Editors

Mohamed Daoudi

Anuj Srivastava

Remco Veltkamp

3D Face Modeling, Analysis and Recognition

WILEY

Contents

3D FACE MODELING, ANALYSIS AND RECOGNITION

Editors

Mohamed Daoudi
TELECOM Lille 1/LIFL, France

Anuj Srivastava
Florida State University, USA

Remco Veltkamp
Utrecht University, The Netherlands

WILEY

This edition first published 2013

Registered office
John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, United Kingdom

For details of our global editorial offices, for customer services and for information about how to apply for permission to reuse the copyright material in this book please see our website at www.wiley.com.

Library of Congress Cataloging-in-Publication Data

Daoudi, Mohamed, 1964–
3D face modeling, analysis, and recognition / Mohamed Daoudi, Anuj Srivastava, Remco Veltkamp.
pages cm
Includes bibliographical references and index.
ISBN 978-0-47066641-8 (cloth)
1. Three-dimensional imaging. 2. Human face recognition (Computer science) 3. Face--Computer simulation. I. Srivastava, Anuj, 1968– II. Veltkamp, Remco C., 1963– III. Title. IV. Title: Three dimensional face modeling, analysis, and recognition.
TA1637.D365 2013
006.6′93–dc23
2013005799

A catalogue record for this book is available from the British Library

ISBN: 9780470666418

Preface

Introduction

The human face has long been an object of fascination, investigation, and analysis. It is so familiar to our visual cognition system that we can recognize a person's face in difficult visual environments, that is, under arbitrary lighting conditions and pose variation. A common question to many researchers is whether a computer vision system can process and analyze 3D face as the human vision system does. In addition to understanding human cognition, there is also increasing interest in analyzing shapes of facial surfaces for developing applications such as biometrics, human–computer interaction (HCI), facial surgery, video communications, and 3D animation.

Because facial *biometrics* is natural, contact free, nonintrusive, and of psychological interest, it has emerged as a popular modality in the biometrics community. Unfortunately, the technology for 2D image-based face recognition still faces difficult challenges. Face recognition is made difficult by data variability caused by pose variations, lighting conditions, occlusions, and facial expressions. Because of the robustness of 3D observations to lighting conditions and pose variations, face recognition using shapes of facial surfaces has become a major research area in the last few years. Many of the state-of-the-art methods have focused on the variability caused by facial deformations, for example, those caused by face expressions, and have proposed methods that are robust to such shape variations.

Another important use of 3D face analysis is in the area of *computer interaction*. As machines become more and more involved in everyday human life and take on increasing roles in both their living and work spaces, they need to become more intelligent in terms of understanding human moods and emotions. Embedding these machines with a system capable of recognizing human emotions and mental states is precisely what the HCI research community is focused on. Facial expression recognition is a challenging task that has seen a growing interest within the research community, impacting important applications in fields related to HCI. Toward building human-like emotionally intelligent HCI devices, scientists are trying to include identifiers of the human emotional state in such systems. Recent developments in 3D acquisition sensors

have made 3D data more readily available. Such data help alleviate problems inherent in 2D data such as illumination, pose, and scale variations as well as low resolution.

The interest in 3D facial shape analysis is fueled by the recent advent of cheaper and lighter scanners that can provide high resolution measurements of both geometry and texture of human facial surfaces. One general goal here is to develop computational tools for analyzing 3D face data. In particular, there is interest in quantifiably comparing the shapes of facial surfaces. This can be used to recognize human beings according to their facial shapes, to measure changes in a facial shape following a surgery, or to study/capture the variations in facial shapes during conversations and expressions of emotions. Accordingly, the *main theme of this book is to develop computational frameworks for analyzing shapes of facial surfaces*. In this book, we use some basic and some advanced tools from differential geometry, Riemannian geometry, algebra, statistics, and computer science to develop the desired algorithms.

Scope of the book

This book, which focuses on 3D face modeling, processing, and applications, is divided into five chapters.

Chapter 1 provides a brief overview of successful ideas in the literature, starting with some background material and important basic ideas. In particular, the principles of depth from triangulation and shape from shading are explained first. Then, an original 3D face (static or dynamic) modeling-guided taxonomy is proposed. Next, a survey of successful approaches that have led to commercial systems is given in accordance with the proposed taxonomy. Finally, a general review of these approaches according to factors that are intrinsic factors (spatial and temporal resolutions, depth accuracy, sensor cost, etc.) and extrinsic (motion speed, illumination changes, face details, intrusion and need for user cooperation, etc.) are provided.

Chapter 2 discusses the state of the art in 3D surface features for the recognition of the human face. Particular emphasis is laid on the most prominent and recent contributions. The features extracted from 3D facial surfaces serve as means for dimensionality reduction of surface data and for facilitating the task of face recognition. The complexity of extraction, descriptiveness, and robustness of features directly affect the overall accuracy, performance, and robustness of

the 3D recognition system.

Chapter 3 presents a novel geometric framework for analyzing 3D faces, with specific goals of comparing, matching, and averaging their shapes. In this framework, facial surfaces are represented by radial curves emanating from the nose tips. These curves, in turn, are compared using elastic shape analysis to develop a Riemannian framework for full facial surfaces. This representation, along with the elastic Riemannian metric, seems natural for measuring facial deformations and is robust to data issues such as large facial expressions. One difficulty in extracting *facial curves* from the surface of 3D face scans is related to the presence of noise. A possible way to smooth the effect of the noise without losing the effectiveness of representations is to consider aggregates of facial curves, as opposed to individual curves, called *iso-geodesic stripes*.

Chapter 4 presents an automatic and efficient method to fit a statistical deformation model of the human face to 3D scan data. In a global-to-local fitting scheme, the shape parameters of this model are optimized such that the produced instance of the model accurately fits the 3D scan data of the input face. To increase the expressiveness of the model and to produce a tighter fit of the model, the method fits a set of predefined face components and blends these components afterwards. In the case that a face cannot be modeled, the automatically acquired model coefficients are unreliable, which hinders the automatic recognition. Therefore, we present a bootstrapping algorithm to automatically enhance a 3D morphable face model with new face data. The accurately generated face instances are manifold meshes without noise and holes, and can be effectively used for 3D face recognition. The results show that model coefficient based face matching outperforms contour curve and landmark based face matching, and is more time efficient than contour curve matching.

Although there have been many research efforts in the area of 3D face analysis in the last few years, the development of potential applications and exploitation of face recognition tools is still in its infancy. Chapter 5 summarizes recent trends in 3D face analysis with particular emphasis on the application techniques introduced and discussed in the previous chapters. The chapter discusses how 3D face analysis has been used to improve face recognition in the presence of facial expressions and missing parts, and how 3D techniques are now being extended to process dynamic sequences of 3D face scans for the purpose of facial expression recognition.

We hope that this will serve as a good reference book for researchers and students interested in this field.

Mohamed Daoudi, *TELECOM Lille 1/LIFL, France*
Anuj Srivastava, *Florida State University, USA*
Remco Veltkamp, *Utrecht University, The Netherlands*

List of Contributors

Faisal Radhi M. Al-Osaimi, Department of Computer Engineering, College of Computer & Information Systems, Umm Al-Qura University, Saudi Arabia

Mohsen Ardabilian, Ecole Centrale de Lyon, Département Mathématiques – Informatique, France

Boulbaba Ben Amor, TELECOM Lille1, France

Mohammed Bennamoun, School of Computer Science & Software Engineering, The University of Western Australia, Australia

Stefano Berretti, Dipartimento di Sistemi e Informatica Università degli Studi di Firenze, Italy

Alberto del Bimbo, Dipartimento di Sistemi e Informatica, Università degli Studi di Firenze, Italy

Liming Chen, Ecole Centrale de Lyon, Département Mathématiques Informatique, France

Mohamed Daoudi, TELECOM Lille1, France

Hassen Drira, TELECOM Lille1, France

Frank B. ter Haar, TNO, Intelligent Imaging, The Netherlands

Pietro Pala, Dipartimento di Sistemi e Informatica, Università di Firenze, Italy

Anuj Srivastava, Department of Statistics, Florida State University, USA

Remco Veltkamp, Department of Information and Computing Sciences, Universiteit Utrecht, The Netherlands

1

3D Face Modeling

Boulbaba Ben Amor,[1] Mohsen Ardabilian,[2] and Liming Chen[2]

[1]Institut Mines-Télécom/Télécom Lille 1, France

[2]Ecole Centrale de Lyon, France

Acquiring, modeling, and synthesizing realistic 3D human faces and their dynamics have emerged as an active research topic in the border area between the computer vision and computer graphics fields of research. This has resulted in a plethora of different acquisition systems and processing pipelines that share many fundamental concepts as well as specific implementation details. The research community has investigated the possibility of targeting either end-to-end consumer-level or professional-level applications, such as facial geometry acquisition for 3D-based biometrics and its dynamics capturing for expression cloning or performance capture and, more recently, for 4D expression analysis and recognition. Despite the rich literature, reproducing realistic human faces remains a distant goal because the challenges that face 3D face modeling are still open. These challenges include the motion speed of the face when conveying expressions, the variabilities in lighting conditions, and pose. In addition, human beings are very sensitive to facial appearance and quickly sense any anomalies in 3D geometry or dynamics of faces. The techniques developed in this field attempt to recover facial 3D shapes from camera(s) and reproduce their actions. Consequently, they seek to answer the following questions:

- How can one recover the facial shapes under pose and illumination variations?
- How can one synthesize realistic dynamics from the obtained 3D shape sequences?

This chapter provides a brief overview of the most successful existing methods in the literature by first introducing basics and background material essential to understand them. To this end, instead of the *classical* passive/active taxonomy of

3D reconstruction techniques, we propose here to categorize approaches according to whether they are able to acquire faces in action or they can only capture them in a static state. Thus, this chapter is preliminary to the following chapters that use static or dynamic facial data for face analysis, recognition, and expression recognition.

1.1 Challenges and Taxonomy of Techniques

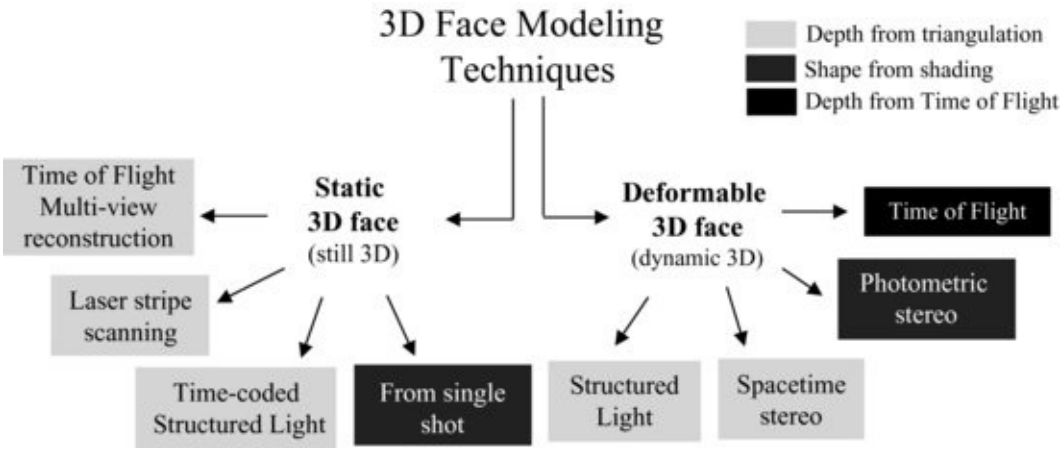
Capturing and processing human geometry is at the core of several applications. To work on 3D faces, one must first be able to recover their shapes. In the literature, several acquisition techniques exist that are either dedicated to specific objects or are general. Usually accompanied by geometric modeling tools and post-processing of 3D entities (3D point clouds, 3D mesh, volume, etc.), these techniques provide complete solutions for 3D full object reconstruction. The acquisition quality is mainly linked to the accuracy of recovering the *z*-coordinate (called depth information). It is characterized by loyalty reconstruction, in other words, by data quality, the density of 3D face models, details preservation (regions showing changes in shapes), *etc.* Other important criteria are the acquisition time, the ease of use, and the sensor's cost. In what follows, we report the main *extrinsic* and *intrinsic* factors which could influence the modeling process.

- *Extrinsic factors*. They are related to the environmental conditions of the acquisition and the face itself. In fact, human faces are globally similar in terms of the position of main features (eyes, mouth, nose, etc.), but can vary considerably in details across (i) their variabilities due to facial deformations (caused by expressions and mouth opening), subject aging (wrinkles), etc, and (ii) their specific details as skin color, scar tissue, face asymmetry, *etc.* The environmental factors refer to lighting conditions (controlled or ambient) and changes in head pose.
- *Intrinsic factors*. They include sensor cost, its intrusiveness, manner of sensor use (cooperative or not), spatial and/or temporal resolutions, measurement accuracy and the acquisition time, which allows us to capture moving faces or simply faces in static state.

These challenges arise when acquiring static faces as well as when dealing with faces in action. Different applications have different requirements. For instance, in the computer graphics community, the results of performance capture should exhibit a great deal of spatial fidelity and temporal accuracy to be

an authentic reproduction of a real actors' performance. Facial recognition systems, on the other hand, require the accurate capture of person-specific details. The movie industry, for instance, may afford a 3D modeling pipeline system with special purpose hardware and highly specialized sensors that require manual calibration. When deploying a 3D acquisition system for facial recognition at airports and in train stations, however, cost, intrusiveness, and the need of user cooperation, among others, are important factors to consider. In ambient intelligence applications where a user-specific interface is required, facial expression recognition from 3D sequences emerges as a research trend instead of 2D-based techniques, which are sensitive to changes and pose variations. Here, also, sensor cost and its capability to capture facial dynamics are important issues. Figure 1.1 shows a new 3D face modeling-guided taxonomy of existing reconstruction approaches. This taxonomy proposes two categories: The first category targets 3D static face modeling, while the approaches belonging to the second category try to capture facial shapes in action (i.e., in 3D+t domain). In the level below, one finds different approaches based on concepts presented in section 1.2. In static face category, the multiview stereo reconstruction uses the *optical triangulation* principle to recover the depth information of a scene from two or more projections (images). The same mechanism is unconsciously used by our brain to work out how far an object is. The correspondence problem in multiview approaches is solved by looking for pixels that have the same appearance in the set of images. This is known as *stereo-matching* problem. Laser scanners use the *optical triangulation* principle, this time called *active* by replacing one camera with a laser source that emits a stripe in the direction of the object to scan. A second camera from a different viewpoint captures the projected pattern. In addition to one or several cameras, time-coded structured-light techniques use a light source to project on the scene a set of light patterns that are used as *codes* for finding correspondences between stereo images. Thus, they are also based on the *optical triangulation* principle.

Figure 1.1 Taxonomy of 3D face modeling techniques



The moving face modeling category, unlike the first one, needs fast processing for 3D shape recovery, thus, it tolerates scene motion. The structured-light techniques using one complex pattern is one solution. In the same direction, the work called *Spacetime faces* shows remarkable results in dynamic 3D shape modeling, by employing random colored light on the face to solve the stereo matching problem. Time-of-flight-based techniques could be used to recover the dynamic of human body parts such as the faces but with a modest shape accuracy. Recently, photometric stereo has been used to acquire 3D faces because it can recover a dense normal field from a surface. In the following sections, this chapter first gives basic principles shared by the techniques mentioned earlier, then addresses the details of each method.

1.2 Background

In the projective pinhole camera model, a point *P* in the 3D space is imaged into a point *P* on the image plane. *P* is related to *P* with the following formula:

(1.1) $p = MP = KR[I|t]P,$

where *P* and *P* are represented in homogeneous coordinates, *M* is a 3×4 projection matrix, and *I* is the 3×3 identity matrix. *M* can be decomposed into two components: the intrinsic parameters and the extrinsic parameters. Intrinsic parameters relate to the internal parameters of the camera, such as the image coordinates of the principal point, the focal length, pixel shape (its aspect ratio), and the skew. They are represented by the 3×3 upper triangular matrix *K*. Extrinsic (or external) parameters relate to the pose of the camera, defined by the 3×3 rotation matrix *R* and its position *t* with respect to a global coordinate system. *Camera calibration* is the process of estimating the intrinsic and extrinsic parameters of the cameras.

3D reconstruction can be roughly defined as the inverse of the imaging process; given a pixel *P* on one image, 3D reconstruction seeks to find the 3D coordinates of the point *P* that is imaged onto *P*. This is an ill-posed problem because with the inverse imaging process a pixel *P* maps into a ray *v* that starts from the camera center and passes through the pixel *P*. The ray direction $\vec{v}$ can be computed from the camera pose *R* and its intrinsic parameters *K* as follows;

(1.2) $$\vec{v} = \frac{R^{-1}K^{-1}p}{\|R^{-1}K^{-1}p\|}$$

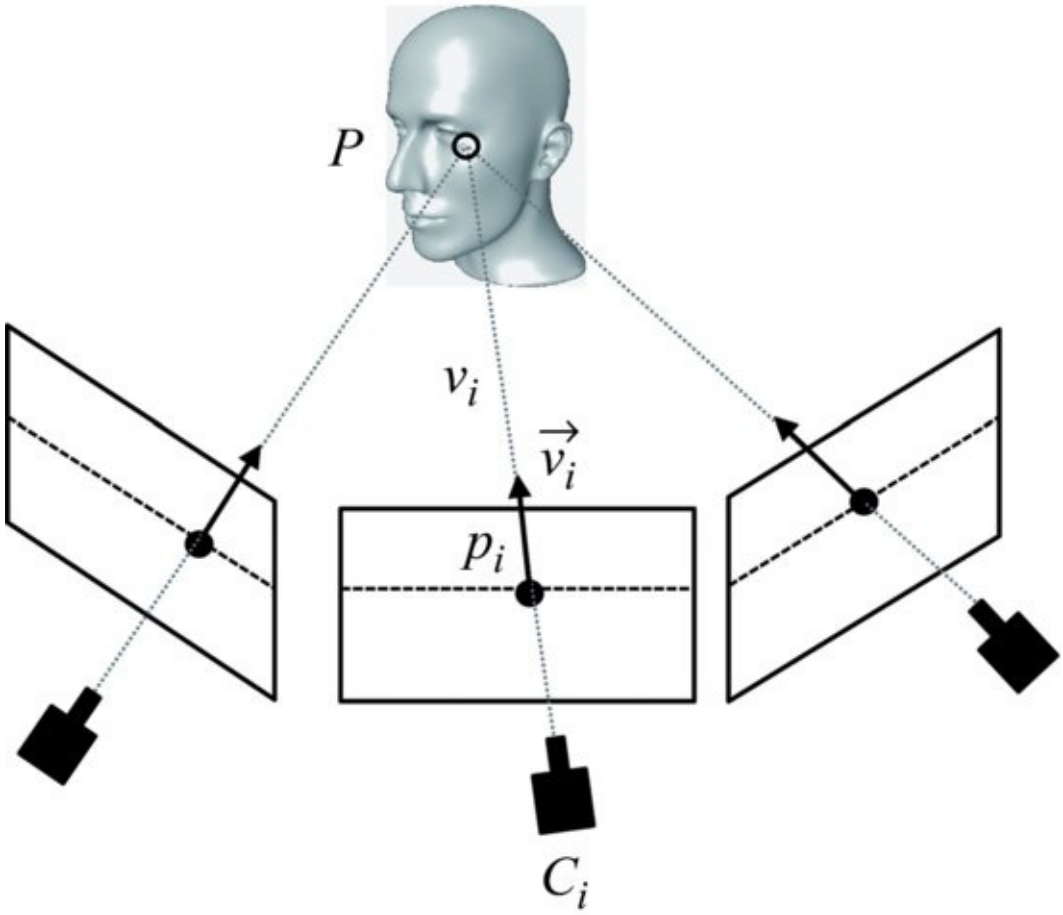
1.2.1 Depth from Triangulation

If *q* is the image of the same 3D point *P* taken by another camera from a different viewing angle, then the 3D coordinates of *P* can be recovered by estimating the intersection of the two rays, v_1 and v_2, that start from the camera centers passing, respectively, through *P* and *q*. This is known as the *optical triangulation* principle. *P* and *q* are called *corresponding* or *matching* pixels because they are the images of the same 3D point *P*.

A 3D point *P* is the intersection of *n*(*n*>1) rays v_i passing through the optical centers c_i of cameras $\{C_i\}$ where $i = 1, \ldots, n$. This can also be referred to *passive optical triangulation*. As illustrated in Figure 1.2, all points on v_i project to p_i, given a set of corresponding pixels p_i captured by the cameras c_i, and their corresponding rays v_i, the 3D location of *P* can be found by intersecting the rays v_i. In practice, however, these rays will often not intersect. Instead, we look for the optimal value of *P* that lies closest to the rays v_i. Mathematically, if K_i, R_i, t_i are the parameters of the camera c_i, where K_i is the 3×3 matrix that contains the intrinsic parameters of the camera and R_i and t_i are the pose of the *I*th camera with respect to the world coordinate system, the rays v_i originating at c_i and passing through p_i are in the direction of $R^{-1}_i K^{-1}_i p_i$. The optimal value of *P* that lies closest to all the rays $\vec{v}_i$, *P* minimizes the distance:

(1.3) $$\|c_j + d_j\vec{v}_j - p\|^2$$

Figure 1.2 Multiview stereo determines the position of a point in space by finding the intersection of the rays v_i passing through the center of projection c_i of the *I*th camera and the projection of the point *P* in each image, p_i



Methods based on the *optical triangulation* need to solve two problems: *(i)* the matching problem, and *(ii)* the reconstruction problem. The correspondence problem consists of finding matching points across the different cameras. Given the corresponding points, the reconstruction problem consists of computing a 3D disparity map of the scene, which is equivalent to the depth map (*z*-coordinate on each pixel). Consequently, the quality of the reconstruction depends crucially on the solution to the correspondence problem. For further reading on *stereo vision* (cameras calibration, stereo matching algorithms, reconstruction, etc.), we refer the reader to download the PDF of the Richard Szeliski's *Computer Vision: Algorithms and Applications* available at http://szeliski.org.[1]

Existing *optical triangulation*-based 3D reconstruction techniques, such as multiview stereo, structured-light techniques, and laser-based scanners, differ in the way the correspondence problem is solved. Multiview stereo reconstruction uses the triangulation principle to recover the depth map of a scene from two or more projections. The same mechanism is unconsciously used by our brain to work out how far an object is. The correspondence problem in stereo vision is solved by looking for pixels that have the same appearance in the set of images.

This is known as *stereo matching*. Structured-light techniques use, in addition to camera(s), a light source to project on the scene a set of light patterns that are used as *codes* for finding correspondences between stereo images. Laser scanners use the triangulation principle by replacing one camera with a laser source that emits a laser ray in the direction of the object to scan. A camera from a different viewpoint captures the projected pattern.

1.2.2 Shape from Shading

Artists have reproduced, in paintings, illusions of depth using lighting and shading. Shape From Shading (SFS) addresses the shape recovery problem from a gradual variation of shading in the image. Image formation is a key ingredient to solve the SFS problem. In the early 1970s, Horn was the first to formulate the SFS problem as that of finding the solution of a nonlinear first-order Partial Differential Equation (PDE) also called the brightness equation. In the 1980s, the authors address the computational part of the problem, directly computing numerical solutions. Bruss and Brooks asked questions about the existence and uniqueness of solutions. According to the Lambertian model of image formation, the gray level at an image pixel depends on the light source direction and surface normal. Thus, the aim is to recover the illumination source and the surface shape at each pixel. According to Horn's formulation of SFS problem, the brightness equation arises as:

(1.4) $$I(x, y) = R(\vec{n}(x, y)),$$

where, (x, y) are the coordinates of a pixel; R, the reflectance map and I the brightness image. Usually, SFS approaches, particularly those dedicated to face shape recovery, adopt the Lambartian property of the surface. In which case, the reflectance map is the cosine of the angle between light vector $\vec{L}(x, y)$ and the normal vector $\vec{n}(x, y)$ to the surface:

(1.5) $$R = \cos(\vec{L}, \vec{n}) = \frac{\vec{L}}{|\vec{L}|} \cdot \frac{\vec{n}}{|\vec{n}|},$$

where R, $\vec{L}$ and $\vec{n}$ depends on (x, y). Since the first SFS technique developed by Horn, many different approaches have emerged; active SFS which requires calibration to simplify the solution finding has achieved impressive results.

1.2.3 Depth from Time of Flight (ToF)

Time of flight provides a direct way to acquire 3-D surface information of

objects or scenes outputting 2.5 D, or depth, images with a real-time capability. The main idea is to estimate the time taken for the light projected by an illumination source to return from the scene or the object surface. This approach usually requires nano-second timing to resolve surface measurements to millimeter accuracy. The object or scene is actively illuminated with a nonvisible light source whose spectrum is usually nonvisible infrared, *e.g.* 780 nm. The intensity of the active signal is modulated by a cosine-shaped signal of frequency *f*. The light signal is assumed to have a constant speed, *c*, and is reflected by the scene or object surface. The distance *d* is estimated from the phase shift θ in radian between the emitted and the reflected signal, respectively:

(1.6) $$d = \frac{c}{2f}\frac{\theta}{2\pi}$$

While conventional imaging sensors consists of multiple photo diodes, arranged within a matrix to provide an image of, e.g., color or gray values, a ToF sensor, for instance a photon mixing device (PMD) sensor, simultaneously acquires a distance value for each pixel in addition to the common intensity (gray) value. Compared with conventional imaging sensors, a PMD sensor is a standard CMOS sensor that benefits from these functional improvements. The chip includes all intelligence, which means that the distance is computed per pixel. In addition, some ToF cameras are equipped with a special pixel-integrated circuit, which guarantees the independence to sunlight influence by the suppression of background illumination (SBI).

