3702733223

FOR
REFERENCE ONLY

A New Method for Generic Three Dimensional Human Face Modelling for Emotional Bio-Robots

PARK LIBRARY
UNIVERSITY OF GLOUCESTERSHIRE
The Park Campus
Cheltenham GL50 2RH
Tel: 01242 714333

Xu Zhang

**A thesis submitted to
The University of Gloucestershire
in accordance with the requirements of the degree of
Doctor of Philosophy
in the Faculty of Media, Arts and Technology**

January, 2012

NOT TO BE
TAKEN AWAY

Abstract

Existing 3D human face modelling methods are confronted with difficulties in applying flexible control over all facial features and generating a great number of different face models. The gap between the existing methods and the requirements of emotional bio-robots applications urges the creation of a generic 3D human face model. This thesis focuses on proposing and developing two new methods involved in the research of emotional bio-robots: face detection in complex background images based on skin colour model and establishment of a generic 3D human face model based on NURBS. The contributions of this thesis are:

- A new skin colour based face detection method has been proposed and developed. The new method consists of skin colour model for skin regions detection and geometric rules for distinguishing faces from detected regions. By comparing to other previous methods, the new method achieved better results of detection rate of 86.15% and detection speed of 0.4~1.2 seconds without any training datasets.
- A generic 3D human face modelling method is proposed and developed. This generic parametric face model has the abilities of flexible control over all facial features and generating various face models for different applications. It includes:
  - The segmentation of a human face of 21 surface features. These surfaces have 34 boundary curves. This feature-based segmentation enables the independent manipulation of different geometrical regions of human face.
  - The NURBS curve face model and NURBS surface face model. These two models are built up based on cubic NURBS reverse computation. The elements of the curve model and surface model can be manipulated to change the appearances of the models by their parameters which are obtained by NURBS reverse computation.
  - A new 3D human face modelling method has been proposed and implemented based on bi-cubic NURBS through analysing the characteristic features and boundary conditions of NURBS techniques. This model can be manipulated through control points on the NURBS facial features to build any specific face models for any kind of appearances and to simulate dynamic facial expressions for various applications such as emotional bio-robots, aesthetic surgery, films and games, and crime investigation and prevention, etc.

Declaration

I declare that the work in this thesis was carried out in accordance with the regulations of the University of Gloucestershire and is original except where indicated by specific reference in the text. No part of the thesis has been submitted as part of any other academic award. The thesis has not been presented to any other education institution in the United Kingdom or overseas.

Any views expressed in the thesis are those of the author and in no way represent those of the University.

Signed Date

Acknowledgement

I would like to take this opportunity to express my heartfelt and sincere gratitude to my mentor, friend, and my first PhD supervisor, Dr. Shujun Zhang, for his insightful guidance, constant support, and continuous encouragement. During my doctoral study, he always provided invaluable advice to direct my thesis based on his immense knowledge and experiences. Without his great guidance, it would not be possible for me to sit here and write these words.

I would also like to thank my second supervisor, Dr. Kevin Hapeshi and the course leader of our department, Dr. Vicky Bush. They have been very supportive of my studies over the past three years that I have spent in the Department of Computing at the University of Gloucestershire.

My special gratitude is due to my parents. I owe my loving gratitude to my dear wife Julie as well for her years supportive accompany and encouragement. I feel incredibly lucky to have met her, loved her, and to have married her. She always stands there next to me and fights during the tough times together with me.

It has been an honour for me to have the priceless opportunity to be offered the studentship by the University of Gloucestershire. The financial support is gratefully acknowledged. The memories of my tutors and the University of Gloucestershire are precious for me and will last for the rest of my life.

Contents

List of Figures

List of Tables

List of Abbreviations

The abbreviations used in this thesis are listed below. All abbreviations are defined in full text at their first appearances.

Abbreviations	Full Name
2D	Two Dimensional
3D	Three Dimensional
AAM	Active Appearance Model
ANN	Artificial Neural Network
API	Application Programming Interface
ASM	Active Shape Model
AU	Action Units
BCL	Basic Class Library
BCM	Bayesian Classifier Model
Bio-Robots	Bionic Robots
B-Spline	Basis Spline
CAA	Computer Aided Analysis
CAD	Computer Aided Design
CAGD	Computer Aided Geometric Design
CAM	Computer Aided Manufacturing
CDF	Cumulative Distribution Function
CG	Computer Graphics
CLR	Common Language Runtime
CT	Computer Tomography
CV	Computer Vision
CVPR	Computer Vision and Pattern Recognition
ECCV	European Conference on Computer Vision
FG	International Conference on Automatic Face and Gesture Recognition
GLAUX	OpenGL Auxiliary Library
GLU	OpenGL Utility Library
GLUT	OpenGL Utility Toolkit
GUI	Graphic User Interface
HMM	Hidden Markov Model
ICCV	International Conference on Computer Vision

Abbreviations	Full Name
ICIP	International Conference on Image Processing
ICPR	International Conference on Pattern Recognition
IPF	Integral Projection Function
IRIS GL	Integrated Raster Imaging System Graphics Library
LDA	Linear Discriminate Analysis
LMCT	Local Maximum-Curvature Tracking
MCS	Model Coordinate System
MIT	Massachusetts Institute of Technology
MRI	Magnetic Resonance Images
MSE	Mean Square Error
NURBS	Non-Uniform Rational B-Spline
OpenAL	Open Audio Library
OpenCL	Open Computing Language
OpenGL	Open Graphic Library
OpenTK	Open ToolKit
PCA	Principal Component Analysis
PDF	Probability Density Function
PSNR	Peak Signal-to-Noise Ratio
RAD	Rapid Application Development
RBF	Radius Basis Function
SCS	Screen Coordinate System
SGI	Silicon Graphics Inc.
SVM	Support Vector Machine
TPAMI	Transactions on Pattern Analysis and Machine Intelligence
VCS	View Coordinate System
VPF	Variance Projection Function
WCS	World Coordinate System

CHAPTER 1

Introduction

1.1 Background

Human faces, as a nonverbal communication organ, are important for transferring information from one person to another. Facial expressions continually change throughout a conversation and these are constantly monitored by the recipient. According to Mehrabian (Mehrabian, 1968), of the information that is described in language by people "......7% is conveyed by words; 38% by the vocal tones and 55% by facial and body expressions". This study indicates the importance of human faces. With the rapid development and applications of computer-based technology, a lot of research have been carried out to study human faces and to build three dimensional (3D) face models for various applications, especially for emotional bio-robots, aesthetic surgery, crime detection and computer game etc. Since the first facial model was introduced by Parke (1972), numerous and verisimilar face models have been proposed and developed using a variety of approaches (Decarlo *et al.*, 1998; Vetter and Blanz, 1998; Shan *et al.*, 2001; Ansari and Mottaleb, 2003; Chaumont and Beaumesnil, 2005; Kim and Choi, 2008; Yan *et al.*, 2011). Except the motivation of the applications, the human faces with certain pattern and abundant individual diversities are ideal experimental objects to test 3D modelling algorithms and methods.

3D modelling techniques have been widely applied in Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM). There are several software such as 3ds Max (Autodesk., 2009), CATIA (Systemes, 2002), Maya (Autodesk., 2009), SolidWorks (Corp, 2009), and Rhino (2009) which can be used for 3D modelling. Various 3D applications are seen in our daily lives. Producers create 3D models for various film and TV productions to bring immersed sense for audiences (Fox, 2011).

Computer engineers introduces "3D" face models to create vivid characters in video games (Cnix, 2009). In virtual community, 3D face models can enhance the user experience significantly (Research, 2009). Moreover, 3D face model has the potential to be applied for web applications as well (Anaface.com., 2009).

Emotional bio-robot is an important application area of 3D modelling. Emotional bio-robot refers to robots which look like human beings that have a torso, arms, legs, and head. In addition, not only do they appear in the shape of humans, but can even act like humans in the way they move their body, speak, or even demonstrate complicated expressions (Takahashi *et al.*, 2007). The "human-like" automaton, which was dedicated to King Mu of Zhou by an ingenious craftsman named Yan Shi, is regarded as the world's first recorded bio-robot (described by *Lie Zi* in 250 BC) (Blümm and Bozic, 2009). Ever since, although the similar idea of a bio-robot has existed for a long time, bio-robot has progressed relatively slowly through history.

The research of emotional bio-robots has been significantly influenced by the development of 3D modelling techniques. Significant developments of emotional bio-robot research have been made in recent decades (Hashimoto *et al.*, 2006; Takahashi and Sato, 2010). This research has fundamentally supported the evolution of studies in entertainment, teaching activities, and medical therapy. One approach of designing the facial models of bio-robot is to include bones and muscles as human faces, whereby the more artificial muscles that are incorporated, the more realistic the face models are. In light of this trend, Hanson (2008) has demonstrated his own delicate Bio-robot which could speak and show fewer expressions, enabled by their high-tech polymer Frubber^{TM} (Mahyar., 2008). Later another realistic bio-robot, which looks like the renowned scientist Albert Einstein, was invented by Hanson in collaboration with the Institute for Neural Computation at the University of California (Fox, 2009). The Robotic Einstein uses approximately 32 motors to control 48 muscles on its face to reflect emotions by mimicking lifelike expressions such as raising the eyebrows, wrinkling the nose, or smiling (Lawsky, 2009).

However, even though several large strides have been made, the emotional bio-robot's capability of showing expressions is still limited. As shown in Figure 1.1 and Figure 1.2, human faces can be divided into several regions which contain different muscles under the skin. The expressions are too complicated to simulate because even the minor expressions involve multiple muscles. Difficulties also exist in transferring expressions of one bio-robot to another because there is no parametric representation of the face models.

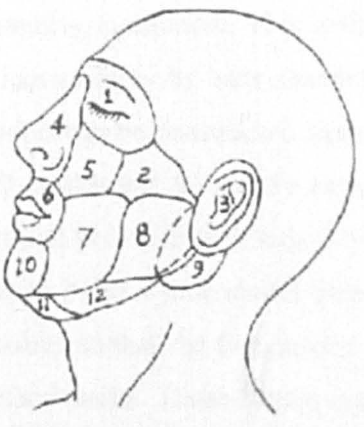


Figure 1.1 Human face anatomy
Source: http://dyfcl.w9.dvbbs.net/Article_Show.asp?ArticleID=3087

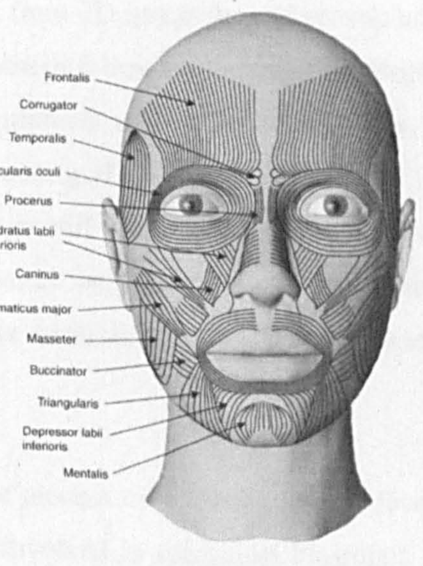


Figure 1.2 Facial muscles
Source: http://www.hypnox.ca/sciencedetail.htm

There are several requirements for emotional bio-robots, such as motion, intelligence and expressions. The intelligence of a bio-robot means it is capable to speak, make decision and detect objects in environment especially human faces. So the emotional bio-robots need the ability of face detection. Besides, the expressions are useful for bio-robots to communicate with human beings emotionally. To build an emotional bio-robot with human-like face, an important step is to design the face by a surface model. The 3D data is necessary for building a surface model. Currently, there are two methods to obtain required 3D data. The one is to use 3D scanner, the other is to extract information from detected faces in two dimensional (2D) images.

For 3D face modelling, the most intuitive method to obtain 3D data of a human face is to scan the face by 3D scanning equipment. This is the active vision method that retrieves 3D data points on human faces by laser scanner or structured lights (Geng, 2011). Then the 3D face models can be constructed based on the data points using a computer. However, the 3D equipment is usually expensive. The number of the collected data of the geometrical points from a face is normally very large, so it is difficult and time-consuming to build a face model from these points directly. The obtained data are scattered points so that the face model is represented by triangular patches rather than a true surface model. These factors restrict the use of the scanning method.

To extract 3D information from 2D image is another way to obtain 3D data. Because 2D image data is easy to obtain following the rapid development of the boom period of the personal digital equipment market in recent years, scholars have begun to explore methods of constructing 3D face models from 2D images (He *et al.*, 2008; Mena *et al.*, 2008). But it is still a challenging problem to recover 3D coordinates from images because images only contain 2D information. To investigate the problem of 3D information extraction, face detection would be the first step that should be studied.

Face detection refers to the process of detecting human faces in the input images. It is an important technique involved in emotional bio-robot. The appearance and the ability of recognising human beings are two sides of emotional bio-robot research: the appearance determined by the face modelling method defines how a robot presents in the world and the ability of recognise human beings determine how the robot observes the outside world. The environment in which the bio-robot is used is usually complicated. In this case, the ability of detect human faces from complex background is very important for emotional bio-robot.

The 3D face model required by emotional bio-robots should be capable of generating faces with arbitrary appearances through manipulating the facial features. The previous modelling methods only provide mechanisms to represent a 3D face from data points (Ahlberg, 2001; He and Qin, 2004). They were not mainly designed for representing facial features and especially their boundary information. Another approach for 3D face modelling is for professional engineers to design the model using 3D modelling software. But the accuracy will depend on the experience of the engineers. It is not easy for an ordinary person to create a face model.

Based on the analysis above, it can be easily seen that without a generic 3D face model, the manufacture of emotional bio-robot would a time consuming job for engineers to create robots with different faces. At this moment, the design process of robots' faces requires experts or experienced computer engineers to carry out. Hence, it is necessary to propose a new method for building a generic 3D human face model. A generic 3D face model can be used to help the engineers design and develop emotional bio-robots rapidly even when the robots' faces are not similar to any existing human beings, because the appearances of the robots can be easily changed by morphing some facial features of the generic model.

To summarise, as discussed above the requirements of emotional bio-robots make it necessary to establish a generic 3D human face model. Furthermore, to distinguish human faces from other objects is essential ability for a bio-robot to perform more humanistic as human beings and provide numerous functions. Therefore, this thesis will focus on two closely related problems: propose and develop a new face detection method and a generic 3D human face model.

1.2 Motivation

With respect to the two tasks of face detection and generic 3D generic face model of emotional bio-robots discussed above, the existing methods for dealing with these two tasks have problems as follows:

- For face detection, most existing methods are proposed based on training samples (Shavers *et al.*, 2006; Zhang *et al.*, 2008; Maglogiannis *et al.*, 2009). To build such a huge library, the permission from the participants should be obtained because of the issue of privacy. As soon as permission is obtained, time is still required to collect thousands of human face pictures. It is a complicated and time consuming process to obtain all the necessary permissions and to take all of the pictures. The performances of existing methods are affected by occlusions and illuminations.

- Different applications have different requirements with respect to 3D face models. Specifically, instant messaging applications or interactive online games may only require the contour of the faces to reflect the characteristics of a role. So 3D face model used in computer games are not necessary to be a true 3D model. Aesthetic surgery simulation requires accurate 3D model that is difficult to build using the existing methods. Emotional bio-robot focuses on the ability of deforming and flexible control over all facial features in order to show various expressions. Although the requirement of accuracy is lower than aesthetic surgery, it is still a challenging problem to generate a great number of robots with subtle differences of face appearances. In some situation, people may want that designed robots do not like any persons in the world, which would be a difficult task without a generic 3D face model.

- In some cases, the target people cannot present for scanning, even an image of the target person is difficult to obtain. For example, the clear frontal images of terrorists are rarely circulated in ordinary circumstances. So it is not feasible to build 3D face models using scanning or the image based methods. Hence, the advantage of generating face models for special target from a generic 3D face model is obvious.

The problems discussed above are the motivations of this thesis. The new face detection method and the proposed generic 3D face modelling method to solve those problems should have the following properties:

- It can be used to detection human faces without any training samples in colour images with complex background. It should be capable to deal with different occlusion and illuminations.

- It should be easy to obtain the source data for modelling. The face surfaces should be fully modelled in 3D.

- The generic 3D face model should incorporate flexible manipulations for each of the facial features. This will enable a specific person's face to be modelled with facial expressions. Moreover, a generic model will allow engineers to model different human faces for different requirements. It can also be used to generate faces that do not look like any persons in the world.

- The approach should be able to deal with asymmetric faces, which means that the regions of the face can be simulated separately. These properties are necessary because the differences between facial features cause diversity of appearances of human faces. This is an important advantage of a generic parametric 3D face model.

1.3 Aim and Objectives

The aim of this thesis is to research, design, and develop a new method for building a generic 3D human face model and a method for face detection in complex background. To fulfil the aim, the objectives of my thesis are as follows:

- To analyse the geometric structure and features of the human face, and explore potential applications of 3D face modelling for emotional bio-robots, aesthetic surgery, games and crime investigation and prevention.

- To propose and develop a face detection method that is based on skin colour information of the input image with complex background. This algorithm can be used for face detection by emotional bio-robots. It can also be applied for extracting the feature points from input image to generate a specific 3D face model.

- To investigate and evaluate existing methods of 3D human face modelling for various applications, especially for emotional bio-robot design and development.

- To explore the B-Spline (Basis Spline) technique and its extended form representation NURBS (Non-Uniform Rational B-Spline) for investigating the feasibility of apply B-Spline technique for building parameterised face model.

- To propose and develop a new method for generic 3D human face modelling based on NURBS techniques, including the curve model and surface model.

- To design and implement a software environment for verifying and validating the proposed method of constructing a generic 3D face model and a face detection algorithm.

This thesis will seek to make several original contributions to the new knowledge. The expected outcomes include:

- To propose and develop a new face detection method in colour images based on skin colour information, which can be used to provide face detection function for bio-robot to locate human faces in the images captured by camera. The other potential application of this method is for capturing facial feature boundaries data to build specific face model.

- To propose and develop a new method for building a generic 3D face model, using a curve model and a surface model. The generic face model can be then used to construct any human face model. The facial features of the built 3D human face model can be manipulated to represent different expressions through. The control points, knot vectors and weight coefficients on the curve and surface models.

- To determine the control points, knot vectors and weight coefficients, a reverse computation algorithm of NURBS curves and surfaces will be proposed and developed for building a generic 3D face model. This reverse computation algorithm can be used to reversely determine the necessary parameters to evaluate the curves and surfaces, including the control points, knot vectors, and weight coefficients from the data points of a human face.

- To design and implement software environment to verify and validate the proposed new generic 3D face model. This software environment enables the users to validate the functions provided by the generic 3D face model, such as facial feature deformation and generating specific human faces.

Among these expected outcomes, the contributions to new knowledge generation are:

- A new skin colour based human face detection method in complex background images will be proposed and developed. The proposed algorithm is consists of skin colour model for detecting skin regions and the geometric rules for distinguishing faces from detected regions.

- A new method will be proposed and developed for generic 3D human face modelling. Based on this method, a generic 3D human face model will be designed and developed. This model is geometrically parameterised and flexibly controlled by users, and hence can be used to build various 3D face models for different applications. It includes:
  - A human face will be divided into 21 surface features. These surfaces have 34 boundary curves. This featured-based segmentation enables the independent manipulation of different geometrical regions of human face.

  - The NURBS curve face model and NURBS surface face model will be specially designed and developed. These two models are built up based on a new cubic NURBS reverse computation algorithm.

  - A new method for building generic 3D human face models will be proposed and implemented based on bi-cubic NURBS through analysing the characteristic features and boundary conditions of NURBS technology. This model can be controlled through control points on the boundary NURBS curves to build any specific face models for any kind of face appearances and to simulate dynamic facial expressions for various applications such as emotional bio-robots, aesthetic surgery, game and crime investigation and prevention etc.

1.4 Methodology

This thesis focuses on the establishment of a face detection method and a generic 3D human face model for design and development of emotional bio-robots. A literature review is needed in order to gather knowledge which will contribute to the aims of this thesis. It explores the information related to 2D image processing for face detection, NURBS techniques, 3D representing and manipulation.

The research process is both exploratory and experimental. For this thesis, a research routine of theoretical investigation, proposition of new algorithms and their implementation through computer programming will be adopted.

For face detection, a number of skin colour pixels will be collected. The statistical information of these skin colour pixels will be analysed. Based on the analysis, a new 2D skin colour based human face detection method will be proposed based on theoretical study and literature review. To improve, verify and validate the proposed method, a demo programme will be design and implemented in Matlab.

For 3D face modelling method, a new method and two algorithms will be firstly proposed through theoretical study of 3D geometry computation, NURBS technology and systematically literature review. Then this method will be used to build a generic 3D face model. A software environment will be design and developed in C++ to modify, verify and validate the proposed method.

1.5 Thesis Outline

The rest of this thesis is organised as follows. Chapter 2 reviews the algorithms and approaches related to this thesis. Popular face detection approaches are classified into four main categories, and their advantages and disadvantages are discussed. 3D modelling methods are also explored in the chapter after examining approaches to

3D data acquisition. The applications of generic 3D face model for emotional bio-robots and other directions are discussed as well in Chapter 2.

Chapter 3 discusses related work about face detection and NURBS techniques. The transformations between different colour spaces are deduced for constructing skin colour models used for face detection in the thesis. Popular image processing techniques including smoothing, edge detection, binarisation, and morphological operations are also described. This chapter also introduces the method to build up skin colour models through studying the characteristics of skin colours. In the second part of this chapter, the NURBS techniques are discussed. As the generalised representation of Bézier and B-Spline, NURBS has geometric invariability and affine invariability to make it suitable for 3D modelling. The process of forming a surface from curves is introduced at the end of the chapter.

In Chapter 4, a new face detection method based on skin colour model is proposed and implemented. Two colour spaces are combined together to provide strong rules to restrict the skin colour representation. Firstly, all face candidates are segmented by the skin colour model. Then, geometrical properties of the eyes region are used to verify the face candidates and improve the efficiency of the algorithms. The implementation and evaluation are carried out. By comparing to other previous face detection methods, the figures shows that the new proposed method can achieve comparable detection rate (86.15%) and detection speed (0.4~1.2 seconds) without any training samples. This new method could be applied to emotional bio-robots for detecting human faces, which could enhance the functionalities of emotional bio-robots.

In Chapter 5, the author proposes and implements a new method for building a generic 3D human face model. Firstly, based on the face characteristics analysis, the author defines 34 NURBS curve features and 21 surface features on a human face. Then, the chapter introduces the design of classes and data structures that hold the information of NURBS curves and surfaces. Next, the author proposes and develops reverse computation algorithms for both NURBS curves and surfaces. An example of NURBS curve reverse computation process is represented to test the performance of reverse computation algorithm. The times spent on reverse computation never exceed 16 milliseconds. Finally, the generic 3D face model is established based on the reverse computation method. The advantages of the proposed method for building a generic 3D human face model are discussed by comparing to two typical existing 3D modelling methods.

Chapter 6 shows the integrated implementation to verify the approach of constructing the generic 3D face model. Based on the OpenGL and .NET framework, the application provides a group of interactive interfaces for model manipulation and facial feature deformation. This work is for demonstration, verification and validation of the proposed 3D modelling methods.

Finally, Chapter 7 summarises the while thesis. The contributions and the limitations are critically reviewed. The further work to improve the quality of my thesis is also discussed.

CHAPTER 2

Literature Review

This chapter provides a literature review about the two tasks of emotional bio-robots: face detection and 3D human face modelling. Section 2.1 presents the historical developments of face detection approaches, Section 2.2 critically discusses the 3D data acquisition methods, and Section 2.3 reviews 3D modelling techniques including Bézier B-Spline and NURBS. The applications of 3D face modelling are discussed in Section 2.4.

2.1 Face Detection Approaches

The face detection is a research area that involves in human-computer interaction, pattern recognition, and computer vision (CV) or machine vision. It is the fundamental and preliminary step of other human face processing techniques, such as face recognition, facial feature extraction, 3D face modelling, and 3D facial expression recovery. Generally speaking, face detection is the process of looking for human faces in the input images or video clips as well as outputting the information (locations, sizes, poses, etc.) of detected faces.

The face detection was initially introduced as a key problem inherent in automatic face recognition systems (Talele and Kadam, 2009; Or and Wilson, 2010). As the initial phase of any face processing system, the incipient face recognition problem is based on the hypothesis that a frontal "mugshot" picture, which is a photograph of someone's face region, especially a criminal's, or one taken for official purposes, with strong constraints has been obtained. However, given the rapid developments of image recording equipments, this assumption is no longer valid. To detecting human faces from image remains challenging because no prior knowledge exists about the sizes, positions, directions, or poses of the possible faces. Moreover, different facial expressions, occlusions, and lighting conditions make the task much more difficult to handle. Thus, face detection is now studied as an independent problem separate from face recognition.

The face detection is currently attracting the research interests of a number of universities and institutes throughout the world such as the Massachusetts Institute of Technology (MIT), Carnegie Mellon University, Yale, Cambridge, Oxford, China Academy of Science, The Chinese University of Hong Kong, Tsinghua University, and Peking University etc. In addition, numerous international conferences relate to face detection techniques, such as the International Conference on Pattern Recognition (ICPR), Computer Vision and Pattern Recognition (CVPR), International Conference on Computer Vision (ICCV), European Conference on Computer Vision (ECCV), International Conference on Image Processing (ICIP), Transactions on Pattern Analysis and Machine Intelligence (TPAMI), and International Conference on Automatic Face and Gesture Recognition (FG). Several popular face databases have been established to support these research efforts as well as benchmark the algorithms and methods, such as FERET (NIST., 2008), CVL (Peer, 2003), Yale (Georghiades, 2005), Multi-PIE (CMU., 2009), ORL (Lab, 2006), and CAS-PEAL (CAS., 2005) etc.

According to Yang *et al.* (1999), in terms of the input sources, face detection can be classified into two categories: still image face detection and video sequences face detection. But these two categories can be integrated if the frames of the input video sequences are considered as a series of still images. Thus, only the face detection in still images is discussed in this thesis.

Technically, the face detection is a pattern-matching problem reflecting two minor problems: the need to determine if faces occur within an input image and, if the answer is yes, the need to extrapolate the face area(s) from the entire image. Several issues should be considered to address this demand:

- Information regarding whether the input image contains human face(s) or not is unpredictable; thus, the algorithm should be capable of dealing with both conditions separately.

- The human face is not a fixed pattern as it readily affected by different races, skin colours, ages, and emotions.
- Various accessories such as beards, glasses, hair, and hats make it difficult to detect human faces effectively.
- Due to the environmental illumination, even pictures of the same person vary because the human face has three dimensions.

All existing face detection methods use particular strategies based on various facial features to determine whether an input image contains a human face or not. The face detection methods could be classified into several categories according to the features they are based on, as shown in Figure 2.1. Popular face detection methods and algorithms can be categorised as (1) geometrical knowledge based methods; (2) template matching methods; (3) statistical learning based methods; and (4) skin colour based methods. The remaining parts of this section review the developments and studies for each category.

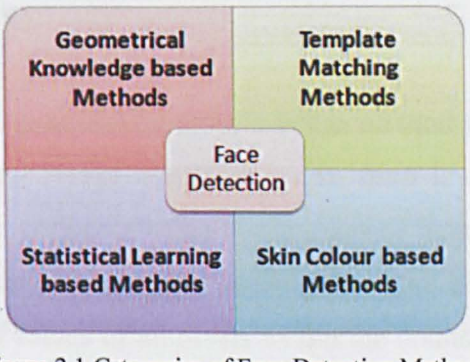

Geometrical Knowledge based Methods
Template Matching Methods
Face Detection
Statistical Learning based Methods
Skin Colour based Methods

Figure 2.1 Categories of Face Detection Methods

2.1.1 Geometrical Knowledge based Methods

Geometrical knowledge based methods are based on the general characteristics of human facial features. Typical human facial features are generalised and encoded to form a universal face rules library that can be used to sift all facial feature candidate regions from the input images (Yap *et al.*, 2009). These candidate regions are then investigated according to the common geometrical rules of human faces. It is suitable to use the geometrical relationship between local features of human faces to detect faces in input images as certain patterns related to local features of human faces do exist (Talele and Kadam, 2009). These patterns include the contour of human faces, which is normally considered as an ellipse; the edges of facial features, which can be fitted by parabolas; the eyes, which always appear symmetrically in the top half of any human face; and the line through the centres of mouth and nose, which is perpendicular to the line through the centres of two eyes. A face candidate is considered to be found while the relative distances and locations between those facial features meet the conditions listed above. These face candidates will be verified one by one to determine if they contain faces or not.

The ellipse contour of the human face is very important geometrical information for detecting faces (Huang and Su, 2004; Hsu *et al.*, 2010). Tan and Wang (2000) extract the edge information from the histogram quantised input images, then try to recover the ellipse shape by connecting those edges using generalised Hough transformation and optimised energy functions. However, this method is a kind of semi-automatic method because the number of the faces should be known before detection. So this method can only used to detect if there exist any faces in an image, but cannot be used to find out automatically how many faces and the other features/organs such as eyes, nose and mouth etc.

Yang and Huang (1994) propose the mosaic image method for face detection. They first decompose the input image into three layers, from lower resolution to higher resolution based on the greyscale value distribution of human faces. Each layer is then divided into a group of small grids of the same size. The value of each grid is assigned to the mean of values of all pixels within the boundaries of the grid. These grids are checked by a set of rules to evaluate the probability of being a human face. Next, by halving the side length of the grid, they construct the next level of the mosaic image, within which the facial features like eyes, nose, and mouth are searched. Finally, all human face candidates in the input images are verified using edge tracking in the binarised images obtained from the first two steps. Singh (2007) uses similar method to recognise faces by a registration and blending procedure through multiple images. For these methods, as the probability between the input image and the gallery of mosaic image are calculated during the detection step, a gallery of mosaic image is required. It is time consuming and difficult to develop the gallery of mosaic image. So its application is very limited.

Kotropoulos and Pitas (1997) introduce a rule-based method that localises facial features using the projection function. The value at position (x, y) in $m \times n$ greyscale image is denoted by $\boldsymbol{I}(x, y)$. The horizontal and vertical projections are defined as $\boldsymbol{I}_H(x) = \sum_{y=1}^{n} \boldsymbol{I}(x, y)$ and $\boldsymbol{I}_V(y) = \sum_{x=1}^{m} \boldsymbol{I}(x, y)$. The left and right edges of the face can be detected by looking for the local minimal values of $\boldsymbol{I}_H$. Similarly, the top and bottom edges of the face can be obtained by looking for the local minimal values of $\boldsymbol{I}_V$. The positions of local minimal values of $\boldsymbol{I}_H$ and $\boldsymbol{I}_V$ are also used to localise facial features like the lips, nose centre, and eyes, comprising the whole face.

Feng and Yuen (1998) developed a method that they named Variance Projection Function (VPF), which is a variation of the projection function (Kanad, 1973). This method makes use of the horizontal and vertical projections of the greyscale input image followed by fitting the edge of facial features to detect faces and get better results (Figure 2.2).

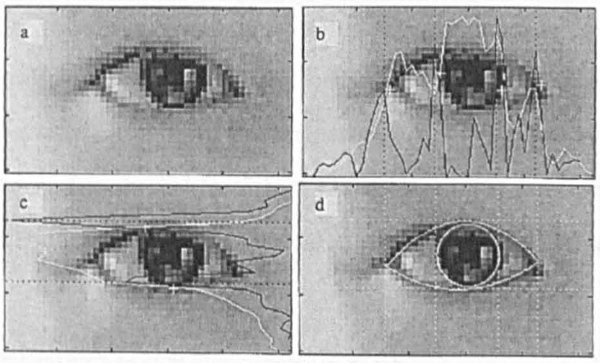


Figure 2.2 VPF method developed by Feng and Yuen

The geometrical knowledge-based methods are easy to implement as these methods require only simple lower level information of input image for face detection. For the frontal face images, Feng and Yuen (Feng and Yuen, 1998) offer an effective verification mechanism with the lower failure rate. Nevertheless, this type of method cannot provide good solutions for images taken from other perspectives. Moreover, such methods also have strict restriction for the input images, which both the in-plane and out-of-plane rotation angles of the face should be small.

Another problem of this type of method is that it is difficult to convert human knowledge into unambiguous rules for the face detection. These methods are not universal algorithms because the effectiveness is strongly based on the set rules of prior knowledge. On the one hand, a face candidate cannot pass the test of the well-defined rules that are too strict; on the other hand, the failure rate rises significantly when the pre-defined rules are too loose.

2.1.2 Template Matching Methods

Template matching is another popular technique used in face detection methods. This technique is based on the fact that the contours of the human face as well as the facial features (including eyes and the mouth) are approximate ellipses. Thus, face candidates can be located by scanning the input images and matching the regions to a pre-defined face template.

Sakai *et al.* (1969) constructed a face template with eyes, nose, mouth, and face contour to detect human faces from frontal face images. Each sub-template is defined using segmentation lines of human faces. The segmentation lines of the input images are extracted using the maximum gradient variation method to match the sub-templates by calculating the similarities between them, one after another. These similarities can be used to evaluate the probability of a face candidate being a real human face. Although it is useful to detect faces in frontal face images, the gradient based method restricts the application of this method to colour images.

A hierarchical template matching method was proposed by Miao *et al.* (1999). Using this method, the input image rotates from -20° to 20° in 5° increments in the first stage. A Laplacian operator is then used to extract edges and obtain a multiple resolution image sequence. The pre-defined face template is composed of the edge information of six facial features: two eyebrows, two eyes, nose, and mouth. Finally, face candidates are verified through a heuristic search of the boundaries. However, like other template matching methods, this method often outputs wrong faces in some situations such as the initial position of the template mismatches the position of a face if the template is not place accurately. So this method is also limited in practical applications.

Liang *et al.* (1999) introduced the method of average face template matching. By considering the special role of eyes' position in the humanities-based process of face recognition, their method applies an eye template as the initial filter before detecting and localising faces in the input image using multiple face templates with different width-height ratios, as shown in Figure 2.3. They process the input images using a fixed compression ratio to tackle the problem of detecting faces in various resolutions. However, this method can only be used for detecting single face by multiple face templates.

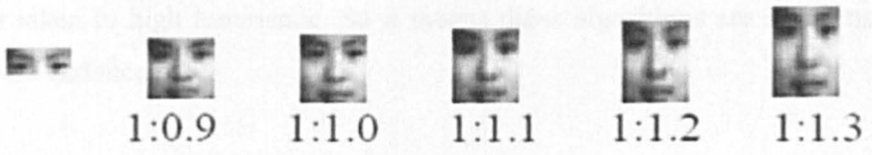


Figure 2.3 Liang *et al.*'s eye template and multiple face templates

Yuille *et al.* (1992) developed a deformable template method for facial features extraction. The deformable template method (Jain *et al.*, 1998) includes an adjustable parametric template, which reflects the shapes of target objects and its energy function designed based on the greyscale information of the input images and shapes of target objects. The concept involves evaluating the energy function and adjusted the parameters of the template while moving the parametric template in the input images. The deformable template is determined by its position and parameters. The template and the face shapes are considered to be well matched while the energy function reaches its minimum. Hence, a face is detected. This method is suitable for dealing with objects of different scales and poses because the parameters of the deformable template are adjustable. However, the accuracies of the pre-defined template methods vary as there is no constraint of parameters.

The Active Shape Model (ASM) coined by Cootes *et al.* (1995) can be used to achieve a higher accuracy for face detection because it makes use of constraints between facial features to restrict the searching shape parameters to reasonable shapes by training the template models. Cootes and Edwards (1998) subsequently developed the Active Appearance Model (AAM) based on ASM. In contrast to ASM, AAM can achieve higher approximate accuracy by combining the shape variation model and texture variation model with prior knowledge to guide the optimisation process.

Hou *et al.* (2006) refined the ASM method based on the gradient statistical characteristics and texture shape constraints. Wu *et al.* (2008) improved ASM's result and robustness by searching the facial feature points processed by Gabor wavelets transformation to reduce the dimensions. Other researchers (Fan *et al.*, 2009; Lu *et al.*, 2010) have also proposed improved algorithms based on ASM and AAM. The algorithms cannot be used to achieve good results when they are applied to pictures taken in high luminance. So it means these algorithms are still sensitive to luminance variance.

As no prior knowledge exists about the size and position of the input images, template matching methods need to scan the input images using standard face templates of different scales and calculate the correlation between the current sub-region and the templates. The correlation determines whether the current sub-region is a face candidate or not. All possible face candidates need to be further verified based on other features. The results highly depend on the pre-defined template. All the parameters of the face template need to be adjusted dynamically during the process of the algorithms of this approach, so that this approach is computation-intensive and time-consuming.

2.1.3 Statistical Learning based Methods

Methods based on statistical learning consider the face region to be a fixed pattern. Some classifiers are constructed by training them with numerous positive (face) and negative (non-face) samples. These classifiers can be used to detect faces by determining the pattern to which all possible face candidates belong. Actually, the problem of the face detection is converted into a dichotomy problem of statistical patterns. Along with the development of the computation abilities of modern computers, these methods have attracted more and more attention. Seven popular statistical learning-based methods are: (1) Artificial Neural Network (Zhang and Fulcher, 1996; Garcia and Delakis, 2004; Nazeer *et al.*, 2007); (2) Principal Component Analysis (Gottumukkal and Asari, 2003; Lang and Gu, 2009; Paul and Gavrilova, 2011); (3) Linear Discriminate Analysis (Kim *et al.*, 2003; Kobayashi and Zhao, 2007); (4) Support Vector Machine (Hyungkeun *et al.*, 2004; Shavers *et al.*, 2006; Zhang *et al.*, 2008); (5) Bayesian Classifier Model (Liu, 2003; Choi and Oh, 2005; Park *et al.*, 2005; Matsui *et al.*, 2008); (6) Hidden Markov Model (Nefian and Hayes, 1998; Nefian and Hayes, 1999; Nefian and Hayes, 2000; Bicego and Murino, 2004; Aleksic and Katsaggelos, 2006); (7) Adaboost (Lang and Gu, 2009; Du *et al.*, 2010; Ma and Du, 2010).

(1) Artificial Neural Network (ANN)

ANN hides the statistical characteristics patterns implicitly in its structural parameters as it has unique advantages to represent complicated and difficult patterns used to describe human faces explicitly (Nazeer *et al.*, 2007). Nazeer's ANN system was developed for learning the complex conditional density patterns of both face and non-face samples. One advantage of this ANN system is that it can be easily constructed as classifiers. The problems introduced by assuming those densities manually can be avoided.

Rowley *et al.* (1996) proposed a two-phase neural network-based face detection system. The first phase of the system is the neural network classifiers determining whether an input image with a fixed size is a face or not. The second phase is duplication testing and merging process based on a single layer neural network. This system can only handle frontal face without rotation. Afterward, Rowley *et al.* (1998) developed a refined method that can deal with human faces with arbitrary rotation angles. However, their method requires a great number of sample images to train the neural network in advance.

Bakry and Stoyan (2004) developed a fast neural network-based face detection method taking into account symmetry. New image is generated by rotating the input image symmetrically. Then, by examining the correlation property of the new image in frequency domain, this algorithm achieves equivalent results with relatively a small number of training samples and neurons in the hidden layer. The problem of training samples still exists because it is necessary for training the network.

(2) Principal Component Analysis (PCA)

PCA (Bakry and Zhao, 2006) is a classic algorithm for pattern recognition. It implements optimal analysis and reconstruction of human faces by mapping high dimensional face vectors to a sub-space composed of several eigenvectors—also known as Eigenface (Turk and Pentland, 1991)—through Karhunen-Loeve (K.L.) transformation. The method based on Eigenface is divided into two steps: (1) The K.L. transformation is applied to the matrix constituted by a large number of sample face vectors; the sub-space formed by the eigenvectors is called Eigenface. (2) During the detection step, the input image is projected to the sub-space to reconstruct a face represented by the sub-space. The level of a desired signal to the level of background noise (signal-to-noise ratio, SNR) between the input image and the reconstructed image is the calculated. The input image is considered to be a face while the SNR is bigger than a specific threshold. The problem of this method is that it is difficult to determine the threshold. It is usually chosen based on personal preferences.

Sung and Poggio (1998) at MIT created a face sub-space using "face" and "non-face" samples with a resolution of 19×19 pixels. A PCA algorithm is applied to the vector of 19×19 dimensions composed by pre-processed positive and negative samples. Then, using a K-Means cluster algorithm (Mahajan *et al.*, 2009), they established six "face" clusters and six "non-face" clusters which surround the "face" clusters in the feature space, so that the boundaries of "face" and "non-face" patterns are clarified clearly. A multiple layer perceptron is trained by the distances from the samples to the clusters' centres. To address the difficult problem of developing rules of choosing the "non-face" samples, they introduced a bootstrap method in which they initialise a classifier using only "face" clusters to detect a group of images. All error results, which are "non-face" but classified as a "face", are added into the pool of "non-face" samples. The new "face" and "non-face" classifiers are then used to detect the input images again. This process repeats until abundant "non-face" samples are picked up. The problem of this method is that both the positive and negative samples should be pre-processed for training the classifiers. Those samples need to be resized to the normalised resolution for detection.

(3) Linear Discriminate Analysis (LDA)

The PCA (or Eigenface)-based approach creates the sub-space according to representation of the original dataset; it usually does not lead to the best classification results as it does not take into account the differences of inter-classes and intra-classes. The Fisher face method was proposed to address this flaw. Unlike the Eigenface method, LDA - which is based on the theory of Fisher Linear Discriminate Analysis (FLDA) (Principe *et al.*, 1997) - can achieve minimal inter-class divergence and maximal intra-class divergence. Therefore, the sub-space generated by FLDA (Kobayashi and Zhao, 2007) is more suitable for the pattern recognition problem than PCA because it is designed for best separability.

Yang *et al.* (2000) developed the classic FLDA method by classifying "face" and "non-face" samples into 25 classes using a Self-Organising Map (SOM) (Kohonen, 1995). The projection matrix of each class's inter-class divergence and intra-class divergence is computed. The faces are detected by sliding a rectangle window on the input image and calculating the probability by maximum likehood estimation.

(4) Support Vector Machine (SVM)

SVM proposed by Vapnik (1995) is another statistical theory. Vapnik argues that the minimal empirical risk cannot always ensure minimal expected risk; thus, he developed the theory of Structural Risk Minimisation (SRM) as well as the concept of the Vapnik-Chervonenkis Dimension (VCD). He demonstrated that the minimal expected risk can only be achieved when the SRM and VCD reach their minimums.

The SVM method was used to achieve some good results for face detection when tested by Osuna *et al.* (1997). Their method processes each 19×19 test window using SVM to distinguish "face" or "non-face". However, SVM training process needs to solve the problem of Quadratic Programming (Qian and Zhanbin, 2008), which is a computation-intensive process. The training speed depends on the number of support vectors. So the execution time of this method can be reduced by reducing the number of support vector, but the detection accuracy of the method will decrease at the same time.

(5) Bayesian Classifier Model (BCM)

BCM is a classifying method based on the probability model (Matsui *et al.*, 2008). It checks all possible image windows according to the posterior probability that a region of the imputed image belongs to a face pattern. Schneiderman and Kanade (1998) from CMU proposed the face detection method based on posterior probability. Their method converts the problem of evaluating posterior probably to likelihood solving problem by Bayesian theory. Due to the difficulty of evaluating posterior probability, they introduced a ratio parameter as the indicator of the detector sensitivity. Each 64×64 pattern region is divided in to 16 sub-regions. The complexities of "face" and "non-face" pattern distribution expressions are reduced by several assumptions, such as the individuality of sub-regions. Finally, these expressions are encoded as a sparse histogram that can be used to check if a region contains human face. For this method, the training samples are necessary and there must be an extra image size standardisation step.

(6) Hidden Markov Model (HMM)

HMM is another probability-based method used for face related research (Samaria, 1993; Ballot, 2005). Many hidden states are evaluated to form a pattern. HMM is trained according to the traditional inter-states probabilities from samples represented as observation sequences. The process of HMM training can be used to maximise the training dataset's probabilities by adjusting the parameters of HMM. The categories are determined according to the output probabilities.

Nefian and Hayes (1999) represented the human face as a 2D embedded HMM based on the fact that the order of features in a frontal face image preserves the changeless nature structure in both top-to-bottom and left-to-right directions. They use 2D discrete cosine transformation (DCT) coefficients as observation vectors. A set of images with different people need to be collected to train the HMM. This method is only useful to process frontal face image.

Chen and Qi (2002) proposed a self-organised HMM method for face detection, training the HMM using multi-view face samples to obtain the initial estimated parameters. The weak connexions between states are then pruned to make the network self-organised as a Multi-Path Left-Right model. The faces are detected using HMM, whose parameters are re-exterminated by Expectation-Maximisation algorithm.

(7) AdaBoost

The boosting method is a statistical learning method that combines weak classifiers into strong classifiers (Dong *et al.*, 2004; Wu *et al.*, 2008). The basic concept is to assign greater weights to the failure training samples so that the learning algorithm focuses on the failure sample more than other samples (Liu *et al.*, 2009). The strong classifiers consist of several selected weak classifiers for detection.

Viola and Jones (Viola and Jones, 2004) proposed a real-time face detection system based on integral image, cascade classifiers and AdaBoost algorithm. The framework includes three parts. First, the Harr-like features are used to represent faces, developing the integral image to enable rapid feature evaluation. Second, some strong classifiers are constructed by weighted voting from several important weak classifiers selected by the AdaBoost algorithm. Finally, the strong classifiers are connected as a cascade classifier that can speed up the detection process effectively. Zhang *et al.* (2002) developed this method and built up the first real-time multi-view face detection system.

Li and Zhang (2004) proposed a coarse-to-fine, simple-to-complex pyramid face detection system named "FloatBoost". Unlike the AdaBoost method, FloatBoost adds a feedback mechanism after each iteration step. The FloatBoost is better than AdaBoost because it aims to minimise the failure rate during the learning process. These authors also proposed an improved training model to achieve a lower failure rate using fewer weak classifiers.

In summary, the method based on statistical and machine learning (Figure 2.4) is a popular face detection method due to the following merits:

(1) It does not rely on the prior knowledge and parameters model so that it can avoid the faults caused by inaccurate and incomplete knowledge.
(2) The parameters of model are learned from samples so that it is more reliable and robust by expanding the samples library.
(3) This method has proven to be effective for detecting human faces from images with complex background.

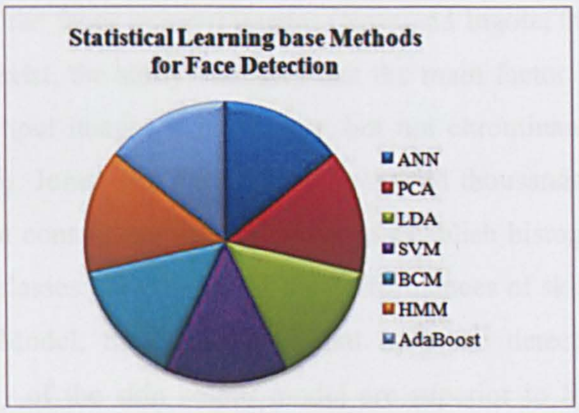


Figure 2.4 Statistical Learning-based Methods for Face Detection

However, this method also has some disadvantages:

(1) The required training samples should be consistent to build up the statistical model. Although it is easy to collect "face" samples, the "non-face" samples are difficult to collect.

(2) Because this method requires an exhaustive search for all possible detecting windows, which introduces extensive computation, the speed of statistical learning based method still has the potential to be improved.

(3) Due to the limitation of detecting speed, it is hard to use all information of the high resolution image. The common way to balance this issue is to reduce the resolution of the detecting window.

(4) The information used for training the model is usually high dimensional statistical information; the relationships among these information and intuitive physical features are ambiguous.

Due to the above problems, the application of statistical based method has been limited.

2.1.4 Skin Colour based Methods

The colours of the human skin are very important information which does not rely on the detailed facial features (Fang and Qiu, 2003). The colours are suitable for distinguishing face regions from backgrounds in most cases as the skin colours of different human races have particular distribution in specific colour spaces (Hore and Ingole, 2010). The colour information of human skin has a good clustering property with a stability in specific colour spaces, which does not vary along with the size, posture, or emotions of the faces in input images (Hore and Ingole, 2010). Although individual differences exist, the study indicates that the main factor influencing the human skin colour in input images is luminance, but not chrominance (Graf *et al.*, 1995; Graf *et al.*, 1996). Jones and Rehg (2002) collected thousands of pictures of human skin regions that contain millions of pixels to establish histogram models of "skin" and "non-skin" classes. By comparing the performances of skin colour model and Mixed Gaussian Model, they concluded that both the detect accuracy and computation complexity of the skin colour model are superior to Mixed Gaussian Model (Jones and Rehg, 2002). Thus, skin colour is an effective property for human face detection.

The skin colour feature is represented by the skin colour model, which closely relies on various colour spaces. Two criteria relate to choosing suitable colour space: (1) Is there a certain colour model that can demonstrate the distribution of skin colour in a given colour space? (2) Can the "skin" region be effectively distinguished from the non-skin region in the given colour space? Popular colour spaces used for human face detection include *RGB*, *nRGB*, and *YCbCr* (Vezhnevets *et al.*, 2003). Gaussian Model, Mixed Gaussian Model, and Histogram Model are common methods in this category.

As the colour information of input images is intuitive, researchers have proposed various approaches for face detection based on colour information. Yoo and Oh (1999) fitted regions segmented by skin colour pixels' connectivity using ellipses. They detect faces based on the ratio between the lengths of the major axis and the minor axis of each ellipse. Cai and Goshtasby (1999) obtained skin colour regions by gradually dilating the local maximum of likelihood computed using pixel values and the skin colour model. The greyscale mean face template is used to verify if a skin region contains a human face. Hsu *et al.* (2002) detected face regions in the *YCbCr* colour space obtained using the light compensation technique and nonlinear colour transformation. They then verify each region and locate facial features according to eye and mouth maps. Lee *et al.* (1996) proposed an automatic face detection system in a complex background based on motion and colour information.

The skin colour based methods have two advantages. First, compared to other methods discussed above, skin colour based methods are very fast. The searching space is significantly reduced as the possible face regions are picked from input images using the skin colour model. Second, because this type of method only relies on skin colour, it can be robustly applied in a wide range of applications. However, the application of this method is limited because it is difficult to summarise a generic skin colour model for different human races. Moreover, the skin colour representation may vary and be sensitive to the light, shadow, and occlusion conditions.

2.2 3D Data Acquisition

3D data acquisition is an important step for 3D modelling. It is not only the first step of digitalising the objects surfaces, but also a key factor of choosing proper methods and algorithms for 3D surface representation and error analysis (Nadia and Bonanni, 2010). The methods for 3D data acquisition can be divided into two main classes: contact measuring method and non-contact measuring method. Contact measuring method appeared earlier than the non-contact measuring method. Zhang *et al. (2004)* have reviewed methods for 3D modelling of biological objects.

For the contact measuring method, a probe moves on the surface of the target object point-by-point to collect 3D data. This method results in higher accuracy and is effective for objects with less characteristic surfaces. However, the system is usually expensive. The measuring process takes time to carry out and may cause damage to the objects. Moreover, human aid is necessary in most situations.

Non-contact measuring is a type of high speed 3D measuring method boosted by the developments of CV techniques (Xiao *et al.*, 2007). This method is suitable for complex surface measurement because there is no need for any 'probe' to come into contact with the objects.

2.2.1 3D Scanning

The 3D scanning method was developed in the 1990s. It is a type of active stereo vision methods. The data obtained using these 3D scanners - so called "clouds" - is complete 3D information including 3D coordinates, colours, and textures rather than the profile images of an object. The number of points in "clouds" varies from hundreds of thousands to millions. Based on these "clouds", different algorithms are developed to build a 3D model. Waters and Terzopoulos (1991) built up a complete realistic modelling and animation of human faces using scanned data from the physical based approach. Lee *et al.* (1995) obtained the entire information for a 3D face using this type of equipment. Guebter (1998) tracked 182 colour points on the human face to record the positions of these points in different emotions. These positions were then used to modify a pre-defined 3D model scanned using 3D scanners to simulate the exact face model. This method proved to be effective in getting good results. However, extra processing had to be applied to eliminate colour information from the texture data to get the final textured model.

Successful commercial examples of 3D scanning systems include Cyberware (2010) (Figure 2.5), Eyetronics (2010), and Bumblebee (Grey, 2010) (Figure 2.6) stereo vision systems. These applications can apply in different resolution requirement scenarios. The accuracy of the face model generated using a 3D scanner depends on the resolution and precision of the scanner. The system has to scan several times to generate a specific face model of different people. Moreover, 3D scanning systems are usually expensive, so the usage of them is restricted. Another limitation is that the person who wants to make a face model must be present for face scanning.

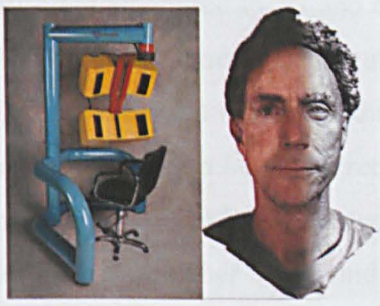


Figure 2.5 Cyberware 3D face and face system
Left: 3D scanner; Right: sample scanned data with more than 1,400,000 polygons

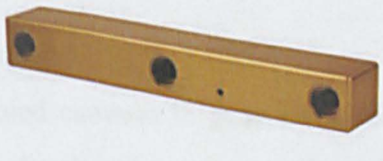


Figure 2.6 Bumblebee XB3 stereo vision camera system

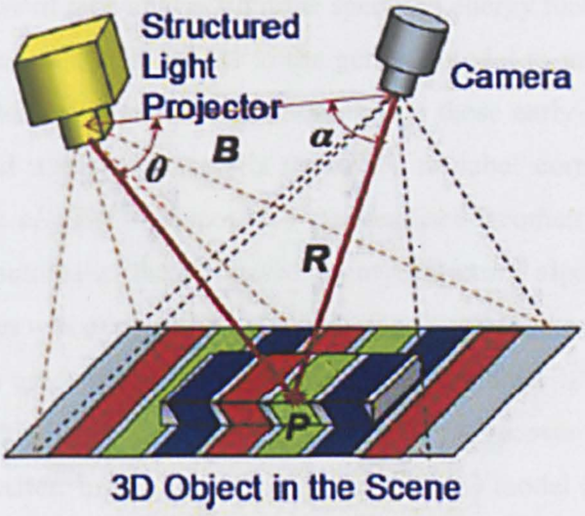


Figure 2.7 Structured light imaging system
Source: Geng, J. (2011) 'Structured-Light 3D Surface Imaging: A tutorial'

2.2.2 Structured Light Method

Structured light is another classic active stereo vision method used to retrieve 3D geometric information. It can extract 3D information from the image of objects illuminated by structured lights (Zhan and Chung, 2010). The structured light system consists of two main parts: the camera and the light projection. The projection irradiates the target with specific coded lights like point light, light strip, light grid, circular light, cross light, and colour coded light. The camera records the images of a scene to recover 3D depth information (Figure 2.7). Geng (2011) reviewed several 3D imaging systems with different structured lights and discussed their applications. Beumier and Acheroy (1999) developed a rapid 3D acquisition system based on this method. They project labelled parallel light stripes on the human face. The absolute 3D coordinates of the human face can then be recovered by taking an image of the face using a calibrated camera. This system has achieved good speed and resolution at that time. However, the system is rather large and with limited depth range. Another limitation of this method is that the measurement accuracy is not as high as laser based scanning methods.

2.2.3 Image based Method

In order to use non-calibrated cameras to generate a 3D model, methods based on multiple images have been developed to parameterise a 3D face model of any faces by extracting face feature points from the multiple images. The model is morphed to match different views of face images until the specified energy function is minimised. The shape and texture are then pasted to the general model to make it more vivid. Due to low computer speed and computation ability, these early-stage methods are semi-automatic, and a user assistant is necessary to label corresponding feature points. Akimoto *et al.* (1993) proposed a personalised geometric face modelling method using an automatic facial feature point extracting algorithm. The shape information for faces was extracted by matching face contour to a dynamic template. Pidhin *et al.* (2002) used similar techniques to label more than 100 feature points in the face pictures taken from five different viewpoints to recover the face pose and absolute 3D coordinates. In this way, the personalised 3D model could be deformed from the standard model using the interpolation of discrete data. Liu *et al.* (2001) used face video as input; they reconstructed a 3D face model related to the video by locating the corner points and labelling five key points (two inner-eye corners, one nose peak, and two corners of the mouth) in two adjacent frames. Lee and Nadia (2000) developed a 3D face reconstruction method from two orthogonal images. This method located facial feature points by constructing both frontal and profile structured Snake models to accomplish the rapid modelling of a human face. Horase and Yin (1996) proposed a method to construct an individualised 3D face model from frontal and side view of face images. A Local Maximum-Curvature Tracking (LMCT) algorithm was developed to extract facial feature points to generate a 3D model. These methods could reconstruct 3D face models automatically, but they are personalised models and can only use for individual models. Especially, some of the 3D models generated are not a true 3D surface model. They are simply a triangulation patches.

To sum up, 3D data acquisition methods are varied with their own merits and disadvantages. The 3D scanning method is accurate, but the scanning equipment is usually expensive and the method of representing the surface from scanned data is triangular patches, which means the obtained model is not a true 3D model. A more realistic and accurate model can only be obtained with massive triangles and vertices, which will increase the computation complexity at the same time. Image based methods can use photos taken by cameras, but there are very strict requirements for the image qualities. Therefore, the existing methods can only be used to obtain the information for generating the face models of individuals because there is no generic model for independent face features manipulation.

2.3 3D Modelling

3D modelling refers to the process of modelling the 3D surfaces of the target objects. The result of 3D modelling is a mathematical representation of the obtained surfaces which can be displayed by computers. The rapid development in 3D data acquisition technology makes the 3D modelling an active research topic for 3D animation (Waters and Terzopoulos, 1991), 3D games, aesthetic surgery, and biomimetics (Zhang *et al.*, 2004). This section reviews several popular methods used for 3D modelling.

In addition to 3D data acquisition, surface representation is another important step for 3D modelling. All surfaces can be classified into two categories. The first is the kind of surfaces that can be composed by primitive analytic elements including plane, cylinder, cone, and sphere. This type of surface is widely used to design machine components because boolean operations such as addition and subtraction, conjunction, disjunction, and complement can be easily applied to obtain complex shapes. The second type is called free form surface. It is in general geometry shape and too complicated to be decomposed into primitive elements. It is obvious that the human face belongs to the latter one, due to the complex structure of human faces. The popular surface models used for surface representation include triangular patches, parametric surface, subdivision surface, implicit surface, and deformable surface.

2.3.1 Triangular Patches Method

The triangular patches method is a surface approximation method. Although the each triangle can be expressed in 2D, numerous triangles in 3D can be connected together to approximate an arbitrary surface. A famous triangular patches face model is CANDIDE (Rydfalk, 1987), which uses hundreds of triangles to represent a 3D human face with several simple facial features. The deformations of the face are obtained using global and local action units (AUs); the global ones correspond to 3D rotation and the local ones control different expressions. There are three versions of the CANDIDE model besides the original one. The original CANDIDE model does not define action units. Figure 2.8 shows the differences among these versions.

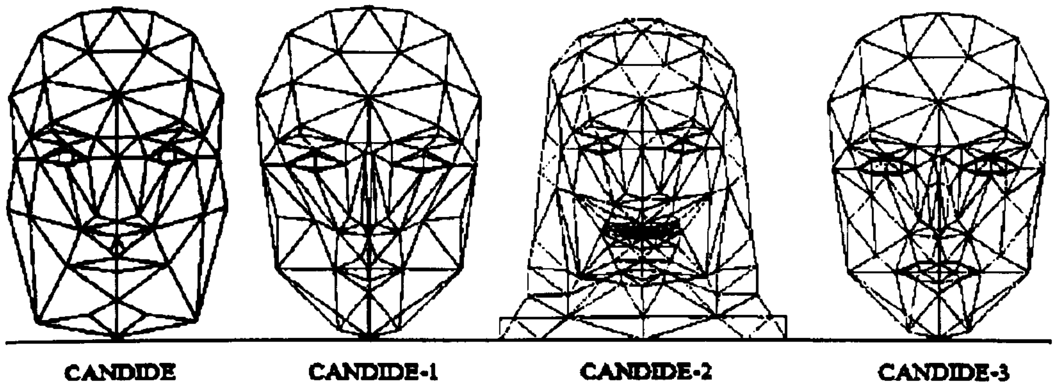


Figure 2.8 Appearances of variations of CANDIDE model

The original CANDIDE model contains 75 vertices and 100 triangles. CANDIDE-1 contains 79 vertices, 108 triangles, and 11 action units. CANDIDE-2 includes hair, teeth, and shoulders so that the number of vertices and triangles increases to 160 and 238, respectively, but it only has six action units. CANDIDE-3 is derived from the original CANDIDE model. It has 113 vertices and 168 triangles. There are 65 action units to correspond to facial animation parameters defined by MPEG-4 (Table 2.1). In the triangular model developed by Peng *et al.* (2006), there are 11,362 triangular and 22,720 vertices initially. They reduced the number significantly to 202 and 400, respectively. However, it is clearly that the triangular patches method is only the approximation of the surface of human face, not the true 3D model. Because it is a convenient way to represent a surface in 3D through approximating it by lots of triangles, the accuracy of the model depends on the number of triangles. The more triangles are used, the more accurate the models are. So it is virtually impossible to graphically adjust the features as they are actually scattered points.

Table 2.1 Comparison of different versions of CANDIDE parametric model

Model	Vertices	Triangles	Action Units
CANDIDE	75	100	N/A
CANDIDE-1	79	108	11
CANDIDE-2	160	238	6
CANDIDE-3	113	168	65

2.3.2 Statistical Model based Method

The most successful statistical model based method is the morphable model method, which reconstructs individual 3D face models automatically. Blanz and Vetter (1999) established a pixel-level 3D prototype face database. A morphable face model was derived by transforming shape and texture into a vector space representation. They obtained the basis of the linear classes by applying the PCA to the 3D face database. Different faces and expressions can then be modelled using linear combinations of the prototypical faces from the database. The shape and texture constraints studied from the face database were used to guide the modelling and matching algorithms. The fundamental of the morphable model is the concept of linear combination. Ullman and Basri (1991) and Shashua (1994) indicated that arbitrary images of an object can be represented by the linear combination of the images of the object taken from three different viewpoints under the condition of orthographic projection and without considering the self-occlude. Choi *et al.* (1991) and Poggio and Brunelli (1992) found an image representation method to decompose an image into two separate groups: geometric shape vectors and texture vectors. Thus, a group of images could be represented by the linear combinations of these vectors. The linear object classes proposed by Poggio and Vetter (1992) described the necessary condition for a 3D object to be a linear object class. The human face was proved to be a linear object class by approximation. Jones and Poggio (1998) used the morphable model to represent different objects like face, car, and handwritten numbers, achieving good results. Compared to other modelling methods, the morphable model based method is automatic and rapid. However, there are still some problems to be considered. These methods are not suitable for complex lighting conditions. The statistical model based method is based on lots of collected human faces in the database, so the accuracy replies on the scale of the database. The constructed model is individual model, rather than a generic model.

2.3.3 Parametric Surface Method

Parametric surfaces have been the main tools used to represent geometric shapes and surfaces. With the development of Computer Aided Geometric Design (CAGD) and lofting requirements of aircraft and watercraft manufacturing, researchers have sought to find effective representing methods to describe various curves and surfaces by computers. Bézier surface and B-Spline surface are typical methods of parametric surface. In particular, NURBS, which is a generalised form of B-Spline, has been chosen as a STEP[1] standard for describing geometric shapes mathematically in the industry after decades of development.

(1) Bézier Method

Although the Bézier method was first developed by Casteljau (1959), it is named after French engineer Pierre Bézier (1972), who published this method in the famous UNISURF (Bézier, 1986) CAD system at Renault. Bézier curve and surfaces can be easily evaluated by their control points, which enables engineers to design and model complex surfaces very quickly (Figure 2.9 and Figure 2.10). Bézier method has been widely used in industry as it has the properties of geometric invariability and affine invariability. The geometric invariability means that the representation of parametric curves or surfaces does not depend on the choice of a coordinate system. In other words, the properties of parametric curves or surfaces can remain unchanged against any rotation and translation of the coordinate system in which they are located. The affine invariability means that the representation of parametric curves or surfaces does not change for arbitrary affine translation.

Typical applications of Bézier method include surface design for airplanes and ships, costume design, computer animation, font design, and 3D modelling because of the following characteristics: (1) arbitrary shapes can be approximated by Bézier curve manipulation; (2) it is easy to control the Bézier curve by adjusting its control points; (3) there is no singularity in Bézier curves; (4) the whole Bézier curve sits inside the convex hull formed by its control points; and (5) the Bézier curve is tangent to the first and last edge of its control polygon.

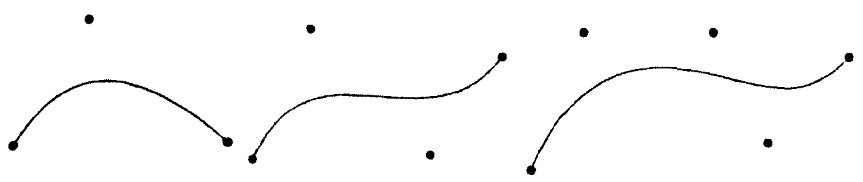


Figure 2.9 Three examples of Bézier curves: quadratic, cubic and quartic

[1] STEP: STandard Exchange of Product data model. It's an international industry standard for products model data exchange.

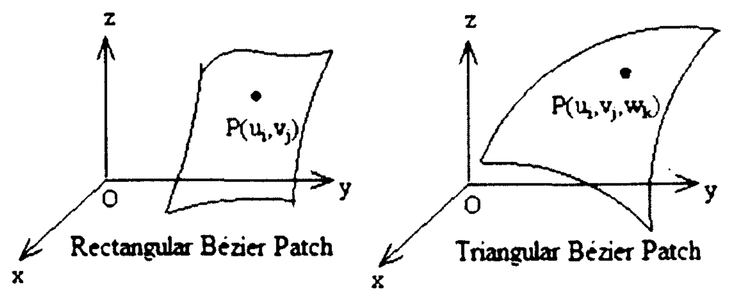


Figure 2.10 Two examples of rectangular and triangular Bézier patches

There are some researches that attempt to model human face by Bézier method. Li and Jiang (2003) used two kinds of approaches to deform a 3D face mesh model. They polished the model through the Bézier method to obtain a realistic 3D face model. Yin and Gao (1998) presented a 3D face model using Bézier surface patches to approximate the facial geometry structure. They also model several facial features by triangular meshes to embed facial expressions. Similarly, Yan and Zhang (2000) presented a method using Bézier patches to create a virtual 3D model of a human face and body. After extracting the front and profile of facial feature points from two orthogonal input images, Bézier patches are applied to create a specific human face model. Some expression is synthesised as well by moving the control points. Morera *et al.* (2007) developed the "geodesic Bézier curve", which is a type of generalised Euclidean curve based on the Bézier curve. Compared with the traditional Bézier curve, the geodesic Bézier curve is smoother and easier to apply operations interactively, such as cutting or filling and texture mapping or colouring. Su and Kumar (2005) use quartic triangular Bézier patches to interpolate and approximate a triangle mesh model. They apply a hierarchical approximation to obtain all features' information, including edges and vertices, and then re-calculate the features on the interpolated Bézier patches model. Based on the retaining feature edges, the approach enables the tangent values of the adjacent non-characteristic edges to change smoothly. Thus, the accuracy of approximation is significantly improved while the original shape information is effectively retained. Bajaj *et al.* (1995) described a surface reconstruction method from dense regular sampled data using implicit Bernstein-Bézier patches without imposing any restrictions about convexity or differentiability on the original dataset. Their approach proved to be simple and efficient, providing a uniform method for both CAD model and scalar field. Guan and Zhang (2008) proposed a 3D facial feature points localisation method based on the local search and global restriction pattern of ASM. They constructed the local surface model using Bézier surfaces obtained from matrix decomposition that can precisely represent 3D surfaces of the human face. Their method localises 3D feature points of human faces by combining the global shape model and local surface model. However, the computation of local surface features is rather heavy so that the localisation time of feature points still has the potential to speed up.

Although the Bézier curve and surface have many geometric advantages, the shape of the Bézier curve and surface will change by moving any control point. The combination of Bézier curves is comparatively complex. Because the order of the Bézier curve and surface is determined by the vertices of its control polygon, this method will be numerically unstable and computationally intensive when the order number becomes larger. Hence, researchers have proposed a B-spline method to overcome the weaknesses of the Bézier method while retaining its merits.

(2) B-Spline Method

The standard B-Spline algorithm was first proposed by De Boor (1972). Gordon and Riesenfeld (1974) subsequently applied B-Spline theory for shape description. Based on the finding that Bézier is a special case of B-Spline, these authors proposed a generalised representation to unify Bézier and B-Spline theory, successfully overcoming the local manipulation problem of Bézier while preserving its advantages by replacing the Bernstein basis functions with B-Spline basis function. Knot insertion, put forward by Boehm (1980) and Cohen (1980), is one of the most important techniques in B-Spline theory. Another important technique is the degree elevation technique proposed by Forrest (1972) and Prautzsch (1991). These researchers' work has led to the B-Spline method being widely used in the expression and design of free form curves and surfaces.

By comparing with Bézier method, B-Spline method is more suitable for 3D modelling because the shape of the surface can be locally changed by moving any control points, which is called local support property. Eck and Hoppe (1996) introduced an automatic B-Spline surface reconstruction technique from arbitrary topology data. The method first established a network of B-Spline patches, then refined the network adaptively in order to reduce the error to the user-specified tolerance. This method has been tested on both synthetic and real datasets. Yin and Xie (2005) applied the B-Spline interpolation technique to two input images to generate the morphing images between them. He and Qin (2004) introduced a modelling method based on the triangular B-Spline surface. Compared with other surface reconstruction method, the method—relying only on less user interaction—can generate a specified model using a single B-Spline surface. He *et al.* (2008) described a 3D facial reconstruction method using Radius Basis Function (RBF) and B-Spline. First, they adopted stereo vision to extract 3D feature points; next, a face model with obvious contour feature was established by computing RBF for regional facial feature points of frontal and profile face images. Finally, the B-Spline method was used to reconstruct 3D facial surfaces. Wei *et al.* (2007) presented a method for 3D facial expression based on the space vector of the shape statistical model. The statistical model is established through control points of B-Spline surfaces in training dataset. Then, a model fitting procedure is applied to display 3D facial surfaces with various expressions.

The local support property of B-Spline method provides more control flexibility of the surface model. The shape of the surface can be graphically modified by moving the control points. However, existing B-Spline face modelling methods either consider the human face as a single surface, or are based on statistical information. These methods are not suitable for controlling the facial features independently and generating a great number of face models. So it is necessary to propose and develop a new generic method for 3D face modelling.

(3) NURBS

Industrial shape design always deals with many shapes represented by quadratic curves, including multi-arcs, elliptical arcs, parabolic arcs, and other conic arcs and line segments. Quadratic surfaces such as cylindrical surface, conical surface and round torus are also very common in mechanical parts and plastic products. B-Spline method encounters difficulties in representing these quadratic curves and surfaces. The initial solution to solve this problem is to adopt an implicit equation, which results in the problem that two different mathematical models exist at the same time to describe the same geometric model. Versprille (1975) first put forward the rational B-Spline method. NURBS was subsequently studied in depth (Piegl, 1987; Piegl, 1989; Piegl, 1989; Piegl, 1991; Tiller and Piegl, 1997). NURBS has the ability to accurately represent regular curves and surfaces, so that unified mathematical models can be used to describe both regular curves and surfaces and free curves and surfaces; the weight coefficient facilitates controlling the shape by comparing it to Bézier and B-Spline. Because of these outstanding advantages of NURBS, it has not only become the international standard for industrial product data exchange, but is also the only mathematical method to describe the geometry of industrial products. Eventually, it became the most popular modern surface modelling technique.

NURBS provides more flexible control than B-Spline by introducing the weights, so it has been widely applied for complex objects 3D modelling. Xiao *et al.* (2006) built a 3D face model combining NURBS and MPEG-4, establishing the association with control points of NURBS surfaces and the standard feature points defined by MPEG-4. Yan and Zhang (2000) used the 3D point cloud dataset of "SizeChina" survey to establish a homologous 3D face and face model. Their study analysed the average face and facial features parameters of Chinese males and females using the PCA method through the statistical dataset. Different face models can be generated by adjusting the width and height of the face as well as facial height and the parameters of facial features, including the forehead, cheek, and jaw. Carswell and Lavery (2006) proposed a rapid product design and solid modelling method based on NURBS surfaces. The method uses NURBS surfaces established from the profile 2D Spline to generate 3D geometric surfaces of the products. The generated surfaces can be imported directly into other commercial software for further processing so that the product design and modelling process are simplified. Au *et al.* (2008) developed a NURBS based semi-automatic geometric model reconstruction technique which can be applied to construct 3D human faces based on Computer Tomography (CT) images. The disadvantage is that the error of the constructed model depends heavily on the resolution of CT images.

The common problem of these existing NURBS based 3D face modelling methods is that the constructed model are presented by single surface. Although NURBS inherits the local support property from B-Spline, this representation restricts the ability of control over all facial features, especially the edges of any features. By contrast, a generic 3D face model with parametric features can meet these requirements of independent control and modification.

In this section, three popular 3D modelling methods have been discussed. The model obtained by triangular patches method consists of thousands to millions of triangles. It is not a true 3D surface, but a simple way for display the objects in 3D. The statistical model based method can only be used to model a specific person because it is based on the analysis the statistical information of a number of 3D face models in a library. The 3D library needs to be built in advance. The previous parametric surface methods consider the human face surface as a whole surface. Without features segmentation, it is not possible to apply flexible control over facial features to generate arbitrary face model with different appearances. So they are not suitable for 3D face modelling for emotional bio-robots.

2.4 3D Face Modelling Applications

Human face and facial expressions are one of the most important ways of expressing emotions and interpersonal communication. The representation of human face has raised long-term concern of artists and engineers. The development of computer technology provides a new way of 3D human face modelling with can be applied to many fields. This section describes some typical applications of 3D face modelling.

2.4.1 Emotional Bio-robots

Emotional bio-robot is one of the important applications of generic 3D face model. To make an emotional bio-robot, one of the tasks is to design and develop a 3D human face surface. The current methods for design 3D face for emotional bio-robots include to scan the sculpture of human face and to draw the model through 3D modelling software. However, as discussed in Sections 2.2 and 2.3, these existing 3D human face models can only be used to build a 3D face model for individual person. These methods cannot be used to generate a generic 3D face model. Here a generic 3D face model is defined as a model a number of predefined surfaces to represent all typical face features such as faces, eyes, nose and mouth etc. The boundaries of all pre-defined surfaces can be adjusted so that the surface can be modified to any expected geometrical shape. Through adjusting the control points on boundary curves, this generic model can be modified to build a face model of any existing human faces or any imaginary human faces.

By referencing to a generic 3D face model, researchers can generate the specific 3D face for bio-robot. It also accelerates the large scale manufacture of bio-robot by enabling engineers to obtain various face appearances of the robots by adjusting the parameters of the generic 3D face model.

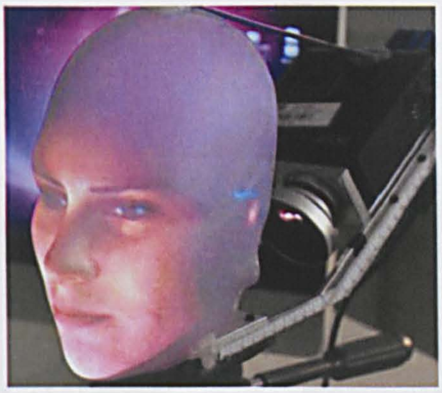


Figure 2.11 *Mask-Bot* created by German and Japanese scientists
Source: http://www.gizmag.com/mask-bot/20416/

2.4.2 Aesthetic Surgery

The traditional method of medicine teaching highly depends on textbooks and autopsy. Books cannot show the actual complex 3D structures of human body, while the autopsy will cause permanent damages to the body so that the body cannot be dissected for multiple times. These drawbacks also exist for facial aesthetic surgery. Applying computer simulation technology to 3D face modelling will be a good solution to deal with these problems. Kumar and Rakesh (2011) propose a 3D face reconstruction method to make the pre-operation visualisation of cranio-facial surgery. By reconstructing 3D facial structure from CT or Magnetic Resonance Images (MRI), they developed a 3D model based on iso-surface extraction and construction. They argue that the 3D structure is more suitable for complex facial skeleton and skull display. Ranal *et al.* (2007) also proposed a similar tool for the interactive simulation of craniofacial osteotomy surgery.

Some of the existing modelling methods for aesthetic surgery only focus on some parts of the face, such as jaw or nose. The others are mainly used for constructing the 3D model to illustrate the anatomy structure of human face, rather than the surface of faces. A generic 3D face model can provide the feature manipulation function as well as the ability of generate a great number of different 3D face models. By demonstrating a generic parametric 3D human face models by computer simulation, surgeons can study the 3D structure of human face. 3D face model can also be used in the communication between surgeons and patients to predict and simulate the post-operation appearance in advance. So the application of generic parametric 3D face model can be used to minimise the risk of aesthetic surgery.

2.4.3 Film and Games

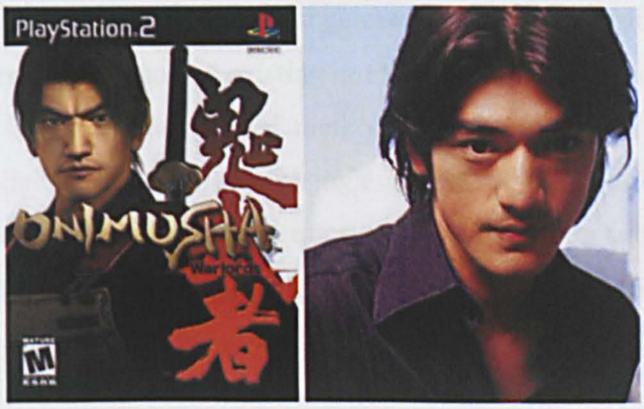


Figure 2.12 Comparison between game character and the star
Source: http://lparchive.org/Onimusha-Warlords/

It becomes more and more common to generate 3D face models for virtual characters in film and game productions. The techniques of 3D face modelling are utilised in many famous films like Avatar (Fox, 2011), Final Fantasy (Cnix, 2009), Ice Age (Fox, 2008), and so on. Computer game is one of the main driving forces of development of CG and computer image. The traditional 2D game scenes and virtual characters limit ability of providing immersive experience to players. The rapid development of graphics hardware, it has gradually become a trend to design and create virtual 3D face models of game characters. An example is the famous Japanese game named *Onimusha* (IGN., 2001). The face of the main character is modelled according to the movie star Takeshi Kaneshiro as the prototype (dy.com.cn., 2010). It can be seen from Figure 2.12, the character of the game is vivid to be similar as the real person.

2.4.4 Crime Investigation and Prevention

Biometric authentication technology progressed rapidly in recent years. Among fingerprint, iris, and other biometric information, facial feature is a natural and straightforward method to be applied in crime investigation and prevention as the images of people can be easily obtained even without being noticed. There are face detection and recognition systems attempt to aid the crime detection (Kar *et al.*, 2006; Frowd *et al.*, 2007; Shan *et al.*, 2007; Tang *et al.*, 2009). However, the images taken from different angles of the same person may appear with great differences. If the 3D face models of criminals are stored beforehand, they can be used to generate photo with specific pose by projecting the 3D face model to 2D plane in which the existing images are. Then the criminals are distinguished easily from ordinary people by comparing the generated photo and existing image. So the 3D face modelling method is more suitable than 2D image based methods for crime control. If this kind of technology is applied in the locations with high security requirement like airports and rail stations, the accuracy and efficiency of criminal investigation and prevention can be significantly increased. There is great potential of development as the research in this area is still in the initial stage.

2.5 Summary

In this chapter, the theories and methods of face detection approaches and 3D face modelling have been reviewed. Among the four kinds of existing face detection methods, the geometrical knowledge based methods face the difficulty of transforming known knowledge to unambiguous rules. The templates matching and statistical learning methods cannot avoid the step of collecting training dataset. It is usually time consuming to training the algorithms. The skin colour base method only depends on the skin colour of human face which remains unchanged while rotations and occlusions happen. However, the skin colour based method is sensitive to illuminations because the colours will change in different illuminations. So, based on the analysis and comparison of the four kinds of face detection methods, the author seeks to propose a new skin colour based face detection method for emotional bio-robots, which does not need any training samples and can deal with images taken in various illuminations. It may also have the potential to be applied for extracting facial features information for 3D human face modelling.

The 3D modelling methods have been examined in Section 2.3. Because the model obtained by triangular patches is not a true 3D surface, it can hardly to be manipulated for changing the appearances. The statistical model based method needs the 3D face model library in advance. It can only be used to generate the face model for a specific person. By comparing to the triangular patches method and statistical model based method, the parametric surface is more suitable for modelling free-form surface like human face. The previous parametric surface methods do not provide flexible control over facial features. Hence, in this thesis the author proposes and developed a new method for building a generic 3D face model based on NURBS. The proposed model has the ability of manipulating all facial features flexibly to get various face models with different appearances. The potential applications of proposed generic 3D human face model are discussed in the last section of this chapter.

CHAPTER 3

Related Work

This thesis will focus on the two tasks of emotional bio-robots: face detection and generic 3D face modelling. This chapter will introduce the related work about the face detection and generic 3D face modelling respectively. Section 3.1 will discuss the theories and techniques for face detection, including colour spaces, image processing and analysis of skin colour information. Section 3.2 is going to discuss the techniques of NURBS which will be used for generic 3D human face modelling.

3.1 Theories and Techniques for Face Detection

Although it is very easy for human beings to distinguish human faces from scenes, even with complex backgrounds, the face detection problem is still a great challenge for computers. This chapter presents the face detection theories in detail. Firstly, several popular colour spaces and their transformations are introduced. Secondly, the image processing techniques for face detection are discussed. Thirdly, the skin colour models are built up in different colour spaces to extract skin region and speed up the detection procedure. Finally, two facial features positioning methods are explicated.

3.1.1 Colour Spaces

This section introduces some popular colour spaces (Ford, 1998; Colantoni, 2001) used by various human face detection methods. Because *RGB* is widely used and the most common space and *YCbCr* explicitly separates the luminance and colour components, only the mathematical transformations between these two colour space are discussed. In this thesis, most equations are from the book of *Multimedia Techniques and Applications* (Feng *et al.*, 2005) and *Digital Colour Image Processing* (Koschan and Abidi, 2008). Without explicit declaration, the term of "colour space" is also written as "space" for short in this section.

According to ITU-R BT.601-4 (1994), the three components of *YCbCr* space are luminance (*Y*) and two chrominance. These two chrominance are colour difference components $Cb = B - Y$ and $Cr = R - Y$.

3.1.1.1 *RGB* to *YCbCr*

The conversion from *RGB* to *YCbCr* is shown in Equation 3.1 (ITU-R., 1994):

$$\begin{aligned}\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} &= \frac{1}{255} \times \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112 \\ 112 & -93.786 & -18.214 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} \\ &= \begin{bmatrix} 0.2568 & 0.5041 & 0.0979 \\ -0.1482 & -0.2910 & 0.4392 \\ 0.4392 & -0.3678 & -0.0714 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix}\end{aligned} \tag{3.1}$$

In the conversions introduced above, the range of luminance *Y* is [16,235] and the range of chrominance *Cb* and *Cr* is [16,240]. These ranges are not convenient to represent images as they are represented in *RGB* space, where the range for *R*, *G*, and *B* components are all [0,255]. To get a full range *YCbCr* colour space, the conversion from *RGB* can be described as Equation 3.2:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.5 \\ 0.5 & -0.419 & -0.081 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix} \tag{3.2}$$

3.1.1.2 *YCbCr* to *RGB*

The reverse conversion from full range *YCbCr* space to *RGB* space can be calculated by solving the Equation 3.2. The result is shown in Equation 3.3:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.0 & 0.0 & 1.4 \\ 1.0 & -0.343 & -0.711 \\ 1.0 & 1.765 & 0.0 \end{bmatrix} \times \left(\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} - \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix} \right) \tag{3.3}$$

Where:

$$\begin{aligned} R, G, B &\in [0,255] \\ Y, Cb, Cr &\in [0,255] \end{aligned}$$

As discussed, the luminance and chrominance are separated by the transformation from *RGB* to *YCbCr* space. The luminance *Y* can be processed independently to adjust the illumination of the original image. So the skin colour model that is constructed in *YCbCr* space has the ability of dealing with wide range of illumination of images for face detection. It can be seen that if both *RGB* and *YCbCr* colour spaces are applied in the integrated way, the advantages can be achieved in effectively processing wide range of illuminations of images captured in any light conditions. This new theoretical finding will be used in the new method of face detection proposed in Chapter 4.

The colour spaces are the primary tools for building up the skin colour models to segment skin regions from input images. Because those skin regions are selected from the image according to their pixel values, they may be independently discrete areas although they should be in one region. Hence, it is necessary to apply image processing technologies to reduce noises, connect discrete areas, and separate connected regions. The regions after the first step processing will be considered as face candidates. So other image processing techniques are required to verify the face candidates. All related image processing technologies will be discussed in the next section.

3.1.2 Image Processing Technologies

The face detection algorithm is applied on images to identify human face from the background. It involves several image processing technologies, such as image smoothing, edge detection, binarisation, and morphological operations, etc. These technologies are discussed below respectively.

3.1.2.1 Image Smoothing

Smoothing is equal to the term of de-noising. Noise exists in all kinds of images obtained using various equipment. It may be introduced in all stages of obtaining the image, including the internal noise of light sensor, blurred noise caused by electrical mechanical movement, and quantisation noise or transmission noise. The noise brings difficulties for further processing to the image, so an effective method for reducing the noise is necessary in most image processing jobs. Generally, the purpose of smoothing is to find a way to eliminate the noise and improve the image quality for processing but the details of the image should remain the same.

If the model for obtaining an image is shown as (Koschan and Abidi, 2008):

$$\boldsymbol{I}(x,y) = \boldsymbol{f}(x,y) + \boldsymbol{n}(x,y) \tag{3.4}$$

Where: $\boldsymbol{f}(x, y)$ is the image without noise, $\boldsymbol{n}(x, y)$ is the noise, and $\boldsymbol{I}(x, y)$ is the obtained image. So the smoothing tries to find a function $\boldsymbol{F}: \boldsymbol{R} \to \boldsymbol{R}$ to fulfil the condition $\boldsymbol{F}(\boldsymbol{I}(x,y)) = \boldsymbol{f}(x,y)$. Actually, the equality can never be achieved because it is impossible to remove the noise to zero in practical applications. Thus, the $\boldsymbol{f}(x, y)$ can only be approximated.

The popular smoothing methods can be classified into two main categories: smoothing in space domain and smoothing in frequency domain. The frequency domain smoothing methods are based on the fact that the noise and signals have different frequency distributions: The signals usually span lower frequency ranges while the noises are with higher frequency ranges. Although the detailed information of the image is also hidden in the higher frequency ranges, the noise can be eliminated if a proper de-noising method is carefully used. The space domain smoothing methods are more intuitive as the image actually represents a 2D space. So in this section two smoothing method in space domain is discussed: mean filter and median filter.

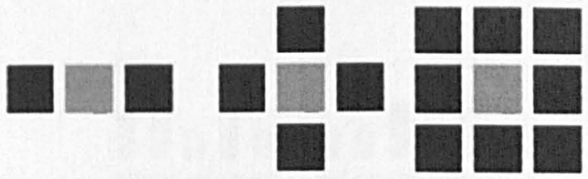


Figure 3.1 Samples of a pixel's neighbourhood
Left: line shape; Middle: 4-neighbourhood; Right: 8-neighbourhood
(target pixel is in grey colour)

Space domain smoothing relates to the pixel neighbourhood, which includes the pixels around the target pixel. Usually the neighbourhood shapes are chosen from line, cross, square, circle, and diamond with certain parameters. Figure 3.1 shows samples of a pixel neighbourhood.

(1) Mean Filter

Mean filter (Gonzalez *et al.*, 2009) is a de-noising method that replaces the value of a target pixel based on the mean of all pixels in its neighbourhood. Usually the neighbourhood is chosen as a $N \times N$ (N=1, 3, 5 …) square window. The filtered image can be represented using Equation 3.5:

$$g(x,y) = \frac{1}{m} \sum_{(i,j)\in s} \boldsymbol{I}(i,j) \tag{3.5}$$

Where: $m=N \times N$ and $\boldsymbol{s}$ is the neighbourhood of pixel at (x, y).

(2) Median Filter

Median filter is another local processing method like mean filter. The difference between median filter and mean filter is that the median filter uses the median of values of pixels in the target pixel's neighbourhood (Tyan, 1981). As shown in Figure 3.2, the median is the value in the middle of a sorted list.

The model of the median filter can be expressed by Equation 3.6:

$$g(x,y) = MED\{\boldsymbol{I}(i,j) \mid (i,j) \in s\} \tag{3.6}$$

Where: (i, j) is the pixels in the neighbourhood of (x, y).

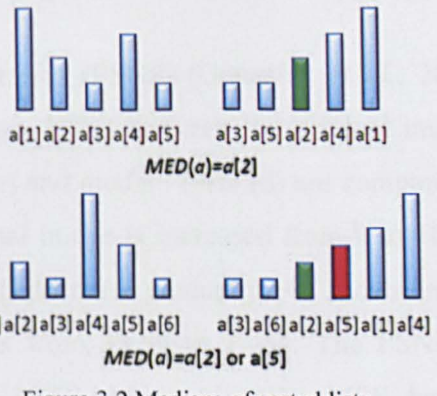


Figure 3.2 Medians of sorted lists

(3) Comparison of Different Smoothing Methods

To compare mean filter and median filter, the author has implemented experiments using Matlab. Figure 3.3 shows the comparison between different image smoothing methods. The original image on the left is polluted by adding "salt & pepper" noise to it. Then the image with noise is smoothed by mean filter and median filter respectively. The detailed regions of left eye in all pictures are magnified and displayed in the second line. It is easy to find that the median filter can achieve better result than mean filter. The reason is that the result value of each pixel is replaced by a "real" value exists in its neighbourhood in the image processed by median filter. By contrast, the result value which is the mean of all pixel values in the target pixel's neighbourhood may be different to any pixel values in its neighbourhood.

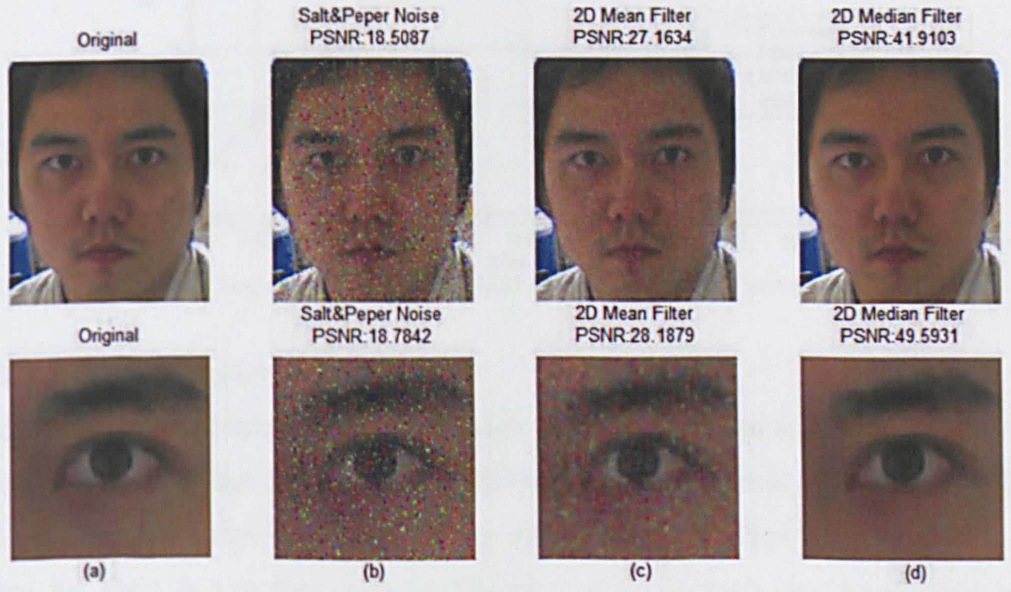


Figure 3.3 Comparison of image smoothing methods
(a) Original image; (b) Polluted image with 'salt & pepper' noise;
(c) Result of applying mean filter; (d) Result of applying median filter

The peak signal-to-noise ratios (PSNR) (Gonzalez *et al.*, 2009) and mean square error (MSE) (Nagabhushana, 2006) between the original image (a) and the results smoothed by mean filter (c) and median filter (d) are compared in Figure 3.4. While the level of noise in original image is increased from 0 to 10%, the PSNR of mean filter and median filter will decrease gradually. As shown in Figure 3.4, the PSNR between (a) and (c) drops from 35.59 to 24.34. The PSNR between (a) and (d) decreases from 44.39 to 38.50. Meanwhile, the MSE between (a) and (c) are significantly increased from 3.3 to 41.90 while the noise level rises. But the MSE between (a) and (d) remains within the range of 0.6 to 1.1. It means that the median filter can achieve better result than mean filter for image smoothing. So in the proposed face detection algorithm represented in Chapter 4, median filter is chosen for image smoothing.

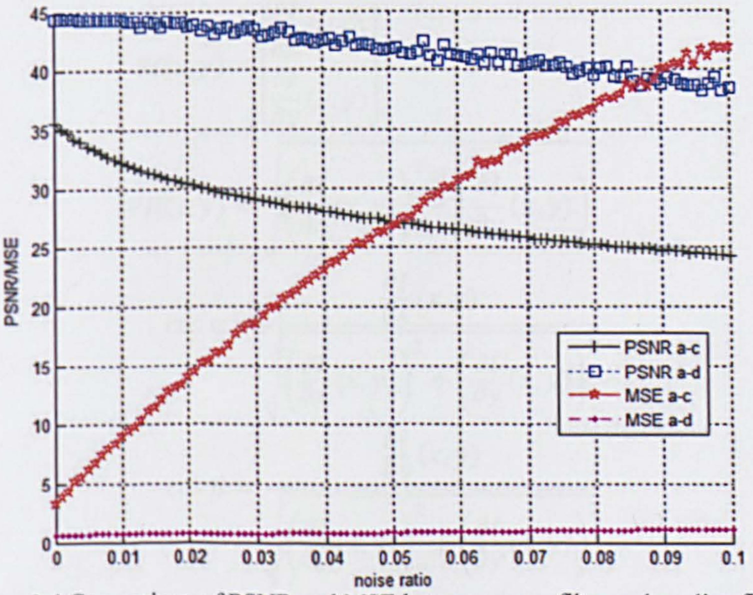


Figure 3.4 Comparison of PSNR and MSE between mean filter and median filter

3.1.2.2 Edge Detection

Edge carries lots of information in an image. The human vision system is sensitive to the edges in an image (Figure 3.5). It represents the discontinuity of the image. The values of pixels around the edges change suddenly. The gradient vector (gradient is used for short in this thesis) is introduced to measure such changes (Burger and Burge, 2008). More specifically, the magnitude of the gradient indicates the significance of the change while the direction of the change is denoted by the direction of gradient.

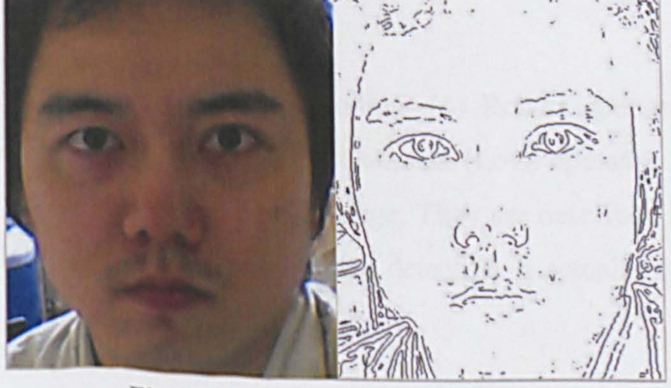


Figure 3.5 A face image and its edges

In an image, the gradient is defined using the partial derivatives along its two axes x and y. The gradient is a function shown in Equation 3.7:

$$\nabla I(x,y) = \begin{bmatrix} \frac{\partial I}{\partial x}(x,y) \\ \frac{\partial I}{\partial y}(x,y) \end{bmatrix}$$
$$|\nabla I|(x,y) = \sqrt{\left(\frac{\partial I}{\partial x}(x,y)\right)^2 + \left(\frac{\partial I}{\partial y}(x,y)\right)^2}$$
$$\cos\alpha = \frac{\frac{\partial I}{\partial x}(x,y)}{\sqrt{\left(\frac{\partial I}{\partial x}(x,y)\right)^2 + \left(\frac{\partial I}{\partial y}(x,y)\right)^2}} \tag{3.7}$$
$$\cos\beta = \frac{\frac{\partial I}{\partial y}(x,y)}{\sqrt{\left(\frac{\partial I}{\partial x}(x,y)\right)^2 + \left(\frac{\partial I}{\partial y}(x,y)\right)^2}}$$

Where: α and β are the angles between the gradient vector and x and y axes. For digital images, the partial derivatives are easy to compute because the digital images are quantised to discrete values. Sometimes the first-order differential is used as the approximation to avoid the square computation, as expressed in Equation 3.8.

$$|\nabla I|(x,y) = \sqrt{\left(\frac{\partial I}{\partial x}(x,y)\right)^2 + \left(\frac{\partial I}{\partial y}(x,y)\right)^2} \approx \left|\frac{\partial I}{\partial x(x,y)}\right| + \left|\frac{\partial I}{\partial x(x,y)}\right|$$
$$\frac{\partial I}{\partial x}(x,y) = \Delta f_x(x,y) = I(x,y) - I(x-1,y) \tag{3.8}$$
$$\frac{\partial I}{\partial y}(x,y) = \Delta f_y(x,y) = I(x,y) - I(x,y-1)$$

All popular edge detection operators including: (1) Robert operator, (2) Prewitt operator, (3) Sobel operator, (4) Laplace of Gaussian (LoG) operator, and (5) Canny operator work on the gradient map of the image. They are described as convolution kernels because the local neighbourhood edge detection is actually the convolution operation.

(1) Robert Operator

Robert operator (Chen *et al.*, 1999) is a very simple edge detector for it only uses the 2×2 neighbourhood of the current pixel. The convoluted kernels of Robert operator are:

$$\Delta f_x(x,y) = \begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix}, \Delta f_y(x,y) = \begin{bmatrix} 0 & +1 \\ -1 & 0 \end{bmatrix} \tag{3.9}$$

The magnitude of gradient can be calculated by Equation 3.10:

$$\begin{aligned} |\nabla I(x,y)| &\approx \left|\frac{\partial I}{\partial x}\right| + \left|\frac{\partial I}{\partial y}\right| \\ &= |\Delta f_x| + |\Delta f_y| \\ &= |I(x,y) - I(x+1,y+1)| + |I(x+1,y) - I(x,y+1)| \end{aligned} \tag{3.10}$$

As discussed above, the noise exists in any images in practical applications. Due to the fact that the size of Robert operator is small (2×2), this operator is too sensitive to noise. So it is not a good choice for edge detection in the images containing much edge information like human faces.

(2) Prewitt Operator

Prewitt operator (Chen *et al.*, 1999) uses 3×3 masks to compute the gradient. The convolution kernels of Prewitt operator are:

$$\Delta f_x(x,y) = \begin{bmatrix} +1 & +1 & +1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}, \Delta f_y(x,y) = \begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix} \tag{3.11}$$

Compared to Robert operator, Prewitt operator can reduce the effect cause by noise, because it considers the neighbourhood of the target pixel by averaging the pixel values covered by the masks.

(3) Sobel Operator

Sobel operator (Chen *et al.*, 1999) is similar to Prewitt operator, but it is a weighted method. The convolution kernels are suitable for horizontal and vertical edges.

$$\Delta f_x(x,y) = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, \Delta f_y(x,y) = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \tag{3.12}$$

According to the definition of Sobel operator, the pixels in the mask do not have the same effect on the target pixel. As shown in Equation 3.12, it is clear that the computation of target pixel value involves its 8 neighbourhood pixels. However, four pixels in the left, right, top and bottom (centre pixels) are assigned with higher weights by comparing to the four pixels at the corner (corner pixels) of the neighbourhood. So the values of centre pixels will have more influences on the result than the values of corner pixels. Thus, the result of Sobel operator is better than Robert and Prewitt operators.

(4) Laplace of Gaussian (LoG) Operator

The Laplace operator (Chen *et al.*, 1999) is a kind of the second-order derivative, so it is sensitive to noises. The Laplace operator is defined as:

$$\nabla^2 I(x,y) = \frac{\partial^2 I}{\partial x^2}(x,y) + \frac{\partial^2 I}{\partial y^2}(x,y) \tag{3.13}$$

The following three approximations of Laplace operator are widely used in image processing. Due to its isotropic characteristics, Laplace operator uses the same masks in both directions. Three popular Laplace operator masks are shown below.

$$LoG_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, LoG_2 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}, LoG_3 = \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix} \tag{3.14}$$

As we can see from Equation 3.14, if there is a bright pixel and the pixel values in its neighbourhood are greatly smaller than its value, the Laplace operator will enhance the contrast between them. Same influence exists for a dark pixel rounded by brighter neighbourhood pixels. So it is sensitive to noises in the image. The Gaussian function is introduced to smooth the image and reduce the effect caused by noise. The combined operator is so called LoG operator.

The LoG operator can be expressed by Equation 3.15:

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left(1 - \frac{x^2+y^2}{2\sigma^2}\right) e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.15}$$

Figure 3.6 shows the shape of Gaussian function with σ=1.2.

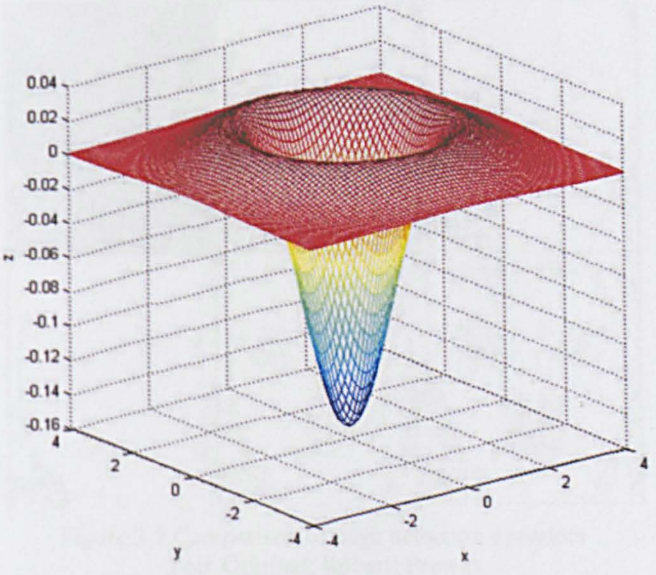


Figure 3.6 Gaussian function with σ=1.2

It can be seen from Figure 3.6 that: the LoG operator can be used to obtain smooth result by adjusting the parameter σ to control the shape of the operator. By this method, the noise can be reduced.

(5) Canny Operator

Canny operator (Canny, 1986) is a multi-stage optimised edge detection operator with smoothing, enhancing, and detecting. Canny proposed several criteria to measure the performance of edge detection methods. Like LoG operator, Canny operator also performs a smoothing operation before computing derivatives.

(6) Comparison of Different Edge Detection Operators

The edge detection results by applying different operators are shown in Figure 3.7. As discussed above, the LoG and Canny operators are based on second-order derivatives. They are sensitive to all possible gradient changes in the original image. So they are too sensitive to noises, therefore, they are not suitable for face detection in complex image backgrounds. The Robert, Prewitt and Sobel edge operators are simple and very easily computed for fast processing. However, as discussed above, the result of Sobel operator is better than Robert and Prewitt operators, hence, the proposed face detection algorithm in Chapter 4 uses the Sobel operator for edge detection according to this comparison.

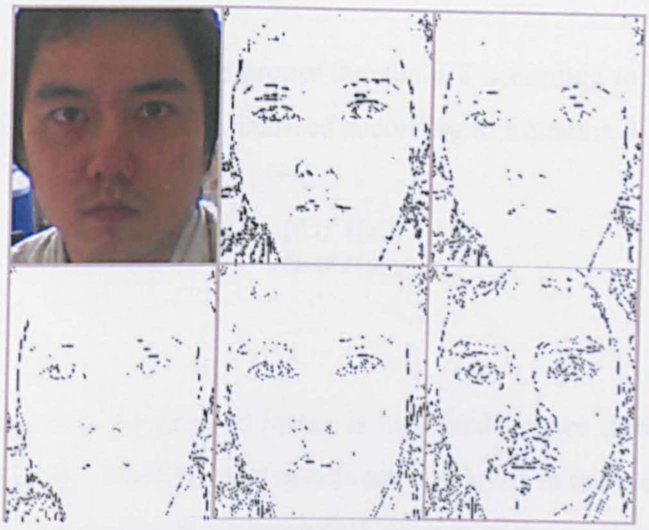


Figure 3.7 Comparison of edge detection operators
Top: Original; Robert; Prewitt
Bottom: Sobel; LoG; Canny

3.1.2.3 Binarisation

Binary image refers to images that have only two values (0 and 1). The regions in binary images are shown as white (1) or black (0). Binarisation—the processing that produces binary images from grey images—is an important technique for image segmentation because it is an effective method to separate the objects and background in an image. Trier and Jain (1995) reviewed classic binarisation methods. This section demonstrates two typical global thresholding methods for binarisation: histogram method and Otsu method.

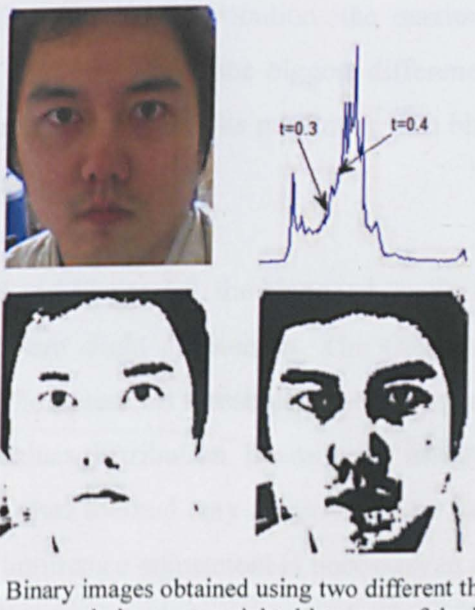


Figure 3.8 Binary images obtained using two different thresholds
Top left: greyscale image; top right: histogram of the image;
Bottom left: binarised with T=0.3; bottom right: binarised with T=0.4

(1) Histogram Method

The histogram method chooses the proper threshold *T* according to the histogram of the grey image. Then the image is binarised according to Equation 3.16:

$$B_I(x,y) = \begin{cases} 0 \ if \ I(x,y) < T \\ 1 \ if \ I(x,y) \geq T \end{cases} \tag{3.16}$$

As shown in Figure 3.8, the original image is binarised by two different thresholds 0.3 and 0.4 respectively. The binarised results are displayed in the second row. It can be seen that bigger threshold will get more dark regions in the result image.

The histogram method for binarisation only involves comparisons of pixel values, so it computes very fast. It is useful especially when the objects and background have significant differences and the objects are not connected with each other. It is also an important step for the face detection algorithm proposed in Chapter 4.

(2) Otsu Method

The Otsu (1979) method is named after the Japanese scholar Otsu, who proposed this method for image binarisation. For an image with a size $M \times N$, the algorithm of the Otsu method is used to find the optimised threshold T_{best}. The fundamental concept of the Otsu method is to maximise the inter-class variance. Because the variance is a kind of measurement for greyscale distribution, the maximal inter-class variance means the two parts of the image have the biggest difference. In other words, the probability of false segmentation reaches its minimum. The binarised result using the Otsu method is shown in Figure 3.9.

By comparing Figure 3.8 and Figure 3.9, the binarised results from the both methods are similar through there are slight differences. The Otsu method is an automatic iterative one for finding the optimised threshold. The histogram method is based on observation of the pixel values distribution. Because the images in practice are taken in various conditions, the Otsu method may sensitive to the luminance of the image. A pre-processing step of luminance adjustment is necessary in most situations. So the complexities between histogram and Otsu methods are roughly same. But it is flexible to choose the threshold through observation on the histogram. So the histogram method is chosen for binarisation in the proposed face detection algorithm

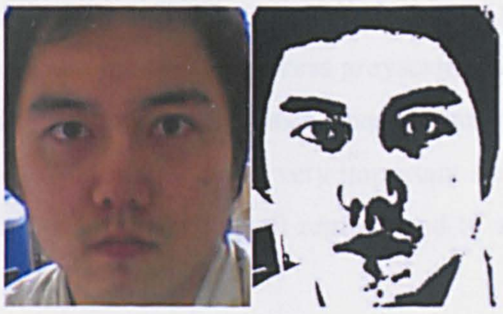


Figure 3.9 Binarised image using the Otsu method

(3) Comparison between Histogram and Otsu methods

These two binarisation methods have been tested over 50 pictures. Their performances are shown in Figure 3.10. The executing times of histogram method vary from 5.17×10^{-4}~2.2×10^{-3} seconds. By contrast, the executing times of Otsu method are in the range of 9.4×10^{-3} to 3.29×10^{-2} seconds. It can be seen that the histogram method is about 4~64 times faster than Otsu method because it can avoid threshold searching iteration. Thus the histogram method will be used in the proposed face detection method in Chapter 4.

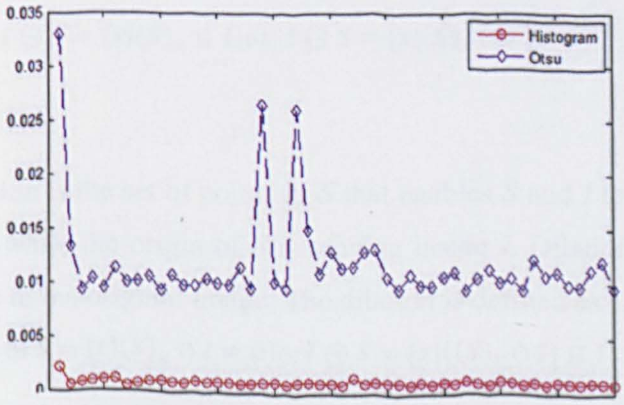


Figure 3.10 Comparison of Histogram and Otsu method

3.1.2.4 Morphology

Originally, morphology was a branch of biology that studied the form and structure of plants and animals (Gonzalez *et al.*, 2009). In this paper, without special instructions, the morphology refers to mathematical morphology, which is the mathematical process used to analyse digital images. The basic idea is to use a certain form of structural elements to measure and extract shape information of the image for the purpose of analysis and recognition. The size of a structural element is usually smaller than the image. Although the structural element moves within the image, a series of logical operations are applied to get the output. Although morphological operations can be used to process greyscale images, it is mainly used to simplify image data, maintain their basic shape characteristics, and remove irrelevant structures in binary image. It is a very important step in image processing to remove the colour areas (noises) of small regions and to separate the connected regions in an image.

Morphological operations primarily involve logical operations: AND (&), OR (|), NOT (~). Using a combination of them, complex logical operations can be applied to process the images. There are four basic morphological operations: (1) erosion, (2) dilation, (3) opening, and (4) closing.

(1) Erosion

The erosion result is the set of reference points of ***S*** that makes ***S*** completely fall within the shapes in image ***I*** while ***S*** moves within ***I***. The effect of erosion involves thinning and shrinking the original shapes. The erosion can be defined as:

$$\boldsymbol{I} \ominus \boldsymbol{S} = \{x|(\boldsymbol{S})_x \subseteq \boldsymbol{I}\} or\ \boldsymbol{I} \ominus \boldsymbol{S} = \{x|(\boldsymbol{S})_x \cap \boldsymbol{I}^c \neq \emptyset\}^1 \tag{3.17}$$

(2) Dilation

The result of dilation is the set of points in ***S*** that enables ***S*** and ***I*** to have at least one common element while the origin of *S* is moving inside ***I***. Dilation can thicken and expand the shapes in the original image. The dilation is defined as:

$$\boldsymbol{I} \oplus \boldsymbol{S} = \{x|(\boldsymbol{S})_x \cap \boldsymbol{I} \neq \emptyset\} or \boldsymbol{I} \oplus \boldsymbol{S} = \{x|((\boldsymbol{S})_x \cap \boldsymbol{I}) \subseteq \boldsymbol{I}\} \tag{3.18}$$

(3) Opening

Opening and closing operations are based on erosion and dilation. The definition of opening is:

$$\boldsymbol{I} \circ \boldsymbol{S} = (\boldsymbol{I} \ominus \boldsymbol{S}) \oplus \boldsymbol{S} \tag{3.19}$$

¹ $\boldsymbol{I}^c$ is the complement set of ***I***. See Moschovakis, Y.N.Moschovakis, Y.N. (2006) *Notes on Set Theory*. Los Angeles, USA: Springer. for more detail.

Although the image *I* is eroded and then dilated by the same structural element *S*, the opening operating cannot restore the original image. It is effective for breaking thin connexions and smoothing the contour by removing the juts.

(4) Closing

Like the opening operation, the closing operation is also based on erosion and dilation, except the order of applying these two operations differs. The closing operation fills the thin chasms and holes smaller than the structural element. The closing operation is defined as Equation 3.20:

$$I \cdot S = (I \oplus S) \ominus S \tag{3.20}$$

Morphological operations are very useful in image processing. They can be used to reduce noises, bridge broken edges and separate connected regions. All morphology operations discussed above are used in the proposed face detection method in Chapter 4.

3.1.3 Investigation of Skin Colour Properties

The skin colour model is the mathematical model of describing skin colour information. Using the skin colour model, it can be determined whether a pixel of input image is a skin colour pixel or not. Because the majority of the human face, except for the eyebrows and eyes region, is covered by skin, the clustering feature of skin colour is ideal for human face detection.

The representation of the skin colour model is closely connected to colour space. Each specific skin colour model must be built in a given colour space. In this thesis, the author will analyse the skin colour information in both *YCbCr* and *RGB* colour spaces.

Using the statistical analysis on large numbers of skin pixels, the model parameters can be calculated to get certain skin colour models. First, several face images under different illumination conditions are selected as samples to build up a skin library (Figure 3.11). Then, the non-skin regions are manually removed to keep only the skin areas in the sample images. A low-pass filter can be adopted to reduce noises. The components of the skin colour samples are then converted to the target colour space. Finally, the skin colour model is established by studying the statistical distribution of all components in the desired colour spaces.

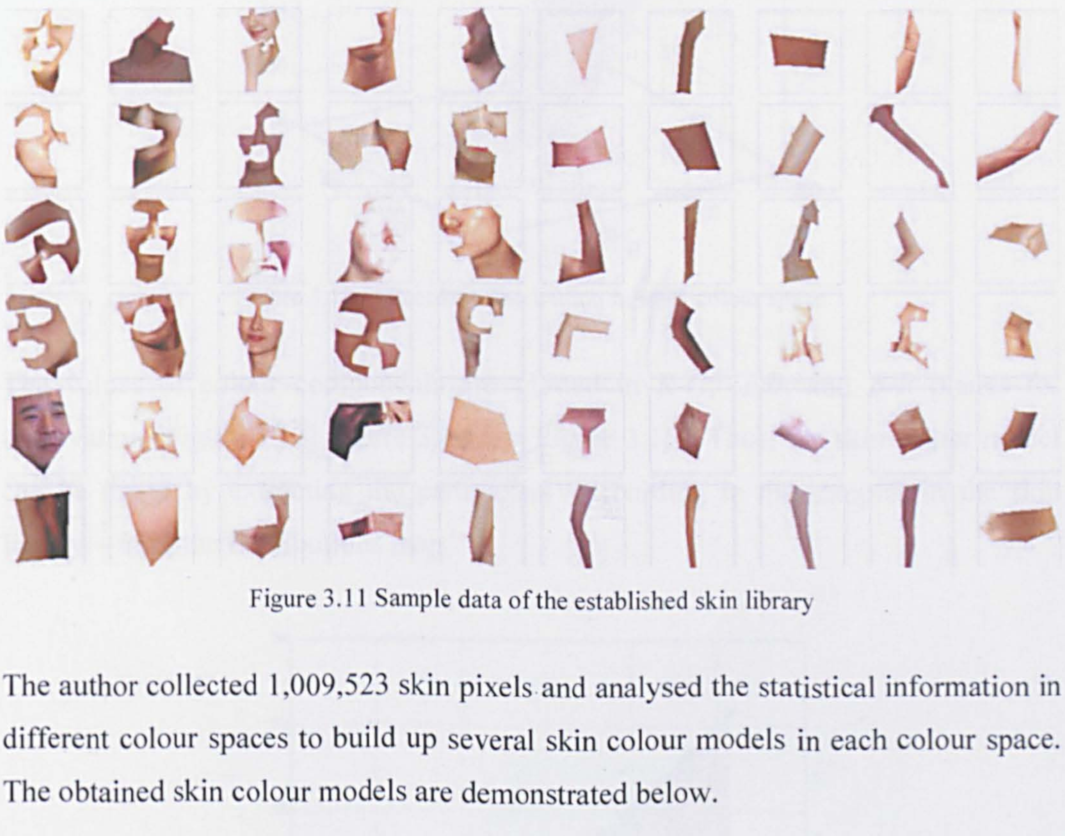


Figure 3.11 Sample data of the established skin library

The author collected 1,009,523 skin pixels and analysed the statistical information in different colour spaces to build up several skin colour models in each colour space. The obtained skin colour models are demonstrated below.

3.1.3.1 *RGB* Skin Colour Model

As shown in Figure 3.12, skin colour pixels have certain distribution in *RGB* colour space, which means the values of the three components are in specific ranges. The statistics of the skin colour values are shown Table 3.1:

Table 3.1 Statistics of *RGB* values of skin colour pixels

	Mean	Standard Deviation	Size
R	129.3703	115.5847	1,009,523
G	119.8056	107.9580	1,009,523
B	110.8314	100.1038	1,009,523

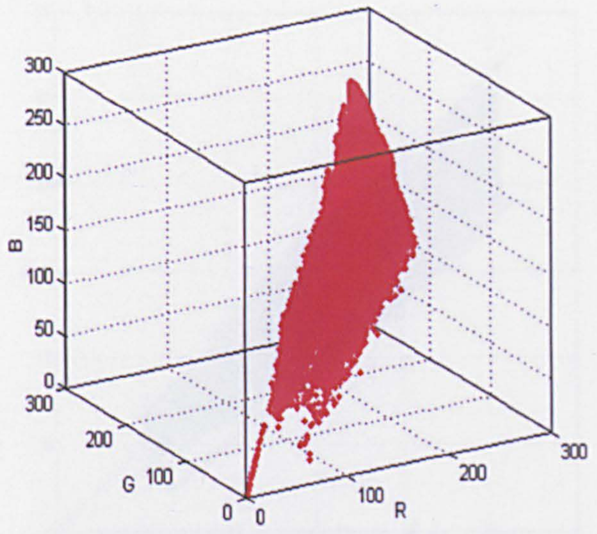


Figure 3.12 Skin colour distribution in *RGB* colour space

The values of colour components are plotted in *R-G*, *G-B*, and *B-R* planes for observation (Figure 3.13, Figure 3.14 and Figure 3.15). Thus, the skin colour model can be found by extracting the parameters—according to the samples in the skin library—from the distributions map.

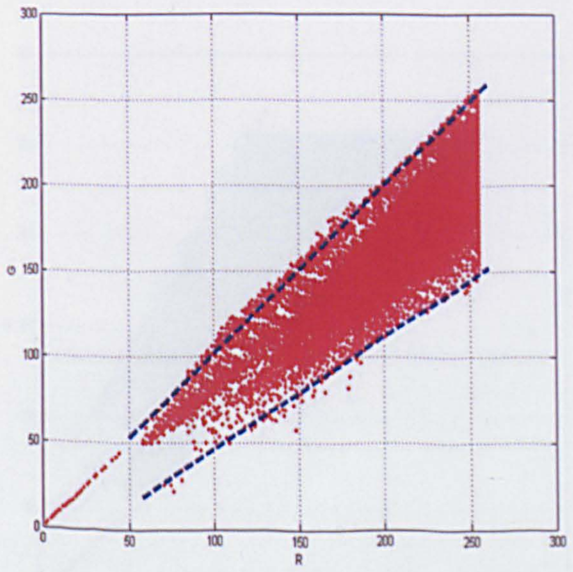


Figure 3.13 Skin colour distribution in the *R-G* plane

The skin colour distribution in the *R-G* plane in the above figure can be described by the following skin colour model.

$$\begin{gathered} r \geq 50 \\ r \geq g \\ 0.645 \cdot r - 14.516 \geq g \end{gathered} \tag{3.21}$$

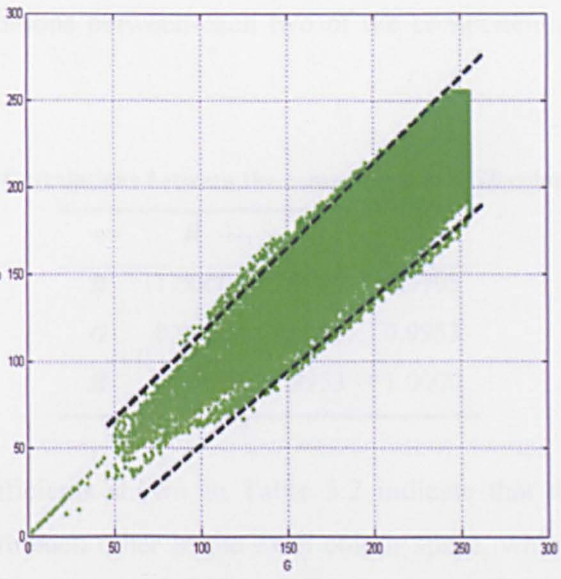


Figure 3.14 Skin colour distribution in the *G-B* plane

The skin colour model in the *G-B* plane of the *RGB* colour space is described as Equation 3.22.

$$\begin{gathered} b \geq 65 \\ 0.923 \cdot g + 19.591 \geq b \\ 0.866 \cdot g - 36.611 \leq b \end{gathered} \tag{3.22}$$

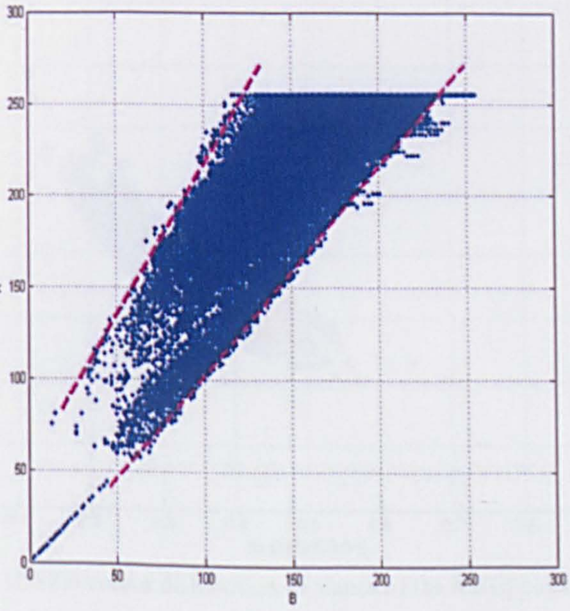


Figure 3.15 Skin colour distribution in the *B-R* plane

Similarly, the skin colour model in the *B-R* plane can be generated as Equation 3.23.

$$\begin{gathered} b \geq 50 \\ r \geq b \\ 1.654 \cdot b + 53.019 \leq r \end{gathered} \tag{3.23}$$

Moreover, the correlations between each two of the components in *RGB* space are checked (Table 3.2).

Table 3.2 Correlations between the components in *RGB* colour space

	R	*G*	*B*
R	1.0000	0.9904	0.9795
G	0.9904	1.0000	0.9953
B	0.9795	0.9953	1.0000

The correlation coefficients shown in Table 3.2 indicate that the components are highly correlated with each other in the *RGB* colour space, which explains why the *nRGB* colour space is developed. The components of *nRGB* space are computed easily as $r = \frac{R}{R+G+B}$, $g = \frac{G}{R+G+B}$ and $b = \frac{B}{R+G+B}$. The distribution map of the skin colours in *nRGB* space is shown in Figure 3.16.

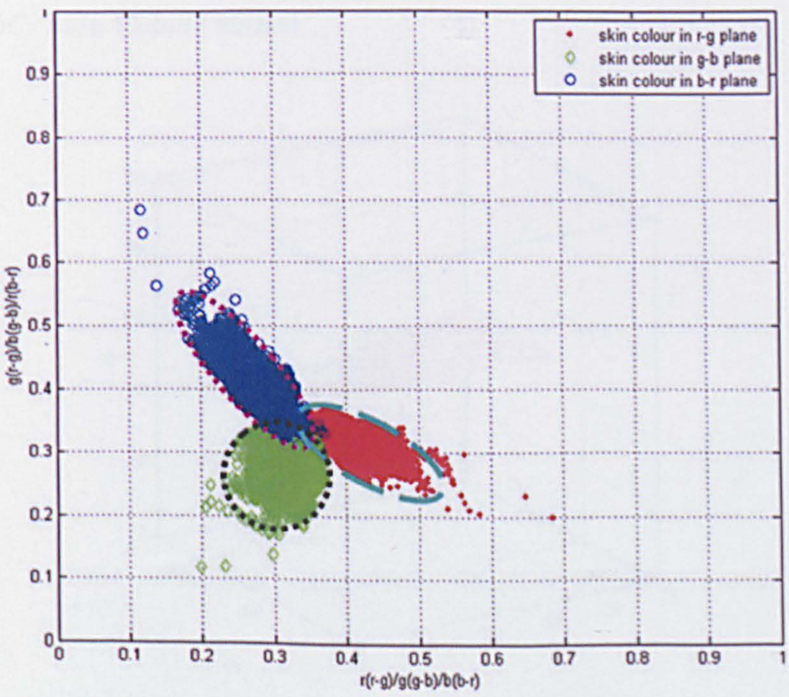


Figure 3.16 Skin colour distribution in planes of the *nRGB* colour space

This clustering property is very intuitive in Figure 3.16. The skin colour model in all planes of the *nRGB* colour space can be represented by ellipse. The models in the *r-g*, *g-b*, and *b-r* planes are listed in Equation 3.24:

$$\begin{aligned} 0.27x^2 + 0.5xy + 0.49y^2 - 0.39x - 0.51y + 0.16 \le 0 \ (in\, r - g\ plane) \\ -0.69x^2 + 0.01xy - 0.53y^2 + 0.41x + 0.28y - 0.1 \le 0 \ (in\, g - b\ plane) \\ -0.45x^2 - 0.58xy - 0.26y^2 + 0.48x + 0.38y - 0.14 \le 0 \ (in\, b - r\ plane) \end{aligned} \quad (3.24)$$

The new correlations of three components in the *nRGB* space are listed in Table 3.3. By comparing Table 3.2 and Table 3.3, it is clear that the correlations in the *nRGB* colour space are smaller than those in the *RGB* space. Although the luminance exists in the three components in both the *RGB* and *nRGB* spaces, the correlations are significantly reduced by converting the colours from one space to another.

Table 3.3 Correlations between the components in the *nRGB* colour space

	r	*g*	*b*
r	1.0000	0.5792	0.9530
g	0.5792	1.0000	0.3049
b	0.9530	0.3049	1.0000

3.1.3.2 *YCbCr* Skin Colour Model

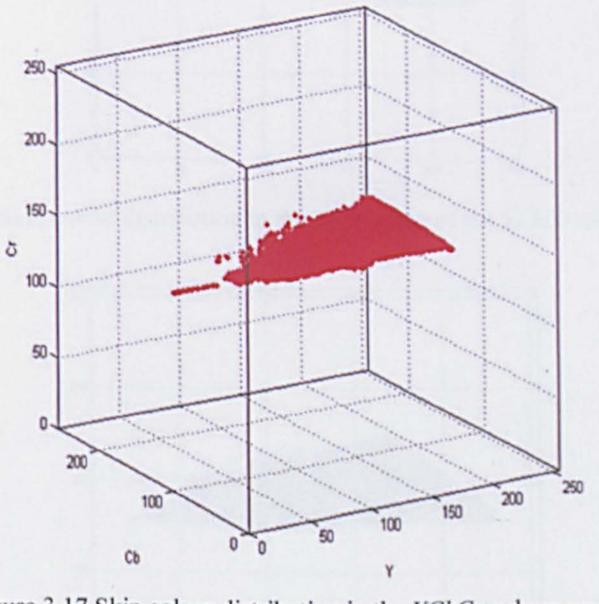


Figure 3.17 Skin colour distribution in the *YCbCr* colour space

The most important difference between the *RGB* and *YCbCr* colour spaces is that the luminance component (*Y*) is distinguished from chrominance components (*Cb* and *Cr*), so that the affection of luminance in various backgrounds is minimised. Figure 3.17 shows the skin colour distribution in the *YCbCr* colour space.

The distributions in the *Y-Cb*, *Y-Cr*, *Cb-Cr* planes projected from the 3D space are shown in Figure 3.18, Figure 3.19 and Figure 3.20. The histograms of the *Cb* and *Cr* components of skin colours are shown in Figure 3.21 and Figure 3.22. As demonstrated in these figures, the *Cb* and *Cr* values of skin colour pixels fall in specific ranges. Thus, skin colour model in the *YCbCr* colour space is generated as:

$$\begin{gathered} 88.97 \leq Cb \leq 130.77 \\ 124.62 \leq Cr \leq 176.12 \\ 32 \leq Y \ \leq 234 \end{gathered} \tag{3.25}$$

The clustered skin colour in the *Cb-Cr* plane can be modelled by Equation 3.26, which represents an ellipse.

$$-0.47x^2 - 0.26xy - 0.32y^2 + 0.56x + 0.48y - 0.26 \leq 0 \tag{3.26}$$

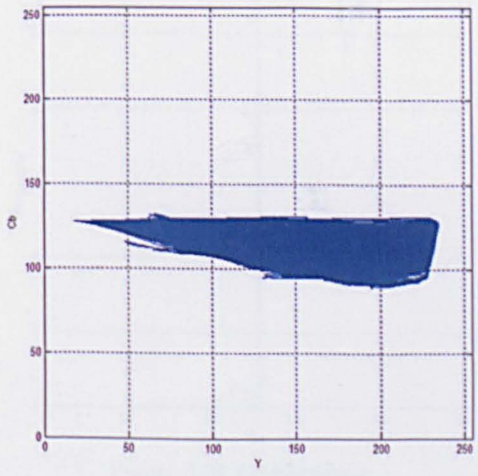


Figure 3.18 Skin colour distribution in the *Y-Cb* plane of the *YCbCr* colour space

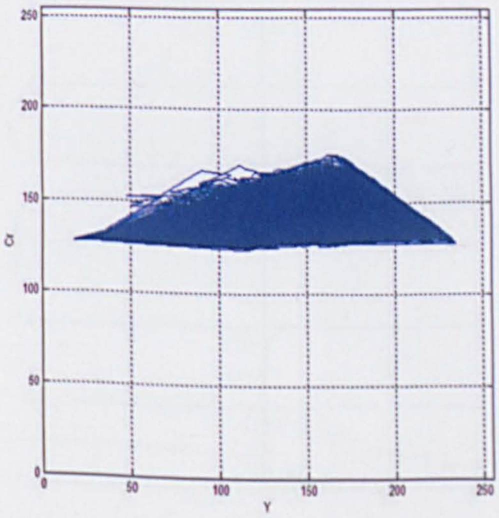


Figure 3.19 Skin colour distribution in the *Y-Cr* plane of the *YCbCr* colour space

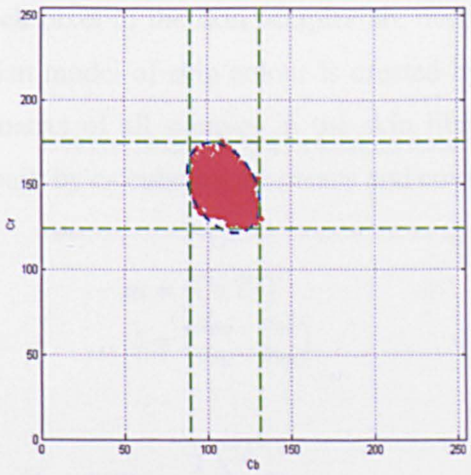


Figure 3.20 Skin colour distribution in the *Cb-Cr* plane in the *YCbCr* colour space

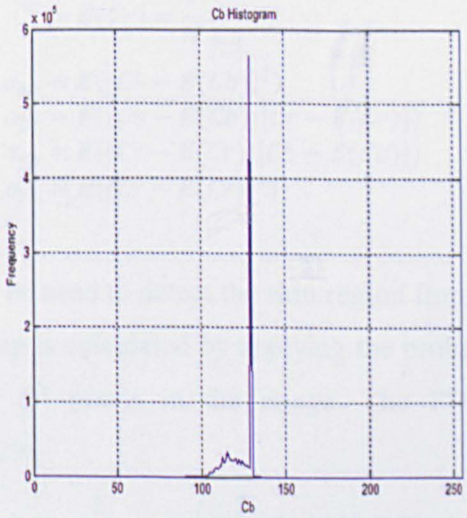


Figure 3.21 *Cb* histogram

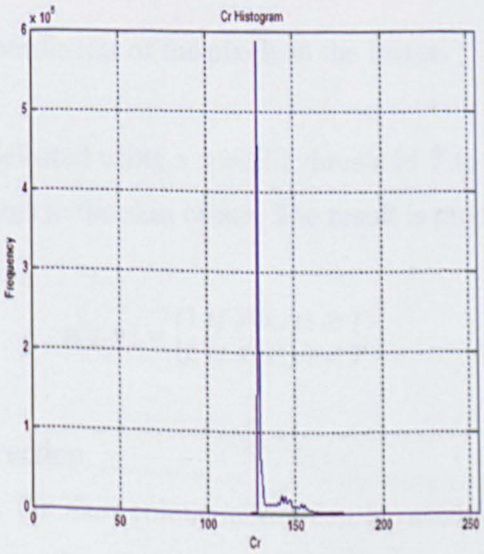


Figure 3.22 *Cr* histogram

If the components of each pixel in the skin samples are denoted as a colour vector $X=(Cb, Cr)^T$, the Gaussian model of skin colour is created by calculating the mean vector and covariance matrix of all samples in the skin library. The 2D Gaussian model of skin colour is built by calculating the means and covariance matrix.

$$\begin{aligned} \boldsymbol{m} &= \left(\overline{Cb}, \overline{Cr}\right)^T \\ \boldsymbol{C} &= \begin{pmatrix} \sigma_{bb} & \sigma_{br} \\ \sigma_{rb} & \sigma_{rr} \end{pmatrix} \end{aligned} \tag{3.27}$$

Where:

$$\begin{aligned} \overline{Cb} &= \boldsymbol{E}(Cb) = \frac{1}{N}\sum_{i=0}^{N-1} Cb \\ \overline{Cr} &= \boldsymbol{E}(Cr) = \frac{1}{N}\sum_{i=0}^{N-1} Cr \\ \sigma_{bb} &= \boldsymbol{E}([Cb - \boldsymbol{E}(Cb)]^2) \\ \sigma_{br} &= \boldsymbol{E}([Cb - \boldsymbol{E}(Cb)][Cr - \boldsymbol{E}(Cr)]) \\ \sigma_{rb} &= \boldsymbol{E}([Cr - \boldsymbol{E}(Cr)][Cb - \boldsymbol{E}(Cb)]) \\ \sigma_{rr} &= \boldsymbol{E}([Cr - \boldsymbol{E}(Cr)]^2) \end{aligned} \tag{3.28}$$

The Gaussian model can be used to detect the skin region from the input images. The skin colour likelihood map is calculated by applying the probability density function (PDF) (Liu, 2003) over all pixels in the image. The PDF value is computed according to Equation 3.29:

$$\boldsymbol{PDF}(x,y) = \frac{1}{\sqrt{2\pi|\boldsymbol{C}|}}exp\left(-\frac{1}{2}(\boldsymbol{X}-\boldsymbol{m})^T\boldsymbol{C}^{-1}(\boldsymbol{X}-\boldsymbol{m})\right) \tag{3.29}$$

Where: x and y are the coordinates of the pixels in the image.

The skin areas are then selected using a specific threshold T to the likelihood map to determine if a pixel belongs to the skin or not. The result is produced by:

$$s(x,y) = \begin{cases} 1 \; if \; f(x,y) \geq T \\ 0 \; if \; f(x,y) < T \end{cases} \tag{3.30}$$

3.1.3.3 Luminance Correction

As previously discussed, the skin colour model can be used to identify criteria for distinguishing skin and non-skin regions. In this way, the searching space is limited and the process is accelerated as well. However, in the real world, the information of obtained colour images are readily affected by environment luminance, background, or the sample device parameters. If the lights are too strong, the colours of most parts of skin regions in the images are still within the skin colour range of the skin colour library. However, when the lights are too weak, the skin colours in the images are significantly affected by the luminance. In this case, the colours in the obtained images may be beyond the ranges of skin colours in the library. Thus, image corrections is a very important pre-processing step before applying the skin colour models to detect faces in images.

(1) Histogram Equalisation (Nagabhushana, 2006)

For an *M*×*N* digital image with *L* level greyscale from *0* to *L*-1, the histogram can be represented by the following discrete function Equation 3.31:

$$h(i) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}(I(x,y) == i?1:0).where\ \ i \in [0, L-1] \tag{3.31}$$

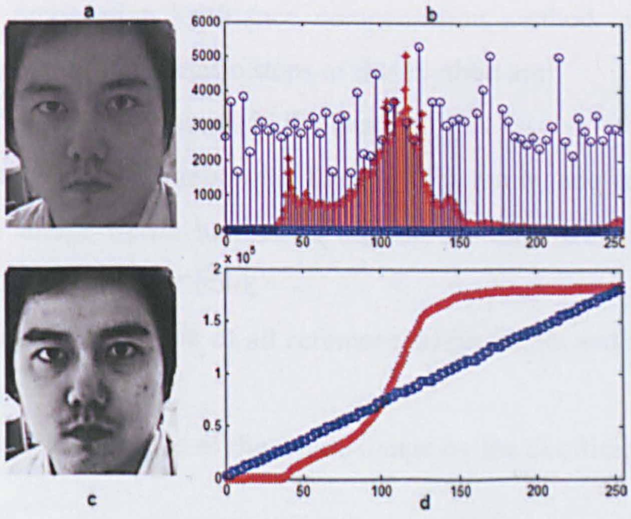


Figure 3.23 Histogram equalisation
a: Original image; b: Original (red) and equalised (blue) histograms;
c: histogram equalised image; d: accumulated histograms of b

If the number of pixels with greyscale i is denoted by n_i, then the normalised histogram is $h(i) = \frac{n_i}{M \times N}$. It is clear that $h(i) \in [0,1]$. From this definition, it can be found that the histogram of discrete digital image is actually the probabilities of all greyscale levels. Because the greyscales of the image have arbitrary distributions, the histograms also vary wildly.

The purpose of histogram equalisation is to re-distribute the greyscales in the image from a narrow range to a wide range to eliminate the affection brought about by the light conditions and differences of devices. It is an effective method to enhance the contrast, promote image qualities, and provide luminance correction.

As show in Figure 3.23, the histogram equalisation is an effective luminance correction method. It can be seen from the histograms in Figure 3.23(b), the distributions of pixels are stretched more uniformly. It is usually applied for images taken in dark environment. Although the histogram equalisation is a greyscale image processing technique, it can also be applied to colour images due to the fact that each component of colour images can be treated as a greyscale image. So it is used in the proposed face detection algorithm.

(2) Reference white

Hsu *et al.* (2002) proposed a luminance compensation method called "reference white" for face detection. The specific steps of this method are:

(1) sort the greyscales of all pixels in the image;
(2) count the number of pixels with the top 5% luminance (called reference white); the image needs luminance adjustment only while the number is bigger than a specific threshold;
(3) calculate the average value of all reference white pixels and stretch it to 255 by a coefficient; and
(4) compensate the luminance of the whole image by the coefficient.

This method is based on two facts: (1) there are "real white" (Poynton, 1996) pixels existing in all images; (2) the dominant bias colours are usually similar to the real white (Hsu *et al.*, 2002). This method uses the reference white luminance compensation method and nonlinear transformation in *YCbCr* colour space to achieve good results for face detection.

To test the effectiveness of this method, the author has applied it by a Matlab programme to 50 colour images. The results of my experiment shown that "reference white" is not as effective as claimed by the previous publications. So this method will not be used in my method.

3.1.4 Features Positioning

The skin colour based face detection method can quickly obtain all the possible face candidates in the input images. Furthermore, skin colour information is not sensitive to size, position, posture, and other interferences. Angelopoulou (2001) pointed out that the reflexion ratios of human skins are highly correlated regardless of race, gender, and age. Jablonski (2000) found that the reflexion ratio of human skins closely relates to ultraviolet radiation levels. He also proved the uniformity of human skin colours in biology. These researchers provided strong evidence related to statistics, physics, and biology. Thus, the skin colour can be applied for both the initial step of the detection system and the accurate location of faces. This section discusses the techniques related to the determination of the positions of features of a face.

The skin colour model can extract all face candidates from the image. The candidates need to be further verified for face or non-face. Two types of methods for verification are feature-based methods and template matching methods. The feature-based methods are bottom-up methods, which try to find facial features in each face candidate according to predefined rules. Human faces vary from person to person, but they have certain patterns with several features for computers, like eyes, eyebrows, mouth, and nose. As such, it is possible to search for these facial features to verify faces. If the features are found, the candidates are verified as a face; otherwise, they are considered as a non-face.

3.1.4.1 Integral Projection

Greyscales integral projection is one of the widely used techniques for face detection (Belhumeur *et al.*, 1997; Feng and Yuen, 1998; Liu, 2003). It can be used not only for finding facial boundaries, but also for locating the eye region. For an

M×*N* greyscale image, the integral projection function (IPF) (Huang and Su, 2004) is defined as:

$$\begin{aligned} \boldsymbol{H}(x) &= \sum_{y=0}^{N-1} \boldsymbol{I}(x,y) \\ \boldsymbol{V}(y) &= \sum_{x=0}^{M-1} \boldsymbol{I}(x,y) \end{aligned} \tag{3.32}$$

Because the eye region has lower greyscale values compared to other facial areas, the eye region can be located by analysing the trough of ***H***(*x*) and ***V***(*x*). Then the face candidates are verified by combining the geometrical location of eyes and other features. It can be seen from Figure 3.24 that the integral projection method is also appropriate for finding the position of the mouth.

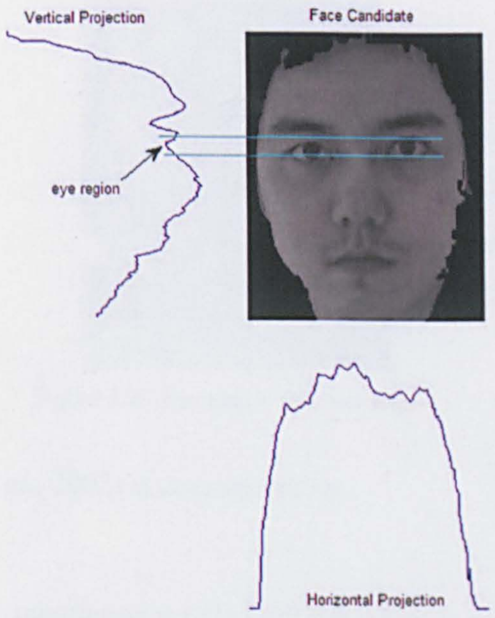


Figure 3.24 Integral projections of a face candidate

The projection functions can be computed effectively in each direction. The calculated projection functions are two 1D vectors. The valley regions can be easily located by analysing the first-order derivatives. This method is used in the proposed face detection method.

3.1.4.2 Colour Map

Hsu *et al.* (2002) proposed colour maps to detect the eyes and mouth from a face candidate based on the experimental observation that higher *Cb* values and lower *Cr* values exist in the eye region in the *Cb-Cr* plane. Because the eyes and mouth are the most apparent regions on human faces, the face boundaries are determined based on geometric restrictions if the positions of eyes and mouth can be found. The colour maps are not involved in the luminance component; thus, this method can deal with images taken in different light conditions.

The colour map (Hsu *et al.*, 2002) of eyes is described by Equation 3.33.

$$eyemap = \frac{1}{3}(Cb^2 + \widetilde{Cr}^2 + \frac{Cb}{Cr}) \tag{3.33}$$

Where: $\widetilde{Cr}$ is the complementary image of *Cr* and all the images are normalised to the range [0,255]. The result of applying the eye map is shown in Figure 3.25:

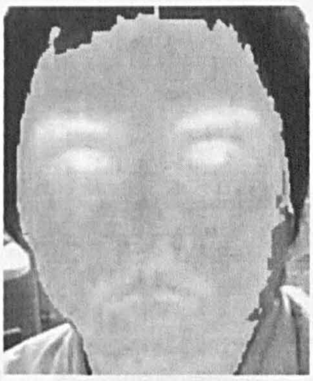


Figure 3.25 Eye map in face candidate

The mouth map (Hsu *et al.*, 2002) is constructed by:

$$\begin{gathered} mouthmap = Cr^2 \cdot \left(Cr^2 - \eta \cdot \frac{Cr}{Cb}\right)^2 \\ \eta = \tau \cdot \frac{\frac{1}{n}\sum_{(x,y)\in S} Cr(x,y)^2}{\frac{1}{n}\sum_{(x,y)\in S} \frac{Cr(x,y)}{Cb(x,y)}} \end{gathered} \tag{3.34}$$

Where: the ratio coefficient τ is set to 0.95 initially in Hsu's method, and all images are normalised to [0,255]. The detected mouth mask is shown in Figure 3.26.

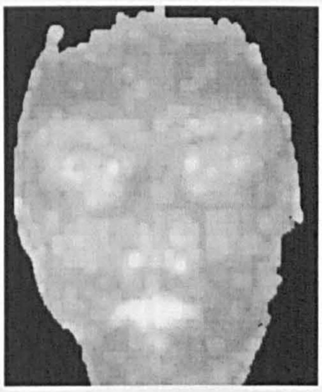


Figure 3.26 Mouth map in face candidate

The results of the eyes and mouth positions are shown in Figure 3.27 and Figure 3.28 for different thresholds, respectively. It can found that the colour maps of eyes and mouth are effective according to the results after applying the eye map and mouth map. The found regions are normally the brightest areas in the result images, which can be located by applying threshold processing. However, the processing is an iterative procedure. The users have to test a considerable number of different thresholds to eventually find the locations of the features (eyes and mouth). So the effectiveness of this is dependent on the users' experiences and it is time-consuming. Therefore, it is not used in the face detection method.

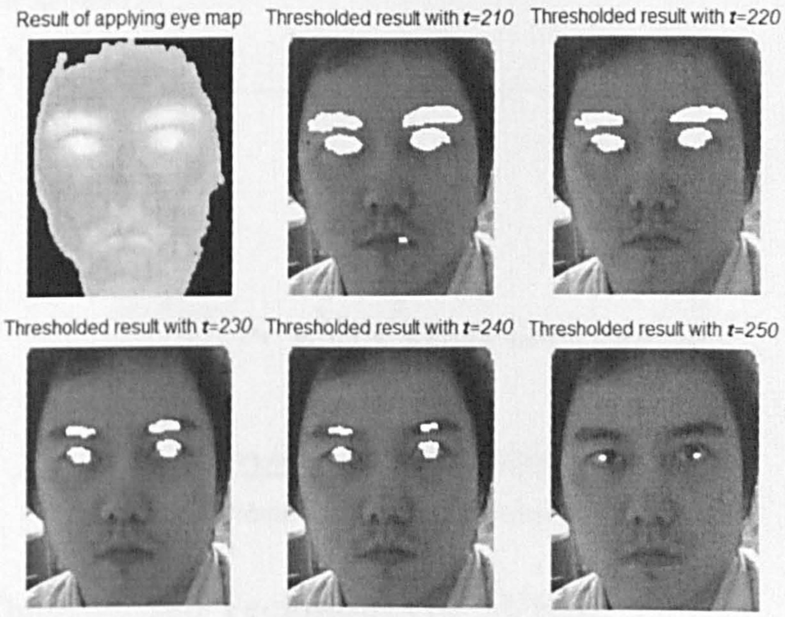

Result of applying eye map
Thresholded result with t=210
Thresholded result with t=220
Thresholded result with t=230
Thresholded result with t=240
Thresholded result with t=250

Figure 3.27 Applying thresholds on eye map result

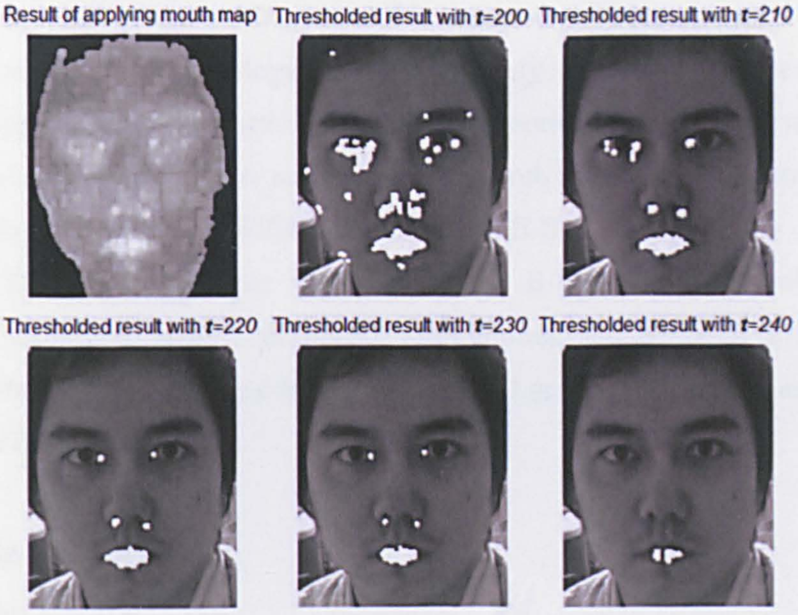

Result of applying mouth map
Thresholded result with t=200
Thresholded result with t=210
Thresholded result with t=220
Thresholded result with t=230
Thresholded result with t=240

Figure 3.28 Applying thresholds on mouth map result

3.1.4.3 Comparison between IPF and Colour Map

The comparison between IPF and colour map methods is illustrated in Figure 3.29. The execution time of IPF is in the range of 0.02~0.05 seconds. The execution time of colour maps method varies from 0.23~0.68 seconds. According to the experiments, The IPF is about 4~30 times faster than colour map. So it will be used for face detection in Chapter 4.

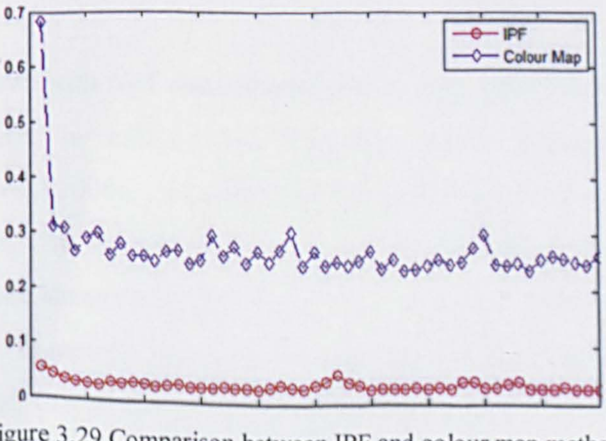


Figure 3.29 Comparison between IPF and colour map method

3.2 Theories and Techniques of NURBS

One of the main tasks for emotional bio-robots is generic 3D human face modelling. This model should be parameterised for flexible control purpose. So it important to investigate what kind of 3D geometry computation techniques is effective in mathematically modelling a 3D generic 3D model. NURNS, as a powerful 3D geometry computation technology, has been widely applied in engineering 3D modelling applications. This section discusses the theories of the NURBS technique. As a generalised form of Bézier and B-Spline method, the NURBS method is ideal for geometric design. The differences between B-Spline and Bézier are also discussed. The NURBS curves which are those B-Spline curves with weight coefficients to provide more flexibility for local control. The B-Spline surfaces and NURBS surfaces, which are used for representing 3D geometric surfaces, are also be discussed in this section.

3.2.1 B-Spline Curve

3.2.1.1 Bézier Curves

Bézier curve is the weighted sum of all control points (Salomon, 2006). Thus, the generalised representation of the Bézier curve defined by a set of N+1 control points $\boldsymbol{P}_0, \boldsymbol{P}_1, \boldsymbol{P}_2, \dots \boldsymbol{P}_N$ is shown in Equation 3.35.

$$
\begin{aligned}
C(u) &= \sum_{i=0}^{N} \boldsymbol{B}_{n,i}(u) \cdot \boldsymbol{P}_i \\
\boldsymbol{B}_{n,i}(u) &= \binom{n}{i} u^i (1-u)^{n-i} \quad , \begin{cases} 0 \le u \le 1 \\ 0 \le \boldsymbol{B}_{n,i}(u) \le 1 \end{cases} \\
\binom{n}{i} &= \frac{n!}{i!\,(n-i)!}
\end{aligned}
\tag{3.35}
$$

The order of the curve with N+1 control points is N. The i-th basis function of order N is $\boldsymbol{B}_{n,i}(u)$, which is called the Bernstein basis function or Bernstein polynomials (Salomon, 2006). According to the generalised representation of the Bézier curve defined by Equation 3.35, it can be seen that each control point contributes to the final shape of the curve. So the curve shape will change if any one of the control points is moved. This is not convenient for building up a generic 3D face model, because the ability of local modification of all facial features are necessary for the generic model.

3.2.1.2 B-Spline Curves

The B-Spline curve is very similar to the Bézier curve, which is calculated according to the weighted sum of control points. The difference is that every single point on the

Bézier curve is computed using all control points, but the points on the B-Spline curve are only affected by some local control points. For a certain parameter value u, some basis functions are zero. Thus, the corresponding control points are not used for computation.

A new variable introduced in the B-Spline is knot vector, which are the elements of a non-decreasing sequence. $U = \{u_0, u_1, u_2, \dots u_m\}$ is called knot vector and $u_i (0 \le i \le m)$ are called knots. Let the appearance of a knot u_i be denoted by k. If k>1, the knot is called a multiple knot with multiplicity k. Otherwise, it is called a simple knot. The range $[u_i, u_{i+1})$ is called knot span on which all B-Spline basis functions are defined.

Unlike the Bézier curve, the degree of the B-Spline curve is independent of the number of control points. The degree decides that the current point on the curve is affected by how many control points. The i-th basis of the B-Spline curve with p degree is (Boor, 1972):

$$\begin{aligned} N_{i,0}(u) &= \begin{cases} 1, if\ u_i \le u < u_{i+1} \\ 0 \quad , otherwise \end{cases} \\ N_{i,p}(u) &= \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \\ Note&: \frac{0}{0} = 0 \end{aligned} \tag{3.36}$$

Equation 3.36 is the de Boor-Cox basis function. Figure 3.30 shows an example of the basis function. Although this is the standard equation for defining the B-Spline, this method is rarely used for real computation.

As demonstrated in Figure 3.30, the B-Spline basis function with zero degree is a step function that is only non-zero in certain knot spans. The higher degree basis functions are computed using the de Boor-Cox basis function. It is clear that, although the degree is zero, the basis functions are only non-zero in one knot span; when the degree is one, the basis functions are non-zero in two knot spans whereas when the degree is two, the basis functions are non-zero in three knot spans. This property can be expanded to an arbitrary degree: The basis function with degree p is non-zero in p+1 knot spans, which means the corresponding control point can affect the parameters in p+1 knot spans.

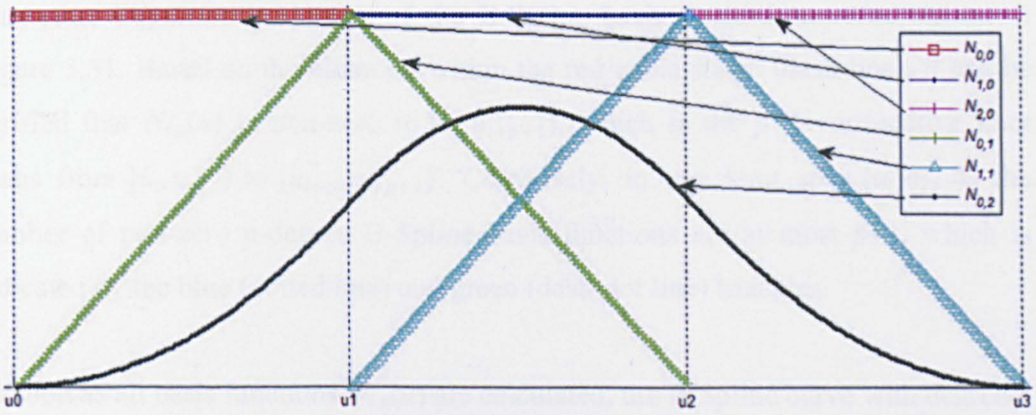


Figure 3.30 B-Spline basis function $N_{i,q}$

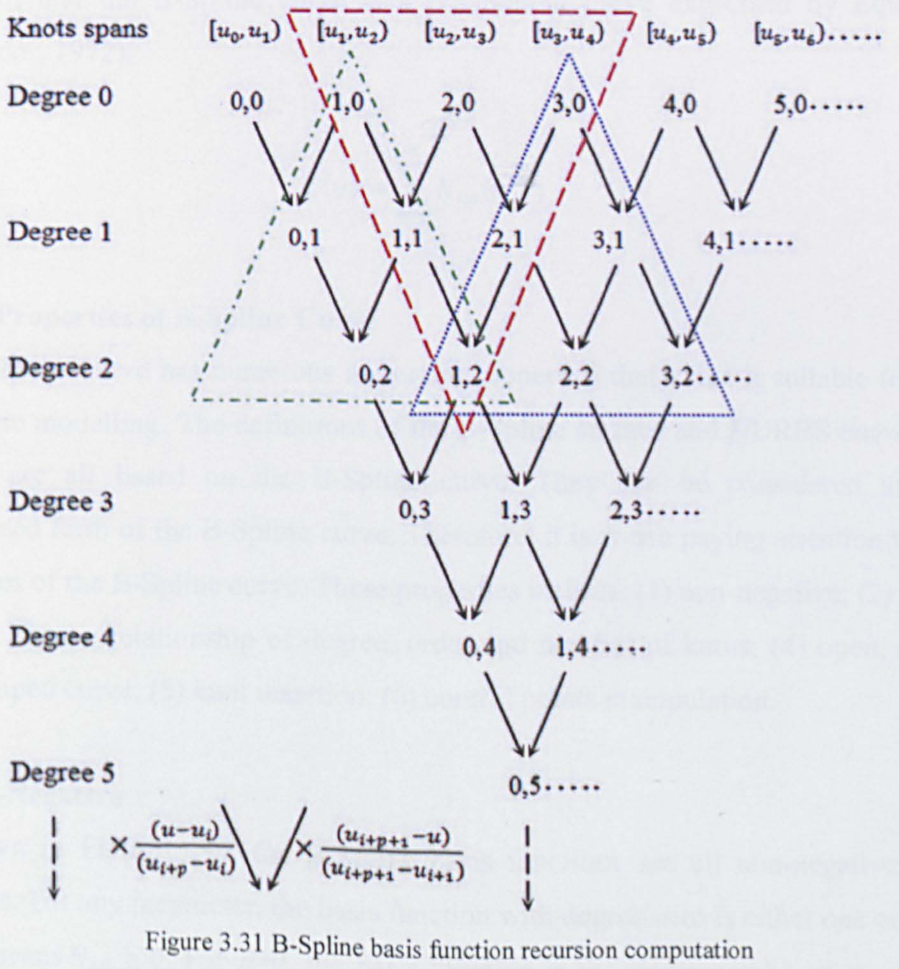


Figure 3.31 B-Spline basis function recursion computation

The recursion computation of the B-Spline basis function can use a similar method as the Bézier curve shown in Figure 3.31. As previously discussed, all zero-degree basis functions, which are associated with certain knot spans, are step functions. Then in the red nabla shape (dash line) started from degree one, each basis is calculated by two from the previous degree level determined using Equation 3.36.

The other important properties of the B-Spline basis function are also shown in Figure 3.31. Based on the elements within the red nabla shape (dash line), it can be verified that $N_{i,p}(u)$ is non-zero in $[u_i, u_{i+p+1})$, which is the p+1 consecutive knot spans from $[u_i, u_{i+1})$ to $[u_{i+p}, u_{i+p+1})$. Conversely, in any knot span $[u_i, u_{i+1})$, the number of non-zero p-degree B-Spline basis functions are at most p+1, which is indicated by the blue (dotted line) and green (dash-dot line) triangles.

As soon as all basis functions $N_{i,p}(u)$ are calculated, the B-Spline curve with degree p and n+1 control points $\boldsymbol{P}_0$, $\boldsymbol{P}_1$, ...$\boldsymbol{P}_n$ can be represented based on the basis function. It is shown that the B-Spline curve is a polynomial curve expressed by Equation 3.37 (Boor, 1972).

$$C(u) = \sum_{i=0}^{n} N_{i,p}(u)\boldsymbol{P}_i \tag{3.37}$$

3.2.1.3 Properties of B-Spline Curve

The B-Spline curve has numerous important properties that make it suitable for 3D geometric modelling. The definitions of the B-Spline surface and NURBS curve and surface are all based on the B-Spline curve. They can be considered as the generalised form of the B-Spline curve. Therefore, it is worth paying attention to the properties of the B-Spline curve. These properties include: (1) non-negative; (2) local support; (3) the relationship of degree, order and number of knots; (4) open, close and clamped curve; (5) knot insertion; (6) control points manipulation.

(1) Non-Negative

As shown in Figure 3.30, the B-Spline basis functions are all non-negative step functions. For any parameter, the basis function with degree zero is either one or zero, which means $N_{i,0} \geq 0$. For p>0, the basis function is the linear combination of two adjacent basis functions in the previous layer, as shown in Figure 3.31. The sum of all basis functions with degree p is one, that is $\sum_{i=0}^{m} N_{i,n}(u) = 1, u \in [u_i, u_{i+1})$.

(2) Local Support

Although the B-Spline curve seems to be the linear combination of all control points, the basis functions are computed only in the knot spans where all involved lower degree basis functions are non-zero. Therefore, the control points of the B-Spline curve can be used to manipulate the curve shape in a certain parameter range, highlighting one of the most important differences between the Bézier and B-Spline methods. As shown in Figure 3.31, the non-zero knot spans of a control point and the non-zero basis functions in a knot span can be obtained by checking the red (dashed), blue (dotted), and green (dash-dot) triangles. This property makes the B-Spline curve flexible for 3D geometrical modelling.

(3) Degree, Order and Knot Vector

Unlike the Bézier curve, the degree of the B-Spline curve is not dependant on the number of control points. The degree is a parameter for generating the B-Spline curve, which becomes smoother and smoother as the degree increases. However, the higher degree causes more computation time. To benefit the flexibility of the B-Spline technique and reduce the computation complexity at the same time, the degree is usually chosen as three, which means it is a cubic B-Spline curve (Sun *et al.*, 2007; Liu and Ning, 2008).

The order equals the degree plus one. Moreover, in a B-Spline curve with m+1 elements knot vector $U=\{u_0,u_1,u_2,...u_m\}$, n+1 control points $\boldsymbol{P}_0,\boldsymbol{P}_1,\boldsymbol{P}_2,...\boldsymbol{P}_n$, and degree p (or order p+1), the relationship between these number is: $m=n+p+1$. The minimum number of control points equals p+1.

In some representation, the values of the non-decreasing sequence of the knot vector are greater than one. For example, [0,0,0,1,2,3,3,4,5,5,5] is a satisfied knot vector. The range of the parameters for evaluating the B-Spline curve is from the minimum to the maximum of the knot values. Multiple knots are acceptable, except that the multiplicity exceeds the order p.

(4) Open, Close, and Clamped Curve

The open curve means the starting and end points are at different positions. In contrast, it is called a close B-Spline curve if the starting and end points are the same.

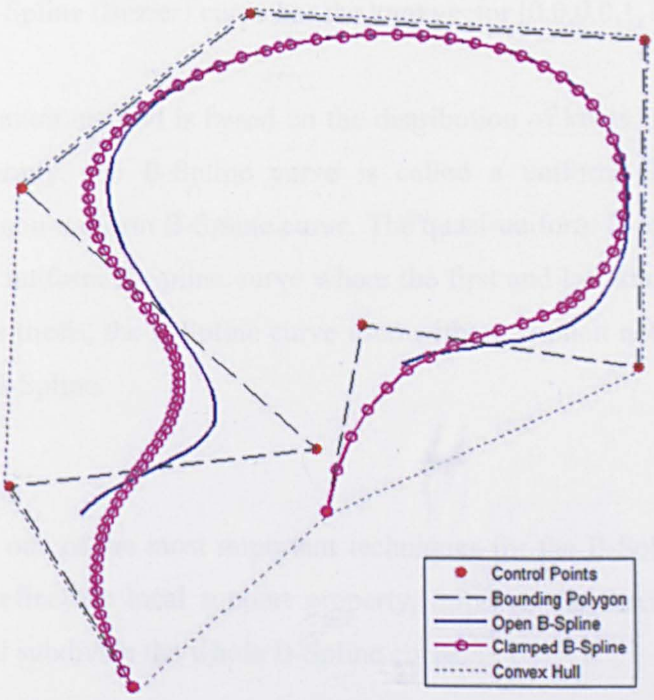


Figure 3.32 Open and clamped B-Spline curves and convex hull (p=3)

As shown in Figure 3.32, the entire B-Spline curve is located inside the convex hull formed by its control points. More specifically, the curve defined by $u \in [u_i, u_{i+1})$ is in the convex hull formed by control points $\boldsymbol{P}_{i-p}, \boldsymbol{P}_{i-p+1}, \dots, \boldsymbol{P}_i$. This property is very important for flexible control for CAGD. If p+1 continuous control points sits on a line, the B-Spline calculated by these control points will become a straight line. If p continuous control points are identical, the curve will path through the control point.

The Bézier curve passes through the first and the last control points, but this is not necessarily true for B-Spline curve. The B-Spline curve is called a clamped curve or quasi-uniform curve if it passes through the first and last control points. These two control points are visited by the B-Spline curve with degree p by arranging the knots vector to make sure the first and last knots repeat p+1 times. The clamped B-Spline curve is tangent to the first and last edges of the control polygon (Figure 3.32).

$$\begin{gathered} u_0 = u_1 = u_2 = \cdots = u_p \\ u_{m-p} = u_{m-p+1} = \cdots = u_m \end{gathered} \tag{3.38}$$

The clamped B-Spline curve is defined within $[u_p, u_{m-p})$. In particular, if there is only one non-zero knot span in a clamped B-Spline knot vector, it becomes a Bézier curve. Thus, the Bézier curve is a special case of a B-Spline curve. For example, a clamped cubic B-Spline (Bézier) curve has the knot vector [0,0,0,0,1,1,1,1].

Another classification method is based on the distribution of knots. If the knots are distributed uniformly, the B-Spline curve is called a uniform B-Spline curve; otherwise, it is a non-uniform B-Spline curve. The quasi-uniform B-Spline curve is a special case of a uniform B-Spline curve where the first and last knots are repeated p+1 times. In this thesis, the B-Spline curve used without explicit notation is mainly a clamped open B-Spline.

(5) Knot Insertion

Knot insertion is one of the most important techniques for the B-Spline and can be used to further refine the local support property, improve the flexibility of shape manipulation, and subdivide the whole B-Spline curve.

Consider the B-Spline curve with degree p (order p+1) and n control points:

$$\boldsymbol{C}(u) = \sum_{j=0}^{n} \boldsymbol{N}_{j,p}(u)\boldsymbol{P}_j. \tag{3.39}$$

If the original knot vector is $\boldsymbol{U} = \{u_0, u_1, u_2, \dots, u_{n+p}\}$, knot insertion is used to insert a new knot $u \in [u_i, u_{i+1}]$ to form the new knot vector $\boldsymbol{U}' = \{u_0, u_1, \dots, u_i, u, u_{i+1}, \dots, u_{n+p}\}$, which is re-labelled as $\boldsymbol{U}' = \{u'_0, u'_1, \dots, u'_i, u'_{i+1}, u'_{i+2}, \dots, u'_{n+p+1}\}$, as shown in Figure 3.33. The new knot vector defines a series of new B-Spline basis functions $\boldsymbol{N}'_{j,p}(u)$ ($j = 0,1,\dots,n+1$). New knot inserted requires a new control points. Then the original curve is represented by the new basis function and new control points $\boldsymbol{P}'_j$, expressed by Equation 3.40.

$$\boldsymbol{C}(u) = \sum_{j=0}^{n+1} \boldsymbol{N}'_{j,p}(u)\boldsymbol{P}'_j \tag{3.40}$$

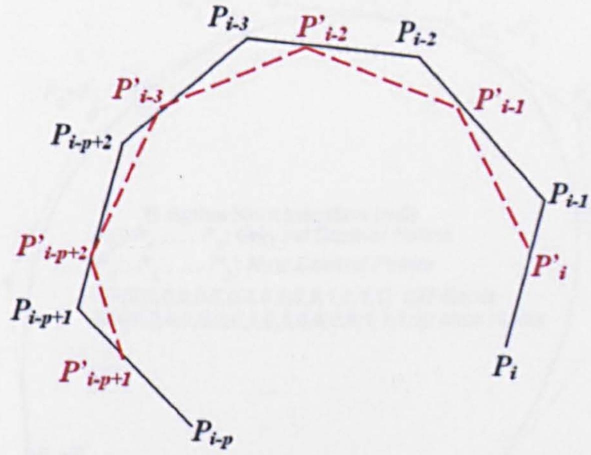


Figure 3.33 Corner cutting to generate new control points for knot insertion

Boehm (1980) described the computation method for the new control points as:

$$\begin{cases} P'_j = P_j, & j = 0,1,\dots,i-p \\ P'_j = (1-\lambda_j)P_{j-1} + \lambda_j P_j, & j = i-p+1,\dots,i \\ P'_j = P_{j-1}, & j = i+1,\dots,n+1 \end{cases} \tag{3.41}$$

Where: $\lambda_j = \frac{u-u_j}{u_{j+p}-u_j}$.

r in Equation 3.41 is the multiplicity of knot u in the original knot vector U. If $u_i<u<u_{i+1}$, then r=0; if r is positive real number and r<p-1, then u=u_i=...=u_{i-r+1}. Two situations of knot insertion are discussed here.

(a) Simple Knot Insertion (r=0)

If r=0, the insertion is the rearrangement of knot elements $u_{i-p+2},\dots,u_{i+p-1}$ and control points $P_{i-p+1},\dots,P_i$. The original control points $P_{i-p+2},\dots,P_{i-1}$ are replaced by new control points $P'_{i-k+2},\dots,P'_i$ (Figure 3.34 and Figure 3.35). Control points in the dotted box (grey) are the original ones that need to be replaced. Those in the dashed box (black) are the new control points. The final control points are in the solid box (red).

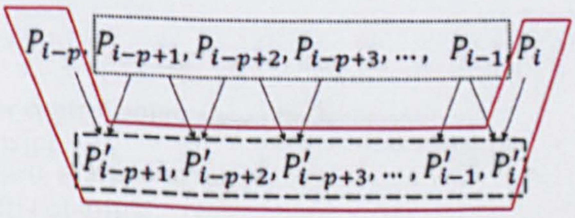

P_{i-p}
$P_{i-p+1}, P_{i-p+2}, P_{i-p+3}, \dots, P_{i-1}$
P_i
$P'_{i-p+1}, P'_{i-p+2}, P'_{i-p+3}, \dots, P'_{i-1}, P'_i$

Figure 3.34 Control points alteration caused by knot insertion with r=0

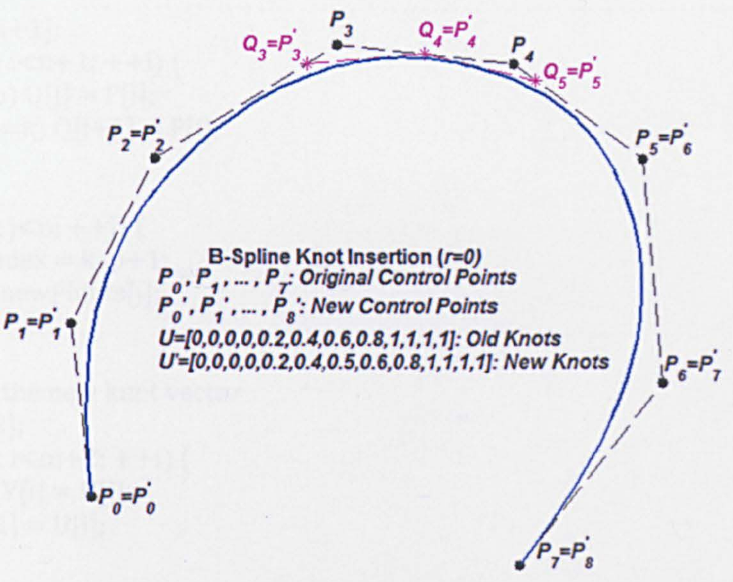

P_3
$Q_3=P'_3$
$Q_4=P'_4$
P_4
$Q_5=P'_5$
$P_2=P'_2$
$P_5=P'_6$
B-Spline Knot Insertion (r=0)
$P_0, P_1, ..., P_7$: Original Control Points
$P_0', P_1', ..., P_8'$: New Control Points
U=[0,0,0,0,0.2,0.4,0.6,0.8,1,1,1,1]: Old Knots
U'=[0,0,0,0,0.2,0.4,0.5,0.6,0.8,1,1,1,1]: New Knots
$P_1=P'_1$
$P_6=P'_7$
$P_0=P'_0$
$P_7=P'_8$

Figure 3.35 Knot insertion (*p*=3, *r*=0, u=0.5)

The algorithm for inserting a single knot to the knot vector is shown below.

```
Input:  control points array P[0:n] with n+1 control
        points: $P_0$, $P_1$, ..., $P_n$; knot vector U[0:m]
        with m+1 knots: $u_0$, $u_1$, ..., $u_m$; degree p
        a new knot t to be inserted
Output: new control points Q[0:n+1]
        new knot vector V[0:m+1]

// 1.  check whether m=n+p+1 is satisfied
if (m!=n+p+1)
   return;

// 2.  find the knot span where t is located
int k = p;
bool index = false;
for (int i=p; i<m-p; ++i) {
   if (t >= U[i] && t < U[i+1]) {
      k = i;
      index = true;
      break;
   }
}

if (!index)
   return;

// 3.  create p new control points $Q_{k-p+1}$ to $Q_k$
Point3D newPoints[p];
for (int i=k-p+1, j=0; i<=k; ++i, ++j) {
   ai = (t-U[i])/(U[i+p]-U[i]);
   newPoint[j] = (1-$a_i$)*P[i-1]+$a_i$*P[i];
}
// 4.  replace the original points $P_{k-p+1}$ to $P_{k-1}$ with $Q_{k-p+1}$ to $Q_k$
```

```
Point3D Q[n+1];
for (int i=0; i<n+1; ++i) {
  if (i<=k-p) Q[i] = P[i];
  else if (i>=k) Q[i+1] = P[i];
}

for (int j=0; j<p; ++j) {
  int startIndex = k-p+1;
  Q[j+k] = newPoints[j];
}

// 5. create the new knot vector
float V[m+2];
for (int i=0; i<m+2; ++i) {
  if (i<=k) V[i] = U[i];
  else V[i+1] = U[i];
}

V[k+1] = t;

return Q and V;
```

(b) Insert Knot with Multiplicity (0<*r*<*p*-1)

If a new knot equal to the *i*-th element in the original knot vector (u=u_i) is to be inserted, let's start with r=1. According to Equation 3.41, $\lambda_i = \frac{u-u_i}{u_{i+p}-u_i} = \frac{0}{u_{i-p}-u_i} = 0$, so $\boldsymbol{P}'_i = (1-\lambda_i)\boldsymbol{P}_{i-1} + \lambda_i \boldsymbol{P}_i = \boldsymbol{P}_{i-1}$. Thus, it is actually not necessary to calculate the last one new control point because $\boldsymbol{P}'_i = \boldsymbol{P}_{i-1}$ (Figure 3.36 and Figure 3.37).

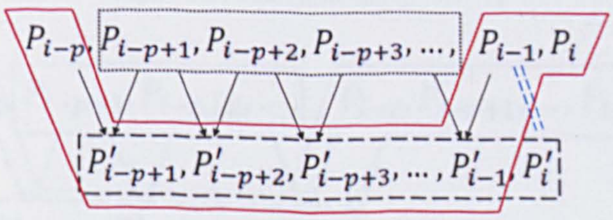

P_{i-p}, $P_{i-p+1}, P_{i-p+2}, P_{i-p+3}, \dots,$ P_{i-1}, P_i
$P'_{i-p+1}, P'_{i-p+2}, P'_{i-p+3}, \dots, P'_{i-1}$ P'_i

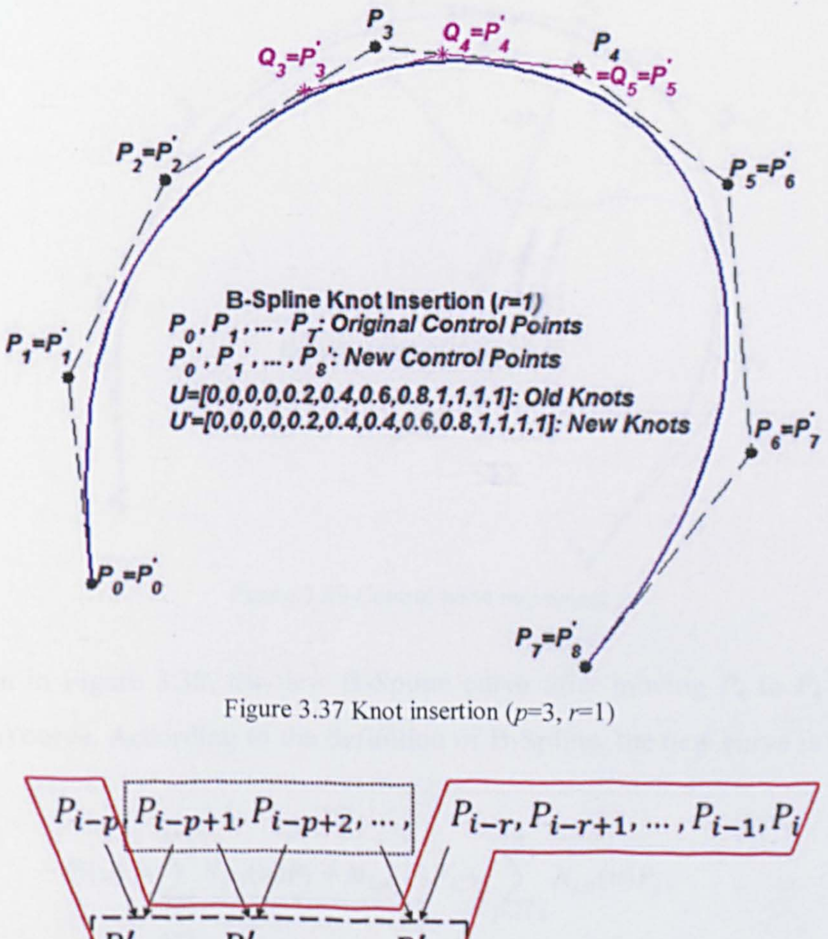
Figure 3.36 Knot insertion with multiplicity or r=1

If r>1, more coefficients λ_j $(i-s+1 \le j \le i)$ will be zero so that the computation of the last r new control points can be avoided, which is illustrated in Figure 3.38.

The algorithm for inserting a knot with multiplicity ***h*** is shown below.

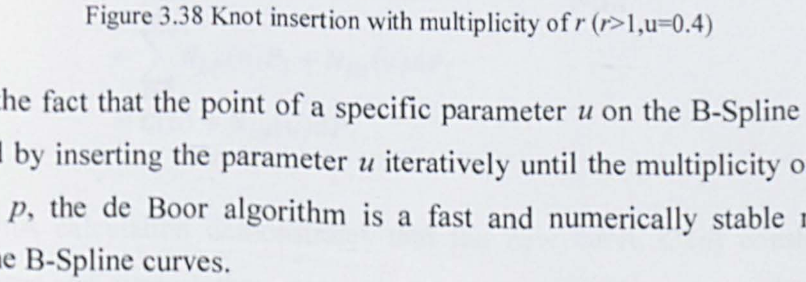
```
Input:  knot t in [Uk, Uk+1) with initial multiplicity s to
        be inserted h times, which h+s<=p; the degree p of
        the given B-Spline curve
Output: a new set of control points Q[0:n+h] after t is
```

```
    inserted h times.

Let control points P_k, P_{k-1}, ..., P_{k-p} be renamed as P_{k,0},
P_{k-1,0}, ..., P_{k-p,0}

for r:=1 to h do
  for i:= k-p+r to k-s do
      Let a i,r = (t-ui) / (ui+p-r+1 - ui);
      Let P i,r = (1-a_{i,r}) P_{i-1,r-1} + a_{i,r} P_{i,r-1}
```



P_3
$Q_3=P'_3$
$Q_4=P'_4$
P_4
$=Q_5=P'_5$
$P_2=P'_2$
$P_5=P'_6$
$P_1=P'_1$
B-Spline Knot Insertion ($r=1$)
$P_0, P_1, \dots, P_7$: Original Control Points
$P'_0, P'_1, \dots, P'_8$: New Control Points
$U=[0,0,0,0,0.2,0.4,0.6,0.8,1,1,1,1]$: Old Knots
$U'=[0,0,0,0,0.2,0.4,0.4,0.6,0.8,1,1,1,1]$: New Knots
$P_6=P'_7$
$P_0=P'_0$
$P_7=P'_8$

Figure 3.37 Knot insertion (p=3, r=1)



P_{i-p}
$P_{i-p+1}, P_{i-p+2}, \dots,$
$P_{i-r}, P_{i-r+1}, \dots, P_{i-1}, P_i$
$P'_{i-p+1}, P'_{i-p+2}, \dots, P'_{i-r}$

Figure 3.38 Knot insertion with multiplicity of r (r>1,u=0.4)

Based on the fact that the point of a specific parameter u on the B-Spline curve can be reached by inserting the parameter u iteratively until the multiplicity of u equals its degree p, the de Boor algorithm is a fast and numerically stable method to evaluate the B-Spline curves.

(6) Control Points Manipulation

The shape of the B-Spline curve changes while moving its control points, which is obviously the easiest way to change the shape of the curve. According to the local support property, the change of control point $\boldsymbol{P}_i$ only affects the curve of parameter range $[u_i, u_{i+p})$. So if the i-th control points $\boldsymbol{P}_i$ of B-Spline $\boldsymbol{C}(u)$ is moving from the original position to $\boldsymbol{P}_i'$, then the curve section defined by $[u_i, u_{i+p})$ changes with the same trend as the moving vector $\Delta\boldsymbol{P}_i$ of $\boldsymbol{P}_i$.

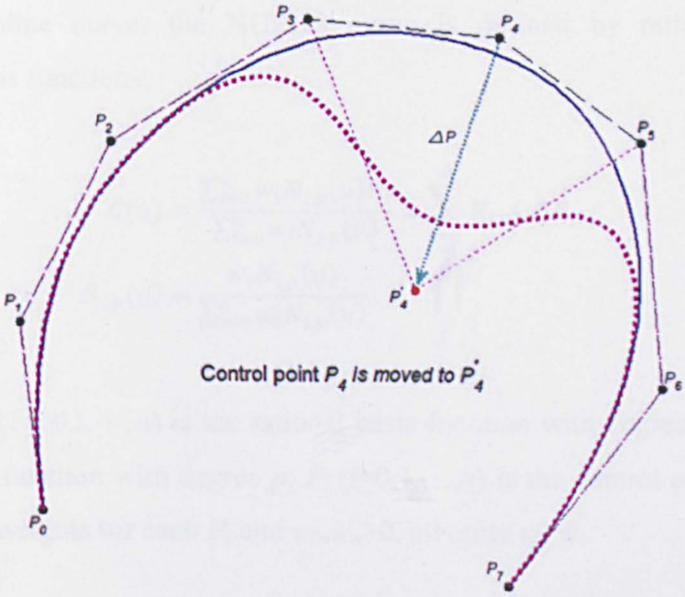


Figure 3.39 Control point movement

As shown in Figure 3.39, the new B-Spline curve after moving $\boldsymbol{P}_4$ to $\boldsymbol{P}_4'$ is dotted (magenta) curve. According to the definition of B-Spline, the new curve is computed as:

$$\begin{aligned} \boldsymbol{C}'(u) &= \sum_{j=0}^{i-1} \boldsymbol{N}_{j,p}(u)\boldsymbol{P}_j + \boldsymbol{N}_{i,p}(u)\boldsymbol{P}_i' + \sum_{j=i+1}^{n} \boldsymbol{N}_{j,p}(u)\boldsymbol{P}_j \\ &= \sum_{j=0}^{i-1} \boldsymbol{N}_{j,p}(u)\boldsymbol{P}_j + \boldsymbol{N}_{i,p}(u)(\boldsymbol{P}_i + \Delta\boldsymbol{P}_i) + \sum_{j=i+1}^{n} \boldsymbol{N}_{j,p}(u)\boldsymbol{P}_j \\ &= \sum_{j=0}^{n} \boldsymbol{N}_{j,p}(u)\boldsymbol{P}_j + \boldsymbol{N}_{i,p}(u)\Delta\boldsymbol{P}_i \\ &= \boldsymbol{C}(u) + \boldsymbol{N}_{i,p}(u)\Delta\boldsymbol{P}_i \end{aligned} \tag{3.42}$$

The previous calculation demonstrates that the new curve $\boldsymbol{C}'(u)$ consists of the original curve and a translation vector $\boldsymbol{N}_{i,p}(u)\Delta\boldsymbol{P}_i$. As previously analysed, $N_{i,p}(u)$ is non-zero in the range of $[u_i, u_{i+p})$ so that the original curve is only changed in this parameter range. This property of B-Spline method provides more flexibility of curve shape control than Bézier method. So the B-Spline is suitable for 3D face modelling by enabling the local facial features manipulation. The 3D face model with local feature manipulation property also can be used to express various expressions and generate a great number of models with different appearances.

3.2.2 NURBS Curve

Like the B-Spline curve, the NURBS curve is defined by rational B-Spline polynomial basis functions:

$$C(u) = \frac{\sum_{i=0}^{n} w_i \boldsymbol{N}_{i,p}(u)\boldsymbol{P}_i}{\sum_{i=0}^{n} w_i \boldsymbol{N}_{i,p}(u)} = \sum_{i=0}^{n} \boldsymbol{R}_{i,p}(u)\boldsymbol{P}_i$$
$$\boldsymbol{R}_{i,p}(u) = \frac{w_i \boldsymbol{N}_{i,p}(u)}{\sum_{i=0}^{n} w_i \boldsymbol{N}_{i,p}(u)} \tag{3.43}$$

Where: $\boldsymbol{R}_{i,p}(u)$ $(i = 0,1,\cdots,n)$ is the rational basis function with degree p; $\boldsymbol{N}_{i,p}(u)$ is a B-Spline basis function with degree p; $\boldsymbol{P}_i$ $(i=0,1,\ldots,n)$ is the control points; w_i is the corresponding weights for each $\boldsymbol{P}_i$ and $w_0, w_n>0$, all other $w_i \geq 0$.

NURBS has the same properties of the B-Spline curve, including local support, non-negative basis functions, the relationship among degrees, order and number of knots, open and clamped, and a convex hull. The weights affect the local shape of NURBS as determined by specific control points.

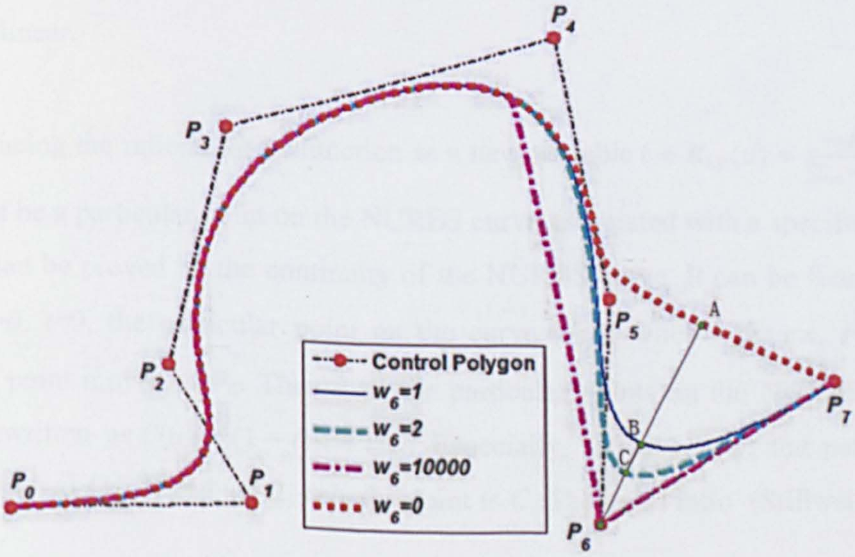


Figure 3.40 NURBS curves with one different weight

As demonstrated in Figure 3.40, all four NURBS curves (red, magenta, blue and cyan) have the same control points and weights except the one assigned to $\boldsymbol{P}_6$. The blue line is the original curve, and all weights are one. The w_6 of the dotted (red) curve is zero; the w_6 of the dashed (cyan) curve is two; and the w_6 of the dash-dot (magenta) curve is 10,000. It is clear that greater weight will pull the curve near the control point and vice versa. Zero weight has no impact on the curve. Infinite (approximated by 10,000) weight causes the curve to pass through the control points. Because the weights are assigned to control points, w_i can only affect the NURBS curve in the range of $[u_i, u_{i+p})$.

Piegl (1989) indicated that the weights have explicit geometric meaning. In Figure 3.40, the points on the curves of different weights w_6 are marked as $\boldsymbol{A}(w_6=0)$, $\boldsymbol{B}(w_6=1)$, $\boldsymbol{C}(w_6=2)$ and $\boldsymbol{P}_6(w_6=10{,}000)$.

$$\begin{aligned} \boldsymbol{C}(u) &= \frac{\sum_{i=0}^{n} w_i \boldsymbol{N}_{i,p}(u)\boldsymbol{P}_i}{\sum_{i=0}^{n} w_i \boldsymbol{N}_{i,p}(u)} \\ &= \frac{w_6 \boldsymbol{N}_{6,p}(u)\boldsymbol{P}_6 + \sum_{6\neq i=0}^{n} w_i \boldsymbol{N}_{i,p}(u)\boldsymbol{P}_i}{\sum_{i=0}^{n} w_i \boldsymbol{N}_{i,p}(u)} \end{aligned} \tag{3.44}$$

According to Equation 3.44, although the control points, degree, knot vector, weights (except w_6), and parameter value assigned to $\boldsymbol{P}_6$ remain unchanged, the NURBS curve becomes a univariant function of w_6. The curve is a rational linear function because the degree of w_6 is one in both numerator and denominator, so $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, and $\boldsymbol{P}_6$ are collinear.

By introducing the rational basis function as a new variable $t = \boldsymbol{R}_{i,p}(u) = \frac{w_i \boldsymbol{N}_{i,p}(u)}{\sum_{i=0}^{n} w_i \boldsymbol{N}_{i,p}(u)}$, there must be a particular point on the NURBS curve associated with a specific value t, which can be proved by the continuity of the NURBS curve. It can be found that, when $w_6=0$, $t=0$, the particular point on the curve is $\boldsymbol{A}$; when $w_6 \rightarrow +\infty$, $t=1$, the particular point moves to $\boldsymbol{P}_6$. Therefore, the particular points on the NURBS curve can be rewritten as $\boldsymbol{C}(u) = (1-t)\boldsymbol{A} + t\boldsymbol{P}_6$. Especially, if $w_6=1$, $t=a$, the particular point is $\boldsymbol{B}=(1-a)\boldsymbol{A}+a\boldsymbol{P}_6$ and $w_6=2$, $t=b$, the point is $\boldsymbol{C}$. The cross ratio (Stillwell, 2010) of points P_6, C, B, A is $\frac{\overline{P_6B}}{\overline{BA}}:\frac{\overline{P_6C}}{\overline{CA}} = \frac{1-a}{a}:\frac{1-b}{b} = w_6$, which is the weight of the control point P_6.

According to the analysis above and Figure 3.40, it can be seen that the weights of NURBS method provides more flexibility for curve shape control without the requirement of moving the control points by comparing with B-Spline method. This property is very useful for 3D modelling. In the new generic 3D face model proposed in Chapter 5, the boundaries of all facial features are constructed by NURBS curves. They can be used for changing the appearance of the face model.

3.2.3 B-Spline Surface

The B-Spline technique can also be applied in 3D space to generate B-Spline surfaces. If the two dimensions are denoted by u and v, the control points along u and v are represented as a 2D array (also called "control net" (Ganesh., 2008)) $P_{i,j}$ $(0 \le i \le m, 0 \le j \le n)$; the knot vector along them are denoted as $U=\{u_0,u_1,\ldots,u_a\}$and $V=\{v_0,v_1,\ldots,v_b\}$, and the degree along u and v are p and q, respectively. Thus, the following relationships between these variables hold:

$$\begin{aligned} a &= m + p + 1 \\ b &= n + q + 1 \end{aligned} \tag{3.45}$$

The B-Spline surface is defined as (Carswell and Lavery, 2006):

$$S(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} M_{i,p}(u)N_{j,q}(v)P_{i,j} \tag{3.46}$$

Where: $M_{i,p}(u)$ and $N_{j,q}(v)$ are the B-Spline basis functions in the directions of u and v. Figure 3.41 shows a Bicubic clamped B-Spline surface with 2D and 3D view.

Like B-Spline curves, B-Spline surfaces are classified into open surface, clamped surface, and closed surface according to different types of knot vectors. A clamped B-Spline surface's edges are the B-Spline curves defined by the first and last rows and columns of its control net.

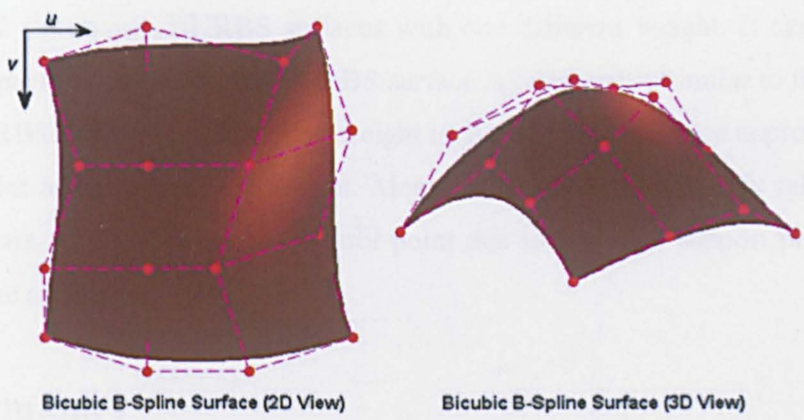

u
v
Bicubic B-Spline Surface (2D View)
Bicubic B-Spline Surface (3D View)

Figure 3.41 Bicubic clamped B-Spline surface with 2D and 3D view

3.2.4 NURBS Surface

The NURBS surface is based on the B-Spline surface by introducing weights as a parameter for flexible control. If $P_{i,j}$ is the control net, $w_{i,j}$ is the corresponding weights array; $M_{i,p}(u)$ and $N_{j,q}(v)$ and p and q are the degree along the directions of u and v, respectively. The NURBS surface is defined as (Shi, 1994):

$$\begin{aligned} \boldsymbol{S}(u) &= \frac{\sum_{i=0}^{m}\sum_{j=0}^{n} w_{i,j}\boldsymbol{M}_{i,j}(u)\boldsymbol{N}_{i,j}(v)\boldsymbol{P}_{i,j}}{\sum_{i=0}^{m}\sum_{j=0}^{n} w_{i,j}\boldsymbol{M}_{i,j}(u)\boldsymbol{N}_{i,j}(v)} \\ &= \sum_{i=0}^{m}\sum_{j=0}^{n} \boldsymbol{R}_{i,p,j,q}(u,v)\boldsymbol{P}_{i,j} \end{aligned} \tag{3.47}$$

Where: $\boldsymbol{R}_{i,p,j,q}(u,v)$ is the bivariant rational basis function; $w_{0,0}, w_{0,n}, w_{m,0}, w_{m,n} > 0$, all other $w_{i,j} \geq 0$, and

$$\boldsymbol{R}_{i,p,j,q}(u,v) = \frac{w_{i,j}\boldsymbol{M}_{i,p}(u)\boldsymbol{N}_{j,q}(v)}{\sum_{s=0}^{m}\sum_{t=0}^{n} w_{i,j}\boldsymbol{M}_{s,p}(u)\boldsymbol{N}_{t,q}(v)} \tag{3.48}$$

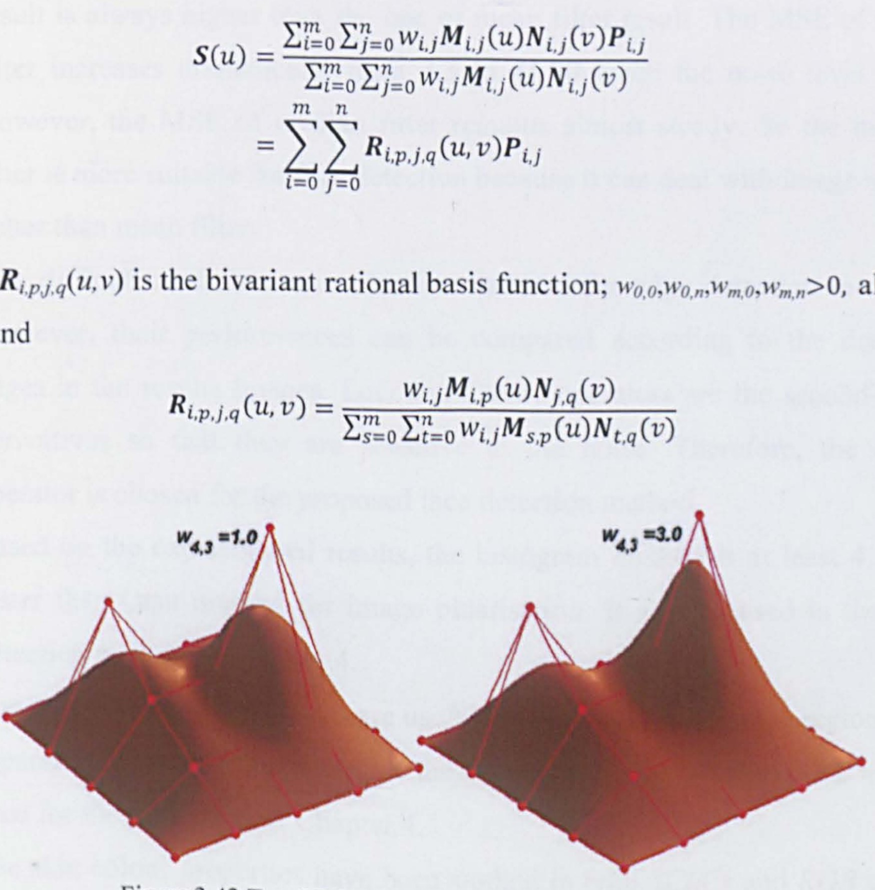


Figure 3.42 Two NURBS surfaces with one different weight

Figure 3.42 shows two NURBS surfaces with one different weight. It can be seen that the impact of weights on the NURBS surface is comparably similar to the impact on the NURBS curve. The greater the weight is, the closer the surface approaches the control point associated to the weight. Meanwhile, this affection only takes place within certain ranges around that control point due to the local support property of the B-Spline technique.

3.3 Summary

In this chapter, the theories of face detection and NURBS are discussed and evaluated. For face detection theories, the transformation between *YCbCr* and *RGB* colour spaces are discussed. The *Y* component can be used to adjust the luminance of the image. Next, the image processing technologies for face detection have been evaluated based on quantitative analysis. It has been found that:

1. While the level of noise in the original image rises from 0 to 10%, the PSNR drops for both mean filter and median filter. But the PSNR of median filter result is always higher than the one of mean filter result. The MSE of mean filter increases dramatically from 3.3 to 41.90 when the noise level rises. However, the MSE of median filter remains almost steady. So the median filter is more suitable for face detection because it can deal with image noises better than mean filter.
2. It is difficult to define an evaluation standard for edge detection methods. However, their performances can be compared according to the detected edges in the results images. LoG and Canny operators are the second-order derivatives so that they are sensitive to the noise. Therefore, the Sobel operator is chosen for the proposed face detection method.
3. Based on the experimental results, the histogram method is at least 4 times faster than Otsu method for image binarisation. It will be used in the face detection method in Chapter 4.
4. The morphological operations are useful for eliminating the noise regions and separating connected areas. All of the four morphological operations will be used for face detection in Chapter 4.
5. The skin colour properties have been studied in both *YCbCr* and *RGB* colour spaces based on the analysis of 1,009,523 skin pixels. Based on the

experimental evaluations, a new skin colour based face detection method will be proposed and developed in Chapter 4.

6. IPF is about 4~30 times faster than colour maps. So in the proposed face detection method, IPF is used for locating face regions.

For NURBS techniques, it has been found that:

1. In Bézier curves, each control point contributes to the final shape of the curve. So the curve shape will change if any one of the control points is moved. This is not convenient for building up a generic 3D face model.
2. B-Spline has the property of local support. This property of B-Spline method provides more flexibility of curve shape control than Bézier method. So the B-Spline is suitable for 3D face modelling by enabling the local facial features manipulation.
3. By introducing weights into B-Spline models, NURBS is a much more powerful model in controlling curves and surfaces without moving the control points by comparing with B-Spline method. This property is very useful for 3D modelling.
4. By comparing the properties of Bézier, B-Spline and NURBS techniques, it is easy to find that NURBS is the most suitable tool for 3D modelling of a face. Besides changing the shape of curves and surfaces by moving the control points, we can also benefit from the freeform control through adjusting the weights associated with control points. So NURBS curve and surface models and their special cases -- B-Spline curve and surface models will be used in proposing and developing the new method for building a generic 3D face model.

CHAPTER 4

A New Method for Face Detection

In this chapter, a new colour image face detection method has been proposed and designed based on human skin colour and eye position detection. The new method has been firstly proposed and developed for face detection in Section 4.1, then skin colour segmentation in *RGB* colour space on the luminance normalised image has been described and implemented in Section 4.2 and several geometric restrictions have been applied to reduce the noises in Section 4.3. Following that, the new algorithm of eye position detection has been developed to further identify which ones are real faces among all face candidates in Section 4.4. Finally, in Section 4.5, the new proposed method in this chapter has been implemented using Matlab. The experimental results and statistics analysis shows that compared with existing algorithms, the new proposed method is more effective and applicable because it only relies on the natural properties of the human face rather than using training datasets.

4.1 A New Face Detection Method

Some existing face detection methods segment the skin region in greyscale images while the others only consider one channel of the colour space (Hjelmas and Low, 2001). Most methods require training dataset (Yang *et al.*, 1999; Liu *et al.*, 2010; Nguyen *et al.*, 2011). So they are not effective in face detection image processing and difficulty to apply without sufficient training images. In this chapter, a new face colour based face detection method is proposed and developed. This method does not require any training data. It can detect human faces in the colour images with complex background.

The diagram of the new proposed method is shown in Figure 4.1. The main advantage of this method over the existing ones is that the method work without any training datasets. Firstly, a luminance normalisation step is introduced to adjust the luminance of the input image. Then, all skin colour regions are detected from the luminance-adjusted image based on the skin colour model developed by Phuong-Trinh *et al.* (2007). These regions are considered as face candidates. Finally, each face candidate is verified using the eye position detection method proposed in Section 4.4. The candidate is considered a face if eye pairs are detected. Otherwise, it is dropped.

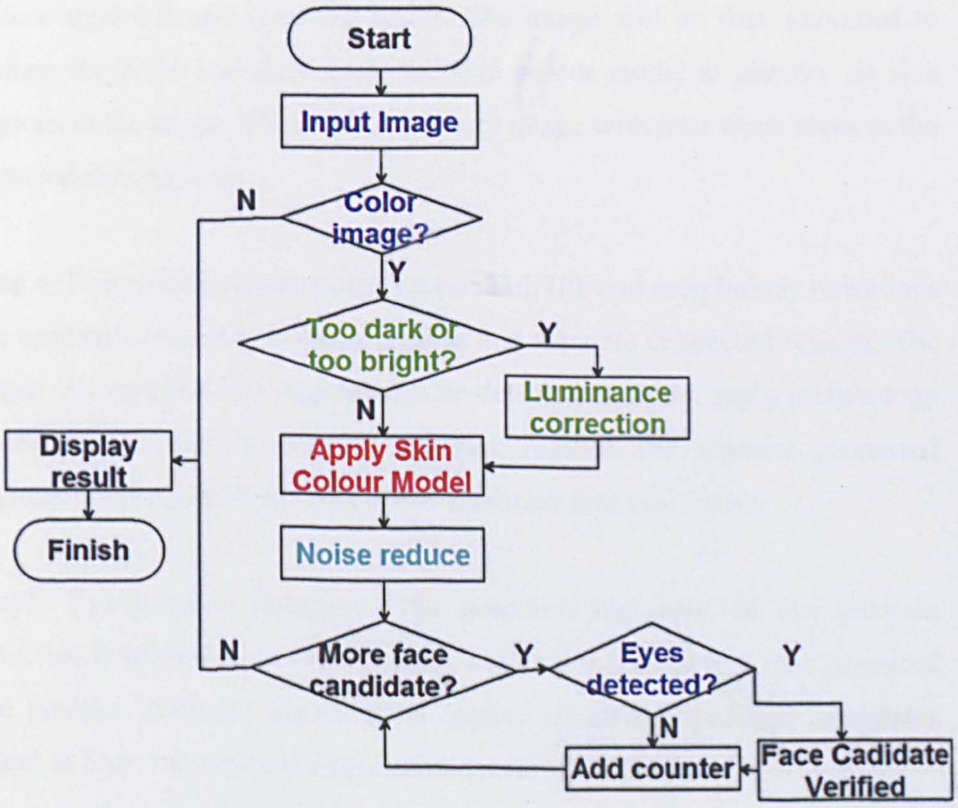


Figure 4.1 The proposed colour based face detection algorithm

As shown in Figure 4.1, the procedure of the proposed method is (each step is represented in a different colour):

- Step 1: Image input and to determine if the input image is a coloured one or not. A Matlab programme is used to input images as a matrix. The input image matrix is analysed. If the dimension of the matrix is two, it is a greyscale image; if the dimension is three, it is a colour image.

- Step 2: Luminance analysis. If it is too bright or too dark, the algorithm is applied to adjust the illumination of the image if necessary. The input image is in *RGB* colour space. This *RGB* space will be mapped into *YCbCr* space. The ratio of *Y* component to the average of *Cb* and *Cr* components will be analysed. If the ratio is within pre-determined range, the luminance of the image is considered to be normal; otherwise, the image luminance should be normalised by adjusting *Y* component. The result of this step is a normal colour image that is suitable for further processing.

- Step 3: Skin colour segmentation. Each pixel is checked based on the skin colour model to get binarised result. The image will be first smoothed to reduce the noise and then apply the skin colour model to identify all skin regions in the image. The result is a binary image with pure white areas as the detected skin regions.

- Step 4: Noise reduce. Edge detection method, IPF and morphology operations are applied to remove non-face regions and separate connected regions. The edges of identified skin regions will be detected and then apply morphology operations and IPF to remove non-face regions and separate connected regions. The results of this step are all qualified face candidates.

- Step5: Eye position detection. The proposed algorithm of eye position detection is applied to verify if a face candidate is true face. A new proposed eye position detection algorithm is applied to all the qualified candidates found in Step 4 to verify if a face candidate is a true face.

4.2 Skin Colour Segmentation

Colour is one of the distinctive properties of human skin. The advantage of detecting skin according to its colour is obvious because skin colour remains invariant when rotation and occlusion occur. Most existing face detection methods consider the luminance component of the image as a greyscale image. They try to segment skin region from the background by applying some values ranges of skin colour pixels in the greyscale image. However, these methods are sensitive to lighting conditions, which means the environment luminance has a significant effect on the pixel values of colour images. Moreover, the statistics of skin colours varies among different human races. So it is not reliable to segment skin region based on one component. More colour components should be taken into account for skin colour segmentation.

There are various skin colour models for face detection (Vezhnevets *et al.*, 2003). For colour based methods, the *RGB* and *YCbCr* are proved to be good spaces to build up the skin colour models (Zhang *et al.*, 2009; Wu *et al.*, 2011). In the method proposed in this thesis, the author proposes a new skin colour model that combined the *RGB* and *YCbCr* colour spaces. This skin colour model is more applicable than those models defined in one colour space.

The skin colour model used in this thesis is defined in the *RGB* and *YCbCr* spaces. The partial model defined in *RGB* space is based on the one developed by Phuong-Trinh *et al.* (2007). By comparing two other skin models (Lin and Fan, 2000; Kovac *et al.*, 2003), this method defines the boundaries of skin colours explicitly within the *RGB* space. This is efficient for avoiding the retaining of non-skin colours (e.g., yellow, white, orange, pink, red, wood-brown, and sand-yellow).

The skin colour model defined in *RGB* space is represented in Table 4.1.

Table 4.1 Skin colour model in *RGB* colour space

R	*(R-G)*	*(G-B)*	*B*	*G*	
[70,85]	[30,55]	[-5,35]	[20,255]	[30,255]	
[86,100]	[30,60]	[-5,40]	[30,255]	[40,255]	
[101,150]	*(R-G)*	*(G-B)*	*(R-B)*	*(R+B-2G)*	*G*
	[0,30]	[-10,45]	[15,75]	[-15,285]	-
	[31,75]	[-5,90]	[-255,120]	[-20,285]	[50,255]
[151,200]	*(R-G)*	*(G-B)*	*(R-B)*	*(R+B-2G)*	*B*
	[15,20]	[-5,40]	[20,255]	[-20,285]	-
	[31,85]	[-15,70]	[20,255]	[0,285]	[40,255]
[201,255]	*(R-G)*	*(G-B)*	*(R+B-2G)*		
	[5,25]	[40,70]	[-30,285]		
	[26,100]	[0,70]	[-15,285]		

Source: Robust Face Detection under Challenges of Rotation, Pose and Occlusion (Pham-Ngoc, 2007)

A luminance normalisation step is added before skin colour segmentation to make the method robust as the luminance will seriously affect the skin segmentation results. This step is used to correct luminance that is too big or too small. The correction rule is expressed by Equation 4.1:

$$Y(x,y) = \begin{cases} Y(x,y) & if\, R\left(\frac{Y}{CbCr}\right) \in [0.9,1] \\ \frac{Y(x,y)}{R\left(\frac{Y}{CbCr}\right)} & otherwise \end{cases} \tag{4.1}$$

Where: $R\left(\frac{Y}{CbCr}\right) = \frac{Y}{0.5(Cb+Cr)}$ is the luminance ratio. ***Y***, ***Cb*** and ***Cr*** are the three components in the *YCbCr* colour space. The correction range [0.9,1] is based on my experiments. The author has written Matlab scripts to process 50 images to decide the appropriate threshold range. According to my test, the image is considered to be too dark if the luminance is smaller than 0.9; it is considered to be too bright if the luminance is greater than 1.

YCbCr colour space is combined to build up the skin colour model. In the *YCbCr* space, the *Y* component is a luminance of the original image. It is essentially a greyscale copy of the original image; the *Cb* and *Cr* components are the chrominance of the image. The *RGB* colour space can be easily transformed to the *YCbCr* colour space (Berbar *et al.*, 2006) according to Equation 3.2

After luminance normalisation step, the skin colour model is applied in the luminance normalised image to segment skin regions. The decision rules of applying the skin model are defined by Equation 4.2:

$$\delta(P(x,y)) = \begin{cases} 1, & if\, S(P(x,y))\, is\, satisfied \\ 0, & otherwise \end{cases} \tag{4.2}$$

Where: (*x*, *y*) is the coordinates of pixels in the image, ***P***(*x*, *y*) is a pixel value of colour image, and condition *S*(***P***(*x*, *y*)) is listed in Table 4.1.

As shown in Table 4.1, the skin model adjusts reasonable differences between R, G, and B components. In addition, after examining the values of skin areas manually marked from more than 1,000,000 pixels, it was found that they should obey such a rule:

$$R > G > B \tag{4.3}$$

This means that the R component is always the strongest one. It may because of the special expression of blood under human skin. The skin region can be extracted from input images by applying Equation 4.2 and 4.3.

After skin segmentation and subsequently labelling the connected skin regions, the noise reducing step is carried out to erase the areas that are smaller than the specified threshold. To deal with images of different sizes, we use a self-adapting threshold according to the input image size. This step serves to improve the processing speed by reducing unreasonable face regions.

$$\delta(S) = \begin{cases} 1, & S \geq t \\ 0, & otherwise \end{cases} \tag{4.4}$$

Where: S is the area of detected regions in a binary image and t is the self-adapting threshold computed with the size of the image.

4.3 Geometric Restrictions

The previously introduced skin colour model is effective for detecting the skin region in the image. However, it is not easy to distinguish faces and other body parts, such as arms and hands. Therefore, some geometric restrictions will be applied to these regions to drop non-face candidates as soon as all skin regions are detected from the images.

Although Séguier (2004) tried to locate elliptic skin regions using Hough transformation and considering them as faces, his method was less efficient than dealing with the condition of skin regions connected with each other. The steps described earlier may also have such face candidates, especially as human faces in images are connected together or with other human parts (e.g., hands, arms) in some cases. Thus, the method used to separate connected skin regions in this thesis is the IPF discussed in Section 3.1.4.1. IPF is used to calculate horizontal and vertical skin pixel intensities. Specifically, based on these projection functions, those small values comparing the width (for horizontal projection) and height (for vertical projection) of the current skin region reflect concave regions, which are the intersections of different parts. These rows or columns are set to zero to separate those parts using Equation 4.5.

$$p(i) = \begin{cases} p(i), & p(i) \geq t \\ 0, & otherwise \end{cases} \tag{4.5}$$

Where: *p*(*i*) is projection value of the *i*-th bin of horizontal or vertical projection functions and *t* is selected coefficient (a self-adapting parameter related to the size of the skin area).

As soon as the connected regions are separated, each region is labelled and considered as an independent face candidate. A series of morphological operators discussed in Section 0 are applied to the binary image to fill the holes in the detected areas.

The author has investigated the properties of human faces. It has been found that the height-width ratios are never bigger than 2.5. Thus, all the non-face skin regions should be rejected based on several geometric conditions, as expressed in Equation 4.6.

$$\delta(S) = \begin{cases} 1, & \dfrac{max(H,W)}{min(H,W)} < 2.5 \\ 0, & otherwise \end{cases} \tag{4.6}$$

Where: *S* is the number of skin pixels belonging to the skin region rectangle, and *H* and *W* are the height and width of the skin region rectangle, respectively.

Several experiments have been implemented in Matlab to test the effectiveness of the method proposed. It has been found that all face areas can be identified. However, some body parts with a similar geometrical shape of the face cannot be removed. From the test results, the author has found that the human hand is a particularly difficult candidate to eliminate if only applying the method. The next section will propose and develop a new method for eliminating the false face candidates using a new proposed eye position detection algorithm.

4.4 Eye Position Detection Algorithm

4.4.1 The Algorithm

Till now, the most potential face candidates have been identified from the original image. To further improve the results, all face candidates will be examined by looking for eye pairs within the areas. If there are eyes found in the area, it will be considered as face region. Otherwise, it will be erased from the previous result. In this section, a new eye position detection method is proposed. The flowchart of the eye position detection algorithm used is illustrated in Figure 4.2.

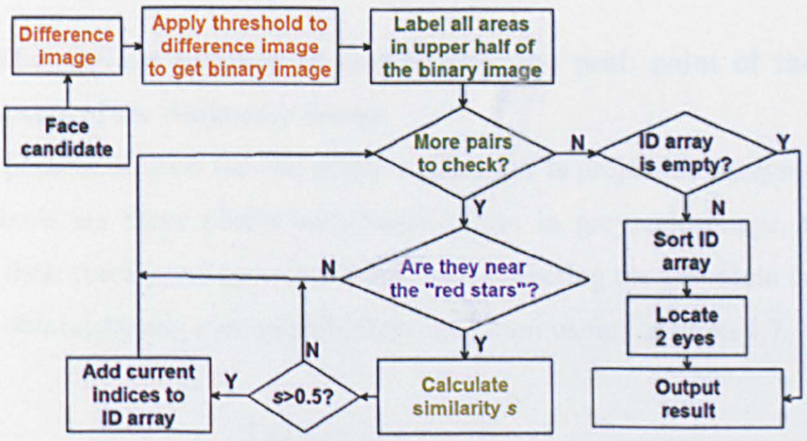

Difference image
Apply threshold to difference image to get binary image
Label all areas in upper half of the binary image
Face candidate
More pairs to check?
N
ID array is empty?
Y
Y
N
Sort ID array
N
Are they near the "red stars"?
Locate 2 eyes
N
Y
Add current indices to ID array
Y
s>0.5?
Calculate similarity s
Output result

Figure 4.2 Eye position detection algorithm

The procedure of the proposed algorithm is:

- Step 1: The difference image binarisation. Apply binarisation method again to the difference images of face candidates to determine those regions with the difference values bigger than the selected threshold. These regions are possibly eye regions.

- Step 2: Region labelling. Label all binarised regions to mark the upper half that possibly contain eyes and check the numbers of the regions. If there are no regions left, algorithm return; otherwise, go to Step 3.

- Step 3: Horizontal projection. Carry out IPF in the horizontal direction and mark the possible eyes position. Only the regions near this position will involve in the calculations in Step 4.

- Step 4: Correlation computation. To calculate the correlation coefficients between the all numbered regions to find out all pairs of eyes.

4.4.2 Eye Position Detection Rules

Based on the anatomical analysis of human face features (Larrabee *et al.*, 2004), it has been found that a pair of eyes on a human face is very unique and with distinguished geometrical characteristics. So if a face candidate image contains a pair of eye, it is very highly reliable to treat this candidate as a face. Several rules are used to match all eye pairs for face candidate verification.

Rule 1: The vertical location of eyes is near the peak point of the vertical projection sum of the difference image.

The most popular method for eye position detection is projection analysis. Because the eye pixels are those pixels with lower values in greyscale image, the author computes their corresponding valley image by subtracting the greyscale image from the image obtained using a morphable close operation using Equation 4.7.

$$\boldsymbol{V}(x,y) = \boldsymbol{F}(x,y) - \boldsymbol{I}(x,y) \tag{4.7}$$

Where: $\boldsymbol{F}(x,y)$ is the image processed by a disk close operator with radius 5 and $\boldsymbol{I}(x,y)$ is the greyscale image.

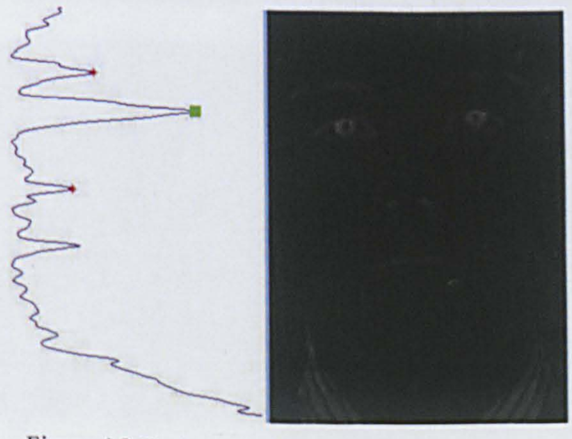


Figure 4.3 Horizontal projection of valley image

The eye area is always the highest peak or with a greater value in the horizontal projection function. Thus, the potential position of the eyes can be localised by computing the horizontal projection of the valley image (the "green stars" in Figure 4.3).

Rule 2: The eyes appear in the upper half in a face and the ratios between the areas of eye blocks and the face candidate are within the range of [0.01,0.1].

In any faces that are not rotated significantly, it is obvious that the first half of the rule should be obeyed. To determine the threshold of the ratios, the author has written and run Matlab programme on 50 test images. By applying the chosen threshold to the difference image, all potential eye areas remain as the white blocks in the binary image (Figure 4.4(d)). To make this method more robust, the binary image is processed using a morphological close operation with disk-shaped structure element with radius 3 (Figure 4.4(e)). Blocks that are with small areas or in the lower half of the binary image are then eliminated. Finally, the remaining blocks are considered to be eye pairs (Figure 4.4(f)).

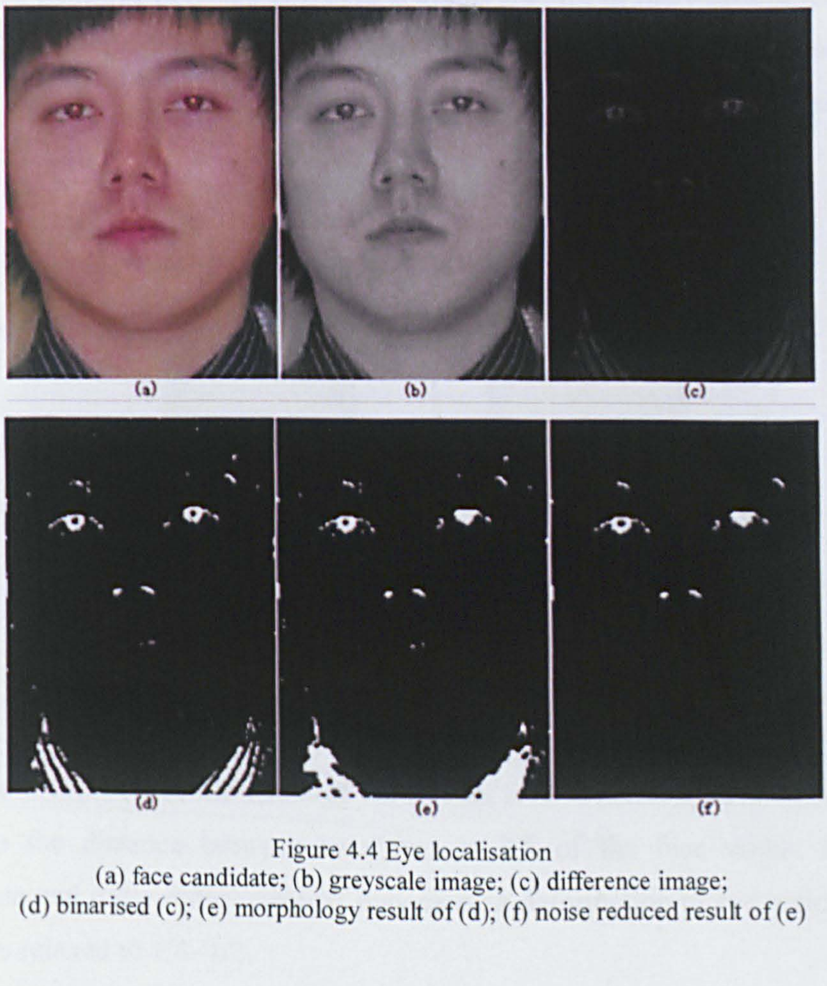


Figure 4.4 Eye localisation
(a) face candidate; (b) greyscale image; (c) difference image;
(d) binarised (c); (e) morphology result of (d); (f) noise reduced result of (e)

Rule 3: The eye blocks are an approximately round shape.

The height-width ratios of all remaining blocks will be computed. Those blocks whose height-width ratios are greater than a given threshold will be eliminated using Equation 4.8. Furthermore, the possible eye pairs should have similar area.

$$\phi(S) = \begin{cases} 1, & e(S) \leq t \\ 0, & otherwise \end{cases} \tag{4.8}$$

Where: S is the eye candidate, $e(S)$ is the eccentricity of region S, and t is a chosen threshold of e. The author has written Matlab script to test 50 images to determine the threshold t.

After applying Rules 1~3, the algorithm will return false if there is no block left. Thus, the current face candidate will be considered as a "false face". Otherwise, the algorithm continues by checking all possible eye pairs to compare their similarity.

Rule 4: The distance between two eyes is about 1/4~1/2 of the width of face.

This rule is used for eliminating impossible eye pairs, which means only blocks with a certain distance will be checked. This rule could reduce the processing time significantly.

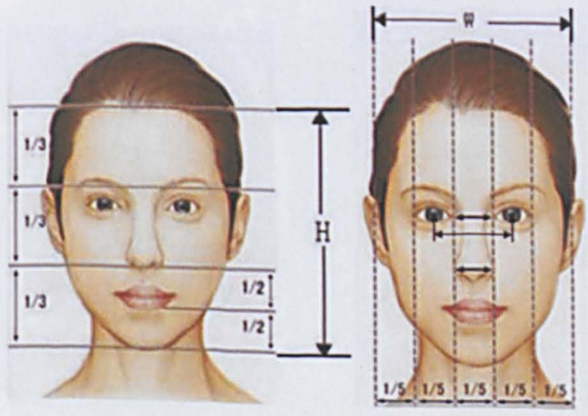


Figure 4.5 Facial ratio
Source: http://caizhuang.abang.com/od/mingcijieshi/a/santingwuyan_p1.htm

As shown in Figure 4.5, the width of the human face is perfectly five times the eye width. So the distance between two eyes is 2/5 of the face width. However, binarisation and noise reduce process may cause a deformation of eye regions. Thus, the ratio is relaxed to 1/4~1/2.

Rule 5: The two eyes are similar in shape and pixel values.

Next, each pair of the remaining blocks is treated as possible eye pairs. According to their centroids, two circle areas with a radius of 20 in the greyscale image of the current face candidate are extracted. The radius 20 is chosen based on the observation on the results obtained by running Matlab scripts on 50 images. Their similarity is then calculated using Equation 4.9. If the similarity is greater than 0.5, their indices are saved in an array. After the loop, the pair with the highest similarity is considered as "real eyes" if the indices' array is non-empty. The current face candidate is then marked as a "real face". Otherwise, it is a "false face". Actually, all similarities can be saved for comparison. The threshold of 0.5 here is introduced for speeding up the algorithm.

$$r = \frac{\Sigma_{i=1}^{m}\Sigma_{j=1}^{n}(A_{i,j}-\overline{A})(B_{i,j}-\overline{B})}{\sqrt{\left(\Sigma_{i=1}^{m}\Sigma_{j=1}^{n}(A_{i,j}-\overline{A})\right)^2\left(\Sigma_{i=1}^{m}\Sigma_{j=1}^{n}(B_{i,j}-\overline{B})\right)^2}} \tag{4.9}$$

Where: $\boldsymbol{A}$ and $\boldsymbol{B}$ are the two areas for comparison and $\overline{\boldsymbol{A}}$ and $\overline{\boldsymbol{B}}$ are the mean of their pixels' values.

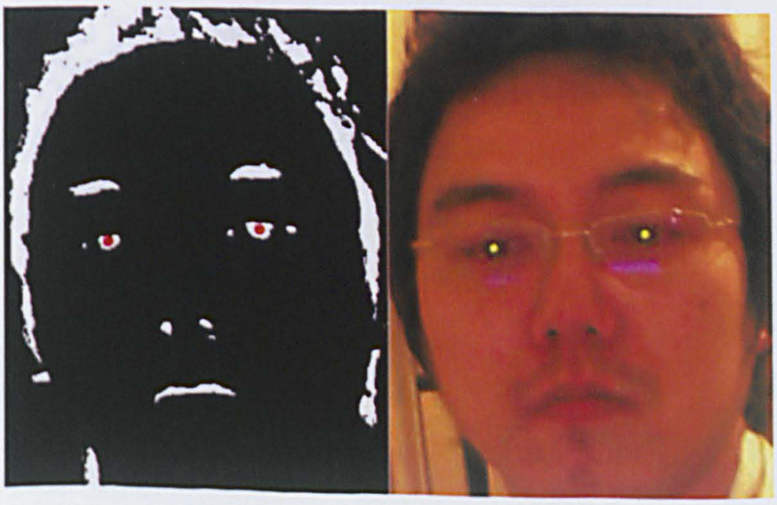


Figure 4.6 Applying rules to locate eyes

Figure 4.6 shows that the proposed eye position detection algorithm is effective. Although there are several pairs of possible eye blocks remain in the noised reduced binarised image, the eye pair has been successfully detected based on the geometrical rules defined.

4.4.3 Determination of Face Region

The face region is a process to determine where the identified faces are in the image and then to draw a solid line box to show the face boundaries. As described in Section 4.4.1, in the algorithm for eye position detection, the distances between the identified pair of eyes are calculated, this distances can be used to crop the face region from the original image. If a face is vertical, it is quite natural to draw surround box. However, there is no guarantee that the faces are vertical. Some of them may be in the tilted poses. The edges of these facial regions are titled as well. So it is necessary to decide the tilted angle of a face region.

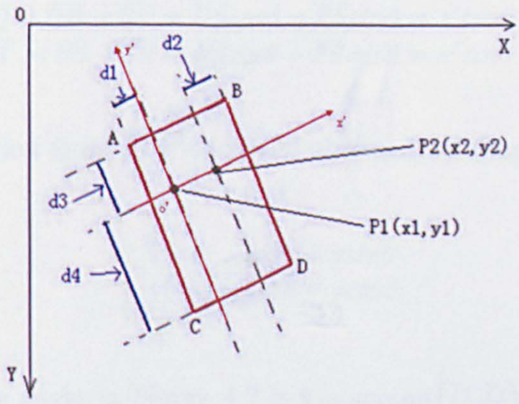


Figure 4.7 Face region in original image

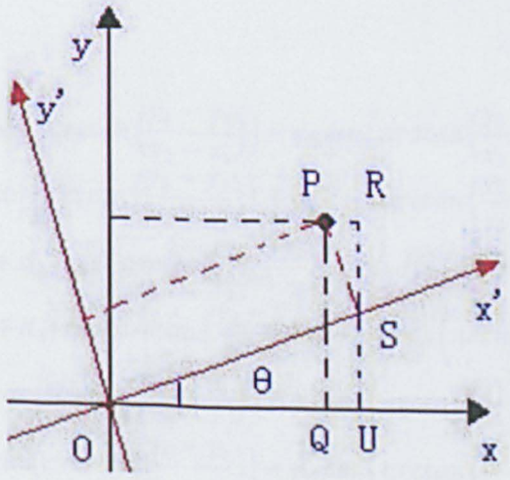


Figure 4.8 Coordinates translation

As demonstrated in Figure 4.7, if two eyes are located at points $P_1(x_1,y_1)$ and $P_2(x_2,y_2)$, the face region is determined by points A, B, C, and D, which can be calculated based on the position of eyes (P_1 and P_2) and four distances d_1, d_2, d_3, and d_4. If the coordinate system is established at P_1, the x' axis is in the direction of $\overrightarrow{P_1P_2}$, while the y' axis is to rotate the x axis about origin $\boldsymbol{P}_1$ anti-clockwise for 90°. In the new coordinate system $x'o'y'$, the face region can be represented by rectangle ***ABCD***, where $\boldsymbol{A}(-d_1, d_3), \boldsymbol{B}(\overrightarrow{\boldsymbol{P}_1\boldsymbol{P}_2} + d_2, d_3), \boldsymbol{C}(-d_1, -d_4)$ and $\boldsymbol{D}(\overrightarrow{\boldsymbol{P}_1\boldsymbol{P}_2} + d_2, -d_4)$.

To locate the face region in the original image, the coordinates need to be translated from $x'o'y'$ to xoy. As shown in Figure 4.8, if the angle between the axis of x' and x is θ, the coordinates of point $\boldsymbol{P}$ is (x,y) in xoy and (x',y') in $(x'oy')$. According to the geometric relationship, the translation between the two coordinate systems is shown in Equation 4.10:

$$\begin{aligned} x &= \overline{\boldsymbol{OQ}} = \overline{\boldsymbol{OU}} - \overline{\boldsymbol{QU}} = \overline{\boldsymbol{OS}}cos\theta - \overline{\boldsymbol{PS}}sin\theta = x'cos\theta - y'sin\theta \\ y &= \overline{\boldsymbol{UR}} = \overline{\boldsymbol{US}} + \overline{\boldsymbol{SR}} = \overline{\boldsymbol{OS}}sin\theta + \overline{\boldsymbol{PS}}cos\theta = x'sin\theta + y'cos\theta \end{aligned} \tag{4.10}$$

The reverse translation from $x'oy'$ to xoy is obtained by simply replacing θ with $-\theta$ Equation 4.11.

$$\begin{aligned} x' &= xcos\theta + ysin\theta \\ y' &= -xsin\theta + ycos\theta \end{aligned} \tag{4.11}$$

Because the rotation angle in Figure 4.7 is $\theta = arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)$, the coordinates of the corners of the face region in the original image (***A***, ***B***, ***C*** and ***D***) are computed using Equation 4.12:

$$\begin{aligned}
x_A &= -d_1 cos\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) - d_3 sin\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) \\
y_A &= -d_1 sin\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) + d_3 cos\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) \\
x_B &= (d_{12}+d_2) cos\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) - d_3 sin\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) \\
y_B &= (d_{12}+d_2) sin\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) + d_3 cos\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) \\
x_C &= -d_1 cos\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) + d_4 sin\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) \\
y_C &= -d_1 sin\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) - d_4 cos\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) \\
x_D &= (d_{12}+d_2) cos\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) + d_4 sin\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) \\
y_D &= (d_{12}+d_2) sin\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right) - d_4 cos\left(arctan\left(\frac{y_2-y_1}{x_2-x_1}\right)\right)
\end{aligned} \tag{4.12}$$

Where: $d_{12} = |\overrightarrow{\boldsymbol{P}_1\boldsymbol{P}_2}|$ is the distance between $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$.

4.5 Experimental Results and Evaluation

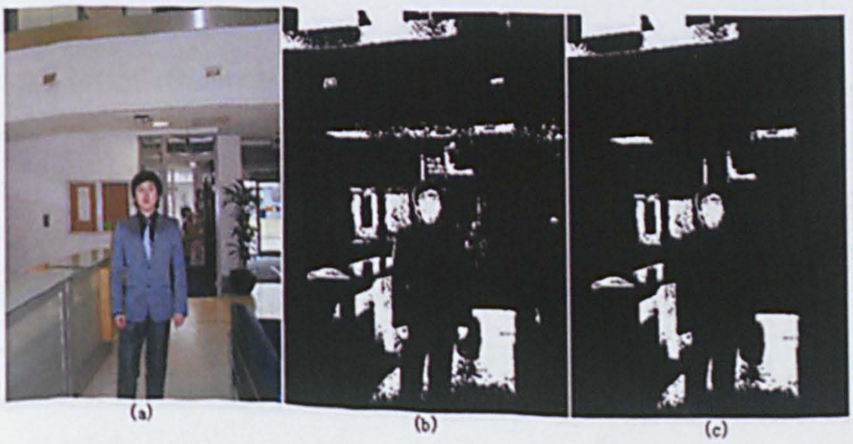
4.5.1 Experimental Results

The proposed colour based face detection method is implemented by Matlab R2009b. The implementation is 1,174 lines of Matlab code which is available upon request. It has been tested on 40 different images taken in various conditions. All images are with complex backgrounds and different lighting conditions. Figure 4.9 shows a complete face detection example and the intermediate steps during the algorithm running.

The testing images include 65 faces. As shown in Table 4.2, the method detects 56 faces successfully, while 9 faces are missed and 4 false faces are detected. Figure 4.10 shows the execution time and the average detection time. It has been found that the average speed of identifying one face is about 0.1 second for an image with multiple faces and about 0.4 second for the image with a single face. The experiment results show that the proposed method is effective to detect multiple faces in the crowded images comparing with the existing methods. It is believed that this speed is fast enough for most face detection image processing applications such as on-site monitoring and game play applications.

Table 4.2 Statistics of face detection algorithm

	Images	Faces	Detected	Missed	False
Number	40	65	56	9	4
Ratio			86.15%	13.85%	6.15%



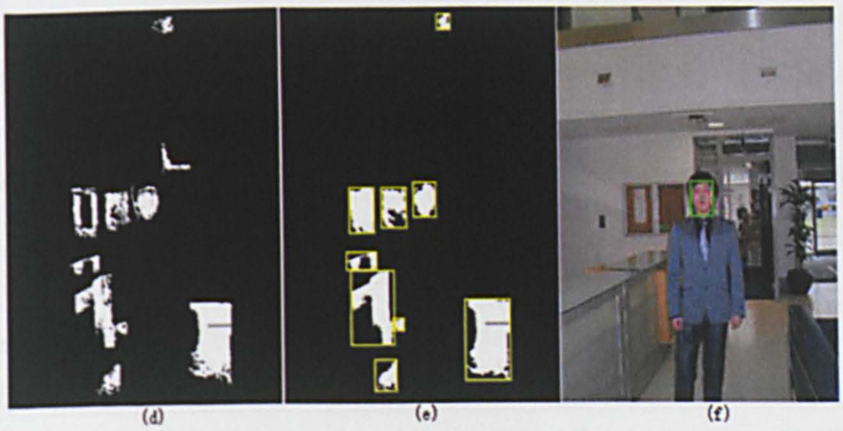


Figure 4.9 A complete face detection example
(a) Original colour image; (b) Skin colour segmentation; (c) Noise reduced result;
(d) Apply geometric restrictions; (e) Face candidates; (f) Detected face.

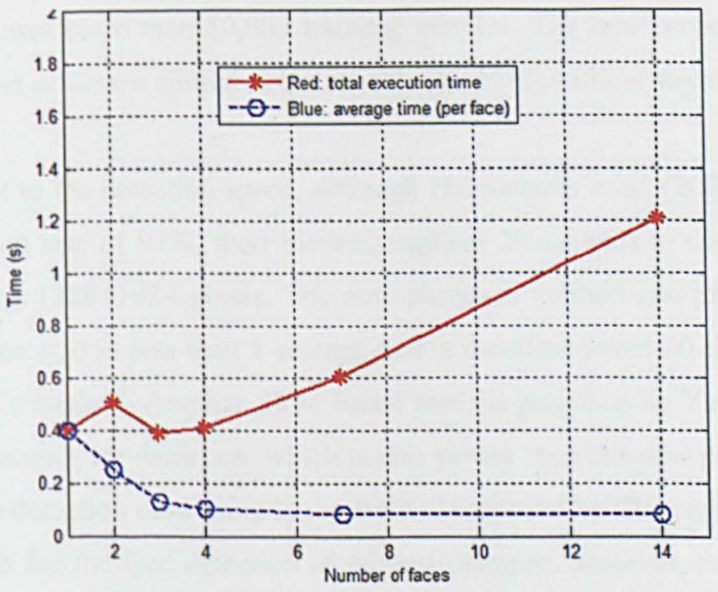


Figure 4.10 Execution time of face detection algorithm

4.5.2 Evaluation

The most important advantage of the new proposed method is that it does not need any training samples. In contrast, the training samples required for existing methods vary from a few thousands to tens of thousands. For example, the method proposed by Sung *et al.* (1998) requires 47,316 negative and 4,150 positive training samples. While Huang *el al.* (2003) requires 56,007 negative and 2,990 positive samples, Yang *et al.* (2008) requires 5,000 negative and 1,400 positive samples. To build such a huge library, it is necessary to obtain permission from the participants because of the issue of privacy. As soon as permission is obtained, time is still required to collect thousands of human face pictures as positive samples. It is a complicated and time consuming process to obtain all the necessary permissions and to take all of the pictures. There are no reports of how long it took to collect all the legal pictures in the studies described above; however a conservative estimate is that it would take a few months to collect a few thousand pictures in a legal way. The proposed method in this thesis does not need training samples.

For the detection rate, Huang *et al.* (2003) tested their method and achieved detection rates from 84.6% to 86%. As mentioned above, the detection rate of this new proposed method is 86.15%, so it can be said that the method here performs better than Huang's best result of 86%. The detection rate of Sung's (1998) method is in the range of 79.9%~96.3%, with an average detection rate of 88.1%. But Sung's method requires more than 50,000 training samples. The new proposed method in this thesis can achieve a similar detection rate (86.15%) without any training samples.

With respect to the detection speed, although Phimoltares *et al.* (2007) achieved the high detection rate of 97%, their method requires 20 seconds to detect faces in an image of size 1280×1024 pixels. The new proposed method can process an image with the same size in less than 1 second, and is therefore about 20 times faster than Phimoltares's method. Another SVM based method proposed by Yang *et al.* (2008) requires 3 seconds for detection, which is also slower than this new proposed method. The average detection time using in the method proposed by Chen and Lien (2009) is 0.81 seconds for the face detection of the test pictures. However, their results were achieved by training the system using 5,920 samples. The new method proposed in this thesis takes 0.4~1.2 seconds for the face detection using the same pictures without training samples.

As discussed above, the detection rate of the new proposed method is comparable to other previous methods. By comparing to Sung's (1998) average detection rate of 88.1% achieved based on more than 50,000 training samples, the new proposed method can achieve a detection rate of 86.15% without any training samples. It also performs better than Huang's (2003) best result of 86%. The detection speed (0.4~1.2 seconds, 0.8 on average) is faster than Phimolotares's (2007) method (20 seconds) and Yang's (2008) method (3 seconds). The speed of new proposed method is similar to Chen's (2009) training sample based method (0.81 seconds). But the new method achieved these 0.4~1.2 seconds (0.8 seconds on average) without any training samples.

4.6 Summary

A new skin colour based face detection method has been proposed and implemented in this chapter. The method includes two steps: skin region segmentation based on a new skin colour model combining the *RGB* and *YCbCr* colour spaces, and an eye position detection algorithm based on a group of geometrical rules. The new proposed method is comparable to other previous face detection methods. The most important advantage of this method is that it does not need any training dataset. The method has been tested over 40 images with 65 human faces. By comparing to other previous methods, the new method has advantages in both detection rate and detection speed. It achieved a detection rate of 86.15% and detection speed of 0.4~1.2 seconds. This new method can be applied to emotional bio-robot for detecting human faces.

CHAPTER 5

A New Method for Generic 3D Human Face Modelling

This chapter presents the proposition and development of a new method for build a generic 3D human face model and how this model can be used to build any personalised 3D face model of any persons or any imagery human faces. Section 5.1 firstly describes the new proposed and implemented method for building a generic 3D human face model. Section 5.2 discusses the face characteristics and divides the whole face into 21 geometrical surfaces for the purpose of graphically expressing the whole face with feature based parameterised geometrical models while Section 5.3 defines data structures and their classes for all these features. Based on the work in the previous sections, Section 5.4 proposes and develops a new method for reverse computation of facial features and builds a generic 3D face model. Finally, Section 5.5 discusses the applications of the proposed and developed method and compared it with the other methods.

5.1 Proposed Method for Generic 3D Face Modelling

As discussed in Chapter 1 and Chapter 2, it is necessary to propose and develop a method that can be used to build a generic 3D human face model for emotional bio-robots. This model can be used generate a 3D model of any human faces or any imaginary human faces. The functional requirements of a generic 3D model should be:

- It should be defined by a group of geometrical parameters. The users can build the any 3D face model they like by modifying these parameters.
- The facial features can be manipulated independently to enable users generate various face models with different appearances.

So here, the key issues of the new method are:

1) To divide the whole face into a number of pre-defined geometrical features that can be modified to simulate the geometrical characteristics of individual appearances.
2) To capture the required geometrical information that can be used to define NURBS curves model. The curves model is the boundaries of all pre-defined geometrical features of a human face.
3) To define the necessary data structure and classes to hold all information of the NURBS curves and surfaces.

The procedure of the new proposed method is shown in Figure 5.1.

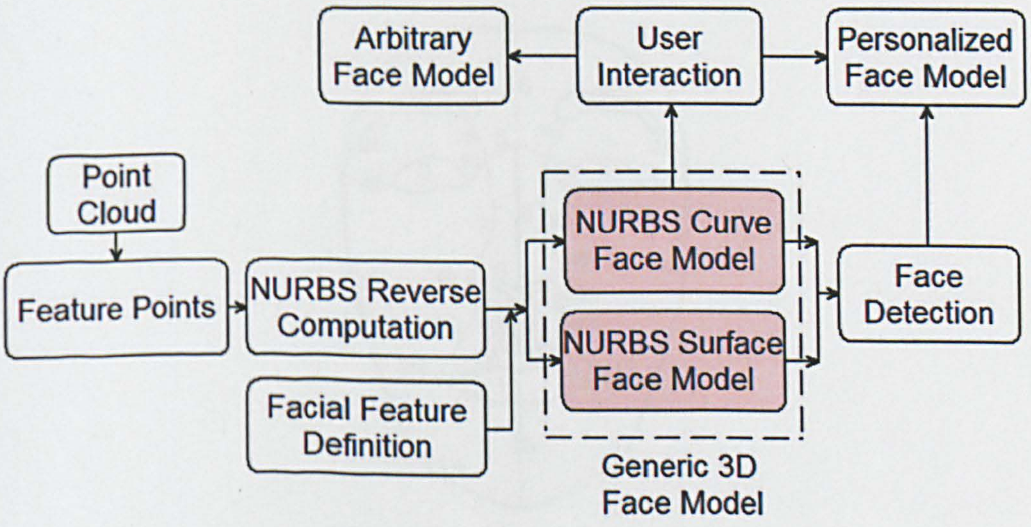


Figure 5.1 Proposed generic 3D face modelling method

In this method, the core part is the generic 3D face model including NURBS curve face model and NURBS surface face model. The features of curve model are the boundaries of the feature of surface model. These curve models are determined by NURBS reverse computation by processing a 3D point cloud of a face to find all the necessary geometrical information of both control and knot points. Through user interaction, the generic 3D face model can used to build either an arbitrary face model that does not look like any specific person or the personalised face model according to a person's photo by face detection.

5.2 Face Characteristics and its Geometrical Features

To build up a generic 3D face model, a human face should be divided into a number of geometrical surfaces for the purpose of graphically expressing the whole face with feature based parameterised geometrical models Based on the features analysis (such as nose, mouth, eyes and chin etc.) on a human face (Rose, 1998; Larrabee *et al.*, 2004) and the convenience and effectiveness of manipulating the generic 3D model, the author defined 34 curve features on a human face. These curve features can be combined together to form 21 surface features.

The curve and surface features are shown in Figure 5.2 and Figure 5.3, respectively. These features should be labelled for referencing them by the face model object.

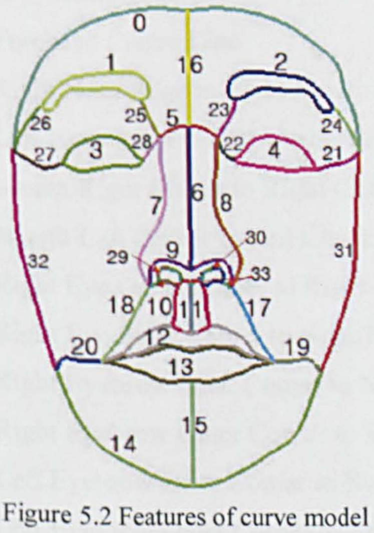


Figure 5.2 Features of curve model

The labeled features are distinguished by their unique indices. Their correspondence between the index and description is shown in Table 5.1. Please note that the right and left sides are opposite to the real directions for both enumeration definitions, just as if one sees oneself in the mirror.

Table 5.1 Labels of curve features

ID	Feature Description
0	Forehead
1	Left Eyebrow
2	Right Eyebrow

ID	Feature Description
3	Left Eye
4	Right Eye
5	Nose Top Edge
6	Nose Centre Line
7	Nose Left Edge
8	Nose Right Edge
9	Nose Bottom Edge (not the real bottom)
10	Philtrum Boundary
11	Philtrum Centre Line
12	Upper Lip
13	Lower Lip
14	Chin Profile
15	Jaw Centre Line
16	Forehead Centre Line
17	Right Nose Edge to Right Mouth Corner
18	Left Nose Edge to Left Mouth Corner
19	Mouth Right Corner to Right Cheek Edge
20	Mouth Left Corner to Left Cheek Edge
21	Right Eye Outer Corner to Right Temple
22	Right Eye Inner Corner to Nose Right Edge
23	Right Eyebrow Inner Corner to Nose Right Edge
24	Right Eyebrow Outer Corner to Right Temple
25	Left Eyebrow Inner Corner to Nose Left Edge
26	Left Eyebrow Outer Corner to Left Temple
27	Left Eye Outer Corner to Left Temple
28	Left Eye Inner Corner to Nose Left Edge
29	Left Nostril
30	Right Nostril
31	Right Cheek
32	Left Cheek
33	Nose Bottom Line (adjacent to Philtrum)

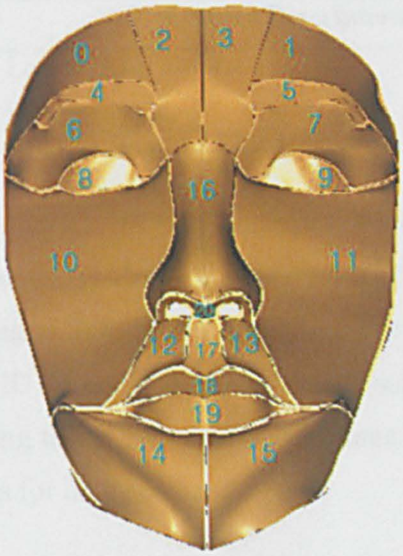


Figure 5.3 Features of surface model

The meanings of surface indices are listed in Table 5.2.

Table 5.2 Labels of surface features

ID	Feature Description
0	Area above Left Eyebrow
1	Area above Right Eyebrow
2	Left Forehead
3	Right Forehead
4	Left Eyebrow
5	Right Eyebrow
6	Area between Left Eyebrow and Left Eye
7	Area between Right Eyebrow and Right Eye
8	Left Eye
9	Right Eye
10	Left Cheek
11	Right Cheek
12	Area between Nose and Mouth on the Left
13	Area between Nose and Mouth on the Right
14	Left Lower Jaw
15	Right Lower Jaw
16	Nose
17	Philtrum
18	Upper Lip

ID	Feature Description
19	Lower Lip
20	Nostril

The 3D surface model has 21 NURBS surface features. The boundaries of these surfaces are NURBS curves, as shown in Figure 5.2. The number of these surfaces can be adjusted according to demands of flexibility. For example, the flat region like the chin (surfaces with ID 14 and 15) and forehead (surface with ID 0, 1, 2, and 3) can be unified by merging the adjacent surface features. The eyes region can also be divided into two features for high accuracy.

Through segmentation of the human face, all features are defined. They can be referenced by the face model according to their index. For computation process, special data structures need to be defined to represent these features. The next section will discuss these data structures.

5.3 Data Structures and Classes

To represent the features of the 3D face model, in this thesis, the author designed the special data structures and corresponding classes to describe the curves and surface features and models. For the details of the file structures of the generic 3D model, please refer to Appendix A. Data File Structure Description.

5.3.1 Enumerate Structures

To reference a curve in the curve model or a surface in the surface model, a unique index is necessary. The numbers of these curves and surface have been determined to 34 and 21, respectively. So the possible values of the indices should be within certain range of either [0,33] or [0,20]. The enumeration (its keyword is *enum*) data type in C++ is ideal for holding this type of information (Norman, 2009) because of the fact that: (1) it is pre-defined, so it cannot be assigned a value during the programming running. Any attempts of assigning operation will be checked by debugger automatically. This can minimise the coding errors; (2) the value of *enum* variable can be exchanged with *int* type, which means it is convenient to use it as an array index. The index can be used to retrieve any features of the face model.

In this thesis, two types of enumerations are defined for tracking the feature IDs of the NURBS curve face model and the NURBS surface face model: the *CurveFeatureID* and *SurfaceFeatureID*. An extra value of -1 is added to each enumeration listed in Table 5.1 and Table 5.2 to represent invalid feature index.

(1) Features of NURBS Curve Face Model

The *CurveFeatureID* enumeration is defined to identify different NURBS curve features of the NURBS curve face model. The names of those curve features are easy to understand: each element in the enumeration starts with "*cf*", which means *C*urve *F*eature. The string composed by the remaining characters represents the description of the curve feature. Table 5.3 shows meanings of the elements of the *CurveFeatureID* enumeration.

Table 5.3 Description of *CurveFeatureID*

Feature Definition	ID	Feature Description
cfNull	-1	NULL, invalid for any curve feature ID
cfForehead	0	Forehead
cfLeftEyebrow	1	Left Eyebrow
cfRightEyebrow	2	Right Eyebrow
cfLeftEye	3	Left Eye
cfRightEye	4	Right Eye
cfNoseTop	5	Nose Top Edge
cfNoseCentre	6	Nose Centre Line
cfNoseLeft	7	Nose Left Edge
cfNoseRight	8	Nose Right Edge
cfNoseBottom	9	Nose Bottom Edge (not the real bottom)
cfPhiltrum	10	Philtrum Boundary
cfPhiltrumLine	11	Philtrum Centre Line
cfUpperLip	12	Upper Lip
cfLowerLip	13	Lower Lip
cfChin	14	Chin Profile
cfJawLine	15	Jaw Centre Line
cfForeheadLine	16	Forehead Centre Line
cfRightNoseMouthCorner	17	Right Nose Edge to Right Mouth Corner
cfLeftMouthNoseCorner	18	Left Nose Edge to Left Mouth Corner

Feature Definition	ID	Feature Description
cfMouthRight	19	Mouth Right Corner to Right Cheek Edge
cfMouthLeft	20	Mouth Left Corner to Left Cheek Edge
cfRightEyeOuter	21	Right Eye Outer Corner to Right Temple
cfRightEyeInner	22	Right Eye Inner Corner to Nose Right Edge
cfRightEyebrowInner	23	Right Eyebrow Inner Corner to Nose Right Edge
cfRightEyebrowOuter	24	Right Eyebrow Outer Corner to Right Temple
cfLeftEyebrowInner	25	Left Eyebrow Inner Corner to Nose Left Edge
cfLeftEyebrowOuter	26	Left Eyebrow Outer Corner to Left Temple
cfLeftEyeOuter	27	Left Eye Outer Corner to Left Temple
cfLeftEyeInner	28	Left Eye Inner Corner to Nose Left Edge
cfLeftNostril	29	Left Nostril
cfRightNostril	30	Right Nostril
cfRightCheek	31	Right Cheek
cfLeftCheek	32	Left Cheek
cfNoseBottomLine	33	Nose Bottom Line (adjacent to Philtrum)

(2) Features of NURBS Surface Face Model

The *SurfaceFeatureID* enumeration is defined to identify different NURBS surface features of the NURBS surface face model. Like the *CurveFeatureID*, the name of each element of those surface features starts with "*sf*", which means *S*urface *F*eature. The string composed by the remaining characters corresponds to the description of the surface feature. Table 5.4 shows the meanings of the elements of the *SurfaceFeatureID* enumeration.

Table 5.4 Surface Feature Enumeration

Feature Definition	ID	Feature Description
sfNull	-1	NULL, invalid for any surface feature ID
sfLeftUpperEyebrow	0	Area above Left Eyebrow
sfRightUpperEyebrow	1	Area above Right Eyebrow
sfLeftForehead	2	Left Forehead
sfRightForehead	3	Right Forehead
sfLeftEyebrow	4	Left Eyebrow
sfRightEyebrow	5	Right Eyebrow
sfLeftLowerEyebrow	6	Area between Left Eyebrow and Left Eye

Feature Definition	ID	Feature Description
sfRightLowerEyebrow	7	Area between Right Eyebrow and Right Eye
sfLeftEye	8	Left Eye
sfRightEye	9	Right Eye
sfLeftCheek	10	Left Cheek
sfRightCheek	11	Right Cheek
sfLeftUpperJaw	12	Area between Nose and Mouth on the Left
sfRightUpperJaw	13	Area between Nose and Mouth on the Right
sfLeftLowerJaw	14	Left Lower Jaw
sfRightLowerJaw	15	Right Lower Jaw
sfNose	16	Nose
sfPhiltrum	17	Philtrum
sfUpperLip	18	Upper Lip
sfLowerLip	19	Lower Lip
sfNostril	20	Nostril

Besides data structures defined here, several classes representing the curve model and surface model are also need to be defined for reverse computation of NURBS.

5.3.2 Classes Design

To represent the face model in 3D, the basic classes needed are 3D Point, NURBS curve and NURBS surface. Based on the NURBS curve and surface classes, two higher level classes of NURBS curve face model and NURBS surface face model are defined. As discussed in Section 3.2.1.3, in practice the degrees of NURBS curves and NURBS surfaces are usually chosen as three (Liu, 2008; Sun, 2007). In this thesis, the degrees of NURBS curves and surfaces are set to three initially. There are 5 classes are defined: (1) 3D points; (2) NURBS curves; (3) NURBS surfaces; (4) NURBS curve face model; (5) NURBS surface face model.

(1) 3D Points

The points on 3D face model is determined by x, y, and z coordinates: $\boldsymbol{P}=(x,y,z)^T$. it can be used to hold all 3D points including the control points and the points on the curve and surface.

(2) NURBS Curves

To represent the NURBS curve features, a class of *NurbsCurveD3* based on the 3D point class is defined. *D3* in the class name indicates that the degree is three. From my own simulation tests, it has been noticed that number of the data points on a curve is never bigger than 20 during the reverse computation although this number can be assigned to a bigger value for much more accurate description of a face. In my software programming, the maximum number of control points was set to 100. Hence, according to the definition of cubic NURBS, the maximum number of elements in knot vector is 104. To store the points on computed curve, an array of point class is defined. Its length is chosen as the maximum integer of 16-bit machines that is 65,536, which is big enough to avoid overflow for any curves used in this thesis.

(3) NURBS surface

The NURBS surface features are abstracted as the class *NurbsSurfaceD3*. *D3* in the name indicates that the degrees of the NURBS surface in both directions are three. The control point matrix is a 3D array. The first and second dimensions are the numbers of control points along x and y directions of a surface. The third dimension represents the control points. The information stored along with the third dimension are x,y,z,w that are x,y,z directional position coordinates and the weight associated with the control point.

As soon as the classes of points, curves, and surfaces are defined, the curve face model and surface face model can be defined based on the three basic classes.

(4) Curve Face Model

The NURBS curve face model is defined by class *NurbsCurveFace*. Its features are represented by *NurbsCurveD3* class. The features of the curve face model are stored in an array of *NurbsCurveD3* objects. An extra array of strings are defined to save the description of each curve features.

(5) Surface Face Model

The NURBS surface face model is represented by *NurbsSurfaceFace* class, whose features are denoted by the *NurbsSurfaceD3* class defined above. A parameter (*numSurfaceFeatures*) is used in *NurbsSurfaceD3* class to represent the value of the number of surface features. There is an array of strings for storing the descriptions of all these features.

In this section, all data structures and classes used in this thesis are defined. The parameters of NURBS curves and surfaces can be computed by reverse computation, which will be discussed in next section.

5.4 Reverse Computation of Facial Features

To build up the geometric face model using NURBS features, the parameters of the NURBS curves and surfaces need to be computed. Because the degree is usually chosen as three and all weights coefficients are set to one initially, the reverse computation is the process of calculating the knot vector and control points of NURBS from known geometrical data points. All facial features need to be manipulated to change the appearances of the model and the adjacent features are connected together, so the NURBS curves and surfaces used to represent the generic model should be clamped (see Section 3.2.1.3 for definition of clamped curves and surfaces).

As shown in Figure 5.1, there are two types of methods of getting geometrical data points. The one is to extract the feature points by face detection. The other is to get the points from scanned point cloud. In this thesis, the face identification image processing method has not been developed to automatically capture the geometrical information on the face feature boundaries and it is not suitable to use 2D images alone to recover the 3D information. So 3D scanned data are selected for building up a generic 3D human face model.

First, the features found on the point cloud are grouped and represented by NURBS curves. The adjacent curves are the boundaries of surface features. Then, the control points and knot vectors of these features are generated through a reverse computation for which the data points on the features of the "cloud face" shown in Figure 5.4 are manually labelled.

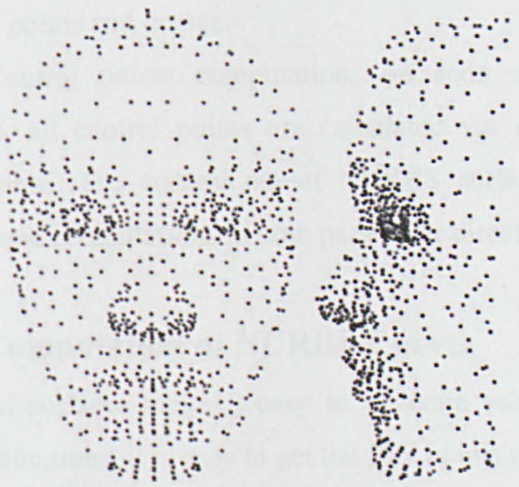


Figure 5.4 Point cloud of a human face

In practical applications, cubic NURBS curves and bicubic surfaces are widely used in order to retain the flexibility of control and reduce the computational complexity. The features of the face model in this thesis are all chosen as cubic curves and bicubic surfaces. Thus, without the loss of generality, the reverse computation takes cubic NURBS curve and bicubic NURBS surface, whose weights equal one, as an example in this section. The process of reverse computation is shown in Figure 5.5:

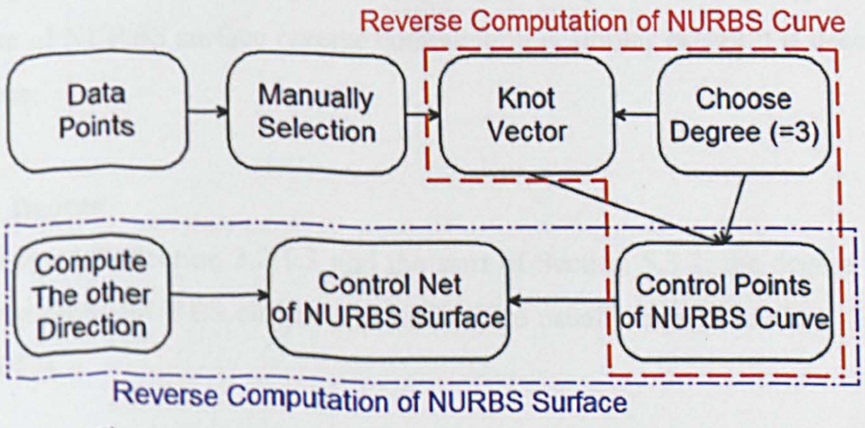

Reverse Computation of NURBS Curve
Data Points
Manually Selection
Knot Vector
Choose Degree (=3)
Compute The other Direction
Control Net of NURBS Surface
Control Points of NURBS Curve
Reverse Computation of NURBS Surface

Figure 5.5 Reverse computation of NURBS curves and surfaces

The procedure of NURBS curves and surfaces reverse computation is:

- Step 1: Data points selection. The needed data points are manually selected from the point cloud.

- Step 2: Degree determination. As discussed in the beginning of this section, the degree of NURBS curves is chosen as three.
- Step 3: Knot vector construction. The knot vector can be computed. Based on the data points and degree.
- Step 4: Control points computation. As soon as the knot vector is determined, all control points are calculated via reverse computation of NURBS curve. The control net of NURBS surface needs two NURBS curves reverse computations in both parameter directions.

5.4.1 Reverse Computation of NURBS Curves

NURBS curves and surfaces make it easy to generate various shapes in CAGD. However, in some situations, it is easy to get the data points on the curves or surfaces rather than get the control points directly. This raises the problem of computing the control points and knot vectors reversely from the data points. The author proposed and developed a new method for obtaining the required geometrical information to define NUBRS curves.

The problem of NURBS curve reverse computation can be described as: given a set of data points $\boldsymbol{Q}=\{q_0,q_1,q_2,\ldots,q_m\}$, to compute the control points $\boldsymbol{P}_i$, weights w_i, and knot vector U that are used to define a clamped NURBS curve with a specified degree p which starts from q_0, ends with q_m, and passes through q_2 to q_{m-1}. The problem of NURBS surface reverse computation is similar except it is described in a 3D space.

5.4.1.1 Degree

As discussed in Section 3.2.1.3 and the start of Section 5.3.2, the degree in reverse computation of NURBS curves and surface are usually chosen as three (Liu, 2008; Sun, 2007).

Because the points of q_j are on the target curve, the correspondence between them and a set of parameter values u exists according to the definition of the NURBS curve.

$$\begin{aligned} q_j = C(u) &= \sum_{j=0}^{m} N_{j,p}(\bar{u}_j)P_j \\ &= \sum_{j=0}^{m} N_{j,3}(\bar{u}_j)P_j \\ j &= 0,1,2,\dots,m \end{aligned} \tag{5.1}$$

Where: p=3, q_j is the data points, $\boldsymbol{P}_j$ is the unknown control points, and $\bar{u}_j$ is the unknown parameter values assigned to each q_j.

Control points and knot vectors are necessary to obtain any NURBS curve. If the $\bar{u}_j$ and knot vector $\boldsymbol{U}$ can be computed, Equation 5.1 will be changed to a set of linear equations that can be solved to get $\boldsymbol{P}_j$. Then the curve can be determined by $\boldsymbol{U}$ and $\boldsymbol{P}_j$. So the next task is to find knot vector.

5.4.1.2 Knot Vector Parameterisation

The curves and surfaces used for this describing the face in this thesis are clamped. Regarding the knot vector $\boldsymbol{U} = [u_0, u_1, u_2, \dots, u_{n+1}, u_{n+2}, \dots, u_{n+p}, u_{n+p+1}]$ of clamped curve, $u_0 = u_1 = u_2 = \cdots = u_p = 0$, $u_{n+1} = n_{n+2} = u_{n+3} = \cdots = u_{n+p+1} = 1$. Only the knots between u_{p+1} and u_n are unknown. Because the clamped NURBS passes through the first and last control points, these two points are set as duplicated control points in this thesis. The relationship between the number of control points n and the number of data points m is n=m+2. So the total number of knots in knot vector is n+p+1=m+2+3+1=m+6, and the total number of control points is n+1=m+3. The unknown knots in the knot vector is from u_{p+1} to u_{m+2} (u_n) can be computed by parameterising knots for each data point q_j. There are three methods can be used to parameterise knots for data point q_j:

(1) Uniform Parameterisation

$$\begin{gathered} u_0 = u_1 = u_2 = u_3 = 0 \\ u_{i+3} = \frac{i}{m} \; for\, i = 1,2,\cdots,m-1 \\ u_{m+3} = u_{m+4} = u_{m+5} = u_{m+6} = 1 \end{gathered} \tag{5.2}$$

(2) Centripetal Parameterisation

$$u_0 = u_1 = u_2 = u_3 = 0$$
$$u_{i+3} = u_{i+2} + \frac{\sqrt{|q_i - q_{i-1}|}}{\sum_{i=1}^{m} \sqrt{|q_i - q_{i-1}|}} \; for\ i = 1,2,\cdots,m-1 \tag{5.3}$$
$$u_{m+3} = u_{m+4} = u_{m+5} = u_{m+6} = 1$$

(3) Cumulative Chord Length Parameterisation (Hartley and Judd, 1978)

$$u_0 = u_1 = u_2 = u_3 = 0$$
$$u_{i+3} = u_{i+2} + \frac{|q_i - q_{i-1}|}{\sum_{i=1}^{m} |q_i - q_{i-1}|} \; for\ i = 1,2,\ldots,m-1 \tag{5.4}$$
$$u_{m+3} = u_{m+4} = u_{m+5} = u_{m+6} = 1$$

Different parameterisation methods lead to different results. The uniform method is suitable for the situation when the edges of polygon formed by the data points are almost equal. The centripetal method is useful while the adjacent edges of the data polygon are sharply angled. The cumulative chord length method is considered to be the best one in most situations because it represents the real distribution of data points, especially while the distribution is not uniform. So this method is selected to calculate knot vector in this thesis.

5.4.1.3 Control Points Computation

According to the analysis in Section 5.4.1.2, the reverse computation can be rephrased as to compute *m*+3 control points and *m*+6 knots of the clamped NURBS curve.

According to Equation 5.1, there are *m*+1 linear equations can be obtained using the knots obtained by knot vector parameterisation discussed in Section 5.4.1.2. The number of the solutions of these *m*+1 equations are only *m*+1. But it requires *m*+3 control points, so two additional equations have to be introduced to find the all solutions. As mentioned in Section 3.2.1.3, the clamped B-Spline curve is tangential to the first and last edge of its control polygon. These two boundary conditions can be used to provide the additional equations needed.

Using two boundary conditions and parameterised knot vector, the linear equation of ***AP=D*** are generated according to Equation 5.1. It can be rewritten as matrix form in Equation 5.5:

$$\begin{bmatrix} a_0 & b_0 & c_0 & & \\ a_1 & b_1 & c_1 & & \\ \ddots & \ddots & \ddots & & \\ & & a_{m+1} & b_{m+1} & c_{m+1} \\ & & a_{m+2} & b_{m+2} & c_{m+2} \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ \vdots \\ P_{m+1} \\ P_{m+2} \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_{m+1} \\ d_{m+2} \end{bmatrix} \tag{5.5}$$

Where: d_0 and d_{m+2} are the bounding conditions at the first and last control point, $d_j=q_{j-1}$ for $j=1,2,\ldots,m+1$, and a_i, b_i, c_i for $i=1,2,\ldots,m+1$ is the Bernstein basis function of three degrees for each $\boldsymbol{u_i}$ previously computed using the cumulative chord length function. The control points vector ***P*** can be computed by $\boldsymbol{P}=\boldsymbol{A}^{-1}\boldsymbol{D}$.

As the degree, knot vector, and control points of the NURBS curve are computed, the unique curve can be evaluated using the de Boor (1972) algorithm. It can also be considered as the NURBS curve whose all weights equal one. It can be changed freely by either control points or weights factors.

5.4.1.4 An Example of NURBS Curve Reverse Computation

The pseudo code of algorithm for NURBS curve reverse computation is show below. All weights coefficients are set to 1.0 initially.

```
Input: degree of NURBS curve p(=3) and data points q[0:m];
Output: knot vector U and control points P[0:m+p+3]=P[0:m+6]

(1) compute the distance array between each pair of data point qj and qj-1
double distance_sum = 0.0;
for (int i=0; i<m; ++i){
  dist[i] = |qi+1-qi|;
  distance_sum += dist[i];
}

(2) compute the knot vector U[0:m+6]
for (int i=0; i<=m+6; ++i){
  if (i<=p) U[i] = 0.0;
  if (i>=m+3) U[i] = 1.0;
  if (p<i<m+3) U[i] = U[i-1]+dist[i]/distance_sum;
}

(3) build up the coefficient matrix A
```

```
for (int i=0; i<=m+2; ++i){
    a[0],b[0],c[0] are the boundary condition of the first control point;
    a[m+2],b[m+2],c[m+2] are the boundary condition of the last control point;
    a[1:m+1] are computed according to the de Boor-Cox function;
}

(4) build up the result vector D
for (int i=0; i<=m+2; ++i){
    d[0] is the boundary condition of the first control point;
    d[m+2] is the boundary condition of the last control point;
    d[1:m+1] is the data points list q[0:m];
}

(5) solve the equation A*P=D by P=inv(A)*D

(6) return U and P.
```

Take the centre line of the nose as an example. The 13 data points are:

x	y	z
0	35.3584	56.9205
0	32.0204	56.2824
0	28.6827	56.9146
0	23.8338	58.9377
0	17.1572	62.5407
0	12.7615	65.4355
0	0.00003	72.9828
0	3.47064	72.1092
0	- 5.02324	71.773
0	- 6.19819	71.1095
0	- 7.37385	70.0651
0	- 9.06913	65.4022
0	- 9.95456	59.9161

Because the x coordinates of all data points are zero, so these 3D points are projected to y-z plane for easy observation (Figure 5.6).

Figure 5.6 Data points on the centre line of nose

For a cubic (degree=3) NURBS curve, the knot vector can be computed according to Equation 5.4. According to the analysis, the number of knots in the knot vector is 13+6=19.

```
[0,0,0,0,0.05825,0.1164,0.2065,0.3366,0.4268,0.6810,0.7423,0.7695,0.7927,0.8196,0.9
047,1,1,1,1]
```

The degree and knot vector are used to calculate the control points. In this thesis, the first and last data points are used as the boundary conditions. According to Equation 5.5, the calculated control points are:

x	y	z
0.0000	35.3584	56.9205
0.0000	35.3584	56.9205
0.0000	31.5255	55.8386
0.0000	28.2434	57.0681
0.0000	23.0111	59.1628
0.0000	17.7758	62.1370
0.0000	10.1017	66.8197
0.0000	3.6795	74.8375
0.0000 -	2.7963	72.1273
0.0000 -	4.9851	71.9057
0.0000 -	6.2783	71.0982
0.0000 -	8.3347	69.3246
0.0000 -	9.0342	65.9038
0.0000 -	9.9546	59.9161
0.0000 -	9.9546	59.9161

It can be seen that the first and last control points are duplicated, so that the NURBS curve will pass through them. The control points and the computed NURBS curve are shown in Figure 5.7.

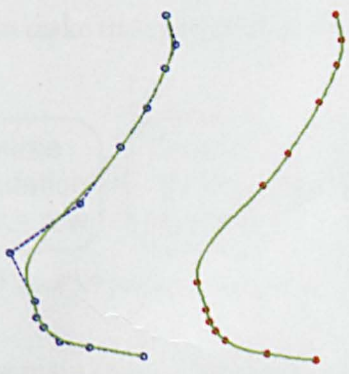
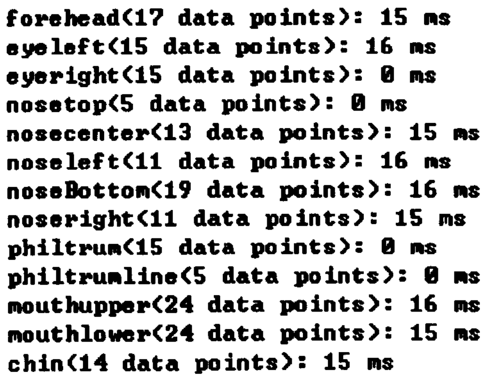


Figure 5.7 NURBS curve by reverse computation
circles (blue) represents the computed control points
dots (red) represents the original data points

The nose feature has 13 data points. The time spent for its reverse computation is 15 milliseconds (ms). The reverse computation performances for other facial features are shown in Figure 5.8. It can be seen that the computed NURBS curve is very fast (less than 16 ms). The obtained NURBS curve passes exactly those data points. So the method for NURBS curve reverse computation discussed above is proved to work for calculating the feature control points.



```
forehead(17 data points): 15 ms
eyeleft(15 data points): 16 ms
eyeright(15 data points): 0 ms
nosetop(5 data points): 0 ms
nosecenter(13 data points): 15 ms
noseleft(11 data points): 16 ms
noseBottom(19 data points): 16 ms
noseright(11 data points): 15 ms
philtrum(15 data points): 0 ms
philtrumline(5 data points): 0 ms
mouthupper(24 data points): 16 ms
mouthlower(24 data points): 15 ms
chin(14 data points): 15 ms
```

Figure 5.8 Executing times of reverse computation for some features

5.4.2 Reverse Computation of NURBS Surface

NURBS curves can be used to represent the curve features of face model. The surface features must be defined by NURBS surface for flexible control. Based on the curve reverse computation, the reverse computation process of NURBS surface is discussed in this section.

NURBS surface reverse computation refers to the problem of giving a series of data points $Q_{i,j}$, i=0,1,...,m; j=0,1,...,n, to construct a NURBS surface that passes through those data points. Because the number of points may vary in each row, new control points need to be inserted to make the computed control points form a control net.

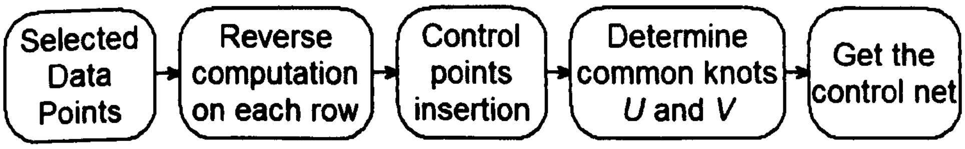


Figure 5.9 Surface reverse computation

As shown in Figure 5.9, the main steps of NURBS surface reverse computation are:

- Step 1: for each row of data points, perform NURBS curve reverse computation to get $C_i(u)$ with control points $\boldsymbol{P}_{i,j}$ and knot vector $\boldsymbol{U}_i$, $i=0,1,2,\ldots,m$;
- Step 2: perform control point insertion for each $C_i(u)$ to make sure they have same number of control points $\boldsymbol{P}_{i,j}$;
- Step 3: compute the common knot vectors $\boldsymbol{U}$ and $\boldsymbol{V}$;
- Step 4: get the control net of the NURBS surfaces.

The surface computation is based on the curves. The results of curves' reverse computation are those boundaries of the surface, although the numbers of control points may be different. However, to evaluate the NURBS surface, the numbers of control points in the boundaries must be same in each pair of opposite boundaries. Thus, an algorithm is used to perform the point insertion to fulfil this requirement.

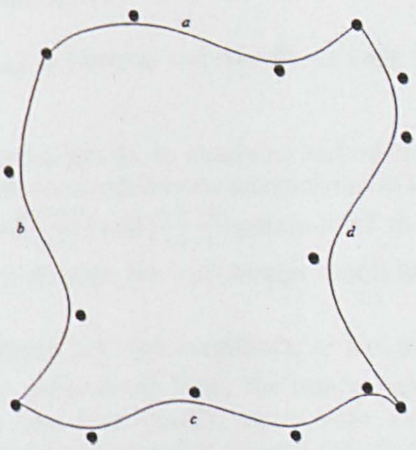


Figure 5.10 Edges and their control points of a NURBS surface

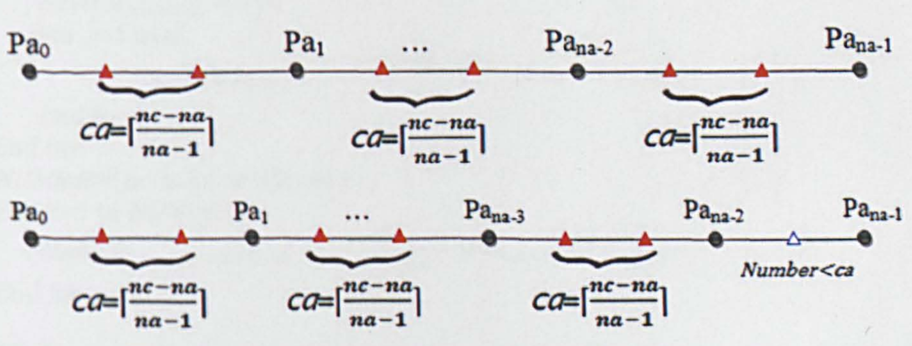


Figure 5.11 Uniform interpolation and non-uniform interpolation

The procedure of the algorithm is shown in Figure 5.12.

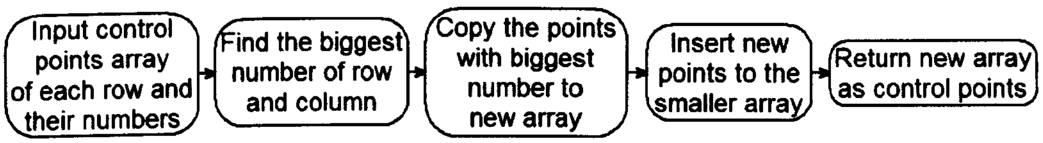

Input control points array of each row and their numbers
Find the biggest number of row and column
Copy the points with biggest number to new array
Insert new points to the smaller array
Return new array as control points

Figure 5.12 Control points insertion algorithm

The pseudo code of algorithm for knot insertions is shown below.

Input: Control points arrays *Pa*, *Pb*, *Pc*, *Pd*, *na*, *nb*, *nc*, *nd* are the numbers of points in each array, where *Pi* is the control points array of edge *i*, $na \neq nc$ and $nb \neq nd$.

Output: NewPa, *NewPb*, *NewPc*, *NewPd*, *n*1 and *n*2, where *NewPi* is the new control points array of edge *i*, *n1* is the number of points in array *NewPa* and *NewPc*, *n2* is the number of points in array *NewPb* and *NewPd*.

(1) $n1=\max(na,nc)$; $n2=\max(nb,nd)$; (assume *na*   *nc* and *nb*   *nd*)

(2) Copy *Pc* to *NewPc*, copy *Pd* to *NewPd*;

(3) Initialise *NewPa* and *NewPb*;

(4) Copy Pa_0 and Pa_{na-1} to $NewPa_0$ and $NewPa_{nc-1}$; copy Pb_0 and Pb_{nb-1} to $NewPb_0$ and $NewPb_{nd-1}$;

(5) For na and nb original points, to obtain nc and nd control points, $(nc-na)$ and $(nd-nb)$ new control points are needed to be interpolated in the $(na-1)$ and $(nb-1)$ spans. It means that there are $\lceil\frac{nc-na}{na-1}\rceil$ and $\lceil\frac{nd-nb}{nb-1}\rceil$ points need to be interpolated in each span respectively, where $\lceil x\rceil$ denotes the least integer that is greater than or equal to *x*.

(6) Let ca=$\lceil\frac{nc-na}{na-1}\rceil$, there are two conditions of the interpolation (Figure 5.11). To compute them in the generalised form, the interpolation is divided into two steps: firstly, interpolating the first $(na-2)$ spans with *ca* points in each span; then interpolating the last span with number $(nc-na)-(na-2)(ca+1)$.

```
For $i=1$ to $na-2$
    $NewPa_{i*(ca+1)} = Pa_i$
    For $j=1$ to ca
        $NewPa_{i*(ca+1)+j} = \frac{(i-1)(ca+1)+j}{ca+1} \cdot |Pa_i - Pa_{i-1}|$
    End for
End for
$NUMleft=(nc-na)-(na-2)(ca+1)$
For $k=1$ to $NUMleft$
    $NewPa_{(na-2)*(ca+1)+k} = \frac{k}{NUMleft+1} \cdot |Pa_{na-1} - Pa_{na-2}|$
End for
```

(7) Repeat techniques in step (6) to compute *NewPb*.

(8) Return *NewPa*, *NewPb*, *NewPc*, *NewPd*, *n*1 and *n*2.

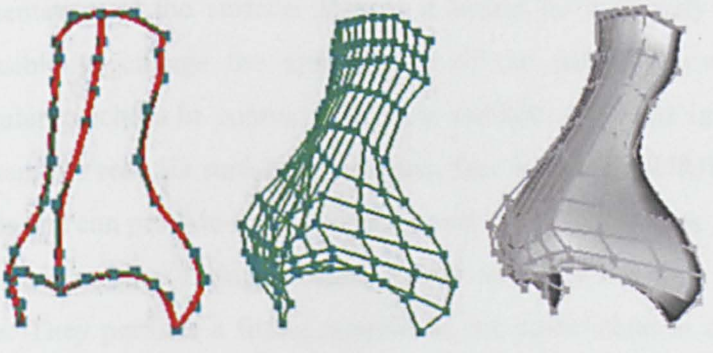


Figure 5.13 Nose surface reverse computation
Left: NURBS curves of the edges of the nose;
Centre: point insertion to make control net;
Right: reverse computed NURBS surface of the nose and its control net.

After the control point insertion, the number of control points on each pair of opposite boundaries will be equal. The NURBS surface is determined by interpolating between each pair of control points on the opposite boundaries. Figure 5.13 shows an example of NURBS surface reverse computation.

The methods discussed in this section can be used to compute the NURBS curves and surfaces reversely from data points. The facial features can be defined by these curves and surfaces. The generic face model can be established based on these features.

5.5 Discussion of the Generic 3D Human Face Model

The new method proposed in this chapter can be used to build a generic 3D human face model with NURBS curves and surface features by applying reverse computation to data points. As far as the author is aware, there are no similar methods available for building a generic 3D face model, so it is impossible to compare the new proposed method with other methods in a quantitative way. So in this section, the new proposed method is compared with two typical existing 3D modelling methods, which are described as follows:

- Triangular patches method: Triangular patches are generated by connecting every three points in a scattered dataset obtained using 3D scanners. Because the dataset is a discrete point cloud, the surface represented by the triangular patches is not a true 3D surface (Rydfalk, 1987; Ahlberg, 2001). It can only be used to show how a surface looks like rather than provide a mathematical representation of the surface. Hence, it would be extremely difficult if not impossible to change the appearances of the surface represented by the triangular patches. In contrast, the new method proposed in this study can represent the real 3D surface of a human face by using NURBS mathematical models and can provide flexible control over all facial features.

- Parametric surface fitting method: These methods are also based on point clouds. They perform a fitting process on the point cloud in order to find the surfaces with minimised errors (Tognola *et al.*, 2003; Marinov and Kobbelt, 2005; Liu *et al.*, 2007). The obtained surfaces are considered as a whole surface without features (Song and Li, 2006). Without explicit facial features, they are not suitable for generating models with various appearances through changing the feature parameters. Moreover, the surface fitting methods need the point cloud for the fitting process. It is very difficult to use these methods to build a face model if the target person cannot be present (for example a VIP). However, the generic 3D human face model built using the method proposed in this thesis consists of 21 NURBS features and 34 NURBS curve features. These features can be manipulated by moving their control points to generate face models with arbitrary appearances such as would be the case for a specific person, or for someone that does not look like anyone else for the application of bio-robots and face aesthetics surgery etc.

5.6 Summary

In this chapter, the author proposed and implemented a new method for building a generic 3D human face model. Firstly, based on the face characteristics analysis, a human face has been divided into various features that are represented with 34 curves and 21 surfaces. These features can be represented by the data structures and classes defined. Then, the author proposed and developed reverse computation algorithms for both NURBS curves and surfaces. An example of NURBS curve reverse computation is represented to test the performance of NURBS reverse computation. The times spent on reverse computation never exceed 16 milliseconds. The proposed reverse computation method is considered to be very efficient for 3D face modelling. Next, the generic 3D face model is established based on the reverse computation method. Finally, the advantages of the proposed method for building a generic 3D human face model are discussed by comparing to two typical existing 3D modelling methods. To test the effectiveness of this method, a software environment is needed to be implemented for simulation, which will be implemented in next chapter.

CHAPTER 6

Prototype and Experiment of the Proposed Generic 3D Face Model

This chapter demonstrates the prototype and experiment of a software environment to test the effectiveness of the proposed method for building a generic 3D human face model. Section 6.1 discusses the development platform, software packages, and libraries. Section 6.2 validates the Graphic User Interface (GUI) design by examining the performances of functions of menus, toolbars, and status bars. Section 6.3 discusses the 3D face model manipulation process. Section 6.4 implements and evaluates the proposed software environment.

6.1 Related Development Packages

To develop the software environment for verifying and validating the generic 3D face modelling method, several programming packages are combined together for design and implementation. These packages include OpenGL (Open Graphic Library) (2009), OpenTK (Open ToolKit) (2009), Tao Framework (Ridge, 2008), and .NET Framework (Microsoft., 2009). The WinForm Application Programming Interface (API) that is originally included in the .NET framework is used to design the GUI as the Rapid Application Development (RAD) tool. OpenGL control provided by OpenTK is embedded as a GUI control to display 3D models. Figure 6.1 shows the hierarchical structure of the software environment.

The different layers of the system are represented by different colours in Figure 6.1. The .NET framework is the fundamental layer (black) to provide all low level APIs as the software environment platform. The second layer (green) consists by two components: WinForm provides various GUI controls to capture user input. OpenGL is the 3D display base library for 3D model representation. Tao is introduced as a dynamic-link library (dll) (Microsoft, 2009) file because OpenTK has not fully implemented the functions included in GLU library (OpenGL Utility Library). The third layer (purple) is the representation layer that includes GUI and 3D display. The fourth layer (orange) of user interactions can be used to manipulate the generic face model in 3D. The main programming language of the software environment to demonstrate the generic 3D face model is VC++.NET.

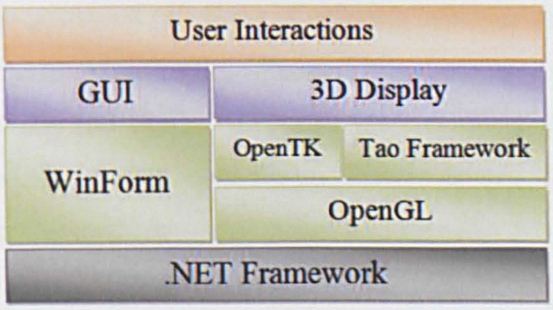


Figure 6.1 Hierarchical structure of software environment packages

6.1.1 OpenGL

OpenGL is a platform-independent high-performance low-level cross-language library and 3D graphic programming interface. It evolved from IRIS GL (Integrated Raster Imaging System Graphics Library) proposed by SGI (Silicon Graphics, Inc.) (Seddon, 2005). Along with the explosive growth of computer games and especially support from Microsoft, OpenGL has become widely used in various areas (Seddon, 2005). Composed of a series of functions, OpenGL can be used to draw complex 3D graphics efficiently. After years of development, the current version of OpenGL standard is 4.2, which provides a great number of advanced features to achieve better user experiences.

As one of the standard tools to deliver high performance and interactive display, OpenGL has several advantages. The most important advantage is the fact that it implements the hardware-independent software interface, which means it offers good portability. The OpenGL interface has been implemented in different platforms, including Windows NT, UNIX, Linux, Mac OS, and OS/2. The OpenGL library has numerous bindings for different computing languages, such as C/C++, Java, Python, C#, Visual Basic, FORTRAN, Perl, and Ada.

OpenGL is actually an open 3D graphics package that includes several extensional libraries. The core library of OpenGL contains more than 300 functions for coordinate transformation, geometric modelling, lighting and material simulation, texture mapping, real-time animations, and interactive operations. Other libraries, including GLU, GLUT (OpenGL Utility Toolkit) and GLAUX (OpenGL Auxiliary Library), enable the advanced functions of drawing complex curves and surfaces and window management. Meanwhile, OpenGL is the 3D engine of numerous commercial modelling and design software, such as Maya (Autodesk., 2009), 3D Studio Max (Autodesk., 2009), and Rhino (2009). In this thesis, the 3D display module of the software environment is based on OpenGL. An example of drawing a ball in 3D with lights is shown in Appendix B. OpenGL 3D Display Example.

6.1.2 OpenTK and Tao Framework

OpenTK and Tao Framework are two different implementation of OpenGL library for .NET supported languages. OpenTK is an advanced, low-level graphic library wrapping of OpenGL, OpenCL (Open Computing Language) (Khronos., 2011), and OpenAL (Open Audio Library), (Labs, 2011). Although OpenTK is written in C#, it can be integrated with other programming languages on .NET platform. OpenTK is open source software distributed under MIT Licence (Initiative, 2009), so it can be used for both commercial and personal purposes absolutely free. Tao Framework is similar as the OpenTK except there is a full implementation of GLU functions. GLU functions are suitable for rapid development because it is a high level drawing routines collection. In this thesis, both OpenTK and Open Tao framework are used for 3D display because the functions in GLU library are used.

6.1.3 .NET Framework

.NET framework is a multi-language integrated development standard designed by Microsoft (2009). It is designed to unify application developments for the Internet as well as desktop and mobile devices. It consists of two main components: Common Language Runtime (CLR) and Basic Class Library (BCL). CLR provides the core services for building and running .NET applications. BCL is integrated object-oriented reusable libraries. Any programming languages that obey the .NET standard can be used to reference BCL to implement advanced functions, such as C++.NET, C#. NET, and VB.NET (Figure 6.2).

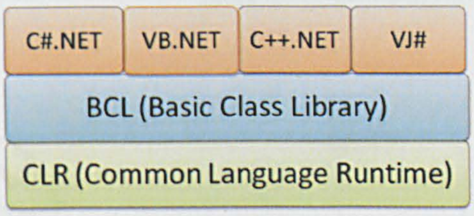


Figure 6.2 Structure of .NET framework

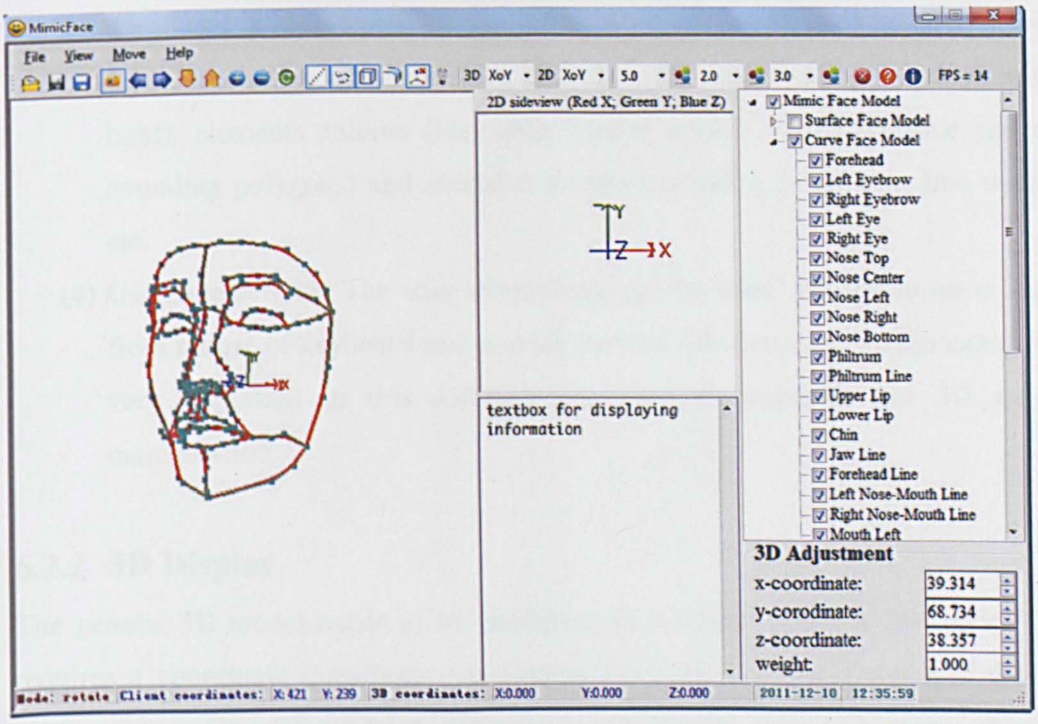
.NET framework has unique advantages for wide applications. It is a set of language standards, so any languages that obey these standards can be run on the platform. This feature encourages the programmers to choose their most familiar programming language. CLR introduces security mechanisms to ensure the reliability of the programme. BCL contains numerous classes and libraries to reuse the code to enhance programming efficiency and advanced features. In this thesis, the WinForm API provided in .NET framework is used for GUI design and user interactions handling.

6.2 GUI Design and 3D Display

To verify and validate the method of building up the generic 3D face model, user interactions are essential to evaluate the effectiveness. In this software environment, the user interactions relates to GUI and 3D display. GUI enables the users to observe and manipulate the 3D model. The 3D model is represented by 3D display that draws the model and projects it to the 2D screen.

6.2.1 GUI Design

The designed GUI of the *MimicFace* is shown in Figure 6.3. The main display area on the left represents the 3D world where the model is represented in. The top centre small window is the 2D planes view of the 3D world. Users can choose between *XOY*, *YOZ*, and *ZOX* planes. An information display textbox is located in the centre bottom. On the right hand side are the features selection treeview control and a group of up-down boxes to adjust the coordinates and the weights in 3D.



```
textbox for displaying
information
```

3D Adjustment
x-coordinate: 39.314
y-corodinate: 68.734
z-coordinate: 38.357
weight: 1.000

Mode: rotate Client coordinates: X:421 Y:239 3D coordinates: X:0.000 Y:0.000 Z:0.000 2011-12-10 12:35:59

Figure 6.3 GUI of the application

The functional requirements and specifications of *MimicFace* are:

(1) 3D model display. The new method is proposed to build a generic 3D face model, but the built model should be displayed on screen for observation. However, the screen is 2D plane, so the 3D model can only be simulated by continuously projecting the model to a fixed plane that is vertical to the camera angles.

(2) 3D model manipulation. The 3D model is feature based to enable the users to change the appearances by adjusting the feature parameters. A mechanism to transform the coordinates between 2D and 3D should be designed. This mechanism contains two stages: (1) to transform the 3D coordinates to 2D screen coordinates; (2) to map the 2D screen coordinates to 3D coordinates. In the first stage, all points on the 3D model are projected to screen to display the model. OpenGL provides the 3D display functions to render the model in 3D. In the second stage, the mouse position on 2D screen is projected

reversely to 3D world to get the selected point on the 3D model. These two closely related stages enable users to manipulate the 3D model interactively.

(3) Environment option setting. There are a group of parameters to define how the model is represented by the software environment, such as environment lights (including ambient light, diffuse light, specular light and emission light), elements colours (including control points, curves, surface and the bounding polygons) and elements widths (including point size, line width), etc.

(4) User interactions. The user interactions are be used to capture users input from mouse or keyboard and provide various functionalities to the users. It is very important in this software environment especially for 3D model manipulation.

6.2.2 3D Display

The generic 3D model needs to be displayed on a screen that is a 2D space. This requires a coordinate transformation process from the 3D world space to the 2D screen space. OpenGL attempts to display 3D objects on a 2D screen through computer graphics pipeline technologies, which is similar to taking photos of the environment using a camera. The world "pipeline" means that there is a fixed order while the graphic card processes and displays data, just like the water flows from one side to the other side of the pipe. So the word of "pipeline" is used to indicate the strict order. The comparison of these two processes is shown in Figure 6.4 (Shreiner, 2009).

To describe a 3D model, the coordinate system is very important for reference. There are four coordinate systems used in OpenGL: model coordinate system (MCS), world coordinate system (WCS), view coordinate system (VCS), and screen coordinate system (SCS). The MCS is useful for defining the local features of a model. The position of the model is originally described in WCS. The remaining three systems are established according to WCS. By applying the model view transformation, it is transformed into VCS. Then it is projected into SCS after coordinate normalisation.

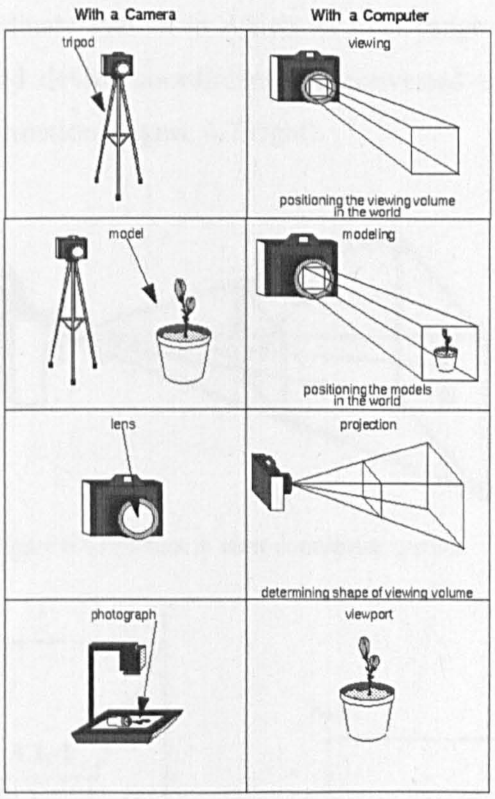

With a Camera
With a Computer
tripod
viewing
positioning the viewing volume in the world
model
modeling
positioning the models in the world
lens
projection
determining shape of viewing volume
photograph
viewport

Figure 6.4 The camera analogy of 3D modelling
Source: *The OpenGL Programming Guide* (Shreiner, 2009)

The OpenGL rendering pipeline is demonstrated in Figure 6.5:

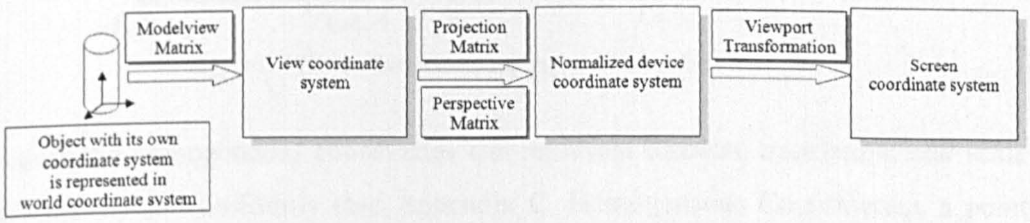


Figure 6.5 OpenGL rendering pipeline

As shown in Figure 6.5, different coordinate systems are used during the process of rendering the OpenGL pipeline. The directions of three axes of WCS point to horizontal right, vertical up, and from screen out. VCS is slightly different from WCS as the *z* axis points inside to the screen, because VCS simulates eyes observing the models. When the model is displayed on screen, only those parts inside the frustum are drawn (Figure 6.6). The projection and perspective transformations are applied to calculate the areas that should be cropped out outside the frustum to get the normalised coordinates (Figure 6.7 left). The normalised coordinate system is a device-independent coordinate system in which all axes range from -1 to 1. In the final step, the normalised device coordinates are converted to screen coordinates through viewport transformation (Figure 6.7 right).

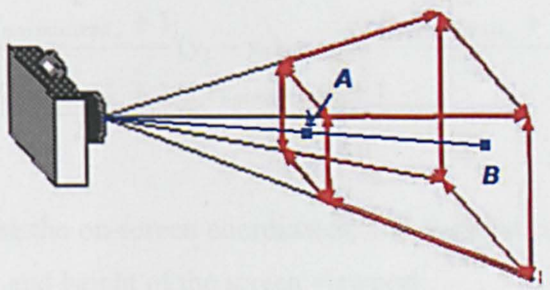


Figure 6.6 Frustum in view coordinate system

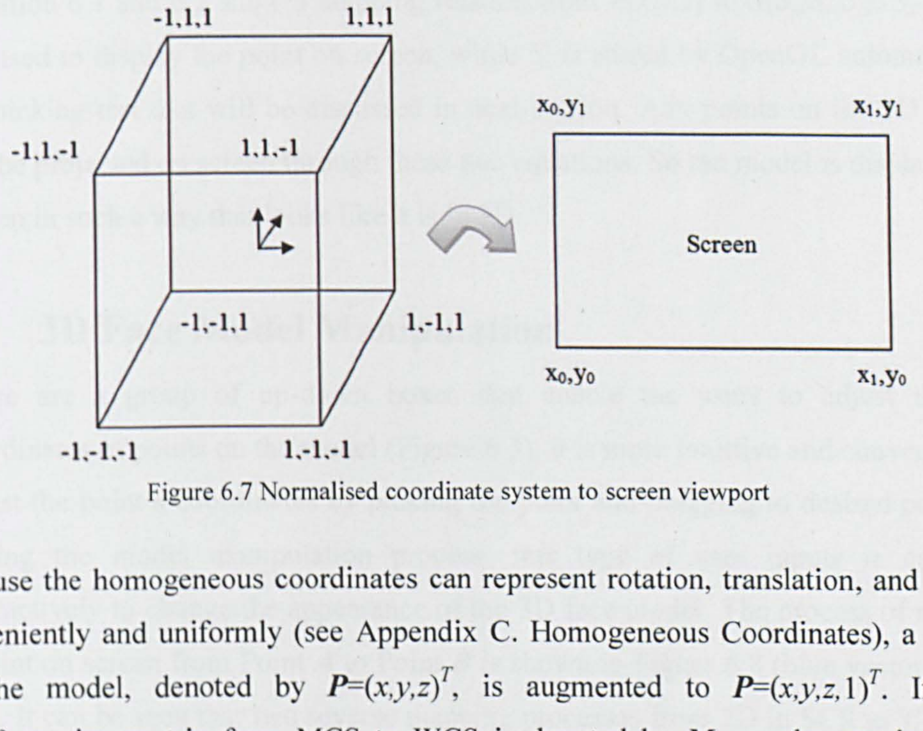


Figure 6.7 Normalised coordinate system to screen viewport

Because the homogeneous coordinates can represent rotation, translation, and scale conveniently and uniformly (see Appendix C. Homogeneous Coordinates), a point on the model, denoted by $P=(x,y,z)^T$, is augmented to $P=(x,y,z,1)^T$. If the transformation matrix from MCS to WCS is denoted by M_{model}, the matrix from WCS to VCS is denoted by M_{view}, usually these two models are represented as a single matrix $M_{modelview}$, called model-view matrix. The matrix of projection process is denoted by $M_{projection}$, the matrix of perspective process is denoted by $M_{perspective}$, and the matrix of viewport transformation is denoted by $M_{viewport}$, the normalised coordinate can be computed according to Equation 6.1:

$$\begin{aligned} P_{normalized} &= M_{viewport}\{M_{perspective}[M_{projection}(M_{view}(M_{model}P))]\} \\ &= M_{viewport}\{M_{perspective}[M_{projection}(M_{modelview}P)]\} \end{aligned} \tag{6.1}$$

The $\boldsymbol{P}_{normalised}$ is the normalised device coordinates. It needs to be transformed to screen coordinates for display by Equation 6.2:

$$
\begin{aligned}
S_x &= \frac{P_{normalized_x}+1}{2}(x_1 - x_0) + x_0 = \frac{P_{normalized_y}+1}{2}w + x_0 \\
S_y &= \frac{P_{normalized_y}+1}{2}(y_1 - y_0) + y_0 = \frac{P_{normalized_y}+1}{2}h + y_0 \\
S_z &= \frac{P_{normalized_z}+1}{2} = \frac{P_{normalized_z}+1}{2}
\end{aligned}
\tag{6.2}
$$

Where: S_x and S_y are the on-screen coordinates, and S_z is the corresponding depth; w and h are the width and height of the screen viewport.

Equation 6.1 and 6.2 show a mapping relation from $\boldsymbol{P}(x,y,z)$ to $\boldsymbol{S}(S_x,S_y,S_z)$. S_x and S_y are used to display the point on screen, while S_z is stored by OpenGL automatically for picking test that will be discussed in next section. Any points on the 3D model can be projected on screen through these two equations. So the model is displayed on screen in such a way that looks like it is in 3D.

6.3 3D Face Model Manipulation

There are a group of up-down boxes that enable the users to adjust the 3D coordinates of points on the model (Figure 6.3), it is more intuitive and convenient to adjust the point's coordinates by picking the point and dragging to desired position. During the model manipulation process, this type of user inputs is captured interactively to change the appearance of the 3D face model. The process of moving a point on screen from Point ***A*** to Point ***B*** is shown in Figure 6.8 (blue vector on the left). It can be seen that two reverse mapping processes from 2D in SCS to 3D WCS are required. The reverse mapping in this thesis refers to the coordinates mapping process from 2D screen point to 3D. Obviously, in the point moving process, the 3D coordinates need to be computed in real-time while the mouse is moving. Therefore, the reverse mapping process should be studied to transform coordinates from 2D to 3D. This section discusses the reverse mapping and local displacement transformation to enable the 3D manipulation of the face model.

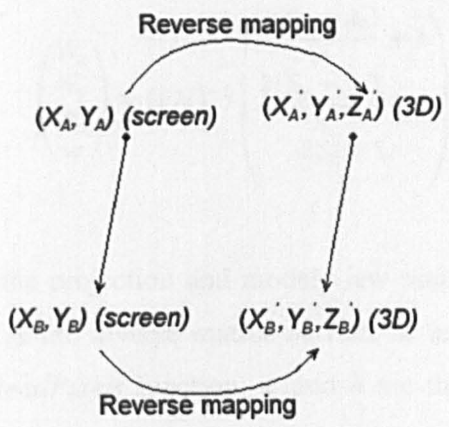


Figure 6.8 Reverse mapping from 2D screen to 3D

6.3.1 Reverse Mapping

When the users click on the screen using the mouse, they actually want to click the model in the 3D world existing in his/her imagination. The "click" action only returns information from the *x* and *y* coordinates. The reverse mapping process from screen coordinate to 3D world coordinate can be divided into two phases: from screen to frustum and from frustum to the 3D world.

As shown in Figure 6.6, Point *A* displayed on screen actually refers to a line segment ***AB*** inside the frustum. All points on ***AB*** with different *z* values appear at the same point on the 2D screen. OpenGL uses a variable depth to store the distance between the point and the near plane of the frustum (Shreiner, 2009).

However, because all points on ***AB*** have the same position on screen (Figure 6.6), so the smallest depth is recorded in OpenGL. This mechanism is efficient and reasonable because the bigger depth means the associated point is behind the other pixel so that it is covered by current pixel. OpenGL provides a function named *glReadPixels* to retrieve the depth of a pixel. This function is:

```
void glReadPixels(GLint x, GLint y, GLsizei width, GLsizei height, GLenum format,
GLenum type, GLvoid * data);
```

The detected *z* depth is returned in the pointer *data*.

Reverse mapping is the reverse process of Equation 6.1 and 6.2. The coordinates of the model in WCS can be computed based on quaternion as Equation 6.3:

$$\begin{pmatrix} W_x \\ W_y \\ W_z \\ w \end{pmatrix} = (PM)^{-1} \begin{pmatrix} \frac{2(S_x - x_0)}{w} + 1 \\ \frac{2(S_y - y_0)}{h} + 1 \\ 2S_z + 1 \\ 1 \end{pmatrix} \tag{6.3}$$

Where: ***P*** and ***M*** are the projection and model-view matrices during the rendering process; $(\boldsymbol{PM})^{-1}$ denotes the inverse matrix of ***PM***; S_z is the depth information of point retrieved by *glReadPixels* function; *w* and *h* are the width and height of the screen viewport.

If users click the mouse on screen, the clicked point on 3D model can be calculated through reverse mapping process. The clicked point on the model can be moved freely to manipulate the generic 3D model. In *MimicFace*, the manipulation is based on the local displacement transformation.

6.3.2 Local Displacement Transformation

To modify the generic face model interactively, the mouse moving event should be captured in real-time. The reverse mapping problem of moving mouse on screen has been solved in Section 6.3.1. However, it needs two reverse mappings. Based on the fact that vector and points has a uniform representation by homogenous coordinates (see Appendix C. Homogeneous Coordinates), this section focuses on invoking the reverse computation only once.

Because all point positions remain unchanged except the one selected by user, the calculation only involves the displacement of the selected point and mapping it in 3D to get the new position of the point. The displacement vector is computed in the local coordinate system of the model.

As shown in Figure 6.9, $x_0y_0z_0$ is WCS and $x_1y_1z_1$ is MCS. According to Euler's rotation theorem (Parent, 2007), one rotation can be transformed into another through three separate rotations about *x*-, *y*- and *z*-axis in a particular order. Because the 3D face model display does not involve the rotation around the *z* axis, the relationship between WCS and MCS can be represented by two rotations around *x* axis and y axis with angles *m_xAngle* and *m_yAngle*, respectively. $x'y'$ is the local SCS. If Point $\boldsymbol{P}$ on the model is picked up and dragged to the new position $\boldsymbol{P'}$, the vector of motion is $\Delta\boldsymbol{P} = \overrightarrow{\boldsymbol{PP'}}$. $\boldsymbol{P'}_{x'}$ and $\boldsymbol{P'}_{y'}$ are the new coordinates in the screen system. To draw the point on the morphed 3D face model, the point $\boldsymbol{P'}$ should be transformed back to $x_0y_0z_0$ (WCS).

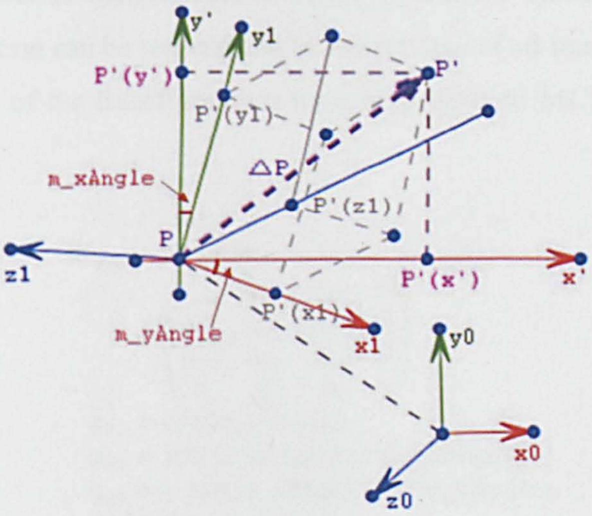


Figure 6.9 3D morphing calculation

The transformation from $x_0y_0z_0$ to the local coordinate system $x'y'z'$ (z' axis points out from the paper) can be achieved with a simple translation. So the transformation matrix is:

$$\boldsymbol{M}_T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -P_{x_0} & -P_{y_0} & -P_{z_0} & 1 \end{pmatrix} \tag{6.4}$$

Where: $\boldsymbol{P}_{x0}$, $\boldsymbol{P}_{y0}$, and $\boldsymbol{P}_{z0}$ are the 3D coordinates of point $\boldsymbol{P}$ in WCS $x_0y_0z_0$. The local coordinate of $\boldsymbol{P}$ is calculated by $\boldsymbol{P}_{local} = \vec{\boldsymbol{P}} \times \boldsymbol{M}_T$, where $\vec{\boldsymbol{P}} = (P_x, P_y, P_z)$ is the position vector of $\boldsymbol{P}$ in WCS.

The transformation from the local coordinate system $x'y'z'$ to MCS $x_1y_1z_1$ can be divided into two separate rotations: one is the rotation about y' axis with angle *m_yAngle*; the other is the rotation about x' axis with angle *m_xAngle*. The homogeneous matrices of these two rotation transformations are:

$$\boldsymbol{M}_y = \begin{pmatrix} cos(m\_yAngle) & 0 & -sin(m\_yAngle) & 0 \\ 0 & 1 & 0 & 0 \\ sin(m\_yAngle) & 0 & cos(m\_yAngle) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$
$$\boldsymbol{M}_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(m\_xAngle) & sin(m\_xAngle) & 0 \\ 0 & -sin(m\_xAngle) & cos(m\_xAngle) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.5}$$

Because the effects of transformations of the coordinate system are accumulative, multi-transformation can be represented by the product of all transformation matrices. Thus, the matrix of the transformation from SCS *x′y′z′* to MCS *x*₁*y*₁*z*₁ is shown in Equation 6.6:

$$\begin{aligned} \boldsymbol{M}_{y\_x} &= \boldsymbol{M}_y \times \boldsymbol{M}_x \\ &= \begin{pmatrix} a_{01} & a_{02} & a_{03} & 0 \\ a_{11} & a_{12} & a_{13} & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ a_{01} &= cos(m\_yAngle) \\ a_{02} &= sin(m\_yAngle)\, sin(m\_xAngle) \\ a_{03} &= -sin(m\_yAngle)\, cos(m\_xAngle) \\ a_{11} &= 0 \\ a_{12} &= cos(m\_xAngle) \\ a_{13} &= sin(m\_yAngle) \\ a_{21} &= sin(m\_yAngle) \\ a_{22} &= -cos(m\_yAngle)\, sin(m\_xAngle) \\ a_{23} &= cos(m\_yAngle)\, cos(m\_xAngle) \end{aligned} \tag{6.6}$$

If Point $\boldsymbol{P}$ moves to a new position, $\boldsymbol{P'}$, the displacement vector in SCS *x′y′z′* is $\boldsymbol{\Delta P} = \left(\Delta P_x, \Delta P_y, \Delta P_z\right)^T where\ \Delta P_z = 0$, the homogenous displacement vector $\boldsymbol{\Delta P'}$ in MCS *x*₁*y*₁*z*₁ can be computed by Equation 6.7.

$$\begin{aligned} \Delta P' &= \boldsymbol{M}_{y\_x} \times \Delta P \\ &= \boldsymbol{M}_{y\_x} \times \begin{pmatrix} \Delta P_x \\ \Delta P_y \\ 0 \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} \Delta P_x\, cos(m\_yAngle) \\ \Delta P_y\, cos(m\_xAngle) + \Delta P_x\, sin(m\_xAngle)\, sin(m\_yAngle) \\ \Delta P_y\, sin(m\_xAngle) - \Delta P_x\, cos(m\_xAngle)\, sin(m\_yAngle) \\ 0 \end{pmatrix} \end{aligned} \tag{6.7}$$

The three components of $\Delta \boldsymbol{P'}$ are the displacement along each direction while moving the mouse. According to the relationship between MCS and WCS, the new position of Point $\boldsymbol{P}$ in WCS is calculated by Equation 6.8:

$$\boldsymbol{P}_{new} = \boldsymbol{P}_{WCS} + \Delta \boldsymbol{P'} \tag{6.8}$$

According to Equation 6.7 and Equation 6.8, the users can perform 3D face model manipulation through the GUI of the software environment to generate different appearances. The software records the rotation angles about *x*- and *y*-axis. While the user chooses a point and moves the point on the screen, the 3D coordinates of the corresponding point on the face model can be computed simultaneously by only one reverse computation of the local displacement.

6.4 Experiment

Based on the discussion above, the software environment is implemented. The implementation is 10,691 lines of C++ code. The code is available upon request. The user can perform manipulation through the provided interfaces. This section introduces the experimental result of *MimicFace* (Figure 6.3). Detailed user interfaces are illustrated in Appendix D. GUI Interface.

As discussed in Chapter 5, the NURBS is very effective for building either a curve face model or a surface face model. The constructed generic 3D face models are vivid and intuitive with defined adjustable features. The generic curve face model and morphed face model are shown in Figure 6.10.

By comparing the two models, it is easy to observe the differences between the features. More specifically, the eyes in the morphed model shrink to become more roundish, and the eyebrows are thinner than those in the generic model. Other changes include the width of the nose, length of the chin, and width of the mouth. So this demonstration proves the effectiveness of the proposed methods and theories used for proposing the new method for 3D human face modelling.

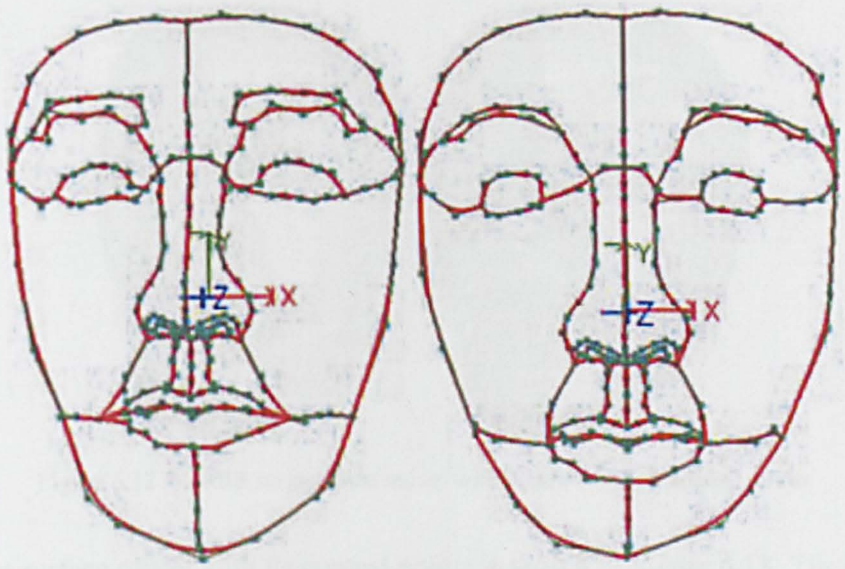


Figure 6.10 Generic and morphed curve face models

The generic 3D surface model and its control net are shown in Figure 6.11.

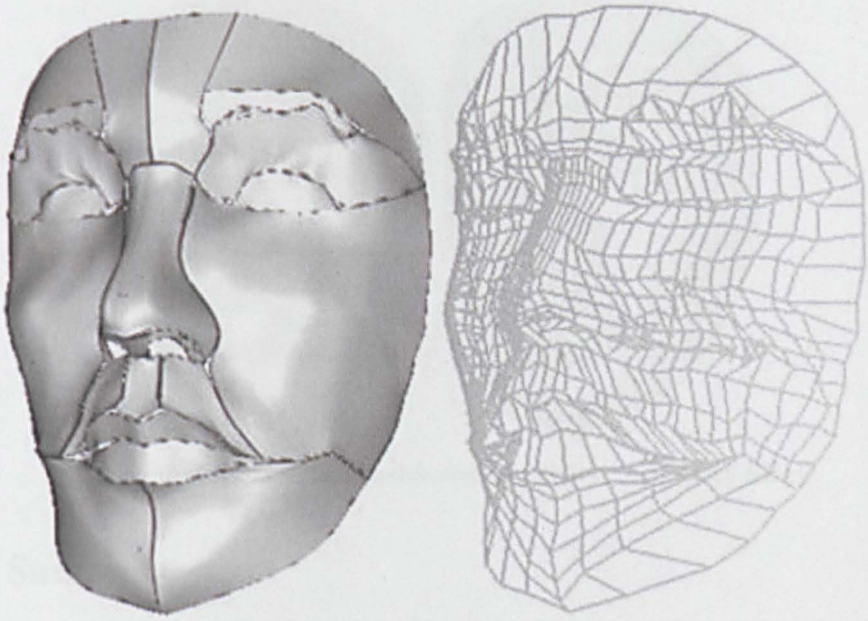


Figure 6.11 NURBS surface face model and its control net

Figure 6.12 illustrates the NURBS surface face model with its control net and control points. All nodes on the net are the control points that can be used to morph the shape of the surface. A personalised face model with various emotions can be generated based on this feature.

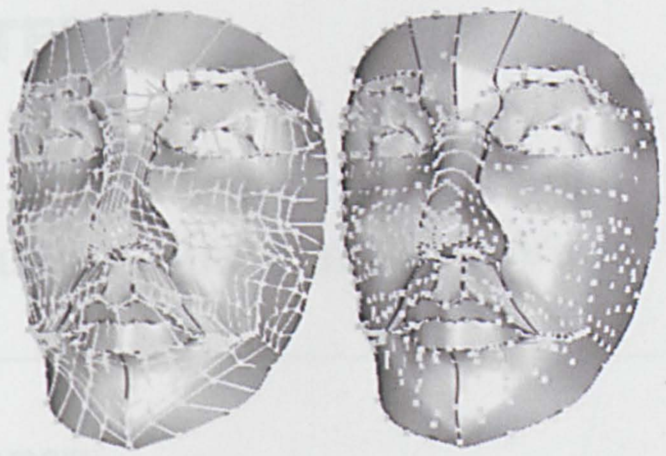


Figure 6.12 NURBS surface face model with control net and control points

A variant surface model with its control points is shown in Figure 6.13. The right eye shrinks slightly by adjusting the control points defining the feature surface. The difference between the left and right eyes is noticeable.

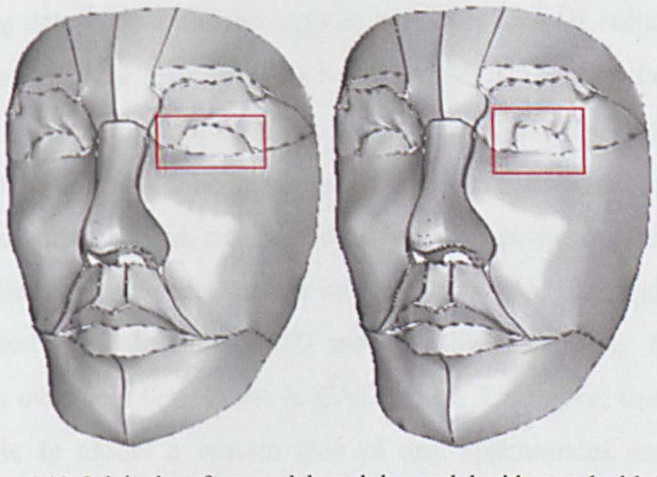


Figure 6.13 Original surface models and the model with morphed left eye

6.5 Summary

A software environment has been developed in this chapter to verify the proposed method for generating a generic 3D face model. The user interfaces can be used to change the appearance of the generic model. It has shown that the generic model can be flexibly manipulated by the control points to generate various face models with different appearances. This generic 3D face model can be applied for emotional bio-robots which enables the engineers to generate various appearances of human face surfaces. It can also be applied for other applications discussed in Section 2.4.

CHAPTER 7

Conclusion and Further Work

7.1 Summary

The human face, as one of the most common objects with complex geometry structure, raises challenges for engineers and researchers to build its 3D model. it is an important research target for emotional bio-robots which are able to show various emotional expressions. Two important requirements of emotional bio-robots are to detect human faces and to build the face model. For face detection, existing methods require training dataset to train the algorithms. They are not suitable for emotional bio-robots because quick responses are required. For 3D human face modelling, the existing 3D modelling technologies have been developed for either CAD/CAM or film and game production. The components used in CAD/CAM mainly consist of primitive geometrical features together with limited free-form surfaces. They do not required very flexible free-form surface modifications. For film and game production, it does not need accurate or true 3D models, the demanding for 3D modelling technologies is even less than those in CAD/CAM. Therefore, the existing methods are not suitable to model a human face of any appearances and expressions for emotional bio-robots. It is very important to propose and develop a new method for building a generic 3D model which can be used to generate personalised models of individuals or a face model with various appearances.

In Chapter 1, through analysing the requirements of emotional bio-robots, the thesis focused on two related problems: face detection and generic 3D human face modelling. It was concluded that it is necessary to build up a generic 3D face model. The background, motivation, and main aims of this thesis were introduced. These are the foundation and main driven force of this thesis. Chapter 1 also described the outline of the whole thesis.

In Chapter 2, the author critically reviewed techniques related to face detection, 3D data acquisition, 3D modelling methods and technologies and their applications. By examining four popular types of existing face detection methods, it was found that the skin colour based method is intuitive and effective. 3D data acquisition methods were discussed as well. The author has shown that the face detection method not only was an important part of emotional bio-robots development, but also can be used to extract facial features to build up the generic 3D face model. By comparing with Bézier and standard B-Spline methods, NURBS was found to be a powerful tool for representing 3D curves and surfaces for its advantages of control flexibility. Based on these reviews and analyses, it was found that it is necessary to propose and develop a method for constructing a generic 3D human face surface model combining the techniques of skin colour based face detection, image processing, and NURBS.

Chapter 3 discussed and evaluated the related work for face detection and NURBS techniques. Firstly, the differences and transformations between several colour spaces were discussed. Image processing technologies including smoothing, edge detection, binarisation, and morphological operations were investigated quantitatively through several experiments. The performances of different techniques have been evaluated and compared. Then, the skin colour properties have been studied based on 1,009,523 skin pixels to build up the skin colour model. The author examined existing methods and technologies and explained how they can be used for a new face detection method proposed and implemented in Chapter 4. Next, the author explored the theories of NURBS techniques. By comparing to Bézier techniques, NURBS does not require more control points while the degree increases. The weight coefficients introduce extra flexibility to change the shapes of NURBS curves and surfaces locally. These theories are the foundation of the reverse computations of NURBS curves and surfaces discussed in Chapter 5.

Chapter 4 proposed and implemented a new skin colour based face detection method. Firstly, skin regions were segmented in the input colour images. Then geometrical rules were applied to verify the face candidates to make the final decisions. The skin colour model used in this method combined the *RGB* and *YCbCr* colour space to generate precise colour information to deal with various luminance conditions. This method has been implemented using Matlab. It is comparable to other previous face detection methods. The most important advantage of this method is that it does not need any training dataset. The new method has been tested over 40 images with 65 human faces. By comparing to other previous methods, the new method also has advantages in both detection rate and detection speed. It achieved a detection rate of 86.15% and detection speed of 0.4~1.2 seconds without any training datasets. These figures indicate that it performs better than some previous methods (see Section 4.5).

Chapter 5 proposed and developed a new method for constructing a generic 3D human face method based on NURBS techniques. Several data structures and classes were defined to store the information of control points and knot vectors of NURBS curves and surfaces, which were used to represent the defined features of 3D human face model. Then, the author proposed and developed a reverse computation algorithm for both NURBS curves and surfaces. An example of NURBS curve reverse computation is represented. The performance of the proposed method has been tested on other facial features as well. The times spent on reverse computation never exceed 16 milliseconds. The proposed reverse computation method is considered to be very efficient for 3D face modelling. Next, the generic 3D face model is established based on NURBS reverse computation. Finally, the advantages of the proposed method for building a generic 3D human face model are discussed by comparing to two typical existing 3D modelling methods.

Based on several computer graphic libraries and programming platform, Chapter 6 developed and implemented a software environment to verify and validate the method of generating the generic 3D face model proposed. After examining the 3D display pipeline of OpenGL, the techniques of the reverse mapping and 3D manipulation were proposed and implemented for my software environment. The GUI interfaces enabled the users to manipulate the 3D model to achieve various appearances. The implementation proved that the generic 3D human face model proposed in this thesis is effective. Compared with all existing methods, the new proposed and developed method can be used to modify each individual face feature so that the 3D face model can be any desired shape or appearances of an existing or any imaginary person.

7.2 Contributions

This thesis focused on two tasks for emotional bio-robots: (1) human face detection and (2) generic 3D human facial modelling. The contributions to the new knowledge generation are:

- A new skin colour based human face detection method was proposed and developed. After examining the differences of skin colour models in various colour spaces, the author proposed a new skin colour model combining *RGB* and *YCbCr* colour space to segment skin regions in the input colour image after luminance normalisation. Based on the skin colour segmentation, face candidates were further verified through the geometrical rules. The method has been tested over 40 images with 65 human faces. By comparing to other previous methods, the new method also has advantages in both detection rate and detection speed. It achieved a detection rate of 86.15% and detection speed of 0.4~1.2 seconds without any training datasets.

- A new generic 3D human face modelling method was proposed and developed. This generic parametric face model can be applied to design and develop a parametric 3D face model that has the abilities of flexible control over all facial features and generating various face models for different applications. It includes:
    - The segmentation of a human face into 21 surface features. These surfaces have 34 boundary curves. This feature based segmentation enables the independent manipulation of different geometrical regions of human face.

    - The NURBS curve face model and NURBS surface face model. These two models are built up based on cubic NURBS reverse computation. The elements of the curve model and surface model can be manipulated to change the appearances of the models by their parameters which are obtained by NURBS reverse computation.

- A new 3D human face modelling method has been proposed and implemented based on cubic NURBS through analysing the characteristic features and boundary conditions of NURBS techniques. This model can be manipulated through control points on the NURBS facial features to build any specific face models for any kind of appearances and to simulate dynamic facial expressions for various applications such as emotional bio-robots, aesthetic surgery, films and games, and crime investigation and prevention, etc.

7.3 Limitations and Further Work

(1) One limitation of this method is associated with the reverse computation. As discussed in Chapter 5, there are various boundary conditions of NURBS reverse computation. By introducing different boundary conditions, different NURBS curves and surfaces can be calculated. Although these curves or surfaces pass the same data points that used for reverse computation, the curvature and derivatives on the data points are different. So the continuity of the reverse computed curves and surfaces are various. C^0 continuities of the reverse computed NURBS curves and surfaces are always ensured by the method discussed in this thesis, but there is no guarantee about other continuities like C^1 and C^2 continuities. However, this problem can be solved by introducing more complex boundary conditions to the reverse computation process or blend the adjacent surfaces boundaries together.

(2) This model is a geometric generic 3D model without texture information. The current research proposed a new method of building up a generic 3D human face model. The author demonstrated that the modelling method for achieving such a model is feasible. The textures are not mapped to the surface because this work focuses on the theoretical possibility of establishing the face model rather than the aesthetic appearances. In the future, it is necessary to study how texture mapping can be used to make the 3D face model realistic. The texture can be cropped from 2D images and mapped to each surface features of the generic 3D face model. Using the proposed face detection method, the facial regions in 2D images could be extracted. The 3D texture is generated by 3D interpolation based on the extracted 2D

information. The face model with texture can be used in the applications of aesthetic surgery and crime detection.

(3) Aesthetic criteria do exist for human face, such as the ratio between the widths of eyes and mouth, the ratio between the width of eyes and the height of nose, the ratio between the eye-to-eye distance and the height of face region, etc. These rules can be defined and embed in to the generic 3D face model to make the generated face model more attractive. By applying these criteria in aesthetic surgery, the generic 3D model can help the patient and surgeon for aesthetic surgeries. It can be applied in the development of emotional bio-robots development. The face models of those bio-robots will be more beautiful and fascinating.

(4) Another limitation occurred during the process of face mode manipulation. At this point, the application enables users to change the model features by themselves. This option is convenient for users, but it may cause unreasonable faces during at the same time. Strict criteria should be introduced to filter those non-faces during the personalisation step. During the process of manipulation the generic 3D face model, AAM or ASM methods can be introduced to get the pre-defined feature points in 2D image. The 3D model can be projected from 3D to 2D plane. All feature points are aligned and get the transformation matrix. The non-feature points will be aligned by applying the affine transformation. By this approach, it is possible to develop a new two stages specific 3D face modelling method to combine the AAM/ASM method with the generic 3D face model proposed in this thesis.

References

Ahlberg, J. (2001) *CANDIDE-3 - an Updated Parameterised Face.* Available at: http: //citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.33.5603&rep=rep1&type=pdf (Accessed: 2 Jan, 2010).

Akimoto, T., Suenaga, Y. and Wallace, R.S. (1993) 'Automatic Creation of 3D Facial Models', *IEEE Computer Graphics & Applications*, 13(5), pp. 16-22.

Aleksic, P.S. and Katsaggelos, A.K. (2006) 'Automatic Facial Expression Recognition Using Facial Animation Parameters and Multistream HMMs', *IEEE Transactions on Information Forensics and Security*, 1(1), pp. 3-11.

Anaface.com. (2009) *Facial Beauty Analysis - Score Your Face.* Available at: http:// www.anaface.com/ (Accessed: 12 Jan, 2010).

Angelopoulou, E. (2001) 'Understanding the Color of Human Skin', *Human Vision and Electronic Imaging VI.* San Jose, USA, 22-25 Jan. pp. 243-251.

Ansari, A.N. and Mottaleb, M.A. (2003) '3-D Face Modeling Using Two Views and a Generic Face Model with Application to 3-D Face Recognition', *IEEE International Conference on Advanced Video and Signal Based Surveillance.* Miami, USA, 21-22 Jul. p. 37.

Au, A.G., Palathinkal, D., Liggins, A.B., Raso, V.J., Carey, J., Lambert, R.G. and Amirfazli, A. (2008) 'A NURBS-Based Technique for Subject-Specific Construction of Knee Bone Geometry', *Computer Methods and Programs in Biomedicine*, 92(1), pp. 20-34.

Autodesk. (2009) *3ds Max - 3D Modeling, Animation, and Rendering Software.* Available at: http://usa.autodesk.com/3ds-max/ (Accessed: 24 Aug, 2009).

Autodesk. (2009) *Maya - 3D Animation, Visual Effect.* Available at: http://usa.auto desk.com/maya/ (Accessed: 22 Aug, 2009).

Bézier, P. (1972) *Numerical Control: Mathematics and Applications.* London, UK: John Wiley & Sons Ltd.

Bézier, P. (1986) *The Mathematical Basis of the UNISURF CAD System.* London, UK: Butterworths.

Bajaj, C.L., Bernardini, F. and Xu, G.L. (1995) 'Automatic Reconstruction of Surfaces and Scalar Fields from 3D Scans', *the 22nd Annual Conference on Computer Graphics and Interactive Techniques.* Los Angeles, USA, 6-11 Aug. pp. 109-118.

Bakry, H.M.E. and Stoyan, H. (2004) 'Fast Neural Networks for Sub-Matrix (Object/Face) Detection', *International Symposium on Circuits and System.* Vancouver, Canada, 23-26 May. pp. 764-767.

Bakry, H.M.E. and Zhao, Q. (2006) 'New Fast PCA for Face Detection', *the 4th International Conference on Information & Communications Technology*. Cairo, Egypt, 10-12 Dec. p. 1.

Ballot, J.S.S. (2005) *Face Recognition Using Hidden Markov Models*. MSc. University of Stellenbosch.

Belhumeur, P.N., Hespanha, J.P. and Kriegman, D.J. (1997) 'Eigenfaces Vs. Fisherfaces: Recognition Using Class Specific Linear Projection', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7), pp. 711-720.

Berbar, M.A., Kelash, H.M. and Kandeel, A.A. (2006) 'Faces and Facial Features Detection in Color Images', *Geometric Modeling and Imaging: New Trends*. London, UK, 05-06 Jul. pp. 209-214.

Beumier, C. and Acheroy, M. (1999) '3D Facial Surface Acquisition by Structured Light', *International Workshop on Synthetic-Natural Hybrid Coding and Three Dimensional Imaging*. Santorini, Greece, 15-17 Sep. pp. 103-106.

Bicego, M. and Murino, V. (2004) 'Investigating Hidden Markov Models' Capabilities in 2D Shape Classification', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2), pp. 281-286.

Blümm, C. and Bozic, I. (2009) *Humanoid Robots*. Available at: http://tams-www.informatik.uni-hamburg.de/lehre/2009ws/seminar/ra/PDF/Humanoid_Robots_Clara_Bluemm_Igor_Bozic.pdf (Accessed: 8 Jan, 2010).

Blanz, V. and Vetter, T. (1999) 'A Morphable Model for the Synthesis of 3D Faces', *the 26th Annual Conference on Computer Graphics and Interactive Techniques*. Los Angeles, USA, 8-13 Aug. pp. 187-194.

Boehm, W. (1980) 'Inserting New Knots into B-Spline Curves', *Computer Aided Design*, 12(4), pp. 199-201.

Boor, C.D. (1972) 'On Calculating with B-Splines', *Journal of Approximation Theory*, 6, pp. 50-62.

Burger, W. and Burge, M. (2008) *Digital Image Processing: An Algorithmic Introduction Using Java*. Berlin, Germany: Springer.

Cai, J. and Goshtasby, A. (1999) 'Detecting Human Faces in Color Images', *Image and Vision Computing*, 18(1), pp. 63-75.

Canny, J. (1986) 'A Computational Approach to Edge Detection', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6), pp. 679-698.

Carswell, D. and Lavery, N. (2006) '3D Solid Fin Model Construction from 2D Shapes Using Non-Uniform Rational B-Spline Surfaces', *Advances in Engineering Software*, 37(8), pp. 491-501.

CAS. (2005) *CAS-PEAL Face Database*. Available at: http://www.jdl.ac.cn/peal/ (Accessed: 4 Oct, 2009).

Casteljau, P.D. (1959) *The Decasteljau Algorithm Outillages Methodes Calcul*. Paris, France: Andre Citroen Automobiles SA.

Chaumont, M. and Beaumesnil, B. (2005) 'Robust and Real-Time 3D-Face Model Extraction', *IEEE International Conference on Image Processing*. Genova, Italy, 11-14 Sep. pp. 461-464.

Chen, C., Pau, L.F., Wang, P.S. and Wang, S. (1999) *Handbook of Pattern Recognition and Computer Vision*. Singapore: World Scientific.

Chen, J.C. and Lien, J.J. (2009) 'A View-Based Statistical System for Multi-View Face Detection and Pose Estimation', *Image and Vision Computing*, 27(9), pp. 1252-1271.

Chen, M. and Qi, F. (2002) 'Face Detection Based on Self-Organized Hidden Markov Model', *Chinese Journal of Computers*, 25(11), pp. 1165-1169.

Choi, C.S., Okazaki, T., Harashima, H. and Takebe, T. (1991) 'A System of Analyzing and Synthesizing Facial Images', *IEEE International Sympoisum on Circuits and Systems*. Singapore, 11-14 Jun. pp. 2665-2668.

Choi, H.C. and Oh, S.Y. (2005) 'Face Detection in Static Images Using Bayesian Discriminating Feature and Particle Attractive Genetic Algorithm', *IEEE International Conference on Intelligent Robots and Systems*. Alberta, Canada, 2-6 Aug. pp. 1072-1077.

CMU. (2009) *The CMU Multi-PIE Face Database*. Available at: http://www.multi pie.org/ (Accessed: 20 Nov, 2010).

Cnix, S. (2009) *Final Fantasy XI*. Available at: http://www.finalfantasyxi.com/ (Accessed: 20 Aug, 2009).

Cohen, E., Lyche, T. and Riesenfeld, R. (1980) 'Discrete B-Splines and Subdivision Techniques in Computer-Aided Geometric Design and Computer Graphics', *Computer Graphics and Image Processing*, 14(2), pp. 87-111.

Colantoni, P. (2001) *Color Spaces*. Available at: http://www.couleur.org/index.php?page=transformations (Accessed: 12 Jun, 2009).

Cootes, T.F., Edwards, G.J. and Taylor, C.J. (1998) 'Active Appearance Models', *European Conference on Computer Vision*. Freiburg, Germany, 2-6 Jun. pp. 484-498.

Cootes, T.F., Taylor, C.J., Cooper, D.H. and Graham, J. (1995) 'Active Shape Model-Their Training and Application', *Computer Vision and Image Understanding*, 61(1), pp. 38-59.

Corp, D.S.S. (2009) *3D CAD Design Software Solidworks*. Available at: http://www. solidworks.com/ (Accessed: 5 Jun, 2009).

Cyberware. (2010) *Head & Face Color 3D Scanner*. Available at: http://www. cyberware.com/products/scanners/px.html (Accessed: 05 Mar, 2010).

Decarlo, D., Metaxas, D. and Stone, M. (1998) 'An Anthropometric Face Model Using Variational Techniques', *SIGGRAPH*. Orlando, USA, 19-24 Jul. pp. 67-74.

Dong, Z., Li, S.Z. and Gatica, P.D. (2004) 'Real-Time Face Detection Using Boosting in Hierarchical Feature Spaces', *the 17th International Conference on Pattern Recognition*. Cambridge, UK, 23-26 Aug. pp. 411-414.

Du, X., Zhu, D. and Zhao, H. (2010) 'Study of Fast Adaboost Face Detection Algorithm', *International Conference on Computer Application and System Modeling*. Taiyuan, China, 22-24 Oct. pp. 136-139.

dy.com.cn. (2010) *Introduction of Takeshi Kaneshiro*. Available at: http://www.dy. com.cn/modules/ActorContent_676.htm (Accessed: 16 Oct, 2010).

Eck, M. and Hoppe, H. (1996) 'Automatic Reconstruction of B-Spline Surfaces of Arbitrary Topological Type', *SIGGRAPH*. New Orleans, USA, 4-9 Aug. pp. 325-334.

Eyetronics. (2010) *3D Eyetronics*. Available at: http://www.eyetronics.com/about. php (Accessed: 07 Mar, 2010).

Fan, X., Peng, Q., Chen, J.X. and Xia, X. (2009) 'An Improved AAM Fast Localization Method for Human Facial Features', *Journal of Electronics and Information Technology*, 31(6), pp. 1354-1358.

Fang, J. and Qiu, G. (2003) 'A Colour Histogram Based Approach to Human Face Detection', *International Conference on Visual Information Engineering*. Surrey, UK, 7-9 Jul. pp. 133-136.

Feng, B., Zhao, Y. and Cui, S. (2005) *Multimedia Techniques and Applications*. Beijing, China: Tsinghua University Press.

Feng, G.C. and Yuen, P.C. (1998) 'Variance Projection Function and Its Application to Eye Detection for Human Face Recognition', *Pattern Recognition Letters*, 19, pp. 899-906.

Ford, A. (1998) *Colour Space Conversions*. Available at: http://www.poynton.com/ PDFs/coloureq.pdf (Accessed: 05 May, 2009).

Forrest, A.R. (1972) 'Interactive Interpolation and Approximation by Bézier Polynomials', *The Computer Journal*, 15(1), pp. 71-79.

Fox, C. (2008) *Ice Age: Continental Drift*. Available at: http://www.iceagemovie. com/ (Accessed: 10 Sep, 2011).

Fox, T. (2009) *It's All Relative: UC San Diego's Einstein Robot Has "Emotional Intelligence".* Available at: http://ucsdnews.ucsd.edu/newsrel/science/02-09Einstein Robot.asp (Accessed: 30 Dec, 2009).

Fox, T.C. (2011) *Avatar Official Website.* Available at: http://www.avatar-movie.co.uk/#/bluray (Accessed: 2 Feb, 2011).

Frowd, C.D., Bruce, V., Gannon, C., Robinson, M., Tredoux, C., Jo, P., Mcintyre, A. and Hancock, P.J.B. (2007) 'Evolving the Face of a Criminal: How to Search a Face Space More Effectively', *Symposium on Bio-inspired, Learning, and Intelligent Systems for Security.* Edinburgh, UK, 9-10 Aug. pp. 3-10.

Ganesh. (2008) *Basics of Computer Aided Geometric Design: An Algorithmic Approach.* New Delhi, India: I.K. International Publishing House Pvt. Ltd.

Garcia, C. and Delakis, M. (2004) 'Convolutional Face Finder: A Neural Architecture for Fast and Robust Face Detection', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11), pp. 1408-1423.

Geng, J. (2011) 'Structured-Light 3D Surface Imaging: A tutorial', *Advances in Optics and Photonics*, 3(2), pp. 128-160.

Georghiades, A.S. (2005) *The Yale Face Database B.* Available at: http://vision.ucsd.edu/~leekc/ExtYaleDatabase/Yale%20Face%20Database.htm (Accessed: 15 Oct, 2009).

Gonzalez, R.C., Woods, R.E. and Eddins, S.L. (2009) *Digital Image Processing Using Matlab.* 2nd edn. New Jersey, USA: Gatesmark Publishing.

Gordon, W.J. and Riesenfeld, R.F. (1974) 'B-Spline Curves and Surfaces', in Barnhill and Riesenfeld (eds.) *Computer Aided Geometric Design.* Waltham, USA: Academic Press, pp. 95-126.

Gottumukkal, R. and Asari, V.K. (2003) 'Real Time Face Detection from Color Video Stream Based on PCA Method', *the 32nd Applied Imagery Pattern Recognition Workshop.* Washington DC, USA, 15-17 Oct. pp. 146-150.

Graf, H.P., Chen, T., Petajan, E. and Cosatto, E. (1995) 'Locating Faces and Facial Parts', *International Workshop on Automatic Face and Gesture Recognition.* Zurich, Switzerland, 26-28 Jun. pp. 41-46.

Graf, H.P., Cosatto, E., Gibbon, D., Kocheisen, M. and Petajan, E. (1996) 'Multi-Modal System for Locating Heads and Faces', *the 2nd International Conference on Automatic Face and Gesture Recognition.* Killington, USA, 14-16 Oct. pp. 88-93.

Grey, P. (2010) *Point Grey - Stereo Vison - Bumblebee XB3.* Available at: http://www.ptgrey.com/products/bbxb3/index.asp (Accessed: 02 Apr, 2010).

Guan, P. and Zhang, L. (2008) 'Three Dimensional Localization of Facial Feature Points Based on Bezier Surface', *Journal of Fudan University (Natural Science)*, 47(1), pp. 117-123.

Guenter, B., Grimm, C., Malvar, H. and Wood, D. (1998) 'Making Faces', *the 25th Annual Conference on Computer Graphics and Interactive Techniques*. Orlando, USA, 19-24 Jul. pp. 55-66.

Hanson, D. (2008) *Creepily Realistic Robot Can Hold Conversations and Answer Questions*. Available at: http://www.vidmax.com/media/video/1100.wmv (Accessed: 20 Dec, 2009).

Hartley, P.J. and Judd, C.J. (1978) 'Parametrization of Bézier-Type B-Spline Curves and Surfaces', *Computer Aided Design*, 10(2), pp. 130-134.

Hashimoto, T., Hiramatsu, S. and Kobayashi, H. (2006) 'Development of Face Robot for Emotional Communication between Human and Robot', *IEEE International Conference on Mechatronics and Automation*. Luoyang, China, 25-28 Jun. pp. 25-30.

He, G., Zhu, B. and Gan, J. (2008) '3-D Face Reconstruction Using Rbf and B Spline Method ', *Congress on Image and Signal Processing*. Sanya, China, 27-30 May. pp. 239-243.

He, Y. and Qin, H. (2004) 'Surface Reconstruction with Triangular B-Splines', *Geometric Modeling and Processing*. Beijing, China, 13-15 Apr. pp. 279-287.

Hjelmas, E. and Low, B.K. (2001) 'Face Detection: A Survey', *Computer Vision and Image Understanding*, 83(3), pp. 236-274.

Horace, H.S.I. and Yin, L. (1996) 'Constructing a 3D Individualized Head Model', *The Visual Computer*, 12(5), pp. 254-266.

Hore, U.W. and Ingole, D.T. (2010) 'Comparative Implementation of Colour Analysis Based Methods for Face Detection Algorithm', *the 3rd International Conference on Emerging Trends in Engineering and Technology*. Jaipur, India, 19-21 Nov. pp. 176-179.

Hou, Y., Fu, Z. and Zhang, Y. (2006) 'Face Feature Points Extraction Based on Refined ASM', *Application Research of Computers*, 11, pp. 255-257.

Hsu, C.Y., Wang, H.F., Wang, H.C., Tseng, K.K. and Tang, Y.J. (2010) 'Automatic Extraction of Face Contours', *International Joint Conference on Neural Networks*. Barcelona, Spain, 18-23 Jul. pp. 1-8.

Hsu, R.L., Mottaleb, M.A. and Jain, A.K. (2002) 'Face Detection in Color Images', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5), pp. 696-706.

Huang, F. and Su, J. (2004) 'Face Contour Detection and Tracking with Complex Backgrounds', *International Conference on Machine Learning and Cybernetics*. Shanghai, China, 26-29 Aug. pp. 3855-3859.

Huang, L.L., Shimizu, A., Hagihara, Y. and Kobatake, H. (2003) 'Face Detection from Cluttered Images Using a Polynomial Neural Network', *Neurocomputing*, 51, pp. 197-211.

Hyungkeun, J., Kyunghee, L. and Sungbum, P. (2004) 'Eye and Face Detection Using SVM', *Intelligent Sensors, Sensor Networks and Information Processing Conference*. Melbourne, Australia, 14-17 Dec. pp. 577-580.

IGN. (2001) *Onimusha: Warlords Review*. Available at: http://uk.ps2.ign.com/objects/011/011520.html (Accessed: 9 Dec, 2009).

Initiative, O.S. (2009) *The MIT License*. Available at: http://www.opensource.org/licenses/mit-license.php (Accessed: 10 Jan, 2010).

ITU-R. (1994) *Encoding Parameters of Digital Television for Studios*. ITU-R BT.601-4. Geneva, Switzerland: International Telecommunication Union Radiocommunication Sector Broadcasting Service (Television).

Jablonski, N.G. and Chaplin, G. (2000) 'The Evolution of Human Skin Coloration', *Journal of Human Evolution*, 39(1), pp. 57-106.

Jain, A.K., Zhong, Y. and Jolly, M.P.D. (1998) 'Deformable Template Models: A Review', *Signal Processing*, 71(2), pp. 109-129.

Jones, M. and Poggio, T. (1998) 'Multidimensional Morphable Models: A Framework for Representing and Matching Object Classes', *International Journal of Computer Vision*, 29(2), pp. 107-131.

Jones, M.J. and Rehg, J.M. (2002) 'Statistical Color Models with Application to Skin Detection', *International Journal of Computer Vision*, 46(1), pp. 81-96.

Kanad, T. (1973) *Picture Processing System by Computer and Recognition of Human Face*. Ph.D. Kyoto University.

Kar, S., Hiremath, S., Joshi, D.G., Chadda, V.K. and Bajpai, A. (2006) 'A Multi-Algorithmic Face Recognition System', *International Conference on Advanced Computing and Communications*. Mangalore, India, 20-23 Dec. pp. 321-326.

Khronos. (2011) *OpenCL - the Open Standard for Parallel Programming of Heterogeneous Systems*. Available at: http://www.khronos.org/opencl/ (Accessed: 20 Oct, 2009).

Kim, J.S. and Choi, S.M. (2008) 'Interactive Cosmetic Makeup of a 3D Point-Based Face Model', *IEICE Transactions on Information and Systems*, E91-D(6), pp. 1673-1680.

Kim, T.K., Kim, H., Hwang, W., Kee, S.C. and Kittler, J. (2003) 'Face Description Based on Decomposition and Combining of a Facial Space with Lda', *International Conference on Image Processing*. Barcelona, Spain, 14-17 Sep. pp. 877-880.

Kobayashi, H. and Zhao, Q. (2007) 'Face Detection with Clustering, Lda and Nn', *IEEE International Conference on Systems, Man and Cybernetics*. Montréal, Quebec, Canada, 7-10 Oct. pp. 1670-1675.

Kohonen, T. (1995) *Self-Organizing Maps*. Berlin, Germany: Springer.

Koschan, A. and Abidi, M.A. (2008) *Digital Color Image Processing*. New Jersey, USA: Wiley-Interscience.

Kotropoulos, C. and Pitas, I. (1997) 'Rule- Based Face Detection in Frontal Views', *IEEE International Conference on Acoustics, Speech, and Signal*. Munich, Germany, 21-24 Apr. pp. 2537-2540.

Kovac, J., Peer, P. and Solina, F. (2003) *Human Skin Color Clustering for Face Detection*. Available at: http://lrv.fri.uni-lj.si/~peterp/publications/eurocon03.pdf (Accessed: 10 Nov, 2009).

Kumar, T.S. and Rakesh, P.B. (2011) '3D Reconstruction of Facial Structures from 2D Images for Cosmetic Surgery', *International Conference on Recent Trends in Information Technology*. Chennai, TamilNadu, India, 3-5 Jun. pp. 743-748.

Lab, A.T. (2006) *The Database of Faces*. Available at: http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html (Accessed: 15 Feb, 2010).

Labs, C. (2011) *OpenAL Homepage*. Available at: http://connect.creativelabs.com/openal/default.aspx (Accessed: 25 Oct, 2009).

Lang, L. and Gu, W. (2009) 'The Face Detection Algorithm Combined Skin Color Segmentation and PCA', *International Conference on Information Engineering and Computer Science*. Wuhan, China, 19-20 Dec. pp. 1-3.

Lang, L. and Gu, W. (2009) 'Study on Face Detection Algorithm Based on Skin Color Segmentation and Adaboost Algorithm', *the 2nd Pacific-Asia Conference on Web Mining and Web-based Application*. Wuhan, China, 6-7 Jun. pp. 70-73.

Larrabee, W.F., Makielski, K.H. and Henderson, J.L. (2004) *Surgical Anatomy of the Face*. Philadelphia, USA: Lippincott Williams & Wilkins.

Lawsky, D. (2009) *Einstein Robot Smiles When You Do*. Available at: http://www.reuters.com/article/2009/02/05/us-robot-einstein-idUSTRE51474520090205 (Accessed: 11 Nov, 2009).

Lee, C.H., Kim, J.S. and Park, K.H. (1996) 'Automatic Human Face Location in a Complex Background Using Motion and Color Information', *Pattern Recognition*, 29(11), pp. 1877-1889.

Lee, S.Y., Chwa, K.Y. and Shin, S.Y. (1995) 'Image Metamorphosis Using Snakes and Free-Form Deformations', *SIGGRAPH*. Los Angeles, USA, 6-11 Aug. pp. 439-448.

Lee, W.S. and Nadia, M.T. (2000) 'Fast Head Modeling for Animation', *Journal of Image and Vision Computing*, 18(4), pp. 355-364.

Li, J. and Jiang, Y. (2003) 'Synthesis of a Realistic Virtual 3-D Human Face', *Journal of Xidian University (Natural Science)*, 30(4), pp. 525-529.

Li, S.Z. and Zhang, Z. (2004) 'Floatboost Learning and Statistical Face Detection', *IEEE Transactions on Patern Analysis and Machine Intelligence*, 26(9), pp. 1112-1123.

Liang, L., Ai, H. and He, K. (1999) 'Single Face Detection Based on Multiple Templates Matching', *Journal of Image and Graphics*, 4(10), pp. 825-830.

Lin, C. and Fan, K.C. (2000) 'A Color-Triangle-Based Approach to the Detection of Human Face', *Biologically Motivated Computer Vision*. Seoul, Korea, 15-17 May. pp. 195-200.

Liu, B. and Ning, S. (2008) 'Constrained Free Form Deformation Driven by Sectional Outline Curve', *International Symposium on Computational Intelligence and Design*. Wuhan, China, 17-18 Oct. pp. 283-286.

Liu, C. (2003) 'A Bayesian Discriminating Features Method for Face Detection', *IEEE Transactions on Patern Analysis and Machine Intelligence*, 25(6), pp. 725-740.

Liu, D., Zhang, H., Luo, L. and Luo, L. (2009) 'Real-Time Face Detection Using Multiple Instances Boosting Cascade', *Chinese Conference on Pattern Recognition*. Nanjing, China, 4-6 Nov. pp. 1-5.

Liu, S., Jin, X., Wang, C.L. and Hui, K.C. (2007) 'Ellipsoidal-Blob Approximation of 3D Models and Its Applications', *Computers & Graphics*, 31(2), pp. 243-251.

Liu, X., Geng, G. and Wang, X. (2010) 'Automatically Face Detection Based on Bp Neural Network and Bayesian Decision', *the 6th International Conference on Natural Computation*. Yantai, Shandong, China, 10-12 Aug. pp. 1590-1594.

Liu, Z.C., Zhang, Z.Y., Jacobs, C. and Cohen, M. (2001) 'Rapid Modeling of Animated Faces from Video', *Journal of Visualization and Computer Animation*, 12(4), pp. 227-240.

Lu, P., Chen, Y. and Chen, W. (2010) 'USAN-AAM Method for Fast Human Facial Features Location', *Application Research of Computers*, 27(8), pp. 3188-3190.

Ma, S. and Du, T. (2010) 'Improved Adaboost Face Detection', *International Conference on Measuring Technology and Mechatronics Automation*. Changsha, China, 13-14 Mar. pp. 434-437.

Maglogiannis, I., Vouyioukas, D. and Aggelopoulos, C. (2009) 'Face Detection and Recognition of Natural Human Emotion Using Markov Random Fields', *Personal and Ubiquitous Computing*, 13(1), pp. 95-101.

Mahajan, M., Nimbhorkar, P. and Varadarajan, K. (2009) 'The Planar K-Means Problem Is NP-Hard. Walcom: Algorithms and Computation', in Das and Uehara (eds.) *Lecture Notes in Computer Science*. Berlin, Germany: Springer, pp. 274-285.

Mahyar. (2008) *Mechanics, Materials, Software & Arts*. Available at: http:// mahyarghf.blogspot.com/2009/09/mechanics-materials-software-arts.html (Accessed: 29 Dec, 2009).

Marinov, M. and Kobbelt, L. (2005) 'Optimization Methods for Scattered Data Approximation with Subdivision Surfaces', *Graphical Models*, 67(5), pp. 452-473.

Matsui, A., Clippingdale, S. and Matsumoto, T. (2008) 'Bayesian Sequential Face Detection with Automatic Re-Initialization', *the 19th International Conference on Pattern Recognition*. Tampa, USA, 8-11 Dec. pp. 1-4.

Mehrabian, A. (1968) 'Communication without Words', *Psychology Today*, 2(9), pp. 52-55.

Mena, C.J.P., Macedo, I., Velho, L. and Cesar, R.M. (2008) 'PCA-Based 3D Face Photography', *XXI Brazilian Symposium on Computer Graphics and Image Processing*. Campo Grande, Brazil, 12-15 Oct. pp. 313-320.

Miao, J., Yin, B., Wang, K., Shen, L. and Chen, X. (1999) 'A Hierarchical Multiscale and Multiangle System for Human Face Detection in a Complex Background Using Gravity-Center Template', *Pattern Recognition*, 32, pp. 1237-1248.

Microsoft. (2009) *Microsoft .Net Framework*. Available at: http://www.microsoft. com/net (Accessed: 12 Sep, 2009).

Morera, D.M., Carvalho, P.C. and Velho, L. (2007) 'Geodesic Bezier Curves: A Tool for Modeling on Triangulations', *XX Brazilian Symposium on Computer Graphics and Image Processing*. Minas Gerais, Brazil, 7-10 Oct. pp. 71-78.

Moschovakis, Y.N. (2006) *Notes on Set Theory*. Los Angeles, USA: Springer.

Nadia, M.T. and Bonanni, U. (2010) *Modeling and Simulating Bodies and Garments*. Berlin, Germany: Springer.

Nagabhushana, S. (2006) *Computer Vision and Image Processing*. New Delhi, India: New Age International.

Nazeer, S.A., Omar, N., Jumari, K.F. and Khalid, M. (2007) 'Face Detecting Using Artificial Neural Network Approach', *the 1st Asia International Conference on Modelling & Simulation*. Phuket, Thailand, 27-30 Mar. pp. 394-399.

Nefian, A.V. and Hayes, M.H. (1998) 'Face Detection and Recognition Using Hidden Markov Models', *International Conference on Image Processing*. Chicago, USA, 4-7 Oct. pp. 141-145.

Nefian, A.V. and Hayes, M.H. (1999) 'An Embedded HMM-Based Approach for Face Detection and Recognition', *IEEE International Conference on Acoustics, Speech, and Signal Processing*. Phoenix, USA, 15-19 Mar. pp. 3553-3556.

Nefian, A.V. and Hayes, M.H. (1999) 'Face Recognition Using an Embedded HMM', *IEEE Conference on Audio and Video-based Biometric Person Authentication*. Washington DC, USA, 22-23 Mar. pp. 834-847.

Nefian, A.V. and Hayes, M.H. (2000) 'Maximum Likelihood Training of the Embedded HMM for Face Detection and Recognition', *International Conference on Image Processing*. Vancouver, Canada, 10-13 Sep. pp. 33-36.

Nguyen, T.D., Thanh, Q.T., Duc, T.M., Quynh, T.N. and Hoang, T.M. (2011) 'SVM Classifier Based Face Detection System Using Bdip and Bvlc Moments', *International Conference on Advanced Technologies for Communications*. Danang, Vietnam, 2-4 Aug. pp. 264-267.

NIST. (2008) *The Color Feret Database*. Available at: http://face.nist.gov/colorferet/ (Accessed: 16 July, 2009).

Opengl.org. (2009) *OpenGL - the Industry Standard for High Performance Graphics*. Available at: http://www.opengl.org/ (Accessed: 2 Jun, 2009).

OpenTK.com. (2009) *OpenTK*. Available at: http://www.opentk.com/ (Accessed: 11 Oct, 2009).

Or, C.C.F. and Wilson, H.R. (2010) 'Face Recognition: Are Viewpoint and Identity Processed after Face Detection?', *Vision Research*, 50(16), pp. 1581-1589.

Osuna, E., Freund, R. and Girosi, F. (1997) 'Training Support Vector Machines: An Application to Face Detection', *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Juan, Puerto Rico, 17-19 Jun. pp. 130-136.

Otsu, N. (1979) 'A Threshold Selection Method from Gray-Level Histograms', *IEEE Transactions on Systems, Man and Cybernetics*, 9(1), pp. 62-66.

Parent, R. (2007) *Computer Animation: Algorithms and Techniques*. Burlington, USA: Morgan Kaufmann.

Park, J.H., Choi, H.C. and Kim, S.D. (2005) 'Bayesian Face Detection in an Image Sequence Using Face Probability Gradient Ascent', *IEEE International Conference on Image Processing*. Genoa, Italy, 11-14 Sep. pp. 346-349.

Parke, F.I. (1972) 'Computer Generated Animation of Faces', *ACM National Conference*. Boston, USA, 14-16 Aug. pp. 451-457.

Paul, P.P. and Gavrilova, M. (2011) 'PCA Based Geometric Modeling for Automatic Face Detection', *International Conference on Computational Science and Its Applications*. Santander, Spain, 20-23 Jun. pp. 33-38.

Peer, P. (2003) *CVL Face Database*. Available at: http://www.lrv.fri.uni-lj.si/facedb.html (Accessed: 10 Jan, 2010).

Peng, X., Gao, P.D., Liu, X.L. and Liu, Z.Y. (2006) 'Realistic Facial Modeling Based on Subdivision Surface', *Journal of Computer-aided Design & Computer Graphics*, 18(5), pp. 742-747.

Pham, N.P.T. and Huynh, Q.L. (2007) *Robust Face Detection under Challenges of Rotation, Pose and Occlusion*. Available at: http://www.fas.hcmut.edu.vn/webhn10/Baocao/PDF/YVSM_PhuongTrinh.pdf (Accessed: 2 Jan, 2010).

Phimoltares, S., Lursinsap, C. and Chamnongthai, K. (2007) 'Face Detection and Facial Feature Localization without Considering the Appearance of Image Context', *Image and Vision Computing*, 25(5), pp. 741-753.

Piegl, L. (1987) 'Interactive Data Interpolation by Rational Bezier Curves', *IEEE Computer Graphics and Applications*, 7(4), pp. 45-48.

Piegl, L. (1989) 'Modifying the Shape of Rational B-Splines. Part1: Curves', *Computer Aided Design*, 21(8), pp. 509-518.

Piegl, L. (1989) 'Modifying the Shape of Rational B-Splines. Part2: Surfaces', *Computer Aided Design*, 21(9), pp. 538-546.

Piegl, L. (1991) 'On NURBS: A Survey', *IEEE Computer Graphics and Applications*, 11(1), pp. 55-71.

Pighin, F., Szeliski, R. and Salesin, D.H. (2002) 'Modeling and Animating Realistic Faces from Images', *International Journal of Computer Vision*, 50(2), pp. 143-169.

Poggio, T. and Brunelli, R. (1992) *A Novel Approach to Graphics*. Technical report 1354. Cambridge, USA: MIT Media Laboratory Perceptual Computing Section.

Poggio, T. and Vetter, T. (1992) *Recognition and Structure from One 2D Model View: Observations on Prototypes, Object Classes and Symmetries*. A.I. Memo 1354. Cambridge, USA: MIT.

Poynton, C.A. (1996) *A Technical Introduction to Digital Video*. San Francisco, USA: John Wiley & Sons.

Prautzsch, H. and Piper, B. (1991) 'A Fast Algorithm to Raise the Degree of Spline Curves', *Computer Aided Geometric Design*, 8(4), pp. 253-265.

Principe, J.C., Xu, D. and Wang, C. (1997) 'Generalized Oja's Rule for Linear Discriminant Analysis with Fisher Criterion', *IEEE International Conference on Acoustics, Speech, and Signal Processing*. Munich, Germany, 21-24 Apr. pp. 3401-3404.

Qian, Z. and Zhanbin, C. (2008) 'A Novel Method to Train Support Vector Machines for Solving Quadratic Programming Task', *the 7th World Congress on Intelligent Control and Automation*. Chongqing, China, 25-27 Jun. pp. 7917-7921.

Ranal, M.A., Setan, H., Majid, Z. and Chong, A.K. (2007) 'Simulation of Interactive Cutting Tool for Craniofacial Osteotomy Planning', *Computer Graphics, Imaging and Visualisation*. Bangkok, Thailand, 14-17 Aug. pp. 506-512.

Research, L. (2009) *Virtual Worlds, Avatars, Free 3D Chat, Online Meeting*. Available at: http://secondlife.com/ (Accessed: 10 Jun, 2010).

Rhino3d. (2009) *Modeling Tools for Designers*. Available at: http://www.rhino3d.com/ (Accessed: 10 Jul, 2009).

Ridge, R. (2008) *The Tao Framework*. Available at: http://sourceforge.net/projects/taoframework/ (Accessed: 16 Nov, 2009).

Rose, E.H. (1998) *Aesthetic Facial Restoration*. New York, USA: Lippincott-Raven.

Rowley, H.A., Baluja, S. and Kanade, T. (1996) 'Neural Network-Based Face Detection', *IEEE Transactions On Pattern Analysis and Machine Intelligence*, 20, pp. 23-38.

Rowley, H.A., Baluja, S. and Kanade, T. (1998) 'Rotation Invariant Neural Network-Based Face Detection', *IEEE Conference on Computer Vision and Pattern Recognition*. Santa Barbara, USA, 23-25 Jun. pp. 38-44.

Rydfalk, M. (1987) *CANDIDE, a Parameterized Face*. Report No. LiTH-ISY-R-2326. Linköping, Sweden: Linköping University.

Séguier, R. (2004) 'A Vary Fast Adaptive Face Detection System', *International Conference on Visualization, Imaging and Image Processing*. Marbella, Spain, 6-8 Sep. p. 452.

Sakai, T., Nagao, M. and Fujibayashi, S. (1969) 'Line Extraction and Pattern Detection in a Photograph', *Pattern Recognition*, 1(3), pp. 233-236.

Salomon, D. (2006) *Curves and Surfaces for Computer Graphics*. New York, USA: Springer.

Samaria, F. (1993) 'Face Segmentation for Identification Using Hidden Markov Models', *the 4th British Machine Vision Conference*. Surrey, UK, 21-23 Sep. pp. 65-74.

Schneiderman, H. and Kanade, T. (1998) 'Probabilistic Modeling of Local Appearance and Spatial Relationships for Object Recognition', *IEEE Conference on Computer Vision and Pattern Recognition*. Santa Barbara, USA, 23-25 Jun. pp. 45-51.

Shan, T., Chen, S., Sanderson, C. and Lovell, B.C. (2007) 'Towards Robust Face Recognition for Intelligent-CCTV Based Surveillance Using One Gallery Image', *IEEE Conference on Advanced Video and Signal Based Surveillance*. London, UK, 5-7 Sep. pp. 470-475.

Shan, Y., Liu, Z. and Zhang, Z. (2001) 'Model-Based Bundle Adjustment with Application to Face Modeling', *the 8th International Conference on Computer Vision*. Vancouver, Canada, 7-14 Jul. pp. 644-651.

Shashua, A. (1994) 'Projective Structure from Two Uncalibrated Images: Structure from Motion and Recognition', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(8), pp. 778-790.

Shavers, C., Li, R. and Lebby, G. (2006) 'An SVM-Based Approach to Face Detection', *the 38th Southeastern Symposium on System Theory*. Tennessee, USA, 5-7 Mar. pp. 362-366.

Shi, F. (1994) *CAGD & NURBS*. Beijing, China: Beihang University Press.

Shreiner, D. (2009) *OpenGL Programming Guide: The Official Guide to Learning OpenGL*. Boston, USA: Addison Wesley Professional.

Singh, R., Vatsa, M., Ross, A. and Noore, A. (2007) 'A Mosaicing Scheme for Pose-Invariant Face Recognition', *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 37(5), pp. 1212-1225.

Song, Y. and Li, B. (2006) 'Single B-Spline Patch 3D Modelling for Facial Analysis', *the 8th International Conference on Advanced Concepts for Intelligent Vision Systems*. Antwerp, Belgium, 18-21 Sep. pp. 786-798.

Stillwell, J. (2010) 'Projective Geometry', in Axler and Ribet (eds.) *Mathematics and Its History*. Berlin, Germany: Springer, pp. 133-135.

Su, Y. and Kumar, S. (2005) 'Generalized Surface Interpolation for Triangle Meshes with Feature Retention', *Computer Aided Design & Applications*, 2(1-4), pp. 193-202.

Sun, N., Ayabe, T. and Nishizaki, T. (2007) 'Efficient Spline Interpolation Curve Modeling', *the 3rd International Conference on Intelligent Information Hiding and Multimedia Signal Processing*. Splendor Kaohsiung, Taiwan, 26-28 Nov. pp. 59-62.

Sung, K.K. and Poggio, T. (1998) 'Example-Based Learning for View-Based Human Face Detection', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1), pp. 39-51.

Systemes, D. (2002) *CATIA - Virtual Product*. Available at: http://www.3ds.com/products/catia/ (Accessed: 15 May, 2009).

Takahashi, Y., Hatakeyama, M. and Kanno, M. (2007) 'Experiments on Human Facial Expressions for Improvement of Simplified Robot Face', *Annual Conference SICE*. Kagawa, Japan, 17-20 Sep. pp. 480-483.

Takahashi, Y. and Sato, H. (2010) 'Compact Robot Face with Simple Mechanical Components', *International Conference on Control Automation and Systems.* Gyeonggi-do, Korea, 27-30 Oct. pp. 2300-2303.

Talele, K.T. and Kadam, S. (2009) 'Face Detection and Geometric Face Normalization', *IEEE Region 10 Conference TENCON.* Suntec City, Singapore, 23-26 Nov. pp. 1-6.

Tang, Y., He, Z., Chen, Y. and Wu, J. (2009) 'ATM Intelligent Surveillance Based on Omni-Directional Vision', *World Congress on Computer Science and Information Engineering.* Los Angeles, USA, 31 Mar-2 Apr. pp. 660-664.

Tiller, W. and Piegl, L. (1997) *The NURBS Book.* New York, USA: Springer.

Tognola, G., Parazzini, M., Svelto, C., Ravazzani, P. and Grandori, F. (2003) 'A Fast and Reliable System for 3D Surface Acquisition and Reconstruction', *Image and Vision Computing,* 21(3), pp. 295-305.

Trier, D. and Jain, A.K. (1995) 'Goal-Directed Evaluation of Binarization Methods', *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 17(12), pp. 1191-1201.

Turk, M.A. and Pentland, A.P. (1991) 'Face Recognition Using Eigenfaces', *IEEE International Conference on Computer Vision and Pattern Recognition.* Maui, Hawaii, USA, 3-6 Jun. pp. 586-591.

Tyan, S. (1981) 'Median Filtering: Deterministic Properties', in Huang (ed.) *Two-Dimensional Digital Signal Prcessing II.* Berlin, Germany: Springer, pp. 197-217.

Ullman, S. and Basri, R. (1991) 'Recognition by Linear Combinations of Models', *IEEE Transactions on Patern Analysis and Machine Intelligence,* 13(10), pp. 992-1006.

Vapnik, V. (1995) *Support Vector Networks.* New York, USA: Springer.

Versprille, K.J. (1975) *Computer-Aided Design Applications of the Rational B-Spline Approximation Form.* PhD. Syracuse University.

Vetter, T. and Blanz, V. (1998) 'Estimating Coloured 3D Face Models from Single Images: An Example Based Approach', *the 5th European Conference on Computer Vision.* Freiburg, Germany, 2-6 Jun. pp. 499-513.

Vezhnevets, V., Sazonov, V. and Andreeva, A. (2003) 'A Survey on Pixel-Based Skin Color Detection Techniques', *GraphiCon.* Moscow, Russia, 5-10 Sep. pp. 85-92.

Viola, P. and Jones, M.J. (2004) 'Robust Real-Time Face Detection', *International Journal of Computer Vision,* 57(2), pp. 137-154.

Wang, J. and Tan, T. (2000) 'A New Face Detection Method Based on Shape Information', *Pattern Recognition Letters,* 21, pp. 463-471.

Waters, K. and Terzopoulos, D. (1991) 'Modelling and Animating Faces Using Scanned Data', *The Journal of Visualization and Computer Animation*, 2(4), pp. 123-128.

Wei, Q., Matuszewski, B.J., Shark, L.K. and Djamel, A.B. (2007) '3-D Facial Expression Representation Using B-Spline Statistical Shape Model', *Vision,. Video and Graphics Workshop*. Coventry, UK, 14 Sep. pp. 12-17.

Wu, J., Brubaker, S.C., Mullin, M.D. and Rehg, J.M. (2008) 'Fast Asymmetric Learning for Cascade Face Detection', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3), pp. 369-382.

Wu, K., Huang, L., Lin, H. and Kong, X. (2011) 'Face Detection Based on YCbCr Gaussian Model and KL Transform', *International Symposium on Computer Science and Society*. Kota Kinabalu, Malaysia, 16-17 Jul. pp. 100-103.

Wu, Z., Zhou, Y., Du, C., Yuan, Q. and Ge, X. (2008) 'Research on Facial Feature Points Extraction in Color Images', *Chinese Journal of Electronics*, 36(2), pp. 309-313.

Xiao, B., Zhang, Q. and Wei, X. (2006) 'A NURBS Facial Model Based on MPEG-4', *the 16th International Conference on Artificial Reality and Telexistence Workshops*. Hangzhou, China, 29 Nov-1 Dec. pp. 491-495.

Xiao, Y., Zhan, Q. and Pang, Q. (2007) '3D Data Acquisition by Terrestrial Laser Scanning for Protection of Historical Buildings', *International Conference on Wireless Communications, Networking and Mobile Computing*. Shanghai, China, 21-25 Sep. pp. 5971-5974.

Yan, J. and Zhang, H. (2000) 'Realistic Virtual Face and Body Synthesis', *the IAPR Conference on Machine Vision Applications*. Tokyo, Japan, 28-30 Nov. pp. 91-94.

Yan, L., Ball, R. and Justice, L. (2011) 'The 3D Chinese Head and Face Modeling', *Computer Aided Design*, 44(1), pp. 40-47.

Yang, G. and Huang, T.S. (1994) 'Human Face Detection in a Complex Background', *Pattern Recognition*, 27(1), pp. 53-63.

Yang, J., Ling, X., Zhu, Y. and Zheng, Z. (2008) 'A Face Detection and Recognition System in Color Image Series', *Mathematics and Computers in Simulation*, 77(5–6), pp. 531-539.

Yang, M.H., Ahuja, N. and Kriegman, D. (1999) *A Survey on Face Detection Methods*. Available at: http://vision.ai.uiuc.edu/mhyang/papers/survey.ps.gz (Accessed: 10 Dec, 2009).

Yang, M.H., Ahuja, N. and Kriegman, D. (2000) 'Face Detection Using Mixtures of Linear Subspaces', *the 4th IEEE International Conference on Automatic Face and Gesture Recognition*. Grenoble, France, 26-30 Mar. pp. 70-76.

Yap, M.H., Ugail, H., Zwiggelaar, R., Rajoub, B., Doherty, V., Appleyard, S. and Hurdy, G. (2009) 'A Short Review of Methods for Face Detection and Multifractal Analysis', *International Conference on CyberWorlds*. Bradford, UK, 7-11 Sep. pp. 231-236.

Yin, B. and Gao, W. (1998) 'Modeling of Facial Expressions and Degree of Lip-Rounding Using Bézier Surface', *Chinese Journal of Computers*, 21, pp. 347-350.

Yin, X. and Xie, L. (2005) 'Deformation Based on B-Spline Interpolation', *Journal of Anhui Technical Teachers College*, 19(5), pp. 21-22.

Yoo, T.W. and Oh, I.S. (1999) 'A Fast Algorithm for Tracking Human Faces Based on Chromatic Histograms', *Pattern Recognition Letters*, 20(10), pp. 967-978.

Yuille, A.L., Hallinan, P.W. and Cohen, D.S. (1992) 'Feature Extraction from Faces Using Deformable Templates', *International Journal of Computer Vision*, 8(2), pp. 99-111.

Zhan, S. and Chung, C.K.R. (2010) 'Determining Both Surface Position and Orientation in Structured-Light-Based Sensing', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(10), pp. 1770-1780.

Zhang, J., Zhang, X.D. and Ha, S.W. (2008) 'A Novel Approach Using PCA and SVM for Face Detection', *the 4th International Conference on Natural Computation*. Jinan, China, 18-20 Oct. pp. 29-33.

Zhang, M. and Fulcher, J. (1996) 'Face Recognition Using Artificial Neural Network Group-Based Adaptive Tolerance (GAT) Trees', *IEEE Transactions on Neural Networks*, 7(3), pp. 555-567.

Zhang, Q., Kamata, S.I. and Zhang, J. (2009) 'Face Detection and Tracking in Color Images Using Color Centroids Segmentation', *International Conference on Robotics and Biomimetics*. Bangkok, Thailand, 22-25 Feb. pp. 1008-1013.

Zhang, S., Hapeshi, K. and Bhattacharya, A.K. (2004) '3D Modelling of Biological Systems for Biomimetics', *Journal of Bionic Engineering*, 1(1), pp. 20-40.

Zhang, Z., Zhu, L., Li, S.Z. and Zhang, H. (2002) 'Real-Time Multi-View Face Detection', *IEEE International Conference on Automatic Face and Gesture Recognition*. Washington DC, USA, 20-21 May. pp. 149-154.

Appendix

A. Data File Structure Description

There are two types of user defined file structure in this thesis: *ZCM* file and *ZSM* file. *ZCM* denotes the ***Zhang's** NURBS **Curve face Model*** file, and *ZSM* denotes the ***Zhang's** NURBS **Surface face Model*** file. Both of *ZCM* and *ZSM* files are saved as plain text files, so that they can be revised by all plain text processing programmes. One of the most important advantages of plain text file is that they are compatible with different operation system architectures like Windows, Linux and UNIX although the differences between the line break symbols among those systems do exist.

The detailed structures of *ZCM* and *ZSM* files are illustrated respectively.

(a) *ZCM* File Structure Description

ZCM is the file structure defined for storing the data structures which are used to handle the NURBS curve face model. There are very limited symbols and letters used in the *ZCM* file. They are # (number sign), – (hyphen), = (sign of equality), *h*, *f*, *c*, and *u*. Each of them leads a single line or multiple lines of different data. The meanings of each catalogue are:

Lines started with # (number sign)

Those lines started with # (number sign) are all comment line. They show basic information of the face model file, including title, author, created date and time, and so on. A sample line starts with # is:

```
# This is a comment line
```

Lines started with *h*

There is only one line starts with *h*. It indicates that this is a face model file (either curve face model – *ZCM* or surface face model – *ZSM*). This line, which is like "*h NUM*", always appears as the first non-comment line in *ZCM* (or *ZSM*) file. The integer *NUM* after letter *h* specifies how many curve features are predefined in the

ZCM file. As stated in Section 5.2, there are 34 features defined in this thesis. A sample line starts with ***h*** is:

```
h 34
```

Lines started with *f*

The letter *f* leads a line of curve feature definition with pattern "*f ID DESCRIPTION*". The integer *ID* after letter *f* is the feature index corresponding to the enumerate structure defined in Section 5.3. The string *DESCRIPTION composed* by the rest characters in the line is a brief description of the curve feature. A sample line starts with *f* is:

```
f 0 forehead
```

Lines started with – (hyphen)

Those lines started with – (hyphen) is used for indicating the number of control points of a specific curve feature. The line pattern is "– *ID NUM*". *ID* has the same meaning as the integer in the Lines started with *f*, which means the feature index of the curve feature. *NUM* is an integer shows how many control points the curve feature has. A sample line starts with – is:

```
- 0 19
```

Lines started with = (sign of equality)

= (sign of equality) leads those lines that indicate the number of elements of knots vector of a specific curve feature. These lines have similar pattern like those leading by – (hyphen): "= *ID NUM*". The integer *ID* means the feature index of the curve feature. Integer *NUM* indicates the number of elements of the knots vector of the curve feature. If the number of elements of the knots vector of a curve feature is denoted by n_1, the number of control points of the curve feature is denoted by n_2, the degree and order of the curve features are denoted by d and o respectively. Then n_1, n_2, d, and o satisfy such condition: $n_1=n_2+d+1=n_2+o$. In this thesis, the degree and order are 3 and 4 for cubic NURBS curves are chosen to represent the curve features. So $n_1=n_2+4$. A sample line starts with = is:

```
= 0 23
```

Lines started with *c*

The leading letter *c* means the line defines a control point of a NURBS curve. The line pattern is like "*c ID INDEX X Y Z*". *ID* is the feature index of the curve feature mentioned above. *INDEX* denotes the control point's indices in the control points array. *X*, *Y*, and *Z* are the 3D coordinates (*x*,*y*,*z*) of the control point. A sample line starts with *c* is shown as:

```
c 0 4 -47.505901 62.636185 32.970329
```

Lines started with *u*

Lines started with letter *u* indicate these lines are knots definitions with the pattern of "*u ID INDEX t*". *ID* has the same meaning as in Lines started with *c* and *f*. *INDEX* is the index of current knot element in the knots vector. *t* is the value of current element. A sample line starts with *u* is:

```
u 0 6 0.183562
```

(b) *ZSM* File Structure Description

Like *ZCM* file, *ZSM* file is defined to store data structure of NURBS surface features face model. The notations used in *ZSM* file include: *f, s, r, c, d, o,* # (number sign), * (asterisk), | (vertical bar), > (greater than sign), – (hyphen), ^ (caret). The meanings of all letters and signs are:

Lines started with # (number sign)

# has exactly the same meaning as it does for *ZCM* file - leading a comment line. These comment lines provide necessary information about the file type, author, contact information, and creation date and time. This symbol also enables the user to add more information and definitions into the file to expand the functionalities. The comment lines are:

```
# This is the comment line
# Author:  Xu ZHANG
# Created:  22 June 2011
```

Lines started with *f*

The line started with *f* defines the number of surface features of the NURBS surface features face model. The line pattern is "*f NUM DESCRIPTION*". *NUM* indicates the number of surface features of the model. *DESCRIPTION* describes the meaning of this line. A sample line is:

```
f 21 number of features
```

Lines started with *s*

The letter *s* leads lines of surface feature definitions. The pattern of these lines is "*s ID DESCRIPTION*". *ID* is the index of current surface feature of the surface face model. *DESCRIPTION* gives information about the current surface features. A sample line is:

```
s 0 Left Upper Eyebrow
```

Lines started with *r*

The letter *r* leads lines of the row number of current surface feature's control point matrices. The pattern of these lines is "*r ID NUM*". *ID* is the index of current surface feature. *NUM* denotes how many rows there are in the control point matrix of current surface feature. A sample line is:

```
r 0 4
```

Lines started with *c*

The letter *c* leads lines of the column number of current surface feature's control point matrices. The pattern of these lines is "*c ID NUM*". *ID* is the index of current surface feature. *NUM* indicates how many columns there are in the control point matrix of current surface feature. A sample line is:

```
c 0 9
```

Lines started with *d*

The letter *d* is used to lead those lines indicating the dimensions of the current surface feature's control point matrices. The fourth dimension used in *ZSM* files denotes the weights of each control points of the NURBS surface. So in the line pattern "*d ID NUM*", *ID* is the index of current surface feature. And *NUM* is a constant value of 4, which means the attributes of all control points defined in the rest of *ZSM* file comprise 4 elements: *X* coordinate, *Y* coordinate, *Z* coordinate, and weight. A sample line started with *d* is:

```
d 0 4
```

Lines started with *o*

The leading letter ***o*** indicates that this line defines the order of corresponding surface feature. *ID* in the line pattern "*o ID NUM*" is the index of current surface feature. *NUM* is the order of current NURBS surface feature. Generally, the orders, which equal to the value of degree plus one, vary from one surface to another. Furthermore, the orders are usually different in *u* and *v* directions for a specific NURBS surface. However, all NURBS surfaces have the same constant orders in both directions in this thesis. The value of all these order are set to 3 which means these NURBS surfaces used to representing the surface features of the face model are bi-cubic NURBS surfaces. Although the orders are constants here, this parameter is reserved to keep the compatibility of the *ZSM* file in the future version. A sample line is like:

```
o 1 4
```

Lines started with * (asterisk)

The * (asterisk) is used to lead the lines of control points definition. These lines have the line pattern of "** ID ROW COL X Y Z W*". *ID* is the index of current surface feature of surface face model. *ROW* is the row number of current control point and *COL* is the column number of current control point. *X, Y, Z,* and *W* are the 3D coordinates (x, y, z) and weight respectively.

As mentioned above, by contrast with the definition of control points for NURBS curve in *ZCM* file, the control points defined in *ZSM* file are all stored as four dimensional points. The fourth dimension is chosen to save the weights of current control point, rather than a geometrical dimension. A sample line starts with * is:

```
d 0 4
```

Lines started with | (vertical bar)

The symbol of | (vertical bar) leads lines of knots vector in *u* direction. These lines are usually formed like "| *ID NUM*". *ID* is the index of current surface feature of the surface face model. *NUM* is the number of elements of the knots vector in *u* direction of current NURBS surface. The following line is a sample:

```
| 0 8
```

Lines started with > (greater than sign)

The lines started with > (greater than sign) specify elements of the knots vector of a NURBS surfaces in *u* direction. These lines have the format like "> *ID INDEX t*". *ID* is the index of current surface feature of the surface face model. *INDEX* defines the index of current knot element of the knots vector in *u* direction. *t* is the value of the current knot element. A sample line started with > is:

```
> 3 5 0.333333
```

Lines started with – (hyphen)

The lines started with – (hyphen) define the knots vector in *v* direction. The pattern of these lines are "– *ID NUM*". *ID* is the index of current surface feature of the surface face model. *NUM* is the number of knot elements of the knots vector of current NURBS surface. A sample line is:

```
- 5 12
```

Lines started with ^ (caret)

The lines started with ^ (caret) mean that they define the elements of the knots vector of a NURBS surface in *v* direction. The format is usually like "^ *ID INDEX t*". *ID* is the index of current surface feature of the surface face model. *INDEX* is the index of the current knot element of the knots vector in *v* direction. *t* is a float point number that specifies the value of the current knot element. A sample line is shown below:

```
^ 6 4 0.071429
```

B. OpenGL 3D Display Example

A typical OpenGL 3D display programme is shown below.

```
#include<GL/gl.h>
#include<GL/glut.h>

// initialise the environment
  void initGLenv(void)
{
    // set light colour
    GLfloat mat_specular[]={1.0,1.0,0.0,1.0};
    GLfloat mat_shininess[]={50.0};
    GLfloat light_position[]={1.0,1.0,1.0,0.0};
    GLfloat light_ambient[]={1.0,0.0,0.0,1.0};
    GLfloat light_diffuse[]={1.0,1.0,0.0,1.0};
    GLfloat light_specular[]={1.0,1.0,0.0,1.0};

    // set background colour as white
    glClearColor(1.0,1.0,1.0,0.0);
    glShadeModel(GL_SMOOTH);

    // specify the material parameters
    glMaterialfv(GL_FRONT,GL_SPECULAR,mat_specular);
    glMaterialfv(GL_FRONT,GL_SHININESS,mat_shininess);

    // set light parameters
    glLightfv(GL_LIGHT0,GL_POSITION, light_position);
    glLightfv(GL_LIGHT0,GL_AMBIENT, light_ambient);
    glLightfv(GL_LIGHT0,GL_DIFFUSE, light_diffuse);
    glLightfv(GL_LIGHT0,GL_SPECULAR, light_specular);

    // enable light
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);

    // enable depth test
    glEnable(GL_DEPTH_TEST);
  }

// display call-back function
// this function draw a solid sphere on screen
  void display(void)
  {
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
    glutSolidSphere(1.0,80,64);
    glFlush();
  }
  // window resize call-back function
  void reshape(int w, int h)
  {
    // set viewpoort
    glViewport(0,0,(GLsizei) w,(GLsizei) h);

    // projection matrix
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();

    // perspective transformation
    if(w<=h)
      glOrtho(-1.5,1.5,-1.5*(GLfloat)h/(GLfloat)w,1.5*(GLfloat)h/(GLfloat)w,-
10.0,10.0);
    else
      glOrtho(-1.5*(GLfloat)h/(GLfloat)w,1.5*(GLfloat)h/(GLfloat)w,-1.5,1.5,-
10.0,10.0);

    // modelview matrix
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
  }

  // main entry
  int main(int argc,char**argv)
  {
    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB|GLUT_DEPTH);
    glutInitWindowSize(500,500);
    glutInitWindowPosition(100,100);
    glutCreateWindow("OpenGL Test");
    initGLenv();
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutMainLoop();
    return 0;
  }
```

The output of the code is shown in Figure Appendix.1.

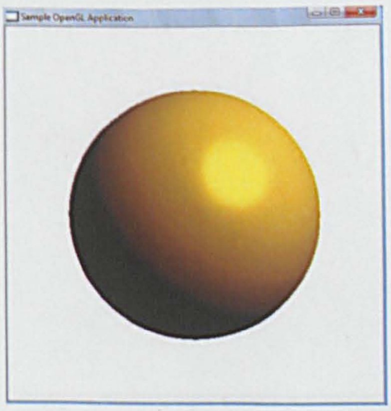


Figure Appendix.1 Output of an example OpenGL application

C. Homogeneous Coordinates

In CG, the homogeneous coordinates are used to represent the linear transformations uniformly. Generally speaking, the concept of homogeneous coordinates is to use n+1 elements to denote a n elements vector. For example, the points on 2D image plane are denoted as $\boldsymbol{P}=(x,y)^T$ in Cartesian coordinates system. The homogeneous coordinates of point $\boldsymbol{P}$ is $\boldsymbol{P}_{homo}=(x,y,1)^T$.

In OpenGL, all points in 3D world are represented by homogeneous coordinates. It is because the transformation of rotation, scale, and translation can be written as the matrix multiplication form.

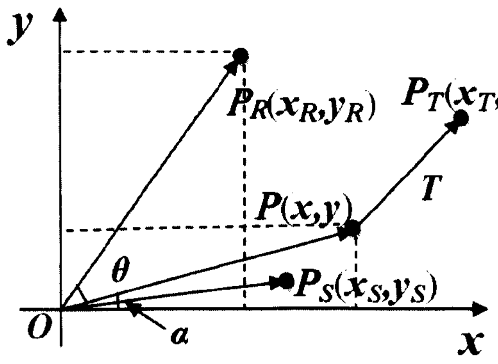


Figure Appendix.2 Transformations in Cartesian coordinate system

As shown in Figure Appendix.2, in Cartesian coordinate system, the original position of point $\boldsymbol{P}$ is (x,y), the new positions after rotating by angle θ, scaling by S_x and S_y, translating by vector $\boldsymbol{T}$ are $\boldsymbol{P_R}$, $\boldsymbol{P_S}$, $\boldsymbol{P_T}$ respectively, the corresponding matrices of these transformations are $\boldsymbol{M_R}$, $\boldsymbol{M_S}$, and $\boldsymbol{M_T}$. These transformations can be computed as:

(1) Rotation

The geometric relation between $\boldsymbol{P}$ and $\boldsymbol{P'}$ during rotation is: $|\boldsymbol{OP}|=|\boldsymbol{OP'}|$. Thus,

$$\begin{aligned} x_R &= |\boldsymbol{OP}| \cdot cos(\alpha+\theta) \\ &= |\boldsymbol{OP}| \cdot (cos\theta \cdot cos\alpha - sin\theta \cdot sin\alpha) \\ &= xcos\theta - ysin\theta \\ y_R &= |\boldsymbol{OP}| \cdot sin(\alpha+\theta) \\ &= |\boldsymbol{OP}| \cdot (sin\alpha \cdot cos\theta + cos\alpha \cdot sin\theta) \\ &= xsin\theta + ycos\theta \end{aligned}$$

The corresponding matrix from of transformation is:

$$\begin{pmatrix} x_R \\ y_R \end{pmatrix} = \begin{pmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \boldsymbol{M}_R\boldsymbol{P}.$$

(2) Scale

If the coefficients along x and y axes are S_x and S_y. Then the following relationship holds:

$$x_S = S_x x$$
$$y_S = S_y y \cdot$$

The matrix form is:

$$\begin{pmatrix} x_S \\ y_S \end{pmatrix} = \begin{pmatrix} S_x & 0 \\ 0 & S_y \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \boldsymbol{M_S P}.$$

(3) Translation

If the point $\boldsymbol{P}$ translates to $\boldsymbol{P_T}$ by a translation vector $\boldsymbol{T} = (T_x, T_y)$, the calculation of the new position is:

$$x_T = x + T_x$$
$$y_T = y + T_y \cdot$$

The matrix form is:

$$\begin{pmatrix} x_T \\ y_T \end{pmatrix} = \begin{pmatrix} T_x \\ T_y \end{pmatrix} + \begin{pmatrix} x \\ y \end{pmatrix} = \boldsymbol{M_T} + \boldsymbol{P}.$$

As shown above, the rotation and scale can be represented similarly by matrix multiplication. However, the translation cannot. Homogeneous coordinate can deal with this issue ideally. By introducing the extra element in the vector, these three translations can be written as:

(1) Rotation

$$\begin{pmatrix} x_R \\ y_R \\ 1 \end{pmatrix} = \begin{pmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}.$$

(2) Scale

$$\begin{pmatrix} x_S \\ y_S \\ 1 \end{pmatrix} = \begin{pmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}.$$

(3) Translation

$$\begin{pmatrix} x_T \\ y_T \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & T_x \\ 0 & 1 & T_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}.$$

More generally, in the 3D space, the affine transformations can be defined by a single matrix:

$$\boldsymbol{M} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & t_x \\ a_{21} & a_{22} & a_{23} & t_y \\ a_{31} & a_{32} & a_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{A} & \boldsymbol{T} \\ 0 & 1 \end{pmatrix}$$

Where: 3×3 matrix A holds all parameters for rotation and scale, $\boldsymbol{T}$ is the translation vector in 3D space.

It is easy to verify that the effect of continuous transformations is superimposed. By using homogeneous coordinates, they equal to multiple the transformation matrices one after another, namely $\boldsymbol{P'} = \boldsymbol{M}_n \ldots \boldsymbol{M}_2 \times \boldsymbol{M}_1 \times \boldsymbol{P}$.

D. GUI Interface

(1) Menus

There are four main menus items: File, View, Move, and Help. Each menu provides a group of operations to control the generic 3D face model or set the environment parameters.

(a) File Menu

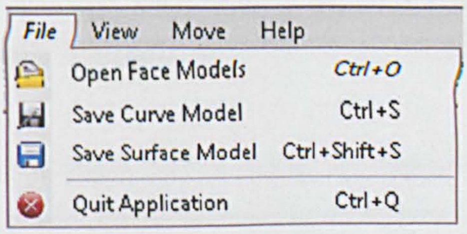


Figure Appendix.3 File menu

The File menu includes four items, with each assigned with the accelerated shortcut keys (Figure Appendix.3). By invoking the "Open Face Model" menu item, an open file dialogue will appear for users to choose from either *ZCM* or *ZSM* data files. File structures have been defined to represent store the data of curves face model and surfaces face model.

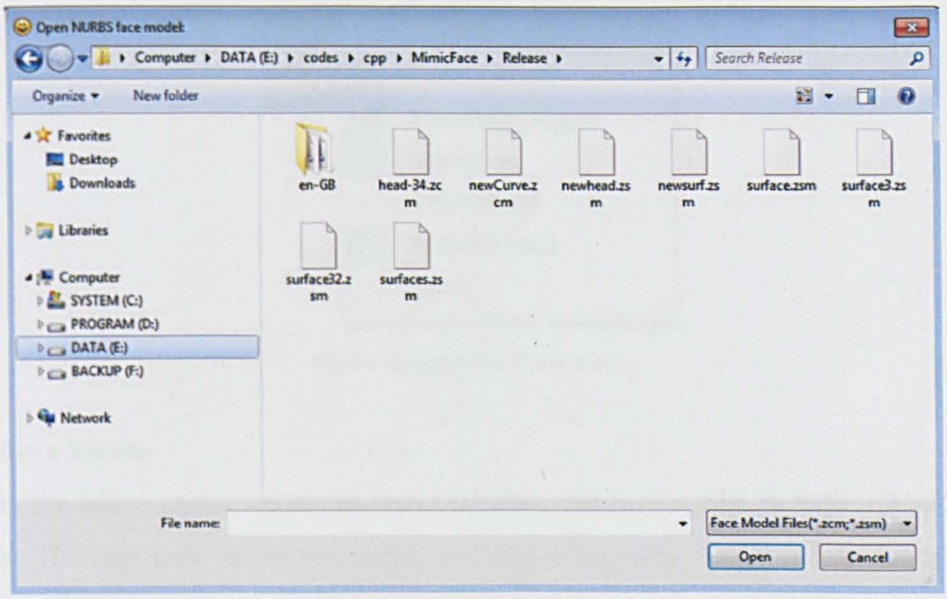
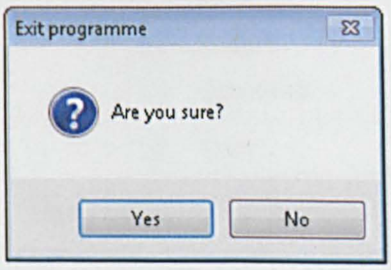


Figure Appendix.4 Open face model dialogue

The second and third items are used to save the current curve model or surface model. Similar to the open file dialogue, each of them has different file type filters to save specific types of file (Figure Appendix.4).

The last item under the File menu is used to exit the application. After clicking it, a message box appears to ask for confirmation (Figure Appendix.5).


Exit programme

Are you sure?

Figure Appendix.5 Exit application

(b) View Menu

The View menu includes six items that set several options for the face model rendering environment, such as light, axes, control points, and bounding polygon display flags. The application offers choices to enable and disable curve model and surface model for convenience (Figure Appendix.6).

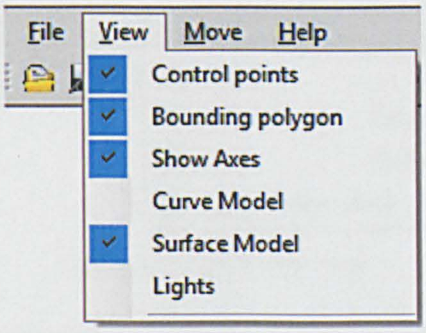


Figure Appendix.6 View menu

(c) Move Menu

Under the Move menu, users can move or scale the face model in both 3D and 2D views. The first item under this menu is used for toggling 3D/2D manipulation. By checking this item, the moving and scale controls take effect in the 3D view window on the left; otherwise, they affect the display in the small 2D view window. The last item can be used to reset these options to the initialised status (Figure Appendix.7).

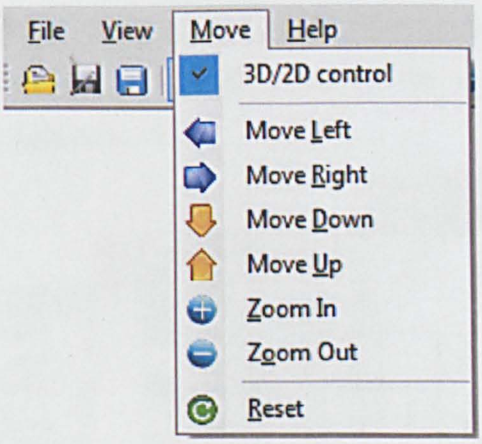


Figure Appendix.7 Move menu

(d) Help Menu

This menu provides the usage and FAQ of the application in detail. The information of the application and the author can be found in the last item (Figure Appendix.8 and Figure Appendix.9).

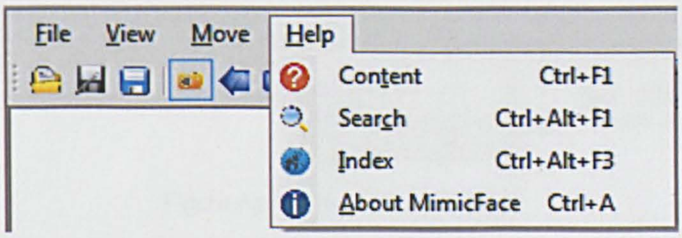
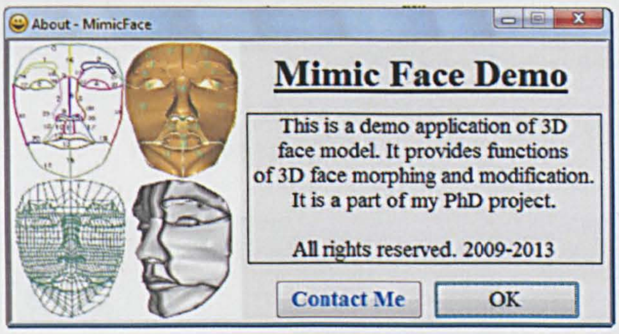


Figure Appendix.8 Help menu


About - MimicFace

Mimic Face Demo

This is a demo application of 3D face model. It provides functions of 3D face morphing and modification. It is a part of my PhD project.

All rights reserved. 2009-2013

Figure Appendix.9 About dialogue of the demo application

(2) Toolbar Items

The toolbar items provide a group of elements for fast access of menus functions. The correspondences between the main items in the menus and toolbar are demonstrated in Figure Appendix.10.

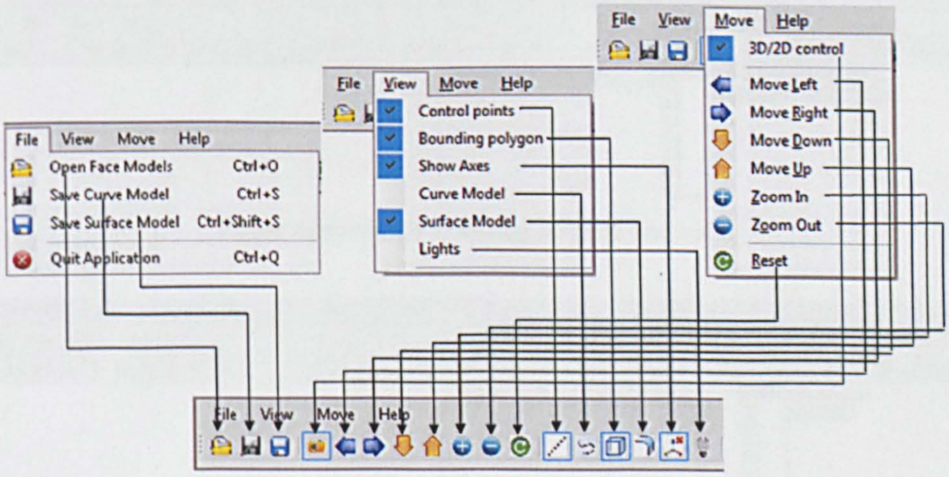


Figure Appendix.10 Correspondences between items in menus and toolbar

Detailed toolbar functions are listed in this section from Figure Appendix.11~Figure Appendix.16.

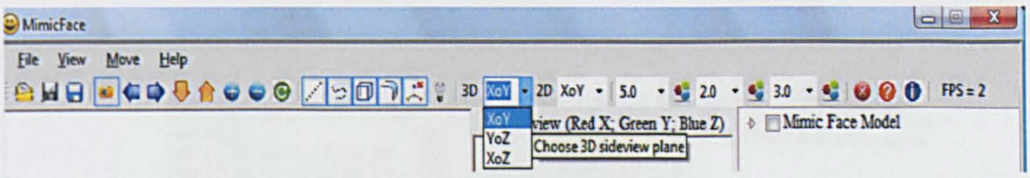


Figure Appendix.11 3D view plane

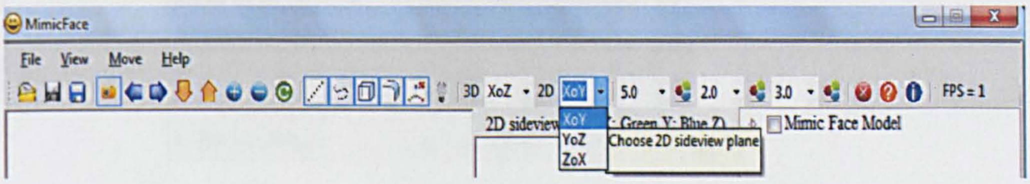


Figure Appendix.12 2D view plane

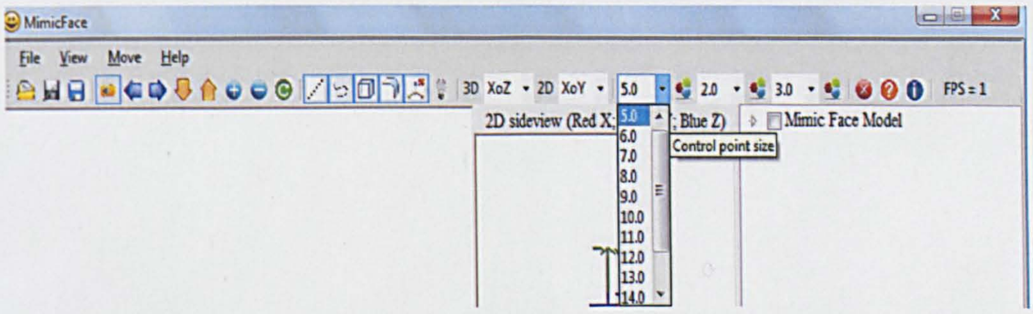


Figure Appendix.13 Control points size

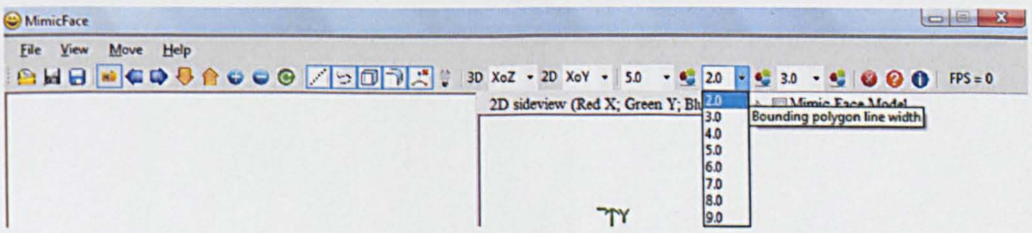

MimicFace
File View Move Help
3D XoZ 2D XoY 5.0 2.0 3.0 FPS = 0
2D sideview (Red X; Green Y; Blu
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
Bounding polygon line width
Y

Figure Appendix.14 Bounding polygon line width

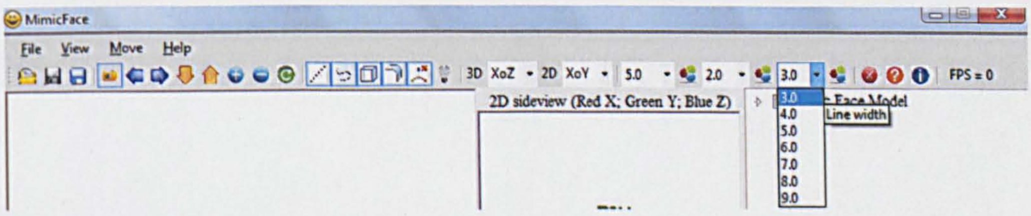

MimicFace
File View Move Help
3D XoZ 2D XoY 5.0 2.0 3.0 FPS = 0
2D sideview (Red X; Green Y; Blue Z)
3.0
4.0
5.0
6.0
7.0
8.0
9.0
Line width
Face Model

Figure Appendix.15 NURBS curve width

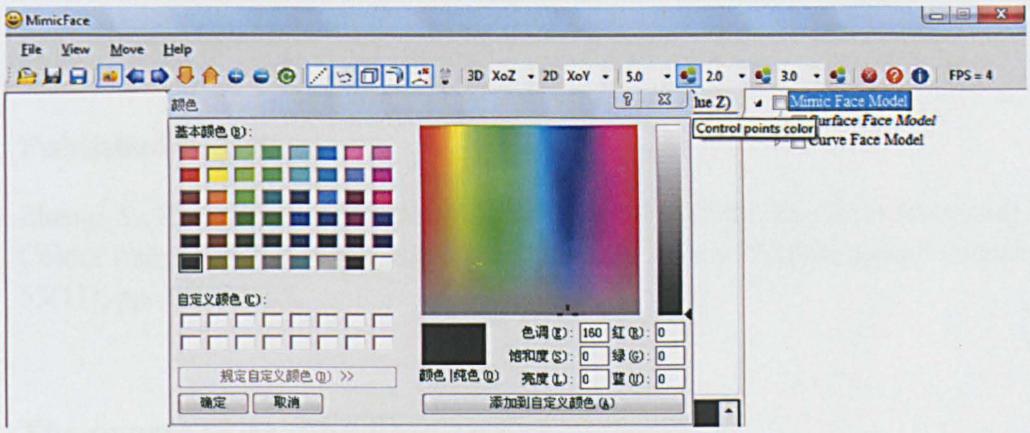


Figure Appendix.16 Control points colour changing

E. Publications

Published paper:

Zhang, X., Zhang, S. and Hapeshi, K. (2010) 'A New Method for Face Detection in Colour Images for Emotional Bio-Robots', *Science China (Technological Sciences)*, 53(11), pp. 2983-2988.

The papers to be submitted and under writing:

Review on Image Processing Theories and Techniques for Face Detection, to be submitted to Computer & Graphics

NURBS and Its Application to 3D Human Face Surface Modelling, to be submitted to Journal of Bionics Engineering

A New Feature-based Generic 3D Human Face Model for Emotional Bio-robots, to be submitted to Journal of Bionics Engineering