such as Internet Explorer and Netscape Navigator.
6.2 Future Works

We have developed the model-based face animation system from the video sequence. But it has not evolved to a sophisticated system to cover all aspects of the concepts presented in this thesis. Therefore, there are more rooms to improve the presented system. In what follows, a number of suggestions are mentioned to overcome the limited abilities as well as to refine the performance of the presented system.

The skin color filtering is tested mainly on Caucasian, and the ranges of the chrominance value for the human skin color are set under the assumption that the speaker is Caucasian or Asian. Consequently, this is not a general solution for all races. Therefore, the chrominance space study of the wide range of the human skin color needs to be evaluated. And the system also needs to be improved to identify the hair portions, which has similar chrominance values to the facial skin area. The designed skin color filtering successfully applied to a number of video sequences but failed to distinguish the hair part of a specific image such as Suzie video sequence.

In many cases, it is natural that a face of an individual in-a video sequence is swayed, tilted, and rotated when he or she is talking. In those case the feature extraction processing becomes not reliable without compensating the orientation. There are a few concerns for the varying orientation. First, we consider when a facial image is rotated less than ±90 degree in X-Y coordinates. In this case, under the assumption that the facial shape is oval, we can apply the PCA analysis to the X-Y coordinates of the skin colored pixels, and find the major axis of the rotated image. Then, the image is reverse-rotated to have the upright face. The other case is when a face is tilted downward or upward. This case requires the upright and non-tilted face. Then, the length in the Y-coordinate of the face can be measured. The difference in length among tilted facial images and the position of the other non-skin colored facial components can provide the information of tilted angle.

The presented system is designed for the off-line applications because the parameter data files need to be compiled using JAVA program to interact with VRML before
the animation starts. We expect the VRML's API, JavaScript, solves this problem. The ICA approach is a robust method for locating major components of a face under certain conditions. The downside of this ICA approach, particularly for real time application, is that the multiple frames are required before the extraction process is started, and the training time needs to be considered.

Currently, the animation was developed for the bottom half of a simple frontal face. The rest of major parts of a face can be designed by the similar technique applied to the mouth area. And, it is possible to generate a simplified hair model using the similarity of the eyebrow to the hair in terms of chrominance components. In the heuristic rules presented, the lip movements in Z-coordinate are considered based on the horizontal length of a mouth. However, the facial muscle movements, such as the Z-coordinate around a cheek, were not addressed. Therefore, the expressions of smiling and laughing cannot be realized with the current rules. In addition, the texture mapping techniques must be applied to the face animation that requires the photo realistic images. When these enumerated subjects are considered, the more realistic presentation of face animation can be achieved.
References