1.3 Static 3D Face Modeling

1.3.1 Laser-stripe Scanning

Laser-stripe triangulation uses the well-known optical triangulation described in section 1.2. A laser line is swept across the object where a CCD array camera captures the reflected light, its shape gives the depth information. More formally, as illustrated in Figure 1.3, a slit laser beam, generated by a light projecting optical system, is projected on the object to be measured, and its reflected light is received by a CCD camera for triangulation. Then, 3D distance data for one line of slit light are obtained. By scanning slit light with a *galvanic mirror*, 3D data for the entire object to be measured are obtained. By measuring the angle $2\pi - \theta$, formed by the baseline *d* (distance between the light-receiving optical system and the light-projecting optical system) and by a laser beam to be projected, one

can determine the *z*-coordinate by triangulation. The angle θ is determined by an instruction value of the galvanic mirror. Absolute coordinates for laser beam position on the surface of the object, denoted by *P*, are obtained from congruence conditions of triangles, by

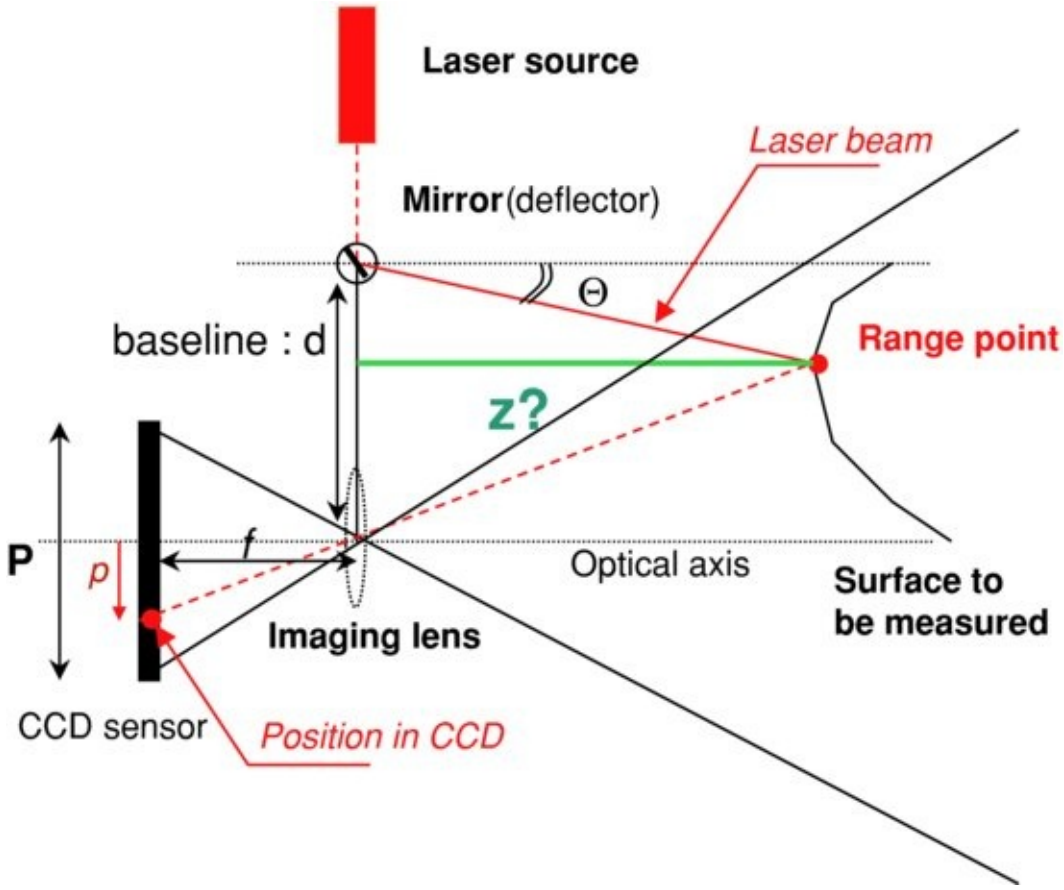
$$\frac{z}{f_0} = \frac{d - z.\tan(\theta)}{p}. \quad (1.7)$$

This gives the *z*-coordinate, by

$$z = \frac{df_0}{p + f_0 \tan(\theta)}. \quad (1.8)$$

Solve question 1 in section 5.5.3 for the proof.

Figure 1.3 Optical triangulation geometry for a laser-stripe based scanner

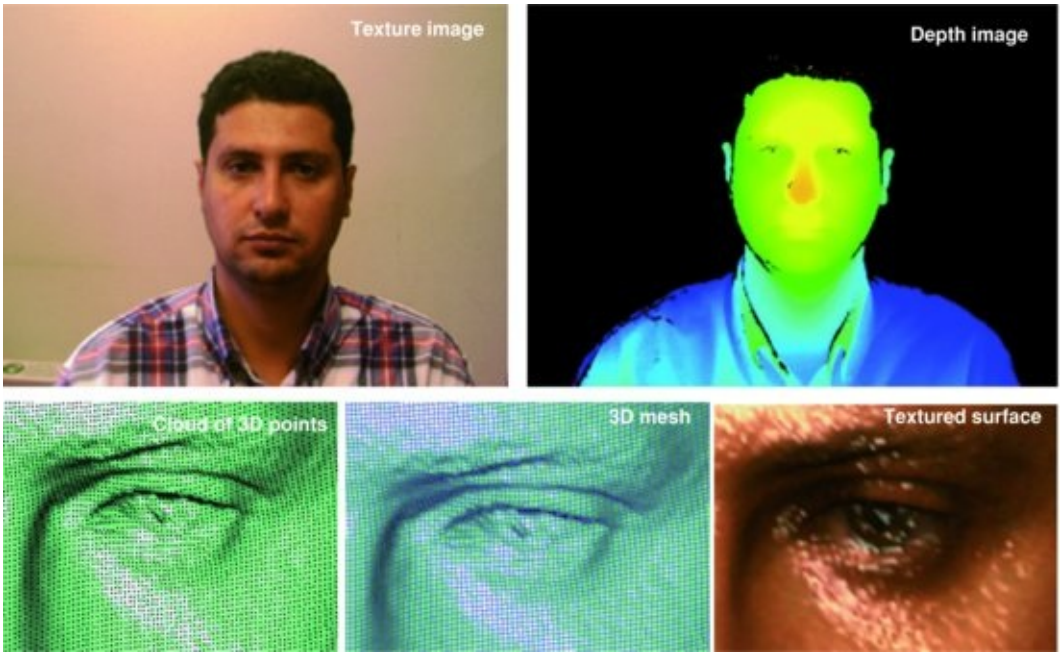


The Charged Couple Device (CCD) is the widely used light-receiving optical system to digitize the point laser image. CCD-based sensors avoid the beam spot reflection and stray light effects and provide more accuracy because of the single-pixel resolution. Another factor that affects the measurement accuracy is the difference in the surface characteristic of the measured object from the calibration surface. Usually calibration should be performed on similar surfaces

to ensure measurement accuracy. Using laser as a light source, this method has proven to be able to provide measurement at a much higher depth range than other passive systems with good discrimination of noise factors. However, this line-by-line measurement technique is relatively slow. The laser-based techniques can give very accurate 3D information for a rigid body even with a large depth. However, this method is time consuming for real measurement since it obtains 3D geometry on a line at a time. The area scanning-based methods such as time-coded structured light (see section 1.3.2) are certainly faster.

An example of acquired face using these technique is given by Figure 1.4. It illustrates the good quality of the reconstruction when office environment acquisition conditions are considered, the subject is distant of 1 m from the sensor and remains stable for a few seconds.

Figure 1.4 One example of 3D face acquisition based on laser stripe scanning (using Minolta VIVID 910). Different representations are given, from the left: texture image, depth image, cloud of 3D points, 3D mesh, and textured shape



1.3.2 Time-coded Structured Light

The most widely used acquisition systems for face are based on structured light by virtue of reliability for recovering complex surface and accuracy. That consists in projecting a light pattern and imaging the illuminated object, a face for instance, from one or more points of view. Correspondences between image points and points of the projected pattern can be easily found. Finally the

decoded points can be triangulated, and depth is recovered. The patterns are designed so that code words are assigned to a set of pixels.

A code word is assigned to a coded pixel to ensure a direct mapping from the code words to the corresponding coordinates of the pixel in the pattern. The code words are numbers and they are mapped in the pattern by using gray levels, color or geometrical representations. Pattern projection techniques can be classified according to their coding strategy: time-multiplexing, neighborhood codification, and direct codification. Time-multiplexing consists in projecting code words as sequence of patterns along time, so the structure of every pattern can be very simple. In spite of increased complexity, neighborhood codification represents the code words in a unique pattern. Finally, direct codification defines a code word for every pixel; equal to the pixel gray level or color.

One of the most commonly exploited strategies is based on temporal coding. In this case, a set of patterns are successively projected onto the measuring surface. The code word for a given pixel is usually formed by the sequence of illumination values for that pixel across the projected patterns. Thus, the codification is called *temporal* because the bits of the code words are multiplexed in time. This kind of pattern can achieve high accuracy in the measurements. This is due to two factors: First, because multiple patterns are projected, the code word basis tends to be small (usually binary) and hence a small set of primitives is used, being easily distinguishable among each other. Second, a coarse-to-fine paradigm is followed, because the position of a pixel is encoded more precisely while the patterns are successively projected.

Figure 1.5 (a) Binary-coded patterns projection for 3D acquisition, (b) *n*-ary-coded coded patterns projection for 3D acquisition

During the three last decades, several techniques based on time-multiplexing have appeared. These techniques can be classified into three categories: binary codes (Figure 1.5**a**), *n*-ary codes (Fig. 1.5**b**), and phase-shifting techniques.

- **Binary codes.** In binary code, only two illumination levels are used. They are coded as 0 and 1. Each pixel of the pattern has its code word formed by the sequence of 0 and 1 corresponding to its value in every projected

pattern. A code word is obtained once the sequence is completed. In practice, illumination source and camera are assumed to be strongly calibrated and hence only one of both pattern axes is encoded. Consequently, black and white strips are used to compose patterns – black corresponding to 0 and white 1, *m* patterns encode 2^m stripes. The maximum number of patterns that can be projected is the resolution in pixels of the projector device; however, because the camera cannot always perceive such narrow stripes, reaching this value is not recommended. It should be noticed that all pixels belonging to a similar stripe in the highest frequency pattern share the same code word. Therefore, before triangulating, it is necessary to calculate either the center of every stripe or the edge between two consecutive stripes. The latter has been shown to be the best choice.

- ***N*-ary codes.** The main drawback of binary codes is the large number of patterns to be projected. However, the fact that only two intensities are projected eases the segmentation of the imaged patterns. The number of patterns can be reduced by increasing the number of intensity levels used to encode the stripes. A first mean is to use multilevel Gray code based on color. This extension of Gray code is based on an alphabet of *n* symbols; each symbol is associated with a certain RGB color. This extended alphabet makes it possible to reduce the number of patterns. For instance, with binary Gray code, *m* patterns are necessary to encode 2^m stripes. With an *n*-ary code, n^m stripes can be coded using the same number of patterns.
- **Phase shifting.** Phase shifting is a well-know principle in the pattern projection approach for 3D surface acquisition. Here, a set of sinusoidal patterns is used. The intensities of a pixel $p(x, y)$ in each pattern is given by:

$$\begin{aligned} I_1(x,y) &= I_0(x,y) + I_{\text{mod}}(x,y)\cos(\phi(x,y) - \theta), \\ I_2(x,y) &= I_0(x,y) + I_{\text{mod}}(x,y)\cos(\phi(x,y)), \\ I_3(x,y) &= I_0(x,y) + I_{\text{mod}}(x,y)\cos(\phi(x,y) + \theta). \end{aligned} \quad (1.9)$$

$I_0(x, y)$ is the background or the texture information, $I_{\text{mod}}(x, y)$ is the signal modulation amplitude, and $I_1(x, y)$, $I_2(x, y)$ and $I_3(x, y)$ are the intensities of the three patterns. $\phi(x, y)$ is the phase value and $\theta = \frac{2\pi}{3}$ is a constant. Three images of the object are used to estimate a wrapped phase value $\hat{\phi}(x, y)$ by:

$$\hat{\phi}(x,y) = \arctan\left\{\sqrt{3}\,\frac{I_1(x,y) - I_3(x,y)}{2\,I_2(x,y) - I_1(x,y) - I_3(x,y)}\right\} \quad (1.10)$$

The wrapped phase is periodic and needs to be unwrapped to obtain an absolute phase value $\phi'(x, y) = \phi(x, y) + 2k\pi$, where *k* is an integer

representing the period or the number of the fringe. Finally the 3D information is recovered based on the projector-camera system configuration. Other pattern configurations of these patterns have been proposed. For instance, Zhang and Yau proposed a real-time 3D shape measurement based on a modified three-step phase-shifting technique (Zhang et al., 2007) (Figure 1.6). They called the modified patterns 2+1 phase shifting approach. According to this approach, the patterns and phase estimation are given by

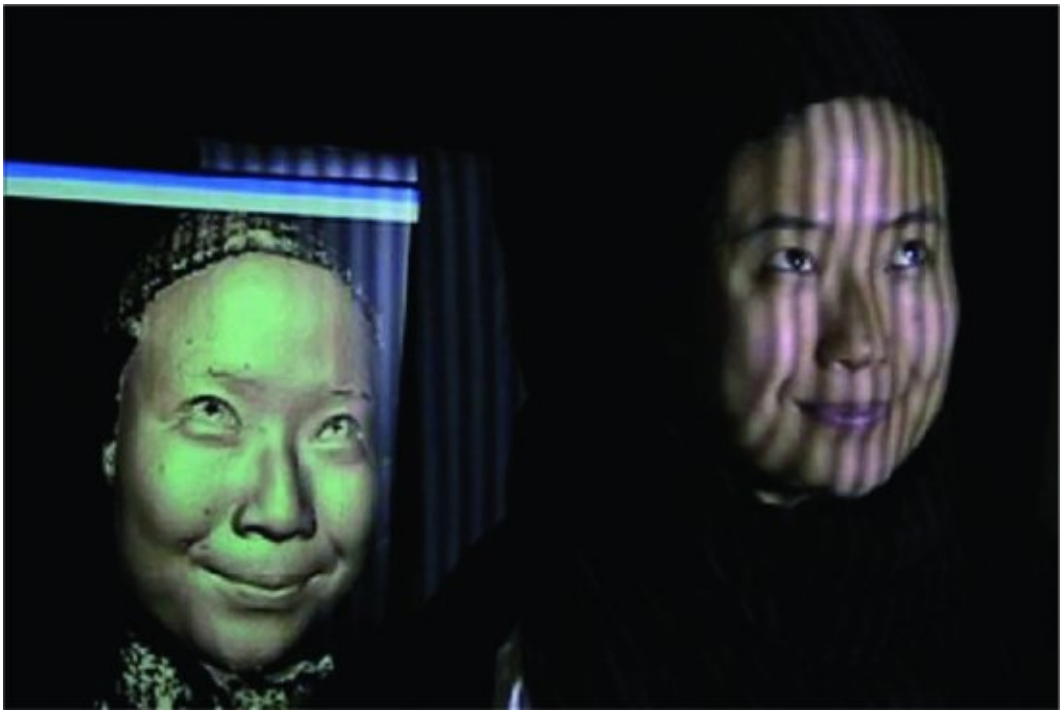
$$I_1(x,y) = I_0(x,y) + I_{\text{mod}}(x,y)\cos\left(\phi(x,y) - \frac{\pi}{2}\right),$$
$$I_2(x,y) = I_0(x,y) + I_{\text{mod}}(x,y)\cos\left(\phi(x,y)\right),$$
$$I_3(x,y) = I_0(x,y), \quad (1.11)$$
$$\hat{\phi}(x,y) = \arctan\left\{\frac{I_1(x,y) - I_3(x,y)}{I_2(x,y) - I_3(x,y)}\right\}. \quad (1.12)$$

A robust phase unwrapping approach called “multilevel quality-guided phase unwrapping algorithm” is also proposed in Zhang *et al.* (2007).

Ouji *et al.* (2011) proposed a cost-effective 3D video acquisition solution with a 3D super-resolution scheme, using three calibrated cameras coupled with a non-calibrated projector device, which is particularly suited to 3D face scanning, that is, rapid, easily movable, and robust to ambient lighting conditions. Their solution is a hybrid stereovision and phase-shifting approach that not only takes advantage of the assets of stereovision and structured light but also overcomes their weaknesses. First, a 3D sparse model is estimated from stereo matching with a fringe-based resolution and a sub-pixel precision. Then projector parameters are automatically estimated through an inline stage. A dense 3D model is recovered by the intrafringe phase estimation, from the two sinusoidal fringe images and a texture image, independently from the left, middle, and right cameras. Finally, the left, middle, and right 3D dense models are fused to produce the final 3D model, which constitutes a spatial super-resolution. In contrast with previous methods, camera-projector calibration and phase-unwrapping stages are avoided.

Figure 1.6 The high-resolution and real-time 3D shape measurement system proposed by Zhang and Yau (2007) is based on the modified 2 + 1 phase-shifting algorithm and particularly adapted for face acquisition. The data acquisition speed is as high as 60 frames per second while the image resolution is 640 × 480 pixels per frame. Here a photograph captured during the experiment is

illustrated. The left side of the image shows the subject, whereas the right side shows the real-time reconstructed geometry



1.3.3 Multiview Static Reconstruction

The aim of multiview stereo (MVS) reconstruction is twofold. Firstly, it allows to reinforce constraints on stereo matching, discard false matches, and increase the precision of good matches. Secondly, spatial arrangement of cameras allows covering the entire face. To reduce the complexity, as well as achieve high quality reconstruction, multiview reconstruction approaches usually proceed in a coarse-to-fine sequence. Finally, multiview approaches involve high resolution images captured in real time, whereas the processing stage requires tens of minutes. MVS scene and object reconstruction approaches can be organized into four categories. The first category operates first by estimating a cost function on a 3D volume and then extracting a surface from this volume. A simple example of this approach is the voxel-coloring algorithm and its variants (Seitz and Dyer, 1997; Treuille et al., 2004). The second category of approaches, based on voxels, level sets, and surface meshes, works by iteratively evolving a surface to decrease or minimize a cost function. For example, from an initial volume, space carving progressively removes inconsistent voxels. Other approaches represent the object as an evolving mesh (Hernandez and Schmitt, 2004; Yu et al., 2006) moving as a function of internal and external forces. In the third category are

image-space methods that estimate a set of depth maps. To ensure a single consistent 3D object interpretation, they enforce consistency constraints between depth maps (Kolmogorov and Zabih, 2002; Gargallo and Sturm, 2005) or merge the set of depth maps into a 3D object as a post process (Narayanan et al., 1998). The final category groups approaches that first extract and matches a set of feature points. A surface is then fitted to the reconstructed features (Morris and Kanade, 2000; Taylor, 2003). Seitz *et al.* (2006) propose an excellent overview and categorization of MVS. 3D face reconstruction approaches use a combination of methods from these categories.

Furukawa and Ponce (2009) proposed a MVS algorithm that outputs accurate models with a fine surface. It implements multiview stereopsis as a simple match, expand, and filter procedure. In the matching step, a set of features localized by Harris operator and difference-of-Gaussians algorithms are matched across multiple views, giving a sparse set of patches associated with salient image regions. From these initial matches, the two next steps are repeated n times (n=3 in experiments). In the expansion step, initial matches are spread to nearby pixels to obtain a dense set of patches. Finally in the filtering step, the visibility constraints are used to discard incorrect matches lying either in front of or behind the observed surface. The MVS approach proposed by Bradley *et al.* (2010) is based on an iterative binocular stereo method to reconstruct seven surface patches independently and to merge into a single high resolution mesh. At this stage, face details and surface texture help guide the stereo algorithm. First, depth maps are created from pairs of adjacent rectified viewpoints. Then the most prominent distortions between the views are compensated by a *scaled-window matching* technique. The resulted depth images are converted to 3D points and fused into a single dense point cloud. A triangular mesh from the initial point cloud is reconstructed over three steps: downsampling, outliers removal, and triangle meshing. Sample reconstruction results of this approach are shown in Figure 1.7.

Figure 1.7 Sample results on 3D modeling algorithm for calibrated multiview stereopsis proposed by Furukawa and Ponce (2010) that outputs a quasi-dense set of rectangular patches covering the surfaces visible in the input images. In each case, one of the input images is shown on the left, along with two views of textured-mapped reconstructed patches and shaded polygonal surfaces.

The 3D face acquisition approach proposed by Beeler *et al.* (2010), which is built on the survey paper, takes inspiration from Furukawa and Ponce (2010). The main difference lies in a refinement formulation. The starting point is the established approach for refining recovered 3D data on the basis of a data-driven photo-consistency term and a surface-smoothing term, which has been research topic. These approaches differ in the use of a second-order anisotropic formulation of the smoothing term, and we argue that it is particularly suited to faces. Camera calibration is achieved in a pre-processing stage.

The run-time system starts with a pyramidal pairwise stereo matching. Results from lower resolutions guide the matching at higher-resolutions. The face is first segmented based on cues of background subtraction and skin color. Images from each camera pair are rectified. An image pyramid is then generated by factor of two downsampling using Gaussian convolution and stopping at approximately 150×150 pixels for the lowest layer. Then a dense matching is established between pairwise neighboring cameras, and each layer of the pyramid is processed as follows: Matches are computed for all pixels on the basis of normalized cross correlation (NCC) over a square window (3×3). The disparity is computed to sub-pixel accuracy and used to constrain the search area in the following layer. For each pixel, smoothness, uniqueness, and ordering constraints are checked, and the pixels that do not fulfill these criteria are reached using the disparity estimated at neighboring pixels. The limited search area ensures smoothness and ordering constraints, but the uniqueness constraint is enforced again by disparity map refinement. The refinement is defined as a

linear combination of a photometric consistency term, d_p, and a surface consistency term, d_s, balanced both by a user-specified smoothness parameter, w_s, and a data-driven parameter, w_p, to ensure that the photometric term has the greatest weight in regions with good feature localization. d_p favors solutions with high NCC, whereas d_s favors smooth solutions. The refinement is performed on the disparity map and later on the surface. Both are implemented as iterative processes.

The refinement results in surface geometry that is smooth across skin pores and fine wrinkles because the disparity change across such a feature is too small to detect. The result is flatness and lack of realism in synthesized views of the face. On the other hand, visual inspection shows the obvious presence of pores and fine wrinkles in the images. This is due to the fact that light reflected by a diffuse surface is related to the integral of the incoming light. In small concavities, such as pores, part of the incoming light is blocked and the point thus appears darker. This has been exploited by various authors (e.g., Glencross et al., 2008)) to infer local geometry variation. In this section, we expose a method to embed this observation into the surface refinement framework. It should be noticed that this refinement is qualitative, and the geometry that is recovered is not metrically correct. However, augmenting the macroscopic geometry with fine scale features does produce a significant improvement in the perceived quality of the reconstructed face geometry.

For the mesoscopic augmentation, only features that are too small to be recovered by the stereo algorithm are interesting. Therefore, first high pass filtered values are computed for all points X using the projection of a Gaussian $\mathcal{N}$:

$$\mu(\mathrm{X}) = \frac{\sum_{c \in \mathcal{V}} \alpha_c \left(\mathcal{I}_c(\mathrm{X}) - \left[\mathcal{N}_{\sum_c} \otimes \mathcal{I}_c \right](\mathrm{X}) \right)}{\sum_{c \in \mathcal{V}} \alpha_c} \tag{1.13}$$

where $\mathcal{V}$ denotes the set of visible cameras, $\sum_c$ the covariance matrix of the projection of the Gaussian $\mathcal{N}$ into camera *c*, and the weighting term α_c is the cosine of the foreshortening angle observed at camera *c*. The variance of the Gaussian $\mathcal{N}$ is chosen such that high spatial frequencies are attenuated. It can either be defined directly on the surface using the known maximum size of the features or in dependence of the matching window *M*. The next steps are based on the assumption that variation in mesoscopic intensity is linked to variation in the geometry. For human skin, this is mostly the case. Spatially bigger skin features tend to be smooth and are thus filtered out. The idea is thus to adapt the

local high frequency geometry of the mesh to the mesoscopic field (X). The geometry should locally form a concavity whenever (X) decreases and a convexity when it increases.

1.4 Dynamic 3D Face Reconstruction

The objective now is to create dynamic models that accurately recover the facial shape and acquire the time-varying behavior of a real person's face. Modeling facial dynamics is essential for several applications such as avatar animation, facial action analysis, and recognition. Compared with a static or quasi-static object (or scene), this is more difficult to achieve because of the required fast processing. Besides, it is the main limitation of the techniques described in Section 1.3. In particular, laser-based scanners and time-coded structured light shape capture techniques do not operate effectively on fast-moving scenes because of the time required for scanning the object when moving or deforming. In this section, we present appropriate techniques designed for moving/deforming face acquisition and post-processing pipeline for performance capture or expression transfer.

1.4.1 Multiview Dynamic Reconstruction

Passive facial reconstruction has received particular attention because of its potential applications in facial animation. Recent research effort has focused on passive multiview stereo (PMVS) for animated face capture sans markers, makeup, active technology, and expensive hardware. A key step toward effective performance capture is to model the structure and motion of the face, which is a highly deformable surface. Furukawa and Ponce (2009) proposed a motion capture approach from video stream that specifically aims at this challenge. Assuming that the instantaneous geometry of the face is represented by a polyhedral mesh with fixed topology, an initial mesh is constructed in the first frame using PMVS software for MVS (Furukawa and Ponce, 2010) and Poisson surface reconstruction software (Kazhdan et al., 2006) for meshing. Then its deformation is captured by tracking its vertices $v_1, \ldots v_n$ over time. The goal of the algorithm is to estimate in each frame f the position v_i^f of each vertex v_i (From now on, v_i^f will be used to denote both the vertex and its position.) Each vertex may or may not be tracked at a given frame, including the first one, allowing the system to handle occlusion, fast motion, and parts of the surface that are not initially visible. The three steps of the tracking algorithm refer to local motion

parameters estimation, global surface deformation, and filtering.

First, at each frame, an approximation of a local surface region around each vertex, by its tangent plane, gives the corresponding local 3D rigid motion with six degrees of freedom. Three parameters encode normal information, while the remaining three contain tangential motion information. Then, on the basis of the estimated local motion parameters, the whole mesh is then deformed by minimizing the sum of the three energy terms.

$$\sum_i \left| v_i^f - \hat{v}_i^f \right|^2 + \eta_1 \left| \left[\zeta_2 \Delta^2 - \zeta_1 \Delta \right] v_i^f \right|^2 + \eta_2 E_r \left(v_i^f \right). \quad (1.14)$$

The first *data* term measures the squared distance between the vertex position v_i^f and the position $\hat{v}_i^f$ estimated by the local estimation process. The second uses the discrete Laplacian operator of a local parameterization of the surface in v_i to enforce smoothness. [The values $\zeta_1 = 0.6$ and $\zeta_2 = 0.4$ are used in all experiments (Furukawa and Ponce, 2009)]. This term is very similar to the Laplacian regularizer used in many other algorithms (Ponce, 2008). The third term is also used for regularization, and it enforces local tangential rigidity with no stretch, shrink, or shear. The total energy is minimized with respect to the 3D positions of all the vertices by a conjugate gradient method. In case of deformable surfaces such as human faces, nonstatic target edge length is computed on the basis of nonrigid tangential deformation from the reference frame to the current one at each vertex. The estimation of the tangential deformation is performed at each frame before starting the motion estimation, and the parameters are fixed within a frame. Thus, the tangential rigidity term $E_r(v_i^f)$ for a vertex v_i^f in the global mesh deformation is given by

$$\sum_{v_j \in N(v_i)} \max \left[0, \left(e_{ij}^f - \hat{e}_{ij}^f \right)^2 - \tau^2 \right], \quad (1.15)$$

which is the sum of squared differences between the actual edge lengths and those predicted from the reference frame to the current frame. The term τ is used to make the penalty zero when the deviation is small so that this regularization term is enforced only when the data term is unreliable and the error is large. In all our experiments, τ is set to be 0.2 times the average edge length of the mesh at the first frame. Figure 1.8 shows some results of motion capture approach proposed in Furukawa and Ponce (2009).

Figure 1.8 The results of motion capture approach, proposed by Furukawa and Ponce (2009), form multiple synchronized video streams based on regularization

adapted to nonrigid tangential deformation. From left to right, a sample input image, reconstructed mesh model, estimated notion and a texture mapped model for one frame with interesting structure/motion for each dataset 1, 2, and 3. The right two columns show the results in another interesting frame. Copyright © 2009, IEEE

Finally after surface deformation, the residuals of the data and tangential terms are used to filter out erroneous motion estimates. Concretely, these values are first smoothed, and a smoothed local motion estimate is deemed an outlier if at least one of the two residuals exceeds a given threshold. These three steps are iterated a couple of times to complete tracking in each frame, the local motion estimation step only being applied to vertices whose parameters have not already been estimated or filtered out.

The face capture framework proposed by Bradley *et al.* (2010) operates without use of markers and consists of three main components: acquisition, multiview reconstruction and geometry, and texture tracking. The acquisition stage uses 14 high definition video cameras arranged in seven binocular stereo pairs. At the multiview reconstruction stage, each pair captures a highly detailed small patch of the face surface under bright ambient light. This stage uses on an iterative binocular stereo method to reconstruct seven surface patches independently that are merged into a single high resolution mesh; the stereo

algorithm is guided by face details providing, roughly, 1 million polygons meshes. First, depth maps are created from pairs of adjacent rectified viewpoints. Observing that the difference in projection between the views causes distortions of the comparison windows, the most prominent distortions of this kind are compensated by a *scaled-window matching* technique. The resulting depth images are converted to 3D points and fused into a single dense point cloud. Then, a triangular mesh from the initial point cloud is reconstructed through three steps: the original point cloud is downsampled using *hierarchical vertex clustering* (Schaefer and Warren, 2003). Outliers and small-scale high frequency noise are removed on the basis of the Plane Fit Criterion proposed by Weyrich *et al.* (2004) and a point normal filtering inspired by Amenta and Kil (2004), respectively. A triangle mesh is generated without introducing excessive smoothing using *lower dimensional triangulation* methods Gopi *et al.* (2000).

At the last stage, in order to consistently track geometry and texture over time, a single reference mesh from the sequence is chosen. A sequence of compatible meshes without holes is explicitly computed. Given the initial per-frame reconstructions G_t, a set of compatible meshes M_t is generated that has the same connectivity as well as explicit vertex correspondence. To create high quality renderings, per-frame texture maps T_t that capture appearance changes, such as wrinkles and sweating of the face, are required. Starting with a single reference mesh M_0, generated by manually cleaning up the first frame G_0, dense optical flow on the video images is computed and used in combination with the initial geometric reconstructions G_t to automatically propagate M_0 through time. At each time step, a high quality 2D face texture T_t from the video images is computed. Drift caused by inevitable optical flow error is detected in the per-frame texture maps and corrected in the geometry. Also, the mapping is guided by an edge-based mouth-tracking process to account the high speed motion while talking.

Beeler *et al.* (2011) extend their MVS face acquisition system, discussed in Section 1.3, to facial motion capture. Their solution, as Bradley's solution, requires no makeup; the temporally varying texture can be derived directly from the captured video. The computation is parallelizable so that long sequences can be reconstructed efficiently using a multicore implementation. The high quality results derive from two innovations. The first is a robust tracking algorithm specifically adapted for short sequences that integrates tracking in image space and uses the integrated result to propagate a single reference mesh to each target frame. The second is to address long sequences, and it employs the "anchor

frame" concept. The latter is based on the observation that a lengthy facial performance contains many frames similar in appearance. One frame is defined as the reference frame. Other frames similar to the reference frame are marked as anchor frames. Finally, the tracker computes the flow from the reference to each anchor independently with a high level of measurement accuracy. The proposed framework operates in five stages:

1. *Stage 1: Computation of Initial Meshes* – Each frame is processed independently to generate a first estimate of the mesh.

2. *Stage 2: Anchoring* – The reference frame is manually identified. Similar frames to the reference frame are detected automatically and labeled as anchor frames.

3. *Stage 3: Image–Space Tracking* – Image pixels are tracked from the reference frame to anchor frames and then sequentially between non-anchor frames and the nearest anchor frame.

4. *Stage 4: Mesh Propagation* – On the basis tracking results from the previous stage, a reference mesh is propagated to all frames in the sequence.

5. *Stage 5: Mesh Refinement* – The initial propagation from Stage 4 is refined to enforce consistency with the image data.

1.4.2 Photometric Stereo

Photometric stereo is a technique in computer vision for estimating the surface normals of objects by observing that object under different lighting conditions. Estimation of face surface normals can be achieved on the basis of photometric stereo assuming that the face is observed under different lighting conditions. For instance, in three-source photometric stereo, three images of the face are given, taken from the same viewpoint and illuminated by three light sources. These light sources emit usually the same light spectrum from three non-coplanar directions. If an orthographic camera model is assumed, the word coordinate system can be aligned so that the *xy* plane coincides with the image plane. *z* axis corresponds to the viewing direction. Hence, the surface in front of the camera can be defined as the height $Z(x, y)$. Now, assuming that ∇z is the gradient of this function with respect to x and y, the vector locally normal to the surface at (x, y) can be defined as

$$n = \frac{1}{\sqrt{1 + |\nabla Z|^2}} \begin{pmatrix} \nabla Z \\ -1 \end{pmatrix}. \quad (1.16)$$

Also, a 2*d* projection operator can be define $P[x] = (x1/x3, x2/x3)$ so that it

follows that $\nabla z = P[\mathbf{n}]$. The pixel intensity $c_i(x, y)$ in the Ith image, for $i = 1, \ldots 3$, can be defined as

(1.17) $$c_i(x, y) = (\mathbf{l}_i^T \mathbf{n}) \int E(\lambda) R(x, y, \lambda) S(\lambda) \, d\lambda,$$

where $\mathbf{l}_i$ is the direction of a light source with spectral distribution $E_i(\lambda)$, illuminating the surface point $(x, y, z(x, y))^T$; $R(x, y, \lambda)$ reflectance function, and $S(\lambda)$ the response of the sensor camera. The value of this integral is known as Albedo ρ, so the pixel intensity can be defined as

(1.18) $$c_i = \mathbf{l}_i^T \rho \mathbf{n}.$$

Using linear constraints of this equation to solve for $\rho\mathbf{n}$ in a least squares sense. The gradient of the height function $\nabla z = P[\rho\mathbf{n}]$ is obtained and integrated to produce the function z. According to three source photometric stereo, when the point is not in shadow with respect to all three lights, three positive intensities c_i can be estimated each of which gives a constraint on $\rho\mathbf{n}$. Thus the following system can be defined as

(1.19) $$\rho\mathbf{n} = \mathbf{L}^{-1}\mathbf{c}.$$

If the point is in shadow, for instance in the 1st image, then the estimated of c_1 cannot be used as constraint. In this case, each equation describes a 3D plane, the intersection of the two remaining constraints is a 3D line given by

(1.20) $$(c_3 \mathbf{l}_2 - c_2 \mathbf{l}_3)^T \mathbf{n} = 0.$$

In a general case, if the point is in shadow in the ith image, this equation can be arranged as

(1.21) $$[\mathbf{c}]_\times^i \mathbf{L}\,\mathbf{n} = \mathbf{0}$$

This equation is derived by Wolff and Angelopoulou (1994) and used for stereo matching in a two view photometric. Fan and Wolff (1997) also used this formulation to perform uncalibrated photometric stereo. Hernandez *et al.* (2011) used that for the first time in a least squares framework to perform three source photometric stereo in the presence of shadows. Figures 1.9 and 1.10 illustrate some reconstruction results with their proposed shading and shape regularization schemes.

Figure 1.9 Two different frames out of a 1000-frame face video sequence Hernandez *et al.* (2011). The left column shows the reconstruction when shadows are ignored. Middle and right columns show the corresponding reconstructions after detecting and compensating for the shadow regions using the shading regularization scheme (middle) and shape regularization scheme

(right). Note the improvement in the regions around the nose reconstruction where strong cast shadows appear (see arrows). Note also how the shape regularization scheme fails to reconstruct some boundary regions (see circle). Copyright © 2011, IEEE

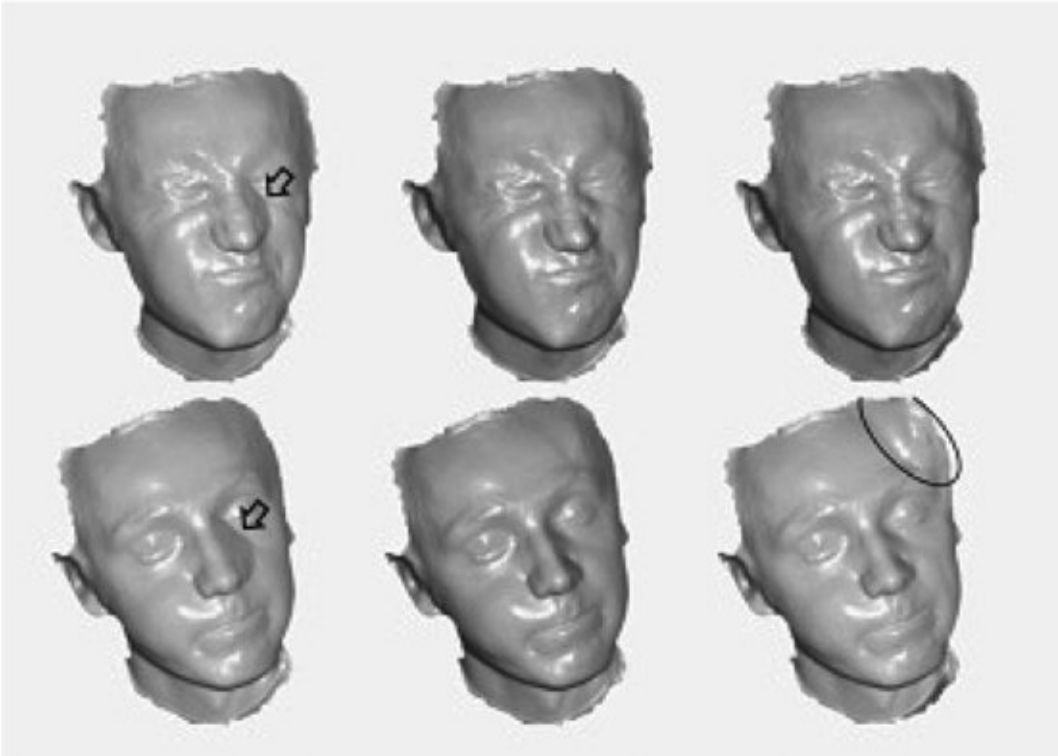
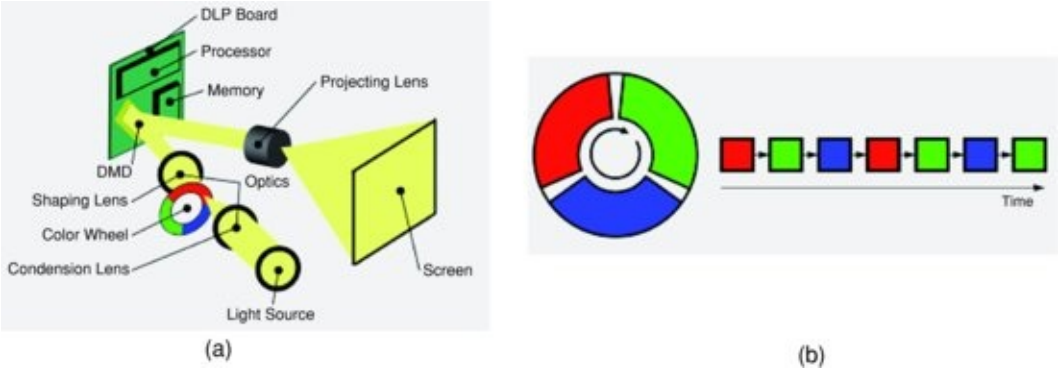


Figure 1.10 Face sequence. Acquisition of 3D facial expressions based on Hernandez *et al.* (2007) and the shadow processing technique described in Hernandez *et al.* (2011). The shadows are processed with the shading regularization scheme. The full video sequence has more than a 1000 frames reconstructed. Copyright © 2011, Springer

Figure 1.11 (a) DLP projecting technology. (b) Single-chip DLP projection mechanism



1.4.3 Structured Light

Structured light–based techniques are reputed to be precise and rapid. However,

3D imaging of moving objects as faces is a challenging task and usually need more sophisticated tools in combination with the existing patterns projection principle. The first strategy consists in patterns projecting and capturing with a synchronized projecting device and camera at a very high rate. The second is to motion modeling and compensation. Finally, the third fuses several 3D models from one or more projector-camera couples to complete them and corrects sensor errors. These three strategies are presented in the following sections. Pan *et al.* (2004) have extensively studied the use of color pattern(s) (RGB) and 3-CCD camera. According to their technique, one single color pattern is used, and the data acquisition is fast. If binary or gray-level patterns are used, they must be switched and projected rapidly so that they are captured in a short period. Rusinkiewicz *et al.* proposed to switch patterns by software (Rusinkiewicz and Levoy, 2001; Rusinkiewicz et al., 2002). To reach fast image switching, Zhang and Yau (2007) proposed to take advantage of the projection mechanism of the single-chip digital-light-processing (DLP) technology. According to their approach, three primary color channels are projected sequentially and repeatedly. This allows capture of three color channel images separately using a synchronized DLP projector device with a digital camera.

A color wheel is a circular disk that spins rapidly. It is composed of R, G, and B filters that color the white light once it passes from in front. Color lights are thus generated. The digital micro-mirror synchronized with the color light, reflects it, and produces three R, G, and B color channel images. Human perception cannot differentiate individual channels as a result of the projection speed. Instead color images are seen. Three phase-shifted sinusoidal patterns are encoded as three primary color channels, R, G, and B of a color image. Three patterns are sent to the single-chip DLP projector from which color filters are removed. A CCD camera is synchronized with the projector and captures each of the three color channels separately into a computer. Unwrapping and phase-to-depth processing steps are applied to the sequence of captured images to recover the depth information. Despite this high speed acquisition, fast motion may still distort the reconstructed phase and hence the reconstructed 3D geometry. Weise *et al.* (2007) proposed to estimate the error in phase shifting, which produces ripples on the 3D reconstructed surface, and to compensate it. Also, this estimation can provide the motion of the reconstructed 3D surface. Three-step phase shifting has been introduced in Section 1.3 where a sinusoidal pattern is shifted by $\frac{2\pi}{3}$ to produce three patterns, the minimum required to recover depth information:

$$I_1(x,y) = I_0(x,y) + I_{\text{mod}}(x,y)\cos(\phi(x,y) - \theta),$$
$$I_2(x,y) = I_0(x,y) + I_{\text{mod}}(x,y)\cos(\phi(x,y)), \text{ and}$$
$$I_3(x,y) = I_0(x,y) + I_{\text{mod}}(x,y)\cos(\phi(x,y) + \theta). \quad (1.22)$$

$I_j,\ j = 1, \ldots 3$, are the recorded intensities, I_0 is the background and I_{mod} is the signal amplitude. $\phi(x,y)$ is the recorded phase value, and θ is the constant phase shift. The phase value corresponds to projector coordinates computed as $\phi = \frac{x_p}{\omega} 2\pi N$, where x_p is the projector *x*-coordinate, ω the horizontal resolution of the projection pattern, and *N* the number of periods of the sinusoidal pattern. The wrapped phase is estimated as

$$\hat{\phi}(x,y) = \arctan\left\{\tan\left(\frac{\theta}{2}\right)\frac{I_1(x,y) - I_3(x,y)}{2I_2(x,y) - I_1(x,y) - I_3(x,y)}\right\}, \quad (1.23)$$

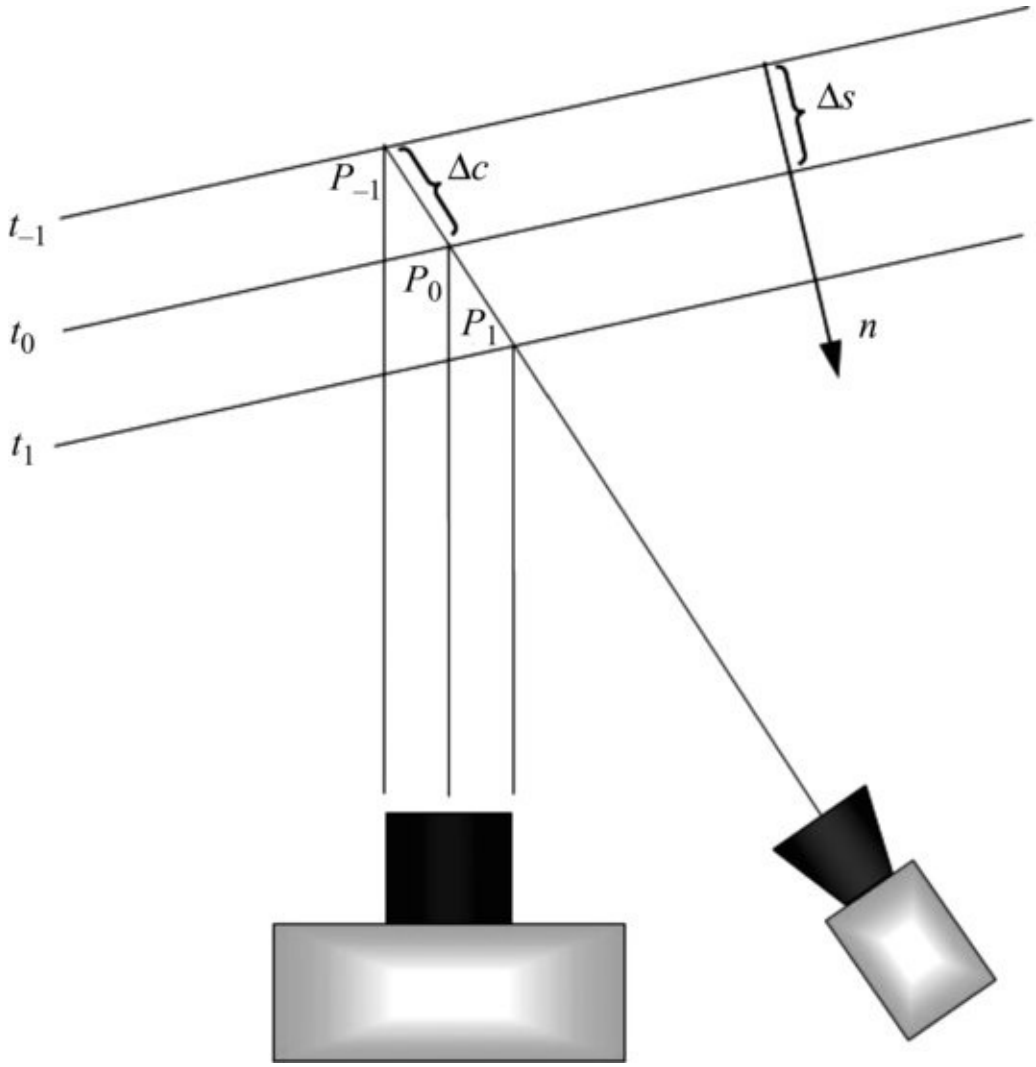
$$I_0(x,y) = \frac{I_1(x,y) + I_2(x,y) + I_3(x,y)}{3}, \text{ and} \quad (1.24)$$

$$I_{\text{mod}}(x,y) = \sqrt{\frac{(I_3(x,y) - I_0(x,y))^2}{3} + \frac{(2I_2(x,y) - I_1(x,y) - I_3(x,y))^2}{9}}. \quad (1.25)$$

Using the estimated phase, the depth is calculated on the basis of triangulation between camera and projection device.

- Motion estimation: Figure 1.12 shows a planar surface and its effects on phase estimation. P_0 is the location observed by the camera at time t_0 and P_1 at t_1. Assuming that $\Delta t = t_0 - t_{-1} = t_1 - t_0$, is a known constant value. If P_0 and P_{-1} are known, the distance vector Δc can be estimated, and thus, the normal motion displacement Δs as the projection of Δc onto the surface normal *n*. From that, the velocity $\frac{\Delta s}{\Delta t}$ of the surface along its normal can be estimated.
- Error estimation and compensation: Now assume P_0, P_{-1}, and P_1 are projector pixel coordinates of P_0, P_{-1}, and P_1. As the camera and projector are mounted horizontally, the projection pattern is invariant vertically, and only the *x*-coordinates are of importance. Hence, the difference between the points in the projection pattern is $\Delta x = p^x_{-1} - p^x_0 \approx p^x_0 - p^x_1$.

Figure 1.12 A planar surface moving towards the camera and its effect on phase estimation (Weise *et al.* (2007)). Here three images are captured at three time steps. The velocity of the surface along its normal is estimated on the basis of the normal motion displacement δ_s as the projection of δ_c, the distance vector, onto the surface normal *n*.



As shown earlier, the intensity of an observed pixel in each of the three images depends on I_0, amplitude I_{mod}, phase $\phi(x, y)$, and shift θ. In case of a planar surface, uniform, and diffuse, I_0 and I_{mod} are locally constant on the observed surface. The shift θ is constant. However, as the observed surface is moving, the $\phi(x, y)$ changes between the three images at three different moments in time. At time t_{-1}, t_0, and t_1 camera observes the intensity as projected by P_{-1}, P_0, and P_1, respectively. By converting Δx into the phase difference we have $\Delta\theta = 2\pi N \frac{\Delta x}{\omega}$; Δx being the width of the projection pattern and N the number of projected wrapped phrase. The true intensities are given by

$$
\begin{aligned}
I_1(x, y) &= I_0(x, y) + I_{\text{mod}}(x, y)\cos(\phi(x, y) - \theta + \Delta\theta), \\
I_2(x, y) &= I_0(x, y) + I_{\text{mod}}(x, y)\cos(\phi(x, y)), \text{ and} \\
I_3(x, y) &= I_0(x, y) + I_{\text{mod}}(x, y)\cos(\phi(x, y) + \theta - \Delta\theta).
\end{aligned} \tag{1.26}
$$

The corrupted shift phase is $\theta - \Delta\theta$. The relative phase error $\Delta\phi$ between

observed distorted phase ϕ_d and true phase ϕ_t is

$$\phi_d = \arctan\left(\tan\left(\frac{\theta - \Delta\theta}{2}\right) g\right), \tag{1.27}$$

$$\phi_t = \arctan\left(\tan\left(\frac{\theta}{2}\right) g\right), \tag{1.28}$$

$$\Delta\phi = \phi_d - \phi_t, \text{ and} \tag{1.29}$$

$$g = \frac{I_1(x, y) - I_3(x, y)}{2\, I_2(x, y) - I_1(x, y) - I_3(x, y)}. \tag{1.30}$$

ϕ_t can be expressed as Taylor expansion of ϕ_d :

$$\phi_t = \phi_d + \sin(2\,\phi_d)\; y - \left(\frac{1}{2}\sin(2\,\phi_d) - \frac{1}{4}\sin(4\,\phi_d)\right) y^2 + O\left(y^3\right), \tag{1.31}$$

where $y = \frac{1}{2}\left(\frac{\tan\left(\frac{\theta - \Delta\theta}{2}\right)}{\tan\left(\frac{\theta}{2}\right)} - 1\right)$, $\Delta\theta = \theta - 2\arctan\left(\tan\left(\frac{\theta}{2}\right)(2y + 1)\right)$. For small motion, only the first-term of the Taylor expansion is enough. In this case, the undistorted phase values can be locally approximated to evolve linearly along a scanline of the camera: $\phi_t(m) = \phi_c + \phi_m\, m$, where *m* is the *x*-coordinate of the pixel. Then a linear least-square fit can be performed in this local neighborhood (7 pixels used in the author's experiments) of each pixel solving for ϕ_c, ϕ_m, and y:

$$\min_{\phi_c,\, \phi_m,\, y} \sum (\phi_c(m) - (m\phi_m(m) - \sin(2\varphi_d(m))\, y))^2. \tag{1.32}$$

For large motion, the first-order Taylor degrades, and instead of using the second-order approximation, a faster solution is to use a simulation that estimates *y* for different values of $\Delta\theta$ and to create a lookup-table (LUT), which is then used to retrieve the true $\Delta\theta$ from an estimated biased *y*. In this case, a median filter is first applied for robustness. Despite high speed acquisition and motion compensation, imperfections essentially due to sensor noise, residual uncompensated motion and acquisition conditions as illumination may persist. To deal with these problems, Ouji *et al.* (2011) proposed to apply a 3D temporal super-resolution for each couple of successive 3D point sets $\mathcal{M}_{t-1}$ and $\mathcal{M}_t$ at time *t*. First, a 3D nonrigid registration is performed. The registration can be modeled as a maximum-likelihood estimation problem because the deformation between two successive 3D faces is nonrigid in general. The coherent point drift (CPD) algorithm, proposed in Andriy Myronenko (2006), is used for the registration the of 3D points set $\mathcal{M}_{t-1}$ with the 3D points set $\mathcal{M}_t$. The CPD algorithm considers the alignment of two point sets $\mathcal{M}_{\text{src}}$ and $\mathcal{M}_{\text{dst}}$ as a

probability density estimation problem and fits the Gaussian Mixture Model (GMM) centroids representing $\mathcal{M}_{\text{src}}$ to the data points of $\mathcal{M}_{\text{dst}}$ by maximizing the likelihood as described in Andriy Myronenko (2006). N_{src} is the number of points of m_{src} and $\mathcal{M}_{\text{src}} = \{sn | n = 1, \ldots, N_{\text{src}}\}$. N_{dst} constitutes the number of points of $\mathcal{M}_{\text{dst}}$ and $\mathcal{M}_{\text{dst}} = \{dn | n = 1, \ldots, N_{\text{dst}}\}$. To create the GMM for $\mathcal{M}_{\text{src}}$, a multivariate Gaussian is centered on each point in $\mathcal{M}_{\text{src}}$. All gaussians share the same isotropic covariance matrix $\sigma^2 I$, I being a 3×3 identity matrix and σ^2 the variance in all directions Andriy Myronenko (2006). Hence the whole point set $\mathcal{M}_{\text{src}}$ can be considered as a GMM with the density $p(d)$ as defined by

(1.33) $$p(d) = \sum_{m=1}^{N_{\text{dst}}} \frac{1}{N_{\text{dst}}} p(d \mid m), \quad d \mid m \propto \mathcal{N}(s_m, \sigma^2 \mathrm{I})$$

The core of the CPD method is forcing GMM centroids to move coherently as a group, which preserves the topological structure of the point sets as described in Andriy Myronenko (2006). The coherence constraint is imposed by explicit re-parameterization of GMM centroids' locations for rigid and affine transformations. For smooth nonrigid transformations such as expression variation, the algorithm imposes the coherence constraint by regularization of the displacement field Myronenko and Song (2010). Once registered, the 3D points sets $\mathcal{M}_{t-1}$ and $\mathcal{M}_t$ and also their corresponding 2D texture images are used as a low resolution data to create a high resolution 3D point set and its corresponding texture. 2D super-resolution technique as proposed in Farsiu *et al.* (2004) is applied, which solves an optimization problem of the form:

(1.34) $$\text{minimize } E_{\text{data}}(H) + E_{\text{regular}}(H).$$

The first term $E_{\text{data}}(H)$ measures agreement of the reconstruction H with the aligned low resolution data. $E_{\text{regular}}(H)$ is a regularization or prior energy term that guides the optimizer towards plausible reconstruction H. The 3D model M_t cannot be represented by only one 2D disparity image since the points situated on the fringe change-over have sub-pixel precision. Also, pixels participate separately in the 3D model since the 3D coordinates of each pixel is retrieved using only its phase information. Thus, for each camera three 2D maps are created, defined by the *x*-, *y*- and *z*-coordinates of the 3D points. The optimization algorithm and the deblurring are applied to compute high resolution images of *x*, *y*, and *z* and texture from the low resolution images. The final high resolution 3D point cloud is retrieved by merging obtained 3D models that are already registered since all of them contain the 3D sparse point cloud. The final result is illustrated in Figure 1.13.

Figure 1.13 Some 3D frames computed by the temporal 3D super resolution approach proposed by Ouji *et al.* (2011)

1.4.4 Spacetime Faces

The vast majority of stereo research has focused on the problem of establishing spatial correspondences between pixels in a single pair of images for a static moment in time. The works presented in Davis *et al.* (2003) and Zhang *et al.* (2003), which presented nearly identical ideas, proposed to introduce the temporal axis (available since they process video sequences) to improve the stereo matching problem. They proposed *spacetime* stereo matching algorithms based on similar ideas. The algorithm proposed in Davis *et al.* (2003) was tested on static objects when varying illuminations. The algorithm proposed in Zhang *et al.* (2003) was tested on moving objects (faces when conveying arbitrary expressions). The following synthesis is based on both works, but the

reconstruction results are taken from Zhang *et al.* (2004) because the object of interest in this chapter is human face. We note that in their experiments, Zhang *et al.* (2004) used four cameras and two projectors. Each side of the face was acquired by one binocular active stereo system (one projector associated to two cameras). By this way, the authors tried to avoid self-occlusions, which can be a challenging problem in stereo vision (even if a textured light were projected).

- *Spatial stereo matching.* The way in which traditional stereo systems determine the position in space of P, is triangulation, that is by intersection the rays defined by the centers c_l, c_r of cameras c_l, c_r and the projection of P in left and right images $I_l(x_l, y_l, t)$ and $I_r(x_r, y_r, t)$, respectively. Thus triangulation accuracy depends crucially on the solution of corresponding problem. This kind of approaches, widely used in literature, operates entirely within the spatial domain (the images). In fact, knowing the cameras positions ((R, t), the stereo extrinsic parameters), one can first apply rectification transformation that projects left image $I_l(x_l, y, t)$ and right image $I_r(x_r, y, t)$ onto a common image plane, where $y_l = y_r = y$. Thus, the establishing correspondence moves from a 2D search problem to a 1D search problem and minimizes the matching 1D function $F(x_r)$ 1.35, to find x_r^*,

  $$F(x_r) = \sum_{V_s} (I_l(V_s(x_l)) - I_r(V_s(x_r)))^2, \quad (1.35)$$

  where V_s is a window of pixels in a spatial neighborhood close to x_l (or x_r). The size of V_s is a parameter, it is well-known that the smoothness/noisy reconstruction depends on larger/smaller used window V_s. $F(x_r)$ given in Equation 1.35 is simply the square difference metric. Other metrics exist is the literature, we refer the reader to the review presented in Scharstein and Szeliski (2002). Figure 1.15**c** shows the reconstructed facial surface from *passive stereo* (left top frame is given Fig. 1.15**a**). Here, neither light pattern is projected on the face. The reconstruction result is noisy due to the texture homogeneity on the skin regions, which leads to matching ambiguities. In contract, an improved reconstruction is given in Figure 1.15**d**, where *active stereo* is used. The projected colored stripes generate texture on the face, which helps the spatial matching process. However, certain facial shape details are smoothed out because of the largeness of the used spatial window (15×15). Frames shown in Figure 1.15**b**) illustrate pattern projections on the face across time.
- *Temporal stereo matching.* In this stereo-matching schema, establishing

correspondence for a pixel (x_l, y, t_0) in frame $\mathcal{M}$ is based, this time, on temporal neighborhood $V_t = t_0 \pm \Delta t$, instead of the spatial window V_s. Thus, one can define the matching function $F(x_r)$ as follows,

$$F(x_r) = \sum_{V_t} (I_l(V_t(x_l, t_0)) - I_r(V_t(x_r, t_0)))^2 \quad (1.36)$$

The previous equation is analogous to Equation 1.35 except that now instead of a spatial neighborhood, one must consider a temporal neighborhood V_t around some central time t_0. Because of the changing of the light patterns over time, this temporal window works. This time, the size of V_t is a parameter, that is, the accuracy/noisy reconstruction depends on larger/smaller of the used window. It should be also adapted to deforming objects speed.

- *Spacetime stereo matching.* This stereo-matching schema combines both spatial matching and temporal one to limit the matching ambiguities. The function $F(x_r)$ is analogous to Equations 1.35 and 1.36 and is given by

$$F(x_r) = \sum_{V_{st}} (I_l(V_{st}(x_l, t_0)) - I_r(V_{st}(x_r, t_0)))^2, \quad (1.37)$$

Here V_{st} represents a spatiotemporal volume instead of a window in a spatial-based matching or a vector in a temporal-based matching. Figure 1.14 illustrates the spatial and the spacetime stereo matchings to establish correspondences between the pixels in I_l and those in I_r. The images are already rectified. Figure 1.15**e** shows the reconstruction result operated by spatiotemporal stereo matching using a volume of size $(9 \times 5 \times 5)$. This time, the spacetime approach cover more shape details than in Fig. 1.15**d**, however, it also yields artifacts due to the over-parametrization of the depth map. An improvement of this reconstruction using a global spacetime stereo matching with the same volume size is given in Fig. 1.15**f**. (See[2] for video illustrations of these reconstructions).

Figure 1.14 Spatial vs. Spacetime stereo matching. The spatial matching uses only spatial axis along y, thus the V_s window to establish correspondence. The spacetime stereo matching extend the spatial window to the time axis, thus the V_{st} is used to compute $F(x_r)$

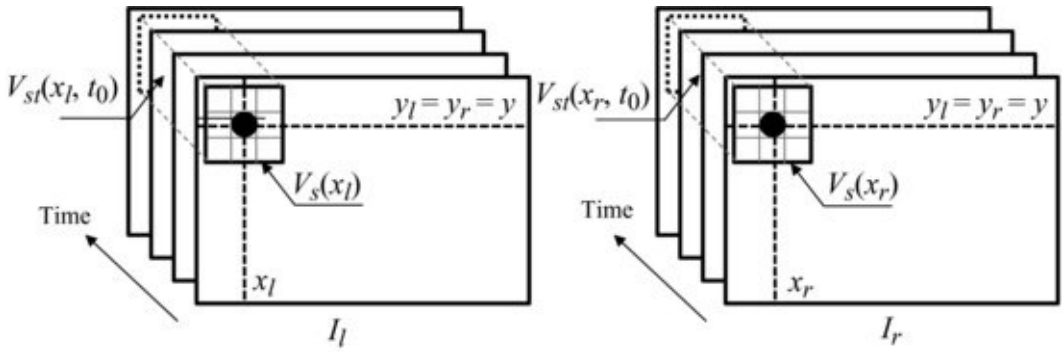
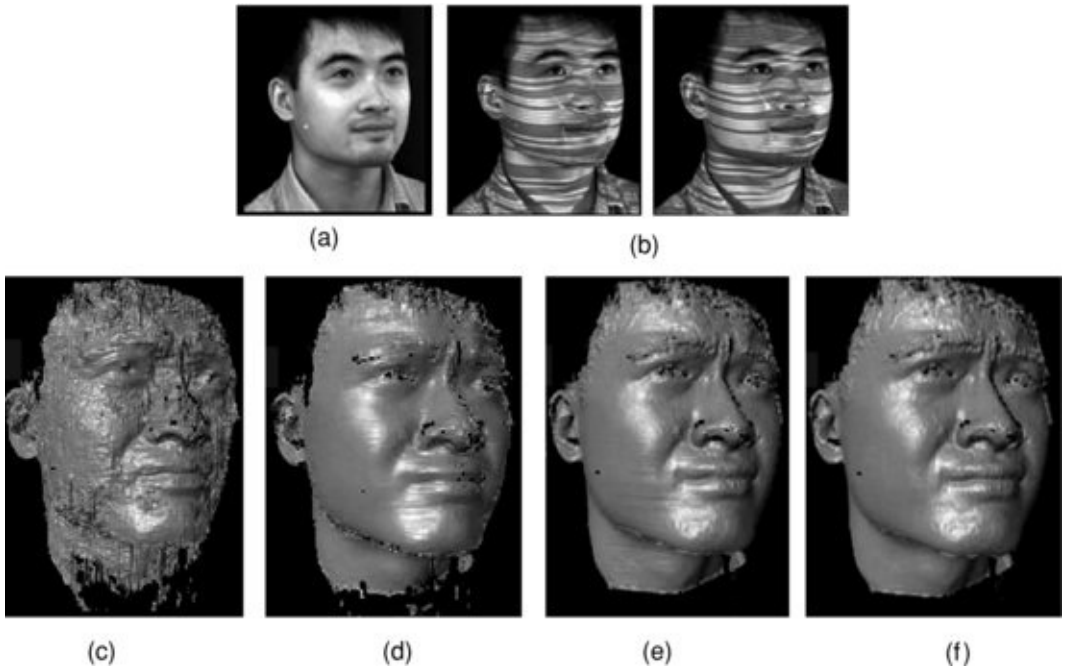


Figure 1.15 Comparison of four different stereo matching algorithms. (a) Top left non-pattern frame, captured in ambient lighting conditions. (b) Sequence of top left pattern frames, captured under patterns projections. (c) Reconstructed face using traditional stereo matching with a $[15 \times 15]$ window achieved on non-pattern left stereo frames. The result is noisy due to the lack of color variation on the face. (d) Reconstructed face using pattern frames (examples are given in (b)) using stereo matching with a $[15 \times 15]$ window. The result is much better because the projected stripes provide texture. However, certain face details are smoothed out due to the need for a large spatial window. (e) Reconstructed face using local spacetime stereo matching with a $[9 \times 5 \times 5]$ window. (f) Reconstructed face using the global spacetime stereo matching with a $[9 \times 5 \times 5]$ window. Global spacetime stereo matching removes most of the striping artifacts while preserving the shape details [from http://grail.cs.washington.edu/projects/stfaces/]



1.4.5 Template-based Post-processing

Recently, template-based approaches emerge due to its simplicity and robustness to noisy range data. Outputs of shape recovery techniques present often imperfections like spikes, holes dues to self-occlusions or the absorption of projected lights by dark regions of the face. The template generic model provides a strong geometric prior and thus leads to high quality reconstructions with automated hole-filling and noise removal. Correspondence estimation is often facilitated by the use of tracked marker points or hand-selected landmarks correspondences. The template-based literature consist on template-to-data registration then fitting and could allowing 3D face tracking and expressions cloning. These stages are described in detail in the following paragraphs. For the rest of this section, let $\mathcal{M}$ denotes the template model and $\mathcal{P}$ denotes the target data.

Landmarks Detection

This step consists on manually or automatically facial *keypoints* detection (eyebrows, eyes, nose, mouth contours, etc.). These facial *keypoints* are important in the following stages. In particular, they could be used in coarse rigid registration to prepare the fine one, and they are often used as control points in the warping/fitting procedure. Automatic 3D face landmarking is one active research topic studied within the 3D face recognition and expression recognition applications. Many approaches are designed and try to face the pose variation and external occlusion problems (Segundo et al., 2010; Mehryar et al., 2010; Zhao et al., 2011).

Rigid Registration

Registration of $\mathcal{M}$ and $\mathcal{P}$ involves estimating an optimal rigid transformation between them, denoted *t*. Here, $\mathcal{P}$ is assumed to remain stationary (the reference data), whereas $\mathcal{M}$ (the source data) is spatially transformed to match it. The *Iterative Closet Point* algorithm (ICP) is the best-known technique for pairwise surface registration. Since the first paper of Besl and McKay (1992) ICP has been widely used for geometric alignment of 3D models and many variants of ICP have been proposed (Rusinkiewicz and Levoy, 2001). ICP is an iterative procedure minimizing the error (deviation) between points in $\mathcal{P}$ and the closest points in $\mathcal{M}$. It is based one of the following two metrics: (i) *the point-to-point*, metric which is the earlier and the classical one, by minimizing in the *K*-th

iteration, the error $E_{\text{reg}}^{k}(T^{k}) = \sum (T^{k}.p_i - q_j)$; $q_j = q | \min_{q \in \mathcal{M}} (E_{\text{reg}}^{k}(T^{k}))$; (ii) *the point-to-plane* introduced later and minimizes $E_{\text{reg}}^{k}(T^{k}) = \sum n(q_j)(T^{k}.p_i - q_j)$. For each used metrics, this ICP procedure is alternated and iterated until convergence (i.e., stability of the error). Indeed, total transformation *t* is updated in an incremental way as follows: for each iteration *K* of the algorithm: $T=T^{k}.T$. One note that ICP performs fine geometric registration assuming that a coarse registration transformation T^{0} is known. The final result depends on the initial registration. The initial registration could be obtained when corresponding detected landmarks in $\mathcal{M}$ and $\mathcal{P}$.

Template Warping/Fitting

A warping of $\mathcal{M}$ to $\mathcal{P}$ is defined as the function *f* such that $F(\mathcal{M}) = \mathcal{P}$. The function *f* is called the *warping function*, which takes $\mathcal{M}$ to $\mathcal{P}$. Given a pair of landmarks (detected as described in Section 1.4.5) with known correspondences, $U_L=(u_i)^{T}_{1<i<L}$ and $V_L=(v_i)^{T}_{1<i<L}$, in $\mathcal{M}$ and $\mathcal{P}$, respectively. One needs to establish dense correspondence between other meshes vertices; u_k and v_k denote the locations of the *K*-th corresponding pair and *L* is the total number of corresponding landmarks. Thus, a warping function, *f*, that warps U_L to V_L subject to perfect alignment is given by the conditions $F(u_i)=v_i$ for $i = 1, 2, \ldots, L$.

- *Thin Plate Spline (TPS)*. TPS Bookstein (1989) are a class of widely used nonrigid interpolating (warping) functions. The thin plate spline algorithm specifies the mapping of points for a reference, $\mathcal{P}$, set to corresponding points on a source set, $\mathcal{M}$. The TPS fits a mapping function $F(u)$ between corresponding point-sets $\{v_i\} \in \mathcal{M}$ and $\{u_i\} \in \mathcal{P}$ by minimizing the following energy function:

  $$E_{tps} = \sum_{i=1}^{L} \|v_i - F(u_i)\|^2 + L\lambda J \quad (1.38)$$

  For a fixed λ which provides trade-off of warp smoothness and interpolation.

  $$J = \iint \left[\left(\frac{\partial^2 F}{\partial u^2} \right)^2 + 2 \left(\frac{\partial^2 F}{\partial u \partial v} \right)^2 + \left(\frac{\partial^2 F}{\partial v^2} \right)^2 \right] du\, dv \quad (1.39)$$

  The interpolation deformation model is given in terms of the warping function $F(u)$, with

$$F(u) = \overbrace{A}^{4\times4} u + \underbrace{W^T}_{4\times L} \overbrace{K(u)}^{L\times1}, \tag{1.40}$$

where A (affine transformation) and W (non-affine warping) are TPS parameters and $K(u) = (|u - u_1|; |u - u_2|; \ldots; |u - u_m|)^T$ is the control point influence vector.

The warping coefficients A and W are computed by the equation:

$$(A|W)\left(\begin{array}{c|c} U & 0 \\ \hline K + L\lambda I & U^T \end{array}\right) = (V|0) \tag{1.41}$$

In other words, any point on $\mathcal{M}$ close to a source landmark v_k will be moved to a place close to the corresponding target landmark u_k in $\mathcal{P}$. The points in between are interpolated smoothly using Bookstein's Thin Plate Spline algorithm Bookstein (1989).

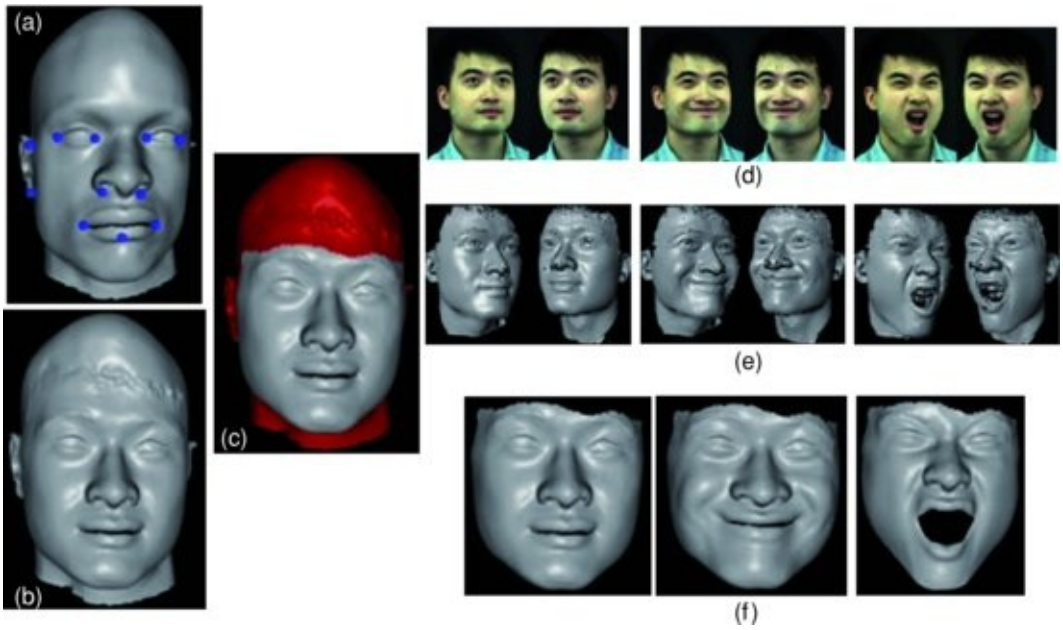
- *Nonrigid ICP*. Register in a nonrigid way a template $\mathcal{M}$ and an input scan $\mathcal{P}$ by nonrigid ICP requires estimating both correspondence and a suitable warping function that matches the shape difference between them. In Allen *et al.* (2003) and Amberg *et al.* (2007) similar ideas are presented for scan-template warping applied on human body in Allen *et al.* (2003) and on human faces in Amberg *et al.* (2007). Both of them proposed an energy-minimization framework, as given by

$$E = \alpha \overbrace{\sum_{v_i \in \mathcal{M}} w_i \text{dist}^2(T^i v_i, \mathcal{P})}^{E_{\text{data}}(T)} + \beta \overbrace{\sum_{i,j|\{vi,vj\}\in \text{edges}(\mathcal{M})} \|T^i - T^j\|_F^2}^{E_{\text{smoothness}}} + \gamma \overbrace{\sum_i \|T^i v_i - u^j\|^2}^{E_{\text{landmarks}}}, \tag{1.42}$$

where minimizing the term E_{data} guarantee that the distance between the deformed template $\mathcal{M}$ and the target data $\mathcal{P}$ is small. The term $E_{\text{smoothness}}$ is used to regularize the deformation. In other words, it penalizes large displacement differences between neighboring vertices. The term $E_{\text{landmarks}}$ is introduced to guide the deformation by using corresponding control points that are simply the anthropometric markers in human body and facial landmarks in the case of face fitting. Similar formulation are presented in Zhang *et al.* (2004) for template fitting. The Figure 1.16 illustrates an example of template fitting results. A similar formulation is used in Weise *et al.* (2009) for personalized template building.

Figure 1.16 Illustration of the template fitting process. (a) A face template with manual landmarks. (b) Obtained mesh after fitting the warped template to the first two depth maps given in (e). (c) Facial region limitation (red colored

regions present unreliable depth or optical flow estimation). (d) A sequence of texture image pairs. (e) A sequence of depth map pairs. (f) Selected meshes after tracking the initial mesh through the whole sequence, using both depth maps and optical flows. The process is marker-less and automatic [from http://grail.cs.washington.edu/projects/stfaces/]



Template Tracking

In Zhang *et al.* (2004), after the template fitting step, the authors proposed a tracking procedure which yields point correspondence across the entire sequence. They obtained time-varying face models (of the deformed template) without using markers. Once this template sequence is acquired, they propose to interactively manipulate it to create new expressions. To achieve template tracking, they first compute optical flow from the sequence. The flow represents vertices motion across the facial sequence and is used to enhance template tracking by establishing inter-frame correspondences with video data. Then, they measure the consistency of the optical flow and the vertex inter-frame motion by minimizing the defined metric. Similar ideas were presented in Weise *et al.* (2009) where a person-specific facial expression model is constructed from the tracked sequences after nonrigid fitting and tracking. The authors targeted real-time puppetry animation by transferring the conveyed expressions (of an actor) to new persons. In Weise *et al.* (2011) the authors deal with two challenges of performance-driven facial animation; accurately track the rigid and nonrigid motion of the user's face, and map the extracted tracking parameters to suitable

animation controls that drive the virtual character. The approach combines these two problems into a single optimization that estimates the most likely parameters of a user-specific expression model given the observed 2D and 3D data. They derive a suitable probabilistic prior for this optimization from pre-recorded animation sequences that define the space of realistic facial expressions.

In Sun and Yin (2008), the authors propose to adapt and track a generic model to each frame of 3D model sequences for dynamic 3D expression recognition. They establish the vertex flow estimation as follows: First, they establish correspondences between 3D meshes using a set of 83 pre-defined key points. This adaptation process is performed to establish a matching between the generic model (or the source model) and the real face scan (or the target model). Second, once the generic model is adapted to the real face model, it will be considered as an intermediate tracking model for finding vertex correspondences. The vertex correspondence across 3D model sequences provides a set of motion trajectories (vertex flow) of 3D face scans. The vertex flow can be depicted on the adapted generic model (tracking model) through the estimation of the displacement vector from the tracked points of the current frame to the corresponding points of the first frame with a neutral expression. The vertex flow is described by the facial motion vector $U = [u_1, u_2, \ldots, u_n]$, where *N* is the number of vertices of the adapted generic model. They used the Hidden Markov Model to model and train facial dynamics.

Expression Transferring

Also known as *expression cloning* or *performance capture* when facial animation uses the performance of an actor to animate virtual models. The steps discussed earlier, namely, template fitting and tracking, allow expression transferring from real-time acquired 3D data to a virtual model or puppetry. Several papers were published to transfer facial animation to templates, puppetry or personalized models, yong Noh and Neumann (2001), Sumner and Popović (2004), Vlasic *et al.* (2005), Pyun *et al.* (2003), Zhang *et al.* (2004), Weise *et al.* (2009), Weise *et al.* (2011), *etc.*

1.5 Summary and Conclusions

Creating 3D face models that look and deform realistically in an important issue is many applications such as person-specific facial animation, 3D-based face recognition, and 3D-based expression recognition. This chapter is a survey of

successful state-of-the-art techniques that sometimes led to commercial systems. These techniques are within a static/dynamic (moving) face modeling–guided taxonomy. Each of the presented techniques is based on one of the following well-known concepts: (i) depth from triangulation, (ii) shape from shading, and (iii) depth from ToF. Obviously, other approaches exist in the literature but we limited our survey to those based on the aforementioned concepts. In this section, we will put forward, a comparative study of the mentioned approaches according to the *intrinsic* and *extrinsic* factors. The *intrinsic factors* are related to the sensor, such as its cost, its spatial (in the case of static modeling) or spatiotemporal resolutions (in the case of dynamic modeling), its measurement accuracy, and its intrusiveness/need user cooperation. The *extrinsic factors* include variations due to illumination changes, motion speed of the observed face, and details in the face (wrinkles, scars, etc.). Figure 1.17 gives an evaluation of approaches according to these criteria.

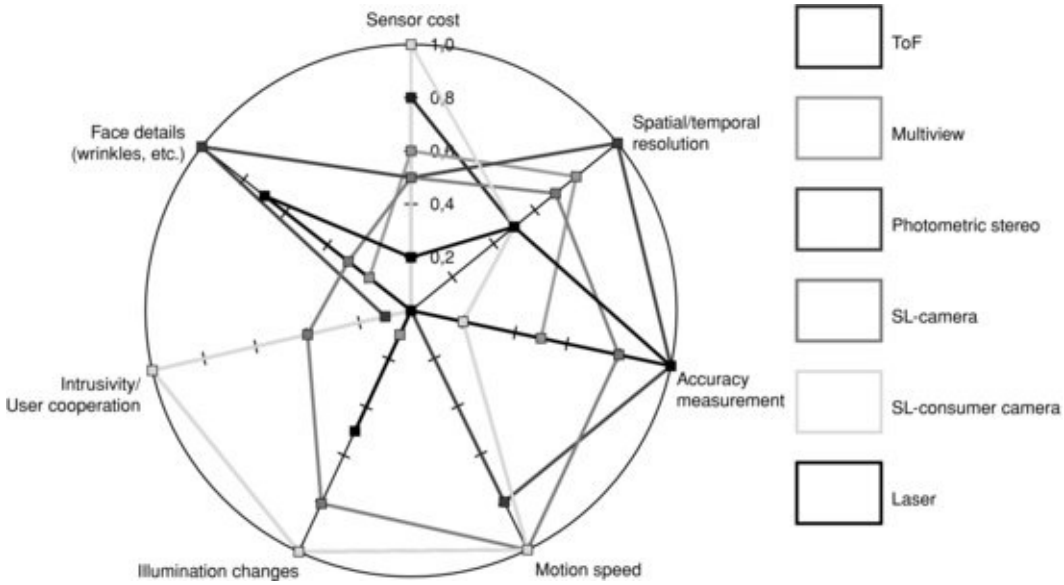
- Laser-stripe scanning is intended for *static faces* due to the processing time required to project the laser stripe on the whole face. The sensor is expensive and needs user cooperation to perform face acquisition (a distance less than 1.5 m is required). Commercial systems such as the Minolta Non-contact 3D Digitizer VIVID910[3] produced texture and depth images of the same resolution 640×480. The system accurately measures the 3D object with a depth-accuracy of around 0.1 mm. It takes 2.5 s for the fine mode and 0.5 s for the fast mode to produce a scan, thus no motion during the scan is tolerated. Laser-based techniques work in office environment lighting conditions. Most of the facial details (wrinkles, scars, and other person-specific markers) are reproduced in the virtual model.
- Structured-light (SL) techniques provided an attractive alternative to the expensive laser-stripe scanning technique. In fact, projected light(s) intend(s) to replace the laser scan. The ARTEC MHT 3D scanner[4] is one commercial system which projects a permanent light pattern and produces 3D video of a rate around 15 fps. Frame resolution is about 500,000 points. The working distance should be in the interval of 0.4–1 m. Depth-measurement accuracy is comparable to laser scanners and is about 0.1 mm. Texture channel is also captured but only when needed. The sensor is cheaper than the laser digitizers.
- Structured-light (SL) consumer depth cameras (which are much cheeper) as MS Kinect[5] and Asus Xtion Pro Live[6] have been recently developed and have been an attractive alternative for expensive sensors. Kinect is based on

the permanent projection of one infrared-laser pattern. It was primarily designed for natural interaction in a computer game environment. In fact, the sensor is less intrusive and only a near-frontal position of the user is needed. However, the characteristics of the data captured by Kinect have attracted the attention of researchers in the field of computer vision and computer graphics. The camera provides depth and texture video with 300,000 points in every frame. The 2D and 3D videos have got a rate of 30 fps. The depth measurement produced by the Kinect was not so accurate, which means that it achieves a coarse reconstruction of the 3D face.

- In photometric stereo approaches, only one image of the face is captured and used to recover the depth information. The face is illuminated with three colored light sources from three different directions. The capture can be made in real time enabling 3D and 4D acquisition. By knowing the surface reflectance properties of the face, the local surface orientation at points illuminated by all three light sources are computed. One of the important advantages of the photometric stereo approaches is that the points do not need to be registered. Thus, this category of approaches does not suffer from the correspondence problem, providing high performance for featureless surfaces as the human skin. On the other hand, the disadvantages of this category of approaches are that they are indirect and practical only for applications in which the illumination is carefully controlled.
- Multiview-based approaches capture images instantly and provide high resolution 3D and 4D textured images. In addition, they have several advantages over active approaches. First is the quality texture image. The acquisition phase does not require pattern projection, and, so, there is a true one-to-one correspondence with every color pixel and every 3D point. The original texture images are always of the highest quality. Second is the absence of holes in the final 3D scans. Dimensional Imaging[7] proposes systems designed specifically to capture high definition 3D surface images of the human face with highly detailed 20-megapixel texture maps using four 10-megapixel cameras and up to 32 cameras with a resolution of up to 21 megapixels.
- ToF cameras are relatively new devices, as the semiconductor processes have only recently become fast enough for such devices. The systems cover ranges of a few meters up to about 60 m. Another advantage of ToF systems is the high rate capture. In return, they have a low resolution and a precision of 1 mm to 1 cm. The Mesa Imaging[8] SwissRanger 4000 (SR4000) is

probably the most well-known ToF camera. It has a range of 5–8 m, 176×144 pixel resolution over $43.6° \times 34.6°$ field of view and operates at up to 54 fps.The PMD Technologies[9] CamCube 2.0 is a less popular one. It has a range of 7 m, 204×204 pixel resolution with $40.0° \times 40.0°$ field of view. It operates at 25 fps.

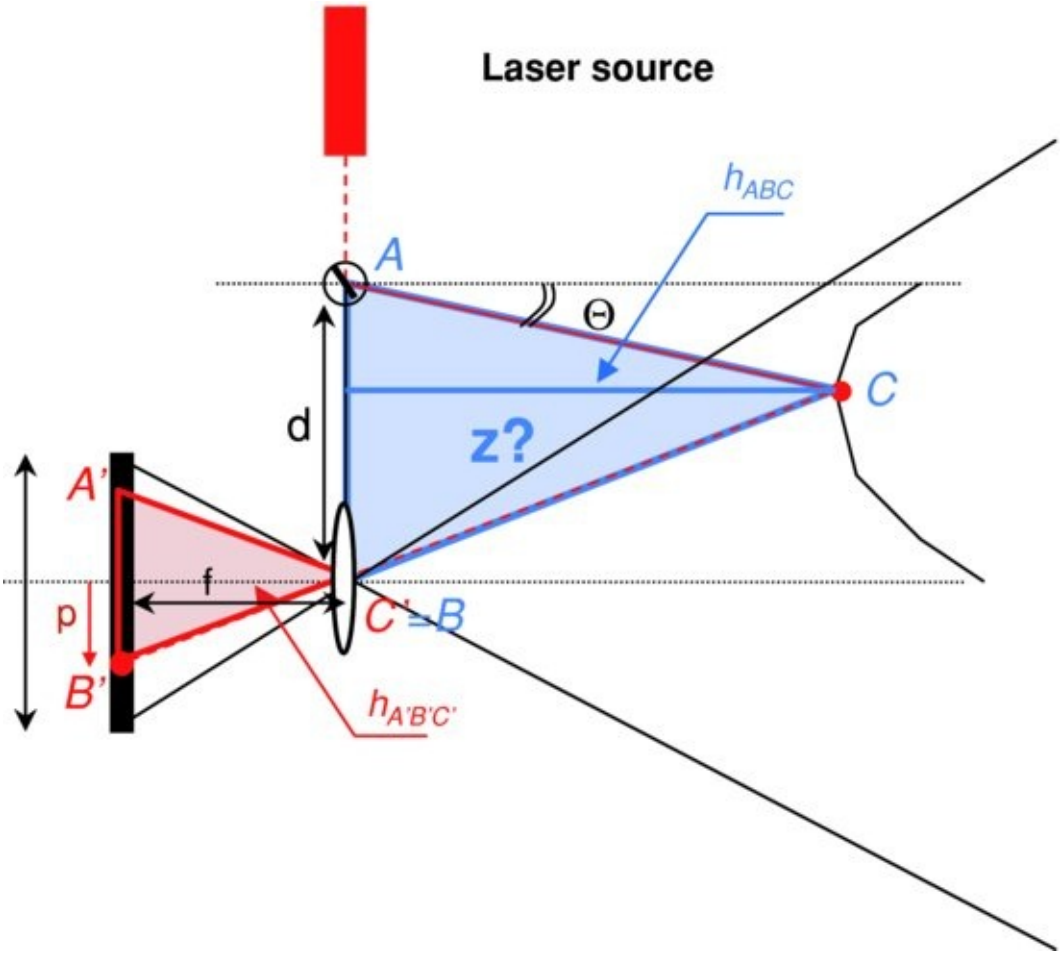
Figure 1.17 Evaluation of different 3D face modeling techniques according to *Extrinsic factors* (Motion speed/illumination changes/intrusivity and need for user cooperation/face details) and Intrinsic factors (spatial and temporal resolutions/accuracy measurement/sensor cost)



Exercises

1. From Figure 1.18 prove that $\frac{AB}{A'B'} = \frac{AC}{A'C'} = \frac{BC}{B'C'} = \frac{h_{ABC}}{h_{A'B'C'}}$; retrieve the z formula given in Equation 1.8.

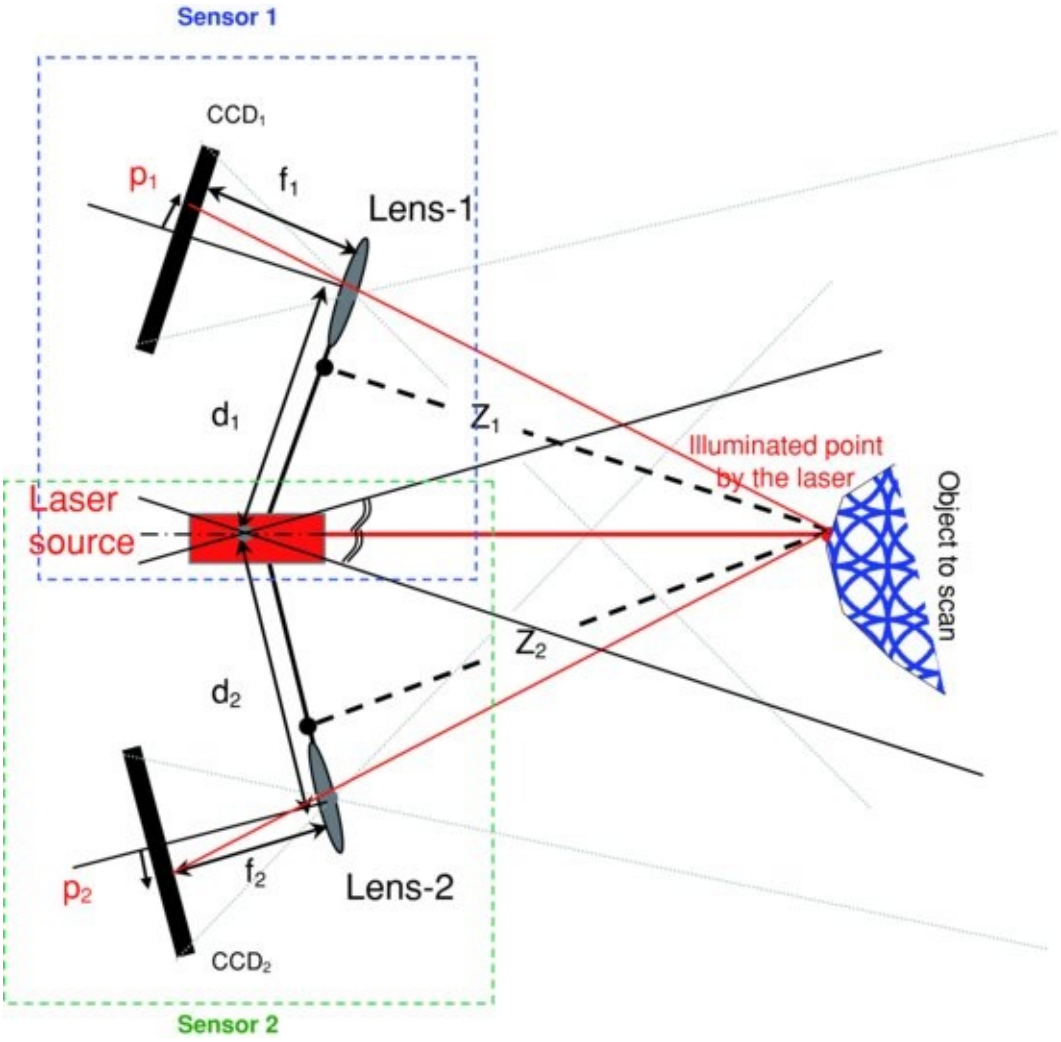
Figure 1.18 Optical triangulation geometry (laser stripe scanning)



2. We need to study the 3D scanning prototype given in Figure 1.19. It consists of a laser source that illuminates the object to be continuously scanned and two cameras that look at the same object. The projected laser stripe is seen by both cameras. The global sensor calculates the depth information, as illustrated in the figure. To capture the full geometry of the object, a manual scan of the surface is required.

- Compute the Z_1-coordinate together with Z_2-coordinate.
- Explain the triangles considered to calculate Z_1-coordinate.
- Why this prototype involves two sensors, each of them capable of measuring the depth. Suggest a depth value z as a function of Z_1 and Z_2; explain your choice.

Figure 1.19 3D scanning system to study



3. We aim to produce patterns associated with Equation 1.10. Assuming that the gray level 0 represents black and gray level 255 represents white, find the value of I_0 and I_{mod}. Solve Equation 1.10 for I_0, I_{mod}, and Equation 1.10.

4. Extend Equation 1.10 to obtain N patterns. Solve these equations for I_0, I_{mod}, and Equation 1.10. What is the impact of using more than three patterns on the accuracy and delay?

5. Assuming that only one pattern from Equation 1.10 is used, resolve I_0, I_{mod}, and Equation 1.10 using Fourier Transfrom.

6. The approach presented in Section 1.4, Equations 1.27 to 1.32, overcomes the major problem of fast phase-shift scanning, namely, motion artifacts. An analysis of the motion error has been introduced to compensate for motion artifacts on the pixel level. Nevertheless, high-frequency texture can still pose problems during motion, as the assumption of invariant surface reflectance is

violated. Investigate the possibility of adding a stereo module and extending the motion compensation to stereo geometry to handle these cases.

References

Allen B, Curless B, Popović Z. The space of human body shapes: reconstruction and parameterization from range scans. ACM Transactions on Graphics 2003;22(3):587–594.

Amberg B, Romdhani S, Vetter T. Optimal step nonrigid icp algorithms for surface registration. 2007;CVPR. IEEE Computer Society.

Amenta N, Kil YJ. Defining point-set surfaces. ACM SIGGRAPH 2004 Papers on SIGGRAPH 04 2004;23(3):264.

Andriy Myronenko, Xubo B. Song MÁCP. Nonrigid point set registration: coherent point drift. NIPS 2006;1009–1016.

Beeler T, Bickel B, Beardsley P, Sumner B, Gross M. High-quality single-shot capture of facial geometry. ACM Transactions on Graphics 2010;29(4):40:1–40:9.

Beeler T, Hahn F, Bradley D, Bickel B, Beardsley P, Gotsman C, Sumner RW, Gross M. High-quality passive facial performance capture using anchor frames. ACM SIGGRAPH 2011 papers on SIGGRAPH11 2011;1(212):1.

Besl PJ, McKay ND. A method for registration of 3-d shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence 1992;14(2):239–256.

Bookstein FL. Principal warps: thin-plate splines and the decomposition of deformations. IEEE Transactions on Pattern Analysis and Machine Intelligence 1989;11(6):567–585.

Bradley D, Heidrich W, Popa T, Sheffer A. High resolution passive facial performance capture. ACM Transactions on Graphics 2010;29:41:1–41:10.

Davis J, Ramamoorthi R, Rusinkiewicz S. Spacetime stereo: a unifying framework for depth from triangulation. IEEE Computer Vision and Pattern Recognition 2003;2:359–366.

Fan J, Wolff LB. Surface curvature and shape reconstruction from unknown multiple illumination and integrability. Journal of Computer Vision and Image Understanding 1997;65(2):347–359.

Farsiu S, Robinson MD, Elad M, Milanfar P. Fast and robust multiframe super

resolution. IEEE Transactions on Image Processing 2004;13(10):1327–1344.

Furukawa Y, Ponce J. Accurate, dense, and robust multiview stereopsis. IEEE Transactions on Pattern Analysis and Machine Intelligence 2010;32(8):1362–1376. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5226635&tag=1.

Furukawa Y, Ponce J. Dense 3D motion capture for human faces. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition; 2009 Jun 20–25; Miami, FL: Computer Vision and Pattern Recognition; 2009. p. 1674–1681. Available at: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5206868.

Gargallo P, Sturm P. Bayesian 3d modeling from images using multiple depth maps. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition; 2005 Jun 20–26; San Diego, CA: Computer Vision and Pattern Recognition; 2005. 2(c). p. 885–891.

Glencross M, Ward GJ, Melendez F, Jay C, Liu J, Hubbold R. A perceptually validated model for surface depth hallucination. ACM Transactions on Graphics 2008;27(3):1.

Gopi M, Krishnan S, Silva CT. Surface reconstruction based on lower dimensional localized Delaunay triangulation. Computer Graphics Forum 2000;19(3):467–478.

Hernandez C, Vogiatzis G, Cipolla R. Overcoming shadows in 3-source photometric stereo. Pattern Analysis and Machine Intelligence, IEEE Transactions on 2011;33(2):419–426.

Hernandez C, Vogiatzis G, Brostow G, Stenger B, Cipolla R. Nonrigid photometric stereo with colored lights. IEEE 11th International Conference on Computer Vision (2007) 2007;0(5):1–8.

Hernandez Esteban C, Schmitt F. Silhouette and stereo fusion for 3d object modeling. Computer Vision and Image Understanding 2004;96(3):367–392.

Kazhdan M, Bolitho M, Hoppe H. Poisson surface reconstruction. Proceedings of the 4th Eurographics Symposium on Geometry Processing; 2006 Jun 26–28; Cagliari, Sardinia, Italy: Eurographics Symposium on Geometry Processing; 2006. p. 61–70.

Kolmogorov V, Zabih R. Multi-camera scene reconstruction via graph cuts. European Conference on Computer Vision 2002;8:82–96.

Mehryar S, Martin K, Plataniotis KN. Automatic landmark detection for 3d face

image processing. IEEE Congress on Evolutionary Computation 2010;1–7.

Morris DD, Kanade T. Image-consistent surface triangulation. Robotics 2000;1:332–338.

Myronenko A, Song X. Point set registration: coherent point drift. IEEE Transactions on Pattern Analysis and Machine Intelligence 2010;32(12):2262–2275.

Narayanan PJ, Rander PW, Kanade T. constructing virtual worlds using dense stereo. Proceedings of the 6th International Conference on Computer Vision; 1998 Jan 4–7; Mumbai, India: Narosa Publishing House; 1998. p. 3–10.

Ouji K, Ardabilian M, Chen L, Ghorbel F. Multi-camera 3d scanning with a nonrigid and spacetime depth super-resolution capability. In: Pedro Real, Daniel Diaz-Pernil, Helena Molina-Abri, et al., editors. Computer Analysis of Images and Patterns. Proceedings of the 14th international conference on computer analysis of images and patterns, Part II; 2011 Aug 29–31 Seville, Spain. Berlin, Heidelberg: Springer-Verlag; 2011. 220–228.

Pan J, Huang P, Zhang S, Chiang FP. Color n-ary gray code for 3-d shape measurement. Proceedings of the 12th International Conference on Experimental Mechanic; 2004 Aug 29–Sep; Bari, Italy.

Ponce J. Dense 3d motion capture from synchronized video streams. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 5(Image and Geometry Processing for 3-D Cinematography); 2008 Jun 24–26; Anchorage, AK: IEEE; 2008. p. 1–8.

Pyun H, Kim Y, Chae W, Kang HW, Shin SY. An example-based approach for facial expression cloning. Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on computer animation, SCA'03; Aire-la-Ville, Switzerland: Eurographics Association; 2003. p. 167–176.

Rusinkiewicz S, Levoy M. Efficient variants of the ICP algorithm. Proceedings of 3rd International Conference on 3D Digital Imaging and Modeling (3DIM); 2001 May 29–Jun 1; Quebec City, Canada: IEEE Computer Society; 2001. p. 145–152.

Rusinkiewicz S, Hall-Holt O, Levoy M. Realtime 3d model acquisition. ACM Transactions on Graphics 2002;21(3):438–446.

Schaefer S, Warren J. Adaptive vertex clustering using octrees. Proceedings of SIAM Geometric Design and Computation; 2003; New York: SIAM. 2003; p. 491–500.

Scharstein D, Szeliski R. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal of Computer Vision 2002;47(1–3):7–42.

Segundo MP, Silva L, Bellon ORP, Queirolo CaC. Automatic face segmentation and facial landmark detection in range images. IEEE Transactions on Systems, Man, and Cybernetics Part B 2010;40(5):1319–1330.

Seitz SM, Dyer CR. Photorealistic scene reconstruction by voxel coloring. Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition 1997;35(2):1067–1073.

Seitz SM, Curless B, Diebel J, Scharstein D, Szeliski R. A comparison and evaluation of multiview stereo reconstruction algorithms. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Volume 1 CVPR06 2006;1(c):519–528.

Sumner RW, Popović J. Deformation transfer for triangle meshes. ACM Transactions on Graphics 2004;23(3):399–405.

Sun Y, Yin L. Facial expression recognition based on 3d dynamic range model sequences. Proceedings of the 10th European Conference on Computer Vision: Part II. Berlin, Heidelberg: Springer-Verlag: ECCV '08; 2008. p. 58–71.

Taylor CJ. Surface Reconstruction from feature based stereo. Proceedings of the 9th IEEE International Conference on Computer Vision; 2003. p. 184–190.

Treuille A, Hertzmann A, Seitz SM. Example-based stereo with general BRDFs Volume 2; 2004; Berlin: Springer-Verlag; 2004. p. 457–469.

Vlasic D, Brand M, Pfister H, Popovi´c J. Face transfer with multilinear models. ACM Transactions on Graphics 2005;24(3):426–433.

Weise T, Bouaziz S, Li H, Pauly M. Realtime performance-based facial animation. ACM Transactions on Graphics (Proceedings SIGGRAPH 2011); 2011.

Weise T, Leibe B, Van Gool L. Fast 3d scanning with automatic motion compensation. IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR '07; 2007. p. 1–8.

Weise T, Li H, Gool LV, Pauly M. Face/off: Live facial puppetry. Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer animation (Proc. SCA'09). Zurich:Eurographics Association, ETH; 2009.

Weyrich T, Pauly M, Heinzle S, Keiser R, Scandella S, Gross M. Post-processing of scanned 3d surface data. Science 2004;1:85–94.

Wolff LB, Angelopoulou E. Threedimensional stereo by photometric ratios. Journal of the Optical Society of America A 1994;11(11):3069–3078.

yong Noh J,Neumann U. Expression cloning. SIGGRAPH'01; 2001. p. 277–288.

Yu T, Xu N, Ahuja N. Shape and view independent reflectance map from multiple views. International Journal of Computer Vision 2006;73(2):123–138.

Zhang L, Curless B, Seitz S. Spacetime stereo: shape recovery for dynamic scenes Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 2; 2003. p. II–367–374.

Zhang L, Snavely N, Curless B, Seitz SM. Spacetime faces: high-resolution capture for modeling and animation. ACM Annual Conference on Computer Graphics; 2004. p. 548–558.

Zhang S, Yau ST. High-speed threedimensional shape measurement system using a modified two-plus-one phase-shifting algorithm. Optical Engineering 2007;46(11):113–603.

Zhang S, Li X, Yau ST. Multilevel quality-guided phase unwrapping algorithm for real-time threedimensional shape reconstruction. Applied Optics 2007;46(1):50–57.

Zhao X, Dellandr´ea E, Chen L, Kakadiaris IA. Accurate landmarking of threedimensional facial data in the presence of facial expressions and occlusions using a threedimensional statistical facial feature model. IEEE Transactions on Systems, Man, and Cybernetics, Part B 2011;41(5):1417–1428.

1. http://szeliski.org/Book/
2. http://grail.cs.washington.edu/projects/stfaces/
3. http://www.konicaminolta.com/instruments/download/catalog/3d/pdf/vivid910_
4. http://www.artec3d.com/3d_scanners/artec-mht
5. http://www.xbox.com/fr-fr/kinect
6. http://www.asus.com/Multimedia/Motion_Sensor/Xtion_PRO_LIVE/
7. http://www.di3d.com
8. http://www.mesa-imaging.ch/
9. http://www.pmdtec.com/

2

3D Face Surface Analysis and Recognition Based on Facial Surface Features

Faisal Radhi M. Al-Osaimi[1] and Mohammed Bennammoun[2]

[1]Department of Computer Engineering, College of Computer and Information Systems, Umm Al-Qura University, Saudi Arabia

[2]School of Computer Science and Software Engineering, The University of Western Australia

This chapter discusses the state of the art in 3D surface features for the recognition of the human face. Particular emphasis is laid on the most prominent and recent contributions. Features extracted from 3D facial surfaces serve as a means for dimensionality reduction of surface data, facilitating the task of face recognition. The complexity of extraction, descriptiveness, and robustness of features directly affect the overall accuracy, performance, and robustness of the 3D recognition system.

2.1 Geometry of 3D Facial Surface

The shape of the human face is a unique class of 3D objects. Many geometrical and topological aspects of the 3D facial surface have been the key ideas behind many facial surface features. Before we proceed with feature extraction, this chapter will first cover (1) the primary 3D surface representations from which feature representations of surfaces are extracted, (2) rigid transformations and decimation of 3D surfaces, and (3) geometries and topologies of the human face that might influence 3D face recognition.

2.1.1 Primary 3D Surface Representations

Before we provide formal definitions and descriptions of the different types of

3D surface representations, we will start with an informal description of these representations and indicate some of their differences.

Scanners usually digitize 3D surfaces in the form of dense 3D point clouds. This is an intuitive output because each surface measurement results in a 3D point. The point cloud representation is capable of *completely* representing open or closed *free-form* (characterized by an irregular shape) 3D surfaces. In some face-recognition systems, the 3D point cloud representation can be used throughout the different modules of the system, namely, preprocessing, feature extraction, and matching. For example, the point cloud representation is sufficient in an iterative closest point (ICP) based face recognition system. (*Note*: Some variants of ICP are based on mesh representation.)

However, converting from the point cloud representation to other *complete* representations such as 3D mesh, range image, or normal map is often needed or desirable. The 3D mesh representation provides a flexible and efficient manipulation of surfaces. This because it stores indexed and precomputed local information of the surface. For instance, once the mesh representation is computed, it enables a more efficient region growing in comparison to a point cloud representation. In addition, the deformation of 3D meshes is more flexible than point clouds. However, 3D meshes require more memory and storage.

For a surface scan of a single view, a range image or a normal map representation can be adequate. To an extent, they resemble 2D gray-level images and thus many 2D image processing and computer vision operations can be similarly and directly applied on them, notably kernel filtering and decimation. These two representations are simpler than 3D meshes, and yet, they can be handy for some types of surface manipulation such as segmentation, computation of curvatures, deformation, and translation. Rotating 3D surfaces might, however, result in the self-occlusion of parts of the surface (in which case some of the 3D points will be over ridden in the range image) or the exposure of previously self-occluded parts (which results in the appearance of surface holes). The following sections provide formal definitions and descriptions of the different 3D surface representations.

Point Cloud Representation

A point cloud representation is merely a set of 3D tuples of the *x*-, *y*- and *z*-coordinates, each representing a 3D point (or a measurement) on the 3D surface. Let $p = (x, y, z) \in \Re^3$ denote a 3D point in a point cloud $\mathcal{P}$. Alternatively, a 3D

point can be represented as a 3D vector **P**, and a point cloud can be represented as a 3× matrix **P**, where *N* is the number of points.

(2.1) $\mathbf{p} = [x \;\; y \;\; z]^\top$.

(2.2) $\mathbf{P} = [\mathbf{p}_1 \ldots \mathbf{p}_N]$.

Textured point clouds are usually represented by attaching to each 3D point a pair of *u* and *v* indices pointing to a position on a texture map. In this case, the tuple representation of a textured 3D point is five dimensional, $p = (x, y, z, u, v) \in \Re^5$. During the manipulation of the textured point cloud, such as rigid transformations or deformations, the *x*-, *y*- and *z*-coordinates of the 3D points change, but the *u* and *v* indices should remain fixed. For this reason, it is preferred to separate the shape **S** from texture **T** in the matrix representation, namely,

(2.3) $\mathbf{s} = [x \;\; y \;\; z]^\top$,

(2.4) $\mathbf{S} = [\mathbf{s}_1 \ldots \mathbf{s}_N]$,

(2.5) $\mathbf{t} = [u \;\; v]^\top$, and

(2.6) $\mathbf{T} = [\mathbf{t}_1 \ldots \mathbf{t}_N]$.

In this matrix representation, the geometric manipulation should be performed on the shape matrix, whereas, the texture matrix stores and maintains the one-to-one correspondence of the 3D points to the texture information. An example of a point cloud and its corresponding texture map is shown in Figure 2.1.

Figure 2.1 (a) A 3D facial scan with texture mapped on top of the 3D surface. (b) 3D facial surface. The image (c) is a point cloud representation and (d) is a polygonal representation of the same facial surface

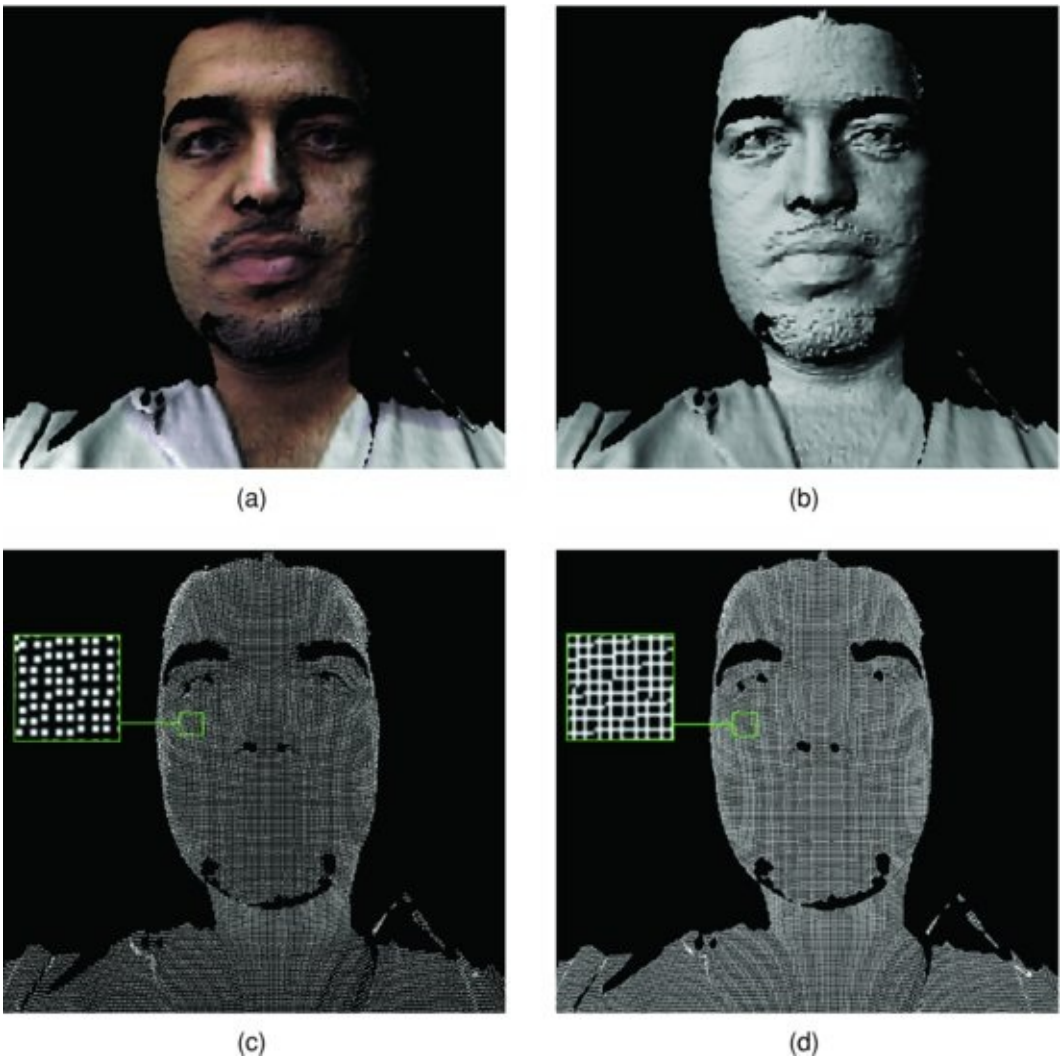


Figure 2.2 The facial surface of the scan shown in Figure 2.1 after cropping and pose-correction in the range image (b) and normal map (c) representations. The normal map is displayed as a color image, where the *x*-, *y*-, and *z*-coordinates of the normals each correspond to the red, green and blue color channels of the image

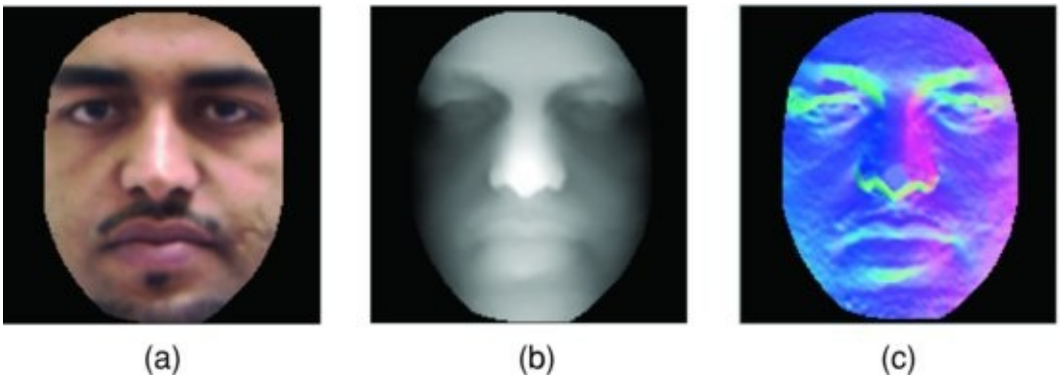


Figure 2.3 Different mesh representations of two surface patches

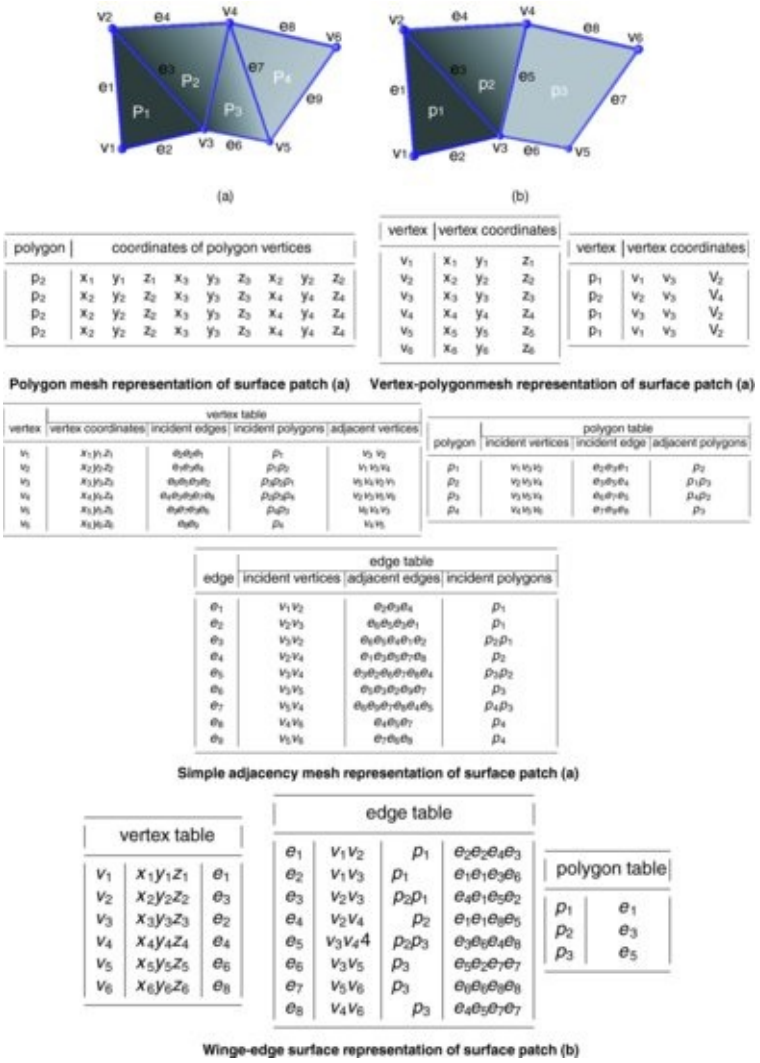


Figure 2.4 (a) A range image of a neutral face and its maximum and minimum principal curvatures, (b), and (c). The red color indicates higher curvatures while the blue color indicates lower curvatures. The image (d) shows the segmented local patches of the facial surface over-laid on the range image. The color codes are red for planar regions, yellow for valleys, green for ridges, blue for saddle regions, cyan for peaks, and magenta for pits

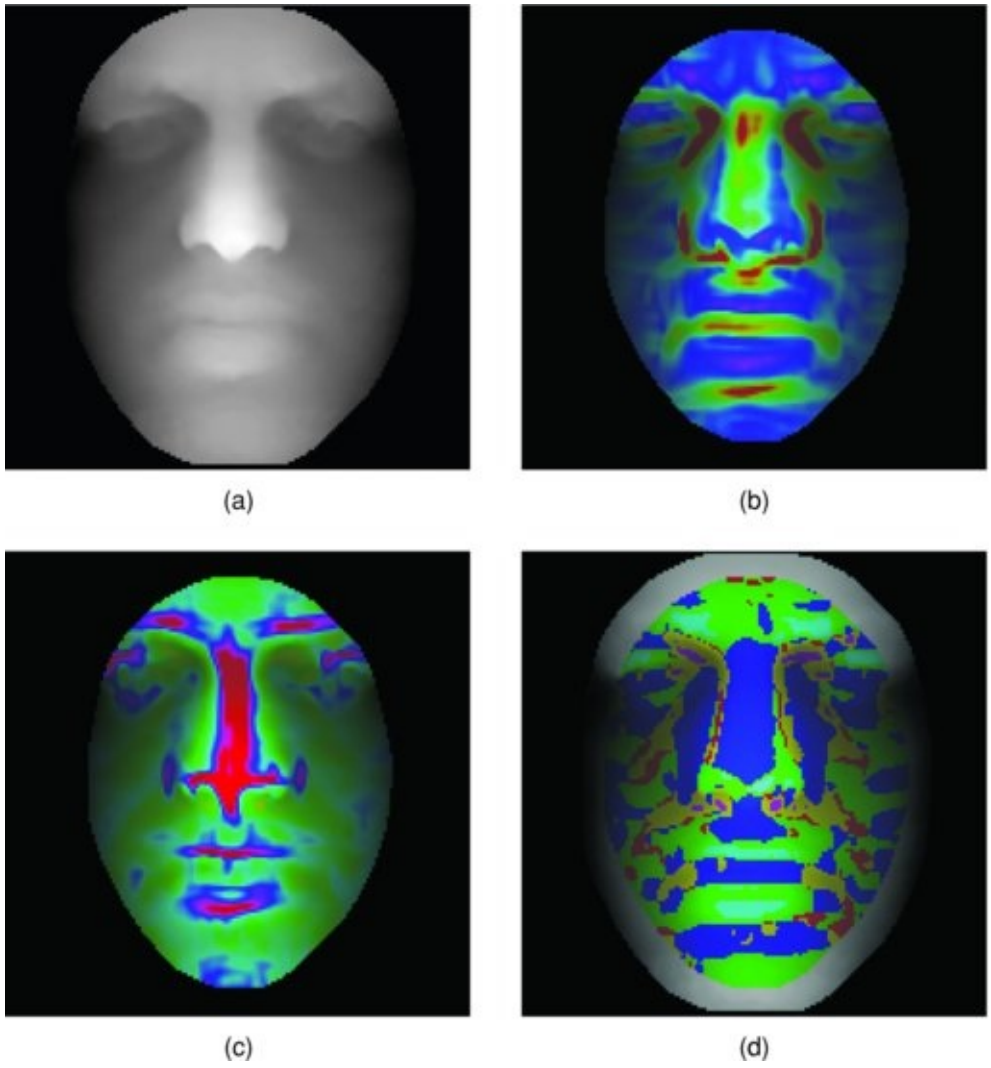


Figure 2.5 The top row shows two neutral facial scans (of the same subject) and their registration using ICP (c). The bottom row shows the registration of a neutral facial scan to a non-neutral scan of the same subject, which appears to be of a higher registration error than the former case

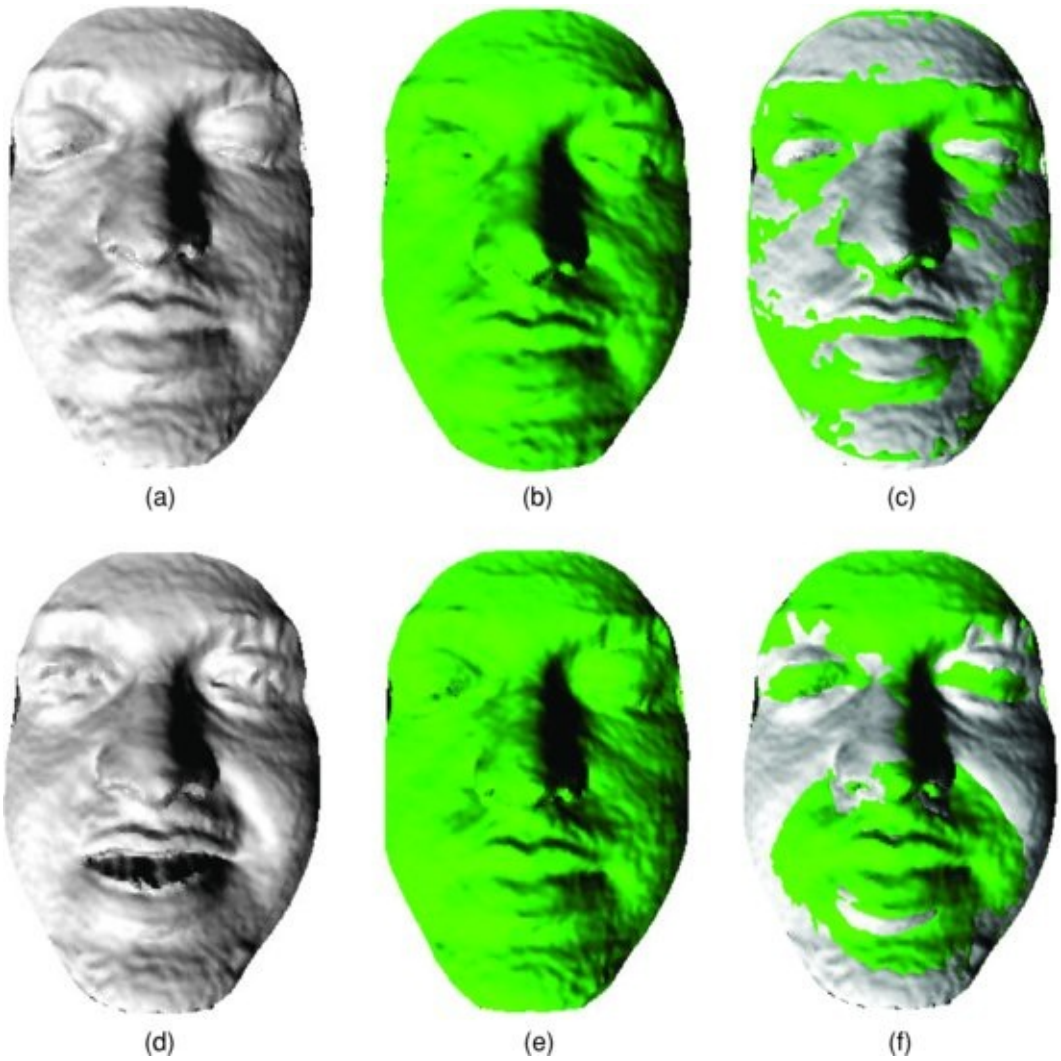
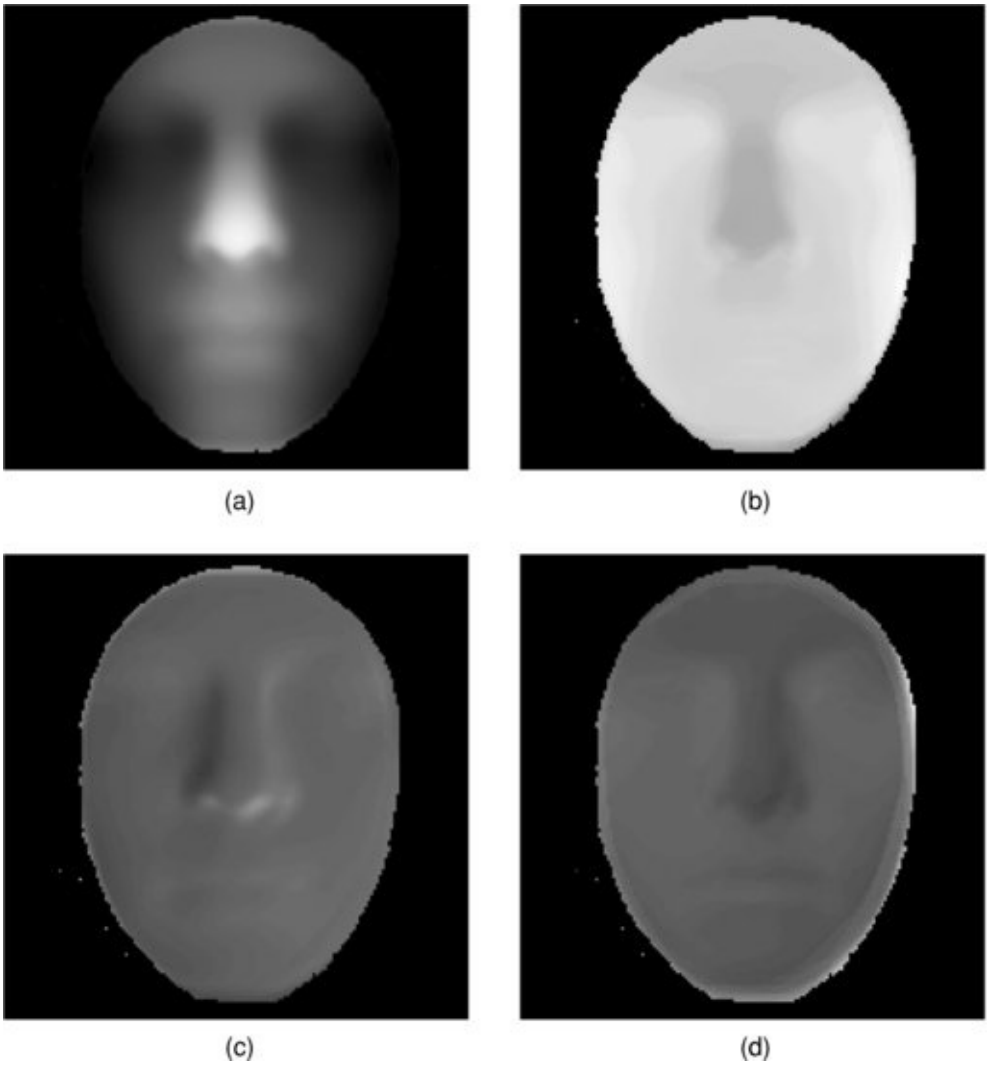


Figure 2.6 An average range face (a) and the first three eigenfaces (eigenvectors of the covariance matrix), (b), (c), and (d), computed from 200 facial range images of different people



3D Polygonal Mesh Representation

A 3D mesh $\mathcal{M}$ represents a 3D surface using sets of mesh elements; vertices $\mathcal{V}$, edges $\mathcal{E}$, and polygons (facets) $\mathcal{F}$ along with incidence and/or adjacency relations, $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$. The mesh vertices are 3D points, $\mathcal{V} \subset \Re^3$. Each edge, $e_i \in \mathcal{E}$, is defined by two distinct vertices, $\mathcal{E} \subset \{e_i = \{v_j, v_k\} \mid v_j, v_k \in \mathcal{V}, j \neq k\}$. While each facet, $f_i \in \mathcal{F}$, is defined by three or more edges such that each pair of edges share a vertex. (The vertex is incident to both edges.) In the case of a triangular mesh (which is the most widely used type of mesh due its relative simplicity), the facet f_i is exactly defined by three edges $\mathcal{F} \subset \{f_i = \{e_j, e_k, e_l\} \mid e_j, e_k, e_l \in \mathcal{E}, j \neq k, j \neq l, k \neq l, \forall e_1, e_2 \in f_i \exists v (v \in e_1 \wedge v \in e_2)\}$. Alternatively, each facet is defined by three distinct vertices, $\mathcal{F} \subset \{f_i = \{v_j, v_k, v_l\} \mid v_j, v_k, v_l \in \mathcal{V}, j \neq k, j \neq l, k \neq l\}$. The use of subsets in the

definitions of the mesh elements signifies that (1) the mesh representation of a 3D surface is not unique. In fact, there are several valid mesh representations for the same 3D surface. (2) For a valid (and to a less extent optimal) mesh representation, further constraints should be imposed on the selection of the mesh element sets and their adjacency relations. For 3D meshes representing compact 2-manifold surfaces (which is the case with facial surfaces), each point of the 3D mesh and its neighborhood should be homeomorphic to an open disc and no holes should be introduced (open discs are removed from the representation). Such topological changes occur for example when (1) adding an excessive number of polygons to $\mathcal{F}$ that are incident to a common vertex or edge (2) dropping internal polygons (away from the border) from a valid mesh representation (creates a hole). Additionally, constraints on the angles of the polygons and the lengths of the edges can result in more optimal mesh representations. Polygons with acute angles are not desirable in mesh representation. There are different types of mesh representations, depending on how the data of the mesh are stored and organized in data structures. Pros and cons of some well-known meshes are briefly discussed in the following paragraphs:

Polygon mesh: It is the simplest 3D mesh in which the polygon data are stored in a table. Each row of the table stores the *x*-, *y*- and *z*-coordinates of all vertices of a polygon. As the vertices that are incident to a polygon can also be incident to many other polygons, this mesh type produces redundancies. Moreover, because polygon meshes do not have adjacency information, mesh traversal is not efficient.

Vertex-polygon mesh: To alleviate the redundancy problem of the polygon mesh, the vertices of the mesh are stored in a separate table. Rather than the actual vertices being stored in the polygon table, their indices to the vertex table are stored instead. Nevertheless, this mesh type stores no adjacency information.

Simple adjacency mesh: In this mesh type, three tables are used to store vertices, edges, and polygons. Along with each individual element, the indices to all incident elements are stored. This mesh enables an efficient traversal but requires extra storage (compared with other mesh types), more processing power, and care during manipulation (e.g., split or merge operations) to maintain mesh consistency.

Partial adjacency mesh: This mesh is similar to the simple adjacency mesh with the exception that not all adjacency information is stored, hence reducing the demand for memory and storage. The unprovided adjacency information can

be efficiently inferred from the one provided; thus, the mesh is efficiently traversable, particularly if the facets are constrained to be triangular (as in the case of a triangular mesh).

Triangular mesh: As previously mentioned, the polygons in this mesh are triangles, and it is the most widely used mesh representation because of its simplicity. The triangular mesh usually stores the vertices and the triangles with indices to all neighboring vertices and triangles in separate tables. The edge information is implicit and can be inferred at any vertex (defined by the neighboring vertices) or triangle (its edges).

Winged-edge mesh: The adjacency information in this mesh is centered around the edges, and the mesh is traversed on an edge-by-edge basis. Along with each edge, indices point to the two incident polygons (left and right) and four other edges; the first encountered edges in the clock wise (CW) and the counter-clock wise (CCW) directions at both vertices of the edge (see Figure 2.3). The split-and-merge operations are flexible and efficient for the winged-edge mesh.

3D meshes can store (or refer to) data pertaining to local mesh elements such as texture (as with point cloud pointers to texture maps are used), normals and curvatures. They can also be customized to better suit (from the point of view of flexibility and/or efficiency) the applications at hand. For more variants and extensive discussions of 3D meshes and mesh operations, the reader is refered to the SIGGRAPH course notes by Botsch *et al.* (2007). Well-thought of designs of data structures for 3D polygonal meshes are provided by Kettner (1999).

Range Image Representation

A range image can be defined as a partial binary function, $r : \Re' \times \Re' \rightarrow \Re$, that maps to the range (depth) of a surface point relative to a reference Cartesian frame (often the frame of the 3D acquisition digitizer). The domain of the range image *R* can be the azimuth $\Theta \subset \Re$ and elevation $\Phi \subset \Re$ angles within the view range of the 3D digitizer, $r : \Theta \times \Phi \rightarrow \Re$. Alternatively, it can be the sets of *x*- and *y*-coordinates, $\mathcal{X} \subset \Re$ and $\mathcal{Y} \subset \Re$, that is $r : \mathcal{X} \times \mathcal{Y} \rightarrow \Re$. The range image from a data structure point view is a 2D matrix, **R**, entries of which correspond to the range data, where its horizontal and vertical indices implicitly define the azimuth and elevation angles (for an angle-based range image) or the *x*- and *y*-coordinates (for the *XY*-based range image), see Figure 2.2.

The angle-based range image representation—which can be the default representation of some 3D digitizers because of the direct relationship with the

sensor orientation—suffers from the limitation that it undergoes a perspective-like transformation. The range readings are as if they were projected on a spherical surface in a way similar to a 2D image plane in the case of 2D imaging. In contrast, the *XY*-based range image does not suffer from this transformation, which makes it a better choice for 3D face recognition. Nevertheless, an angle-based range image is capable of representing multi view surfaces (e.g., closed surfaces). A cylindrical form of a range image can represent the 3D surface around one direction, often the azimuth, $r : \Theta \times \mathcal{Y} \rightarrow \Re$.

Conversion from an angle-based to an *XY*-based range representation: The range is first converted to a point cloud representation. Then the point cloud is converted to an *XY*-based range image. Let an $N+1 \times M+1$ matrix, $\mathbf{R}_{\theta\phi}$, represent an angle-based range image with implicit azimuth angles ranging from $\theta_{\min}$ to $\theta_{\max}$, $\Theta = \{\theta_i = \theta_{\min} + \frac{(\theta_{\max}-\theta_{\min})i}{N} \mid i = 0 \ldots N-1\}$, and elevation angles ranging from $\phi_{\min}$ to $\phi_{\max}$, $\Phi = \{\phi_j = \phi_{\min} + \frac{(\phi_{\max}-\phi_{\min})j}{M} \mid j = 0 \ldots M-1\}$. The point cloud representation is shown in Equation 2.10.

(2.7) $x = R_{ij} \cos\phi_j \cos\theta_i,$

(2.8) $y = R_{ij} \sin\phi_i,$

(2.9) $z = R_{ij} \cos\phi_j \sin\theta_i,$

(2.10) $\mathcal{P} = \{p_{i,j} = (x, y, z) \mid (\theta_i, \phi_j) \in \Theta \times \Phi\}.$

Conversion from a point cloud to an *XY*-based range representation: First, the resolution of the *XY*-based range image $\mathbf{R}_{xy}$ is decided, let it be $N-1 \times M-1$. The implicit information about the *x*- and *y*-coordinates are then decided. One choice is to let the horizontal indices represent the set $\mathcal{X}$ of the *x*-coordinates varying from minimum $x_{\min} = \min_x \mathcal{P}$ to maximum $x_{\max} = \max_x \mathcal{P}$, i.e., $\mathcal{X} = \{x_i = x_{\min} + \frac{(x_{\max}-x_{\min})i}{N} \mid i = 0 \ldots N-1\}$. Similarly, the vertical indices represent the $\mathcal{Y}$-coordinates, $\mathcal{Y} = \{y_j = x_{\min} + \frac{(x_{\max}-x_{\min})j}{M} \mid j = 0 \ldots M-1\}$. The range image pixels correspond to the implicit *x*- and *y*-coordinates according to $R_{ij} = (x_i, y_j) \in \mathcal{X} \times \mathcal{Y}$. The pixel values are the interpolation of the range (*z*-coordinate) at the implicit *x*- and *y*-coordinates. Those pixels not in the 2D convex hull formed by the neighboring *x*- and *y*-coordinates (of 3D points in $\mathcal{P}$) are masked out because not all the implicit *x*- and *y*-coordinates correspond to the 3D surface (or 3D points in $\mathcal{P}$).

Normal Map Representation

A normal map representation can be defined as a partial binary function that maps the horizontal and vertical coordinates to unit normal vectors (or tuples), $\mathbf{n} : \Re' \times \Re' \rightarrow \Re^3$. Similar to range images, the domain of **n** can correspond to the *x*- and *y*-coordinates, $\mathcal{X} \times \mathcal{Y}$. Similar to the case in range images, other 2D parameterizations of the domain can be used with the normal map representation such as the spherical (angle-based, $\Theta \times \Phi$) or the cylindrical ($\Theta \times \mathcal{Y}$) domain. A suitable data structure for a normal map representation **N** is a three-channel matrix (i.e., $N+1 \times M+1 \times 3$). As in the case of range images, the horizontal and vertical matrix indices correspond to the domain. Equally, the normal vectors $\mathbf{N}_{ij}$ can be computed from the positions of the 3D points and their neighbors while the surface is in a point cloud, a 3D mesh, or a range image representation.

For a 3D mesh representation, a normal vector is computed at each polygonal facet f_i by taking the (normalized to unit length) cross product of any two edges of f_i that are incident to a common vertex (e_1=(v_1, v_c), e_2=(v_2, v_c)) as shown in Equation 2.11. Depending on the right hand rule of the cross product the normal can point inward or outward from the surface, where the latter is more appropriate for 3D surface representation.

$$\mathbf{n}_i = \pm \frac{(\mathbf{v}_1 - \mathbf{v}_c) \times (\mathbf{v}_2 - \mathbf{v}_c)}{\|(\mathbf{v}_1 - \mathbf{v}_c) \times (\mathbf{v}_2 - \mathbf{v}_c)\|}. \quad (2.11)$$

The normals at the mesh vertices can be estimated from the facet normals (located the at centroids of the facets) using a weighted sum (according to the distances between the facet centroids and vertices) of all the normals of the facets incident to the vertex or using any other interpolation technique. A normal map can then be computed from the vertex normals in a similar manner to the (previously discussed) conversion from point clouds to range images, with the difference that the normals are used instead of the range values. The normal maps are also used with 3D meshes (typically in computer graphics) where they are referenced from the mesh elements (similar to the case of texture maps as explained earlier).

The normal map can reliably be computed from an *XY*-based range image representation, *R*. The normal $\mathbf{n}_{ij}$ is the normalized cross product between $[1, 0, \frac{\partial}{\partial x} R_{ij}]^\top$ and $[0, 1, \frac{\partial}{\partial y} R_{ij}]^\top$ is shown in Equation 2.12.

$$\mathbf{n}_{ij} = \frac{\left[-\frac{\partial R_{ij}}{\partial x}, -\frac{\partial R_{ij}}{\partial y}, 1\right]^\top}{\left\|\left[-\frac{\partial R_{ij}}{\partial x}, -\frac{\partial R_{ij}}{\partial y}, 1\right]^\top\right\|}. \quad (2.12)$$

All that is needed to compute the normals is the gradient of the range image, ∇R. A sound approach (which is less prone to noise) for finding the gradient of *R* is to convolve it with kernels of partial derivatives of the Gaussian function $G(x, y, \sigma)$, that is,

$$\nabla R = \left(\frac{\partial R}{\partial x}, \frac{\partial R}{\partial y}\right) \simeq \left(\frac{\partial (G * R)}{\partial x}, \frac{\partial (G * R)}{\partial y}\right) = \left(\frac{\partial G}{\partial x} * R, \frac{\partial G}{\partial y} * R\right).$$

The normals of a 3D surface vary when the surface is transformed, although translation and rigid scaling do not affect the orientation of the normals. Instead of recalculating the normals, the original normals can be transformed accordingly. Let **M** be a linear 3×3 transformation matrix (not involving translation). It can be shown that the transformed surface normal equals to the multiplication of the transposed inverse of **M** by the original normal, that is, $\mathbf{n}' = \mathbf{M}^{-1\top}\mathbf{n}$. The linear transformations are discussed in subsection 2.1.2.

2.1.2 Rigid 3D Transformations

3D rigid transformations (namely rotations, translations, or combinations of both) are special cases of linear transformation. Let the vector $\bar{\mathbf{v}}$ represent a 3D point augmented by one, $\bar{\mathbf{v}} = [x\ y\ z\ 1]^\top$, the vector $\bar{\mathbf{v}}'$ represents the linear transformation of $\bar{\mathbf{v}}$, and the vector $\mathbf{t} = [t_x\ t_y\ t_z]^\top$ represents the translation (the displacement of the surface points in the *x*-, *y*- and *z*-directions). The linear transformation is expressed as in Equation 2.13.

$$\bar{\mathbf{v}}' = \mathbf{A}\bar{\mathbf{v}} = \begin{bmatrix} \mathbf{M} & \mathbf{t} \\ \mathbf{o}^\top & 1 \end{bmatrix} \bar{\mathbf{v}}, \quad (2.13)$$

where $\mathbf{o}$ is a vector of zeros, $[0\ 0\ 0]^\top$, and the matrix **A** is square (4×4). Note that it is possible to represent the same linear transformation using a non-square matrix (3 × 4) or just using a 3×3 matrix by multiplying **M** by the point vector then adding the translation vector, $\mathbf{v}' = \mathbf{M}\mathbf{v} + \mathbf{t}$. However, the use of a 4×4 matrix makes it possible to represent a sequence of linear transformations through matrix multiplication, $\mathbf{A}_t = \mathbf{A}_n \ldots \mathbf{A}_2\mathbf{A}_1$, where $\mathbf{A}_i$'s represent successive rigid transformations culminating into a total rigid transformation represented by $\mathbf{A}_t$.

In the most general case of Equation 2.13 (when the values of $\mathbf{M}$ and $\mathbf{t}$ are

arbitrary), the transformation is affine. Applying an affine transformation on a 3D surface may change its shape (including the scale and the shear), its orientation (due to rotations) and its location (due to translations). However, the parallelism between lines (formed by surface points) is preserved. The special case when $\mathbf{M}$ is orthonormal ($\mathbf{M}^{\top} = \mathbf{M}^{-1}$ or equivalently $\mathbf{I} = \mathbf{M}\mathbf{M}^{\top}$) and satisfies $\det(\mathbf{M}) = 1$ results in a rigid transformation. In this case, the matrix $\mathbf{M}$ is called a rotation matrix and is denoted by $\mathbf{R}$, and for such transformation (rigid) only the orientation and the location of the surface may change while the surface shape is preserved. When the matrix **M** is equal to an identity matrix $\mathbf{I}$, it corresponds to no rotations, but the surface translates according to $\mathbf{t}$.

As indicated by Euler's rotation theorem, any arbitrary (or general) rotation can equally be achieved by a sequence of three rotations about the *x*-, *y*- and *z*-axes, each represented by the Euler angles α, β, and γ. The (total) rotation matrix corresponding to these parameters $\mathbf{R}_t$ is given by the following equations, where $\mathbf{R}_{\alpha}$, $\mathbf{R}_{\beta}$, and $\mathbf{R}_{\gamma}$ are the rotation matrices about *x*-, *y*- and *z*-axes.

$$\mathbf{R}_{\alpha} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \tag{2.14}$$

$$\mathbf{R}_{\beta} = \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \tag{2.15}$$

$$\mathbf{R}_{\gamma} = \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.16}$$

The overall rotation $\mathbf{R}_t$ is given by

$$\mathbf{R}_t = \mathbf{R}_{\gamma}\mathbf{R}_{\beta}\mathbf{R}_{\alpha} \tag{2.17}$$

The total rotation ($\mathbf{R}_t$) depends on the order in which the rotation Euler angles were applied. The use of Euler's angles to generate a rotation matrix also suffers from the gimbal lock problem. For some values of the rotation angles, the degree of freedom becomes lower than what it should be. For example, when the angle $\beta = 0$, the degree of freedom of the Euler's rotation decreases from two (the expected) to one as the rotation about the *x*-axis becomes equivalent to a rotation about the *z*-axis (α or γ becomes redundant). Nevertheless, the resulting rotation matrix remains mathematically correct.

Another way of representing rotation matrices is through the use of unit quaternion rotation, a representation that is not affected by the disadvantages of

using Euler's rotation. The unit quaternion rotation is equivalent to a rotation of the 3D points by angle θ about unit vector $\mathbf{u}$ and is represented by unit quaternion $\mathbf{q} = [x\ y\ z\ w]^\top$. Quaternions are a 4D extension of complex numbers. A quaternion is a combination of a real number, *w*, and weighted three imaginary numbers, namely, **q**=*w*+*xi*+*yj*+*zk*. In a parallel way to the ordinary (2D) complex numbers, the conjugate of **q** is defined as $\mathbf{q}^* = w - xi - yj - zk$ and the magnitude of the quaternion is defined as $\|\mathbf{q}\| = \sqrt{\mathbf{q}\mathbf{q}^*} = \sqrt{w^2 + x^2 + y^2 + z^2}$. The complex numbers *i*, *j*, and *k* are related by the fundamental formulae $i^2=j^2=k^2=ijk=-1$. The multiplication of quaternions is not commutative: $\mathbf{q}_1\mathbf{q}_2 \neq \mathbf{q}_2\mathbf{q}_1$. From the fundamental formulae and the non commutative property, the following complex products derive *ij*=*k*, *ji*=−*k*, *jk*=*i*, *kj*=−*i*, *ki*=*j* and *ik*=−*j*. On the basis of that, the product of $\mathbf{q}_1 = [x_1\ y_1\ z_1\ w_1]^\top$ and $\mathbf{q}_2 = [x_2\ y_2\ z_2\ w_2]^\top$ is shown in Equation 2.18.

$$\mathbf{q}_1\mathbf{q}_2 = \begin{array}{l} w_1w_2 - x_1x_2 - y_1y_2 - z_1z_2 \\ +(w_1x_2 + x_1w_2 + y_1z_2 - z_1y_2)i \\ +(w_1y_2 - x_1z_2 + y_1w_2 + z_1x_2)j \\ +(w_1z_2 + x_1y_2 - y_1x_2 + z_1w_2)k \end{array}. \tag{2.18}$$

When representing the quaternions by 4×4 matrices (as in Eq. 2.19), matrix addition and multiplication are equivalent to quaternion addition and multiplication, respectively, and the quaternion conjugate is the matrix transposition.

$$\mathbf{q} = \begin{bmatrix} w & x & y & z \\ -x & w & -z & -y \\ -y & z & w & -x \\ -z & -y & x & w \end{bmatrix}. \tag{2.19}$$

The rotation about unit 3D vector $\mathbf{u}$ by angle θ is represented by the unit quaternion $\mathbf{q} = [\mathbf{u}^\top \sin\frac{\theta}{2}\ \cos\frac{\theta}{2}]^\top$. The rotation of 3D point $\mathbf{v} = [x\ y\ z]^\top$ is shown in Equation 2.20, where the 3D points before and after the rotation are represented by the quaternions $\mathbf{p} = [\mathbf{v}^\top\ 0]^\top$ and $\mathbf{p}' = [\mathbf{v}'^\top\ 0]^\top$, respectively

$$\mathbf{p}' = \mathbf{q}\mathbf{p}\mathbf{q}^{-1} = \mathbf{q}\mathbf{p}\mathbf{q}^*. \tag{2.20}$$

The rotation resulting from the unit quaternion, **q**, is equivalent to that given by the rotation matrix, **R**, as shown in Equation 2.21.

$$\mathbf{R} = \begin{bmatrix} w^2 + x^2 - y^2 - z^2 & 2xy - 2wz & 2xz + 2wy \\ 2xy + 2wz & w^2 - x^2 + y^2 - z^2 & 2yz - 2wx \\ 2xz - 2wy & 2yz + 2wx & w^2 - x^2 - y^2 + z^2 \end{bmatrix}. \tag{2.21}$$

2.1.3 Decimation of 3D Surfaces

Decimation refers to the process of reducing the resolution of digitized signals or, in our case, 3D surface scans/representations. The resolution has impacts on the accuracy, memory storage, and the computational complexity of 3D face recognition. A high resolution representation enables the capture of facial surface details and generally yields to a better recognition accuracy unless it is beyond the finest details. However, in the case of overly high resolution, in addition to the high storage requirement and computational burden, a diminished or even adverse effect on the recognition accuracy may result (possibly as result of the curse of high dimensionality). Acquired facial surfaces are usually of a higher and possibly varying resolution (the resolution depends on the distance of the face from the 3D digitizer). By decimating the 3D surface, a compromise between these requirements can be achieved. Generally, a resolution of 1 or 0.5 mm suffices for 3D face recognition.

Mesh Decimation

Mesh decimation, or mesh simplification as it is sometimes referred to, aims to reduce the number of mesh elements such that the reduced mesh remains as approximate to the original mesh as possible. An overview of some well-known of the various approaches to mesh decimation are provided here. For more in-depth discussions and a wider exposure to mesh decimation algorithms, the reader is referred to the surveys by Botsch *et al.* (2007), Heckbert and Garland (1997), Luebke (2001), and Renze and Oliver (1996).

Mesh decimation algorithms can be contrasted from each other depending on the following factors: iterativeness (iterative versus non iterative), ability to preserve the mesh topology, faithfulness to preserve the mesh details, error measures used to prioritize elements removal or quantification of the closeness of the decimated mesh to the original mesh, regularity (as opposed to adaptiveness) of the decimated mesh, and/or efficiency.

Vertex clustering: Vertex clustering was proposed by Rossignac and Borrel (1993) for the rendering of scenes in computer graphics. Vertex clustering is fast and produces regular decimated meshes that can be of large decimation ratios. Despite the relatively low mesh quality and the possibility of introducing topological changes (e.g., producing handles), the algorithm has potential applications in 3D face recognition. It can be used, for example, along with suitable feature extraction techniques (that are tolerant to its drawbacks) to

perform fast rejection classification in which matching gallery facial surfaces are short-listed not only for a more accurate but also a more computationally expensive classifier.

The 3D space around the surface is first divided into regular cells (a 3D grid). The vertices of the original surface that are within each cell is replaced by a representative vertex associated with that individual cell. When increasing the size of the grid, more vertices are likely to fall into each cell intersected by the surface; hence, the decimation ratio is increased. The accuracy of the decimation is guaranteed to be less than the size of the cell. The mesh edges and triangles that have more than one incident vertex falling into a cell become degenerate. Those elements are removed, and the mesh representation is adjusted accordingly. The representative vertices are possibly selected as the centers of the cells, means or medians of the vertices falling into the cells. However, a more accurate result can be achieved when assigning the representative vertices on the basis of the quadratic error metric (QEM). (Garland and Heckbert, 1997). The QEM minimizes the squared distances from the optimal vertex to the planes $\mathcal{P}$ on which the triangles incident to the vertex reside. Let the *i*th plane be $a_i x+b_i y+c_i z+d=0$, where $\|[a_i b_i c_i]\| = 1$ is represented by the vector $\mathbf{p}_i = [\mathbf{n}_i^\top d_i]^\top$, and the vertex position $\bar{\mathbf{v}}$ equals $[\mathbf{v}^\top 1]^\top$. The optimal vertex position $\bar{\mathbf{v}}_o$ is shown as the following:

$$\bar{\mathbf{v}}_o = \arg\min_{\bar{\mathbf{v}}} \sum_{\forall \mathbf{p}_i \in \mathcal{P}} (\mathbf{p}_i^\top \bar{\mathbf{v}})^\top \mathbf{p}_i^\top \bar{\mathbf{v}} \tag{2.22}$$

$$= \arg\min_{\bar{\mathbf{v}}} \sum_{\forall \mathbf{p}_i \in \mathcal{P}} \bar{\mathbf{v}}^\top \mathbf{p}_i \mathbf{p}_i^\top \bar{\mathbf{v}} \tag{2.23}$$

$$= \arg\min_{\bar{\mathbf{v}}} \bar{\mathbf{v}}^\top \Big(\sum_{\forall \mathbf{p}_i \in \mathcal{P}} \mathbf{p}_i \mathbf{p}_i^\top \Big) \bar{\mathbf{v}} \tag{2.24}$$

$$= \arg\min_{\bar{\mathbf{v}}} \bar{\mathbf{v}}^\top \mathbf{Q} \bar{\mathbf{v}}. \tag{2.25}$$

By taking partial derivatives of $\bar{\mathbf{v}}^\top \mathbf{Q} \bar{\mathbf{v}}$ with respect to the *x*-, *y*- and *z*-coordinates of $\bar{\mathbf{v}}$ and equating to zero, it can be shown (based on Equation 2.25) that the optimal vertex position is shown as the following closed form solution, where q_{ij} is is the *ij*th element in **q**.

$$\bar{\mathbf{v}}_o = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}. \tag{2.26}$$

Vertex removal: The algorithm by Schroeder *et al.* (1992) is the first and

probably the most well-known in this class of algorithms. In general, mesh decimation algorithms of this class iteratively remove vertices on the basis of some metric criteria. The removal of a vertex results in a surface hole, which is then filled by retriangulating the surface patch corresponding to the hole. (This can easily be achieved by iteratively splitting the patch until all of the splits become triangles.) The mesh is then updated accordingly. Schroeder *et al.* used the distance from a candidate vertex to a local plane (computed by taking average of all the triangles incident to vertex weighted by their areas) and the distance from the mesh edge (for near border vertices) as the criteria for prioritizing vertex removal; vertices with the least distances are removed first. Iterative decimation is terminated when a certain decimation ratio is achieved or a certain decimation error is exceeded. This decimation algorithm is fast and can adapt (adjust) vertex density to the level of local surface details. Additionally, it can preserve the topology of the surface.

Edge contraction: Similar to the vertex removal approach, mesh edges can also be removed by replacing the two vertices that are incident to an edge by a single vertex. The triangles that are incident to the edge become degenerate and are then removed from the mesh. The position of the new vertex can be chosen to be either one of the edge vertices. However, more quality is achieved by taking their average or by choosing an optimal position (on the basis of a metric measure). Besides lowering the decimation error, the later choice also has a low pass filtering effect that avoids mesh aliasing. They are also generally similar to vertex-removal approaches in their iterative nature and the use of metric measures to prioritize edge contractions. Unlike vertex-removal algorithms, the mesh topology is not preserved. However, this can sometimes be advantageous. For example, a surface hole can be filled by contracting the edges around it. A number of edge contraction algorithms have been proposed that mostly differ by the metric measures they use. An example of an edge contraction algorithm was developed by Garland and Heckbert (1997) for which the concept QEM (discussed earlier) was initially proposed.

Range Image Decimation

As with 2D images, the simplest and the most efficient approach to range image decimation is to retain every *ri*th horizontal and *rj*th vertical range pixel and drop the outstanding vertices, where *R*, *i*, and *j* are the decimation parameter, the horizontal, and vertical indices of the decimated range pixels. If *R* is a non-integer, then indices *ri* and *rj* can be also non-integers, which can be then

rounded to their nearest integer or interpolated at the location corresponding to the *i*th and *j*th pixel of the decimated range image. However, this approach could compromise the quality of the decimated range image. In fact, the dropped pixels can be important (depending on the local shape of the surface and the level of noise). Additionally, the decimation of range images with high frequency components (abrupt shape variations) can results in shape aliasing. A better decimation quality can be achieved by introducing low pass filtering before decimation. Alternatively, the decimation and low pass filtering are usually performed in one step as in Equation 2.27, where R, R', and H are the range image, the decimated range image, and a low pass kernel (typically a Gaussian).

$$R'(i,j) = \sum_{k,l} R(k,l)H(ri, rj). \quad (2.27)$$

2.1.4 Geometric and Topological Aspects of the Human Face

The recognition of 3D human faces can be categorized as within-class object recognition. However, some geometric and topological aspects that are specific to the human face can pose challenges and/or provide opportunities to enhance the performance of face recognition systems. These aspects and their relevance to 3D face recognition are discussed in the following sections.

The Nonrigidity of the Human Face

The 3D shape of the human face is highly deformable as a result of aging, weight loss/gain and most prominently following variations in facial expressions. These variations pose challenges to 3D face recognition as they can obscure those arising from identity variations (shape variations among different individuals) on the basis of which individuals are recognized. This is because the 3D shape variations of the human face among different individuals are statistically small. In fact, some facial expressions, in addition to the geometric changes, can induce topological changes to the 3D facial surface, such as those involving mouth opening. Nevertheless, the nonrigidity of the human face is not arbitrary, particularly the one related to facial expressions because the anatomical structure of the face remains unchanged. This factor makes the modeling of the facial expressions or the extraction of expression-invariant features for 3D face recognition possible. Another aspect of the nonrigidity due to facial expressions is that some facial regions such as the nose and the forehead

(called the semirigid regions) are (to an extent) less affected by facial expressions. In addition, some facial regions (possibly other than the semi rigid ones) may be less deformed than some others, depending on the facial expression of the face. The two latter aspects have allowed for the development of *rigid* approaches to 3D face recognition (where the 3D face or some of its parts are treated as if they are rigid) that are invariant to facial expressions.

The Symmetry of the Human Face

The 3D shape of the human face is bilaterally symmetric about a plane splitting the face into two mirror, left and right, halves (to a large extent). Interestingly, facial symmetry holds considerably not only for faces under neutral expression but also for most facial expressions, especially those that naturally express emotional states or result from talking. The symmetry of the face has been employed by many 3D face recognition systems for a number of tasks. It was used for pose estimation and correction of 3D faces (e.g., Pan and Wu, 2005), whereby they estimate the facial symmetry plane and define reference points on the symmetry profile (i.e., the intersection between the symmetry plane and the 3D face). It was also used for feature extraction or dimensionality reduction of the facial data (e.g., Gnanaprakasam et al., 2010; Harguess et al., 2008). Other applications in which facial symmetry was exploited include, and is not limited to, the interpolation of holes in raw facial data (for a better estimation of the missing data), the localization of the fiducial points (for more robustness and accuracy), the detection of the face (for an enhanced detection efficiency), and the normalization of illumination of textured 3D faces. Nevertheless, the asymmetry of the human face can also be of importance to face recognition as it may be person specific.

Fiducial Points of the Human Face

There are natural landmark points on the facial surface called the fiducial points, which can be detected even when the face is deformed. Typical fiducial points include the eye corners, the mid point between the eyes, the tip of the nose and its two lower corners, the furthest chin point, and mouth corners. Sometimes, it is needed to establishing point-to-point correspondence between two or more facial scans of the same person or of different people, as is the case in some approaches to expression modeling and expression invariant face recognition. For rigid surfaces, an accurate point-to-point correspondence is possible (e.g., by

using the ICP algorithm). However, for deformable surfaces, the establishment of such correspondences is more difficult. The problem can be undermined by first establishing correspondences between the fiducial points of the facial scans. The fiducial points are also used as standard locations from which local features can be extracted for 3D face recognition. Additional features can be extracted from their relative locations for further enhancement of the recognition.

Self-Occlusions of the Human Face

Depending on the pose of the 3D face relative to the viewpoint of the 3D digitizer, parts of the scanned facial surface can be self-occluded because some of the visible regions block the line of sight between the 3D digitizer and the occluded regions. As a result, the occluded facial surface regions cannot be digitized and will appear as holes in the single view scan (often referred to as 2.5D scan). In profile facial scans, where the face is scanned from roughly either the left or right side, all or large parts of the opposite side can be self-occluded. The side of the nose, which is considered to be a highly discriminative region of the face, is easily occluded even for a moderate side pose. On the other hand, facial scanning from frontal or near frontal viewpoints is least affected by self-occlusions. In this case, all or nearly all of the facial regions are visible to the 3D digitizer. Additionally, near frontal scanning is less susceptible to occlusions by hair and clothing.

On that basis, the frontal facial pose is the choice of most recognition systems that match 2.5D probes scans against a gallery of 2.5D scans. The recovering of occluded regions is possible by performing a scan from different viewpoints. From these multiple scans of the face, a complete facial model can be generated from ear to ear. The online generation of complete models (as probes) is not practical for most applications (complete against complete model matching). However, matching 2.5D probes against complete gallery models is more practical (offline model completion) and reduces the effects of occlusions (limited to the probes) and allows for the matching of both frontal and profile 2.5D probes.

2.2 Curvatures Extraction from 3D Face Surface

3D facial surface curvatures (or curvature-like quantities) have been widely used in 3D face recognition systems. Some 3D face preprocessing approaches (e.g., surface smoothing), 3D face or subregion segmentation, feature extraction, and

the detection of facial fiducial points rely on some (or all) of the different types of surface curvatures. This section first discusses the concepts and mathematical definitions of 3D curvatures and then covers some practical methods for their extraction. For further information on this topic, the reader is referred to textbooks on differential geometry such as those by O'Neill (2006) and Bar (2010).

2.2.1 Theoretical Concepts on 3D Curvatures

Curvature of 3D Curves

The curvature is a measure of how much a curve is bent. In other words, the curvature at a point on the curve is the extent at which the curve locally deviates from the tangent at the point. Curvature κ is quantified as $\pm\frac{1}{R}$, where *R* is the radius of the osculating circle (that locally fits the curve). For a parametric curve embedded in a 3D space, $\gamma = [x(t)\ y(t)\ z(t)]^\top$, the curvature is described as

$$\kappa = \frac{\|\ddot{\gamma} - \ddot{\gamma}\cdot\hat{\dot{\gamma}}\|}{\|\dot{\gamma}\|^2} \tag{2.28}$$

$$= \frac{\|\dot{\gamma}\times\ddot{\gamma}\|}{\|\dot{\gamma}\|^3}, \tag{2.29}$$

where $\dot{\gamma}$ and $\ddot{\gamma}$ are the first and second derivatives of the curve with respect to the parameter *t*. The unit vector $\hat{\dot{\gamma}}$ is the curve tangent, $\hat{\dot{\gamma}} = \dot{\gamma}/\|\dot{\gamma}\|$. The vector $\ddot{\gamma} - \ddot{\gamma}\cdot\hat{\dot{\gamma}}$ is orthogonal to the tangential vector, and when normalized to a unit magnitude, it defines the principal normal of the curve. In theory, 3D curves can be (re)-parameterized so they are unit-speed, that is, $\|\dot{\gamma}\| = 1$. The curvature of unit speed curves simplifies to $\|\ddot{\gamma}\|$, and the principal normal to $\hat{\ddot{\gamma}}$. It should be noted that both the osculated circle and the principal normal are coplanar and the curvature κ is invariant to rigid transformations.

Curvatures of Curves on 3D Surfaces

For curves on 3D surfaces, the second derivative vector of the parametric curve defines two types of curvatures: the geodesic and the normal curvatures. Let $\sigma = [x(u,v)\ y(u,v)\ z(u,v)]^\top$ be a 3D surface parameterized by *u* and *v*. A further parameterization of the surface parameters, with respect to a single parameter *t*, yields a 3D curve on the surface σ, that is,

$\gamma = [x(u(t), v(t))\ y(u(t), v(t))\ z(u(t), v(t))\ ]^\top$. The surface normal $\mathbf{n} = \frac{\sigma_u \times \sigma_v}{\|\sigma_u \times \sigma_v\|}$, where σ_u and σ_v are the derivatives of the surface with respect to *u* and *v* (the two vectors span the tangential plane). The tangent vector of the curve is $\mathbf{t} = \hat{\dot{\gamma}}$, differentiated with respect to *t*. The geodesic and the normal curvature are shown in Equations 2.30 and 2.31.

(2.30) $$\kappa_g = \frac{\ddot{\gamma}}{\|\dot{\gamma}\|^2} \cdot (\mathbf{n} \times \mathbf{t}).$$

(2.31) $$\kappa_n = \frac{\ddot{\gamma}}{\|\dot{\gamma}\|^2} \cdot \mathbf{n}.$$

The curvatures κ_g and κ_n are a quantification of how the curve is bent within and away from the tangential plane of the surface. The normal curvature. κ_n. has a particular importance in 3D face recognition because it is a characteristic of the surface in the tangential direction of the curve, $\mathbf{t}$. While κ_g is solely a characteristic of the 3D curve (as opposed to the surface). From the preceding two equations (Eqs. 2.30 and 2.31) and Equations 2.28 and 2.29, the total curvature κ of the curve is related to κ_g and κ_n by $\kappa = \sqrt{\kappa_g^2 + \kappa^2}$.

Principal curvatures: The principal curvatures are two unique normal curvatures (Eq. 2.31) corresponding the curvature of the curve resulting from the intersection of the 3D surface with a normal plane (orthogonal to the tangential plane) in the tangential directions that give the minimum κ_1 and maximum κ_2 normal curvatures. While normal curvatures have pertinence to both the surface and the parameterization of the curve on the surface, which can yield tangents with one degree of freedom (revolution around surface normal), the two principal curvatures are a sufficient description of the local surface around a point. In fact, the local surface can be locally approximated using the principal curvatures up to the rigid transformations using the paraboloid $z = \frac{1}{2}(\kappa_1 x^2 + \kappa_2 y^2)$.

The principal curvatures are given in terms of the first and second fundamental forms as follows (starting from the Eq. 2.31):

(2.32) $$\|\dot{\gamma}\|^2 \kappa_n = \ddot{\gamma} \cdot \mathbf{n}.$$

(2.33) $$= \frac{\mathrm{d}}{\mathrm{d}t}\dot{\gamma} \cdot \mathbf{n}.$$

$$= \frac{\mathrm{d}}{\mathrm{d}t}\left(\sigma_u \frac{\mathrm{d}u}{\mathrm{d}t} + \sigma_v \frac{\mathrm{d}v}{\mathrm{d}t}\right) \cdot \mathbf{n}.$$

$$= \left[\sigma_u \frac{\mathrm{d}^2 u}{\mathrm{d}t^2} + \sigma_v \frac{\mathrm{d}^2 v}{\mathrm{d}t^2} + \left(\sigma_{uu} \frac{\mathrm{d}u}{\mathrm{d}t} + \sigma_{uv} \frac{\mathrm{d}v}{\mathrm{d}t}\right) \frac{\mathrm{d}u}{\mathrm{d}t}\right. \tag{2.34}$$

$$\left. + \left(\sigma_{uv} \frac{\mathrm{d}u}{\mathrm{d}t} + \sigma_{vv} \frac{\mathrm{d}v}{\mathrm{d}t}\right) \frac{\mathrm{d}v}{\mathrm{d}t}\right] \cdot \mathbf{n}. \tag{2.35}$$

In Equation 2.34, the tangent vector (to the curve) $\dot{\gamma}$ is expressed as a weighted combination of the surface tangents σ_u and σ_v, where the weights are $\frac{\mathrm{d}u}{\mathrm{d}t}$ and $\frac{\mathrm{d}v}{\mathrm{d}t}$. Some terms in Equation 2.35 are tangential, so their dot product with the surface normal reduces to zero. Equation 2.35 is expressed in terms of the first and second fundamental forms, $\mathcal{F}_1$ and $\mathcal{F}_2$, and the weight vector $\mathbf{w} = [\frac{\mathrm{d}u}{\mathrm{d}t} \;\; \frac{\mathrm{d}v}{\mathrm{d}t}]^\top$.

$$\|\dot{\gamma}\|^2 \kappa_n = \dot{\gamma} \cdot \dot{\gamma} \kappa_n = \mathbf{w}^\top \begin{bmatrix} \sigma_{uu} \cdot \mathbf{n} & \sigma_{uv} \cdot \mathbf{n} \\ \sigma_{uv} \cdot \mathbf{n} & \sigma_{vv} \cdot \mathbf{n} \end{bmatrix} \mathbf{w}. \tag{2.36}$$

$$\left(\sigma_u \frac{\mathrm{d}u}{\mathrm{d}t} + \sigma_v \frac{\mathrm{d}v}{\mathrm{d}t}\right) \cdot \left(\sigma_u \frac{\mathrm{d}u}{\mathrm{d}t} + \sigma_v \frac{\mathrm{d}v}{\mathrm{d}t}\right) \kappa_n = \mathbf{w}^\top \begin{bmatrix} \sigma_{uu} \cdot \mathbf{n} & \sigma_{uv} \cdot \mathbf{n} \\ \sigma_{uv} \cdot \mathbf{n} & \sigma_{vv} \cdot \mathbf{n} \end{bmatrix} \mathbf{w}. \tag{2.37}$$

$$\mathbf{w}^\top \begin{bmatrix} \sigma_u \cdot \sigma_u & \sigma_u \cdot \sigma_v \\ \sigma_u \cdot \sigma_v & \sigma_v \cdot \sigma_v \end{bmatrix} \mathbf{w} \kappa_n = \mathbf{w}^\top \begin{bmatrix} \sigma_{uu} \cdot \mathbf{n} & \sigma_{uv} \cdot \mathbf{n} \\ \sigma_{uv} \cdot \mathbf{n} & \sigma_{vv} \cdot \mathbf{n} \end{bmatrix} \mathbf{w}. \tag{2.38}$$

$$\mathbf{w}^\top \begin{bmatrix} E & F \\ F & G \end{bmatrix} \mathbf{w} \kappa_n = \mathbf{w}^\top \begin{bmatrix} L & M \\ M & N \end{bmatrix} \mathbf{w}. \tag{2.39}$$

$$\mathbf{w}^\top \mathcal{F}_1 \mathbf{w} \kappa_n = \mathbf{w}^\top \mathcal{F}_2 \mathbf{w}. \tag{2.40}$$

Using the Equation 2.40, the normal curvature can be computed in any tangential direction specified by $\mathbf{w}$, $\kappa_n = \frac{\mathbf{w}^\top \mathcal{F}_2 \mathbf{w}}{\mathbf{w}^\top \mathcal{F}_1 \mathbf{w}}$. Also, on the basis of Equation 2.40, Equation 2.41 holds indicating that the maximum and minimum principal curvatures are the eigenvalues of ${\mathcal{F}_1}^{-1} \mathcal{F}_2$.

$$\mathbf{w} \kappa_n = {\mathcal{F}_1}^{-1} \mathcal{F}_2 \mathbf{w}. \tag{2.41}$$

Mean curvature: The mean curvature *H* is defined as the mean of the maximum and minimum normal curvatures, $H = \frac{\kappa_1 + \kappa_2}{2}$. Since κ_1 and κ_2 are the eigenvalues of $\mathcal{F}_1$ and $\mathcal{F}_2$, the mean curvature is also shown by

$$H = \frac{1}{2} \operatorname{trace}\left({\mathcal{F}_1}^{-1} \mathcal{F}_2\right). \tag{2.42}$$

Gaussian curvature: The Gaussian curvature *k* is also defined based on κ_1 and κ_2, $K = \kappa_1 \kappa_2$. In terms of $\mathcal{F}_1$ and $\mathcal{F}_2$, the Gaussian curvature is given by

$$K = \det\left({\mathcal{F}_1}^{-1} \mathcal{F}_2\right) = \det\left({\mathcal{F}_1}^{-1}\right) \det(\mathcal{F}_2). \tag{2.43}$$

Another approach of finding *k* is to use a Gaussian map. The Gaussian map is a continuous mapping from the 3D surface σ to unit sphere S^2 (representing normals), each point on the surface is mapped to a point on the unit sphere that represents the surface normal. For an infinitely small local surface patch around a point (with a disc diameter *R*), the absolute value of *k* is given by the ratio of the corresponding area on the unit sphere A_u to the area of the surface patch A_σ.

(2.44) $$|K| = \lim_{r \to 0} \frac{A_u}{A_\sigma}.$$

It should be noted that from both *H* and *k*, the principal curvatures κ_1 and κ_2 can be computed. Therefore, similar to the principal curvatures, the mean and Gaussian curvatures together fully characterize an infinitely small surface patch.

(2.45) $$\kappa_1 = H + \sqrt{H^2 - K}.$$

(2.46) $$\kappa_2 = H - \sqrt{H^2 - K}.$$

2.2.2 Practical Curvature Extraction Methods

As explained in our theoretical discussions about surface curvatures earlier, the values of the curvatures depend on the first and second derivatives of the surface (with respect to the surface parameters). While it is possible to find numerically those derivatives for both 3D point clouds and meshes, they are susceptible to noise, particularly the second derivatives. One way to overcome derivative noise is to (heavily) smooth the surface. For range images, it is more convenient to differentiate a Gaussian kernel (with a low cut off frequency) and convolve it with the range image. However, a heavy smoothing may remove surface details and affect the accuracy of estimating curvatures at locations where the surface is highly curved, which in turn might affect subsequent surface segmentation and recognition modules.

Another approach of finding estimating/calculating surface curvatures at a 3D point is to fit an analytical surface to the local patch (or points) around the point. The curvatures are then computed from the analytical surface. Fitting a truncated Taylor expansion (up to the second order), one to each of *x*-, *y*- and *z*-coordinates of the 3D points (Eqs. 2.47, 2.48, and 2.49), is widely used. This approach can equally be used with 3D point clouds, meshes, and range images. However, for 3D point clouds and meshes, the fitting of the analytical surface to the local 3D points requires a parameterization of the surface. A simple parameterization approach is to project the local 3D points on the tangential plane. Their positions on the tangential plane are then expressed in any orthonormal coordinates, *u* and

v.

$$x = c_{x0} + c_{x1}u + c_{x2}v + \frac{c_{x3}}{2}u^2 + c_{x4}uv + \frac{c_{x5}}{2}v^2. \quad (2.47)$$

$$y = c_{y0} + c_{y1}u + c_{y2}v + \frac{c_{y3}}{2}u^2 + c_{y4}uv + \frac{c_{y5}}{2}v^2. \quad (2.48)$$

$$z = c_{z0} + c_{z1}u + c_{z2}v + \frac{c_{z3}}{2}u^2 + c_{z4}uv + \frac{c_{z5}}{2}v^2. \quad (2.49)$$

2.3 3D Face Segmentation

The segmentation of 3D surfaces consists in the partitioning of the surface into disjoint subsurfaces of the same representation. The segmented subsurfaces may correspond to well-defined surface parts, which are referred to as part-segmentation. Examples of this type include the segmentation of the facial surface from a 3D scan and/or the subparts of the face such as the nose, the forehead, and the eyes. Alternatively, the result of the segmentation may not correspond to natural and visually distinct surface parts, yet the segmentation is still meaningful. In this case, it is referred to as patch-segmentation. The latter segmentation is objectively carried out on the basis of metric criteria and properties of the surface without aiming to obtain segments neatly corresponding to distinctive parts.

There are many applications of 3D surface segmentation, including parameterization (which allows texture mapping and curvature extraction), modeling, and morphing. However, the primary purpose of surface segmentation in 3D face recognition is the extraction (from the segmented subsurfaces) of surface features for matching. Many general approaches for the segmentation of 3D surfaces have been proposed in the literature. To some extent, these approaches fall in a specific class of clustering algorithms but with a different variety of distance measures and constraints pertaining to 3D surfaces. A recent survey of (generic) 3D surface segmentation algorithms is provided by Shamir (2008) and another one for CAD applications is provided by Agathos *et al.* (2007). In the next subsections, we limit our focus to 3D segmentation approaches that are specific to facial surfaces.

2.3.1 Curvature-based 3D Face Segmentation

An number of 3D face recognition systems use surface curvatures for segmentation. As curvatures are independent of the transformation/pose of the face and can be computed at every 3D point of the surface, they are handy as

measures for surface segmentation. They are equally applicable for both patch-segmentation, part-segmentation, and the detection of facial landmarks. For patch-segmentation, either principal curvatures, κ_1 and κ_2, or the mean H and Gaussian k curvatures are computed at every point of the surface and are used to assign each points to a surface segment. Typically, each of the segments has consistent geometric properties such as convexity (peak), concavity (pit), ellipticity (shaped like an oval cup), hyperbolicity (saddle shaped), cylindricity (both concave and convex corresponding to valleys and ridges in the surface when relaxing the requirement of fixed curvature), and flatness. The signs of the curvatures along with (near) zero curvatures (determined using appropriate thresholding) along with region growing algorithm have been widely used for segmentation (Akagunduz and Ulusoy, 2007; Besl and Jain, 1988; Moreno et al., 2003). Table 2.1 maps the signs of curvatures to the geometric types of local patches, see Figure 2.4 for an illustration of such facial surface segmentation.

Table 2.1 Relation between signs of curvatures and geometric shapes of local patches of the surface

	$\kappa_1 > 0$	$\kappa_1 < 0$	$\kappa_1 = 0$	$H > 0$	$H < 0$	$H = 0$	
$\kappa_2 > 0$	pit	n.p.	n.p.	pit	peak	n.p.	$G > 0$
$\kappa_2 < 0$	saddle	peak	ridge	saddle valley	saddle ridge	minimal surface	$G < 0$
$\kappa_2 = 0$	valley	n.p.	plane	valley	ridge	flat	$G = 0$

n.p. means that the combination is not possible.

Alternatively, the values of curvatures or their ranges along with constraints on the segments such as proximity, smooth perimeters of the patches, and their areas could be used for clustering the points into segments. The use of hierarchical clustering or *K*-means algorithms is typical with these approaches.

The curvature-based patch segmentation is closely related to many landmark detection approaches (Akagunduz and Ulusoy, 2007; Dibeklioglu et al., 2008; Segundo et al., 2007), in terms of using the curvatures to detect the landmarks on the basis of their resemblance to the geometric shapes mentioned above. The location of the facial landmarks (or fiducial points) makes it possible to segment the facial surface into parts (part-segmentations).

2.3.2 Bilateral Profile-based 3D Face Segmentation

The tip of the nose (which is an important landmark of the face) and the ridge of the noise lie on the plane of symmetry. Finding the plane of symmetry or its intersection with the facial surface (bilateral profile line) can assist in detecting those landmarks accurately. The detection/localization of these landmarks can

also assist in the detection of the remaining ones (based on relative positions). The extraction of the symmetry plane starts by finding an initial course estimate of the plane (based on the principal directions of the face) then the points of the surface are mirrored across that plane from both sides. The actual and mirror facial surfaces are then iteratively registered. In each iteration, the symmetry plane is adjusted on the basis of the new poses of the two facial surfaces. Zhang *et al.* (2006) use the bilateral profile in addition to the mean curvature to detect the nose tip and the lowest point on the ridge of the nose.

In the work by Mian *et al.* (2007), the nose tip in near frontal facial scans is detected. Then, the facial surface is segmented by dropping the 3D points that lie outside a sphere centered at the tip of the nose with a radius r=80 mm. The pose of the segmented facial surface is then corrected to the frontal view using the principal directions of the face. The frontal view and the knowledge of the location of the tip of the nose enables the segmentation of the parts of the facial surface using image masks. The detection of the tip of the nose starts by slicing the facial surface horizontally from top to bottom. Each resulting line is then searched for the point which has the highest distance (altitude) to the line segment formed by the intersection of a circle with a horizontal slice, which is designated as a potential nose tip. As the potential nose tips are assumed to lie on the nose ridge, a line is fitted to them using the random sample consensus algorithm (RANSAC). The potential nose tip, which lies on the consensus line and has the highest altitude, is considered to be the final estimate of the tip of the nose.

2.4 3D Face Surface Feature Extraction and Matching

Feature extraction in 3D face recognition systems, as in the case of pattern recognition in general, is a core stage that largely influences the performance and reliability of these systems. The extraction of features generally serves the recognition process in a number of ways. It can be considered as a form of data reduction that captures the essential discriminative information from the full-dimensional data. It also serves as a tool for matching facial surfaces. Often, 3D face recognition systems achieve invariance to facial pose and expression variations by means of extracting invariant features.

Many feature extraction approaches have been used in 3D face recognition.

They can be categorized on the basis of the size of the surface area from which they are extracted (holistic, regional, or point features). Other categorizations include, rigid versus deformable feature extraction and uni modal (shape only) versus multi modal (e.g., shape and texture). In this section, the most prominent rigid approaches to feature extraction are discussed. They are organized according to their surface size. Nonrigid approaches are discussed in Section 2.5.

2.4.1 Holistic 3D Facial Features

The holistic approaches extract 3D facial features from the whole surface of the face. Most feature extraction approaches fall in this category.

Iterative Closest Point

The ICP algorithm, proposed by Besl and Mckay (1992) and Chen and Medioni (1991) for corresponding and registering 3D surfaces, was initially widely used for 3D modeling. Later, ICP proved to be equally useful for surface matching, providing a high accuracy particularly for rigid surfaces. On the downside, the computational complexity of ICP is high. Nonetheless, there are efficient variants, for example, the one by Greenspan and Yurick (2003), which uses a K-D tree to speed up the search for closest points (the main bottleneck of ICP). The registration error, expressed as the sum of the Euclidean distances or other distances (e.g., city block) or their histogram, is used for matching.

Description of ICP: The ICP algorithm requires an initial coarse registration. Less accurate registration approaches such as those based on the detection of the fiducial points of the face (either 3D or 2D in case of textured scans) and/or pose correction can be used to provide an initial coarse registration. Starting from that coarse registration, ICP iteratively estimates the rigid transformations (rotations and translations) between the two surface (which minimizes the registration error) and updates one of the two surfaces accordingly. When the registration error from one iteration to another diminishes and/or after a pre-specified number of iterations, the algorithm assume convergence. The convergence of ICP can be influenced by the shape of the surface. Surfaces that can be slid on each other in one or two geodesic directions can yield low registration errors but an incorrect registration. Examples of these surfaces include planar, spherical, and cylindrical (which can be slided bidirectionally). Any 3D surface that can result from the rotation of a curve (rotational surface) or its translation along a straight line, will induce the ICP to potentially converge somewhere along the

direction of rotation or translation of the curve. However, facial surfaces as most free-form surfaces exhibit a decent convergence behavior.

The estimation of the rigid transformations at the *i*th iteration, the central part of ICP, is computed from two sets of points with a one-to-one correspondence. These correspondences are typically formed by taking the vertices of the first surface as the first set of 3D points and the second set as their closest points in the other surface. Alternatively, the closest points along the directions of the surface normals are taken. The latter approach has been shown to demonstrate a higher registration accuracy. Let $\mathbf{P}$ and $\mathbf{Q}$ be $3 \times n$ matrices storing the *x*-, *y*- and *z*-coordinates of the first and second point sets, and let $\mathbf{p}_i$ and $\mathbf{q}_i$ represent the *i*th corresponding pair of points. The transformations ($\mathbf{R}_i$ and $\mathbf{t}_i$) that minimize the registration error, $e = \sum_{i=1}^{n} \|\mathbf{q}_i - \mathbf{R}_i\mathbf{p}_i - \mathbf{t}_i\|$, can be found in a number of ways. A closed form solution can find $\mathbf{R}_i$ and $\mathbf{t}_i$ by first translating the point sets so that their average positions lie on the origin, $\mathbf{p}'_i = \mathbf{p}_i - \bar{\mathbf{p}}_i$ and similarly $\mathbf{q}'_i = \mathbf{q}_i - \bar{\mathbf{q}}_i$ where $\bar{\mathbf{p}}_i$ and $\bar{\mathbf{q}}_i$ are $\bar{\mathbf{p}}_i = 1/n \sum_{i=1}^{n} \mathbf{p}_i$ and $\bar{\mathbf{q}}_i = 1/n \sum_{i=1}^{n} \mathbf{q}_i$. The rotational matrix is then found by orthonormalizing the matrix $\mathbf{A} = \mathbf{P}'^{\top}\mathbf{Q}'$ using single value decomposition $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^{\top}$; the singular values are replaced by ones, and if the determinant of the resulting matrix is −1, the matrix is multiplied by −1, that is $\mathbf{R} = \pm\mathbf{U}\mathbf{I}\mathbf{V}^{\top}$. The translation $\mathbf{t}_i$ is given by $-\mathbf{R}\bar{\mathbf{p}} + \bar{\mathbf{q}}$.

Use of ICP in holistic face recognition: The registration errors between facial surfaces has been used in 3D face recognition as a dissimilarity measure (Lu et al., 2006). Since ICP is an approach designed for the registration of rigid objects, its accuracy in 3D face recognition on the basis of its registration error deteriorates with variations in facial expressions (see Figure 2.5). Matching facial surfaces using the histograms of the registration errors appears to handle expression variations better (Amor et al., 2006). The partial ICP approach to 3D face recognition (Wang et al., 2006), can dynamically extract the regions of the face that are not affected by expressions (vary according to the expressions) and compute the dissimilarity measures of these regions. The partial ICP ranks the correspondences (in each ICP iteration) on the basis of their distances in ascending order and considers only a fixed percentage of them, that is, those which rank first. The ICP is also used for textured 3D face recognition. It should be noted that the extension of ICP to multi modal is straightforward (apart from the complexities related to illuminations variations) as a fixed mapping between the 3D points and their texture should be maintained.

Principal Component Analysis (PCA)

PCA is an effective approach to the compression of high dimensional data (where each observation is represented by a multi dimensional vector). The higher dimensional data can often reside (or approximately reside) in a lower dimensional orthonormal subspace. In this case, the original data can be approximated or represented by a lower number of variables by projection on that subspace, the dimension of which is equal to the number of the reduced variables. These variables were widely used as feature vectors in 2D face recognition and later in 3D face recognition.

Description of PCA: Firstly, a lower subspace of the 3D facial data (usually a collection of range image) is found. As PCA operates on vectorial data, the range images are vectorized in the same manner. The pixels of each range image are stacked in a vector, row by row or, equivalently, column by column. Let the vector $\mathbf{f}_i$ represent the vectorized *i*th range image. The average range face is then computed, $\bar{\mathbf{f}} = 1/n \sum_{i=1}^{n} \mathbf{f}_i$. The covariance matrix Ω is next computed as in Equation 2.50.

$$\Omega = \sum_{i=1}^{n} (\mathbf{f}_i - \bar{\mathbf{f}})(\mathbf{f}_i - \bar{\mathbf{f}})^{\top}. \quad (2.50)$$

The lower dimensional subspace is given by the *k* eigenvectors of Ω with the highest eigenvalues $\mathbf{E} = [\mathbf{e}_1 \dots \mathbf{e}_k]$, where *k* is much lower than the number of range pixels (fixed in each image). This is because the extent of data variation along an eigenvector is indicated by the corresponding eigenvalue, $\alpha_i \mathbf{e}_i = \Omega \mathbf{e}_i$. See Figure 2.6.

The compact representation of an unseen range image (after vectorization and subtracting the average face) $\mathbf{f}_p$ is called the PCA coefficients $\mathbf{c}_p$ (the feature vector) and is given by the projection $\mathbf{c}_p = \mathbf{E}^{\top} \mathbf{f}_p$. The PCA coefficients of the range image $\mathbf{c}_p$ can be matched against another PCA coefficients of another unseen range image $\mathbf{c}_g$ (found in a similar way) on the basis of any metric distance (dissimilarity measure). Typical dissimilarity measures used along with PCA are the Euclidean $\|\mathbf{c}_p - \mathbf{c}_g\|$ and the cosine $\frac{\mathbf{c}_p^{\top} \mathbf{c}_g}{\|\mathbf{c}_p\| \|\mathbf{c}_g\|}$ distances.

Comments on the use of PCA in 3D face recognition: The use of PCA in recognition requires the pose-correction of the facial surfaces (to a predefined pose, typically the frontal one), the cropping of the facial surface. The range images should also be of the same resolution. Variations that could be introduced by these factors affect the computation of the PCA subspace and/or the PCA features which ultimately may undermine the recognition performance.

Another aspect of using PCA features for 3D face recognition is that they are sensitive to facial expressions in that the values of the features vary according to facial expressions. However, as an effective tool for dimensionality reduction of facial data, PCA is the de facto technique for feature extraction in many 3D face recognition approaches that further process these features to counteract feature variations and, as a result, enhance the recognition accuracy. Examples of such approaches are some of those based on linear discriminate analysis (LDA), nonlinear manifolds and support vector machines (SVMs). PCA is also a building block in some approaches to nonrigid 3D face recognition (see Section 2.5).

Linear Discriminant Analysis (LDA)

Description of LDA: Similar to that of PCA, LDA projects observations of multi-dimensional data on an orthonormal subspace. However, as opposed to PCA, which aims for a subspace that captures most of the variance in the data, LDA aims for a better separability among the various classes of the data (the 3D faces). In order to achieve that, LDA finds a subspace $\mathbf{E} = [\mathbf{e}_1 \dots \mathbf{e}_k]$ that maximizes the Fisher's criterion (the ratio of the variance between the classes to the variance within the classes) as shown in Equation 2.51.

$$\mathrm{J}(L) = \mathrm{trace}\left(\frac{\mathbf{E}^\top \Omega_B \mathbf{E}}{\mathbf{E}^\top \Omega_W \mathbf{E}}\right) \tag{2.51}$$

$$= \sum_{i=1}^{k} \frac{\mathbf{e}_i^\top \Omega_B \mathbf{e}_i}{\mathbf{e}_i^\top \Omega_W \mathbf{e}_i} \tag{2.52}$$

$$= \sum_{i=1}^{k} \mathbf{e}_i^\top \Omega_W^{-1} \Omega_B \mathbf{e}_i, \tag{2.53}$$

where Ω_B and Ω_W are the between and within covariance matrices that are defined in terms of a number of *C* class means, $\mu_1 \dots \mu_C$, the overall mean (the average face) $\bar{\mathbf{f}}$, and the vectorized facial data, $\mathbf{f}_{ij}$, which denotes the *i*th facial vector belonging to the *j*th person (class).

$$\Omega_B = \sum_{j=1}^{C} (\mu_j - \bar{\mathbf{f}})(\mu_j - \bar{\mathbf{f}})^\top. \tag{2.54}$$

$$\Omega_W = \sum_{J=1}^{C} \sum_{i=1}^{N_j} (\mathbf{f}_{ij} - \mu_j)(\mathbf{f}_{ij} - \mu_j)^\top. \tag{2.55}$$

From Equation 2.53, it can be seen that $\mathbf{J}$ maximized when the vectors $\mathbf{e}_1 \dots \mathbf{e}_k$

are the eigenvectors of $\Omega_w^{-1}\Omega_B$. Similarly to PCA, the LDA feature vector of an unseen facial vector is computed by subtracting the mean face first and then projecting the difference onto the subspace.

Comments on the use of LDA in 3D face recognition: Because LDA requires the inversion of the within-class covariance matrix, the total number of the training data samples should be higher than the dimension of sample vectors, which is usually large (the number of pixels). Otherwise, the matrix will be singular, referred to as the small sample size (SSS) problem. One way to overcome this problem is to use the pseudo-inverse of Ω_w . A common approach to pseudo-inverse is to decompose the covariance matrix using SVD, $\Omega_w = \mathbf{U}\Lambda\mathbf{U}^\top$ and then invert the non-zero singular values (or only those that are considerably higher than zero and the remaining ones are set to zeros), $\Omega_w^+ = \mathbf{U}\Lambda^+\mathbf{U}^\top$, where Λ^+ is diagonal and each diagonal element is the inverse ($1/\lambda_i$) of the corresponding element in Λ if it is non-zero. A more accepted approach is to reduce the dimensionality of the training data using PCA prior to the application of LDA, called PCA + LDA.

LDA is widely used in both 2D and 3D face recognition and generally demonstrates better recognition accuracies than does PCA. Apart from its relative discriminatory advantage, LDA still suffers from pose and expression variations. There are three prominent approaches to LDA feature extraction in 3D face recognition. First, LDA feature extraction is performed on range images. Second, LDA features are extracted from facial normal maps. Gokberk *et al.* (2006) experimentally have shown that LDA on normal maps outperforms that on range images. Third, in textured 3D face recognition, LDA feature extraction has been performed on the texture, whereas ICP, which is more accurate than 3D LDA, is used for shape feature extraction (Lu et al., 2006). In matching, the dissimilarity measures produced by the shape and texture are combined using a weighted sum rule.

Curvature-based Methods

In comparison to the use of curvatures for regional and point-based feature extraction, 3D face detection and pose estimation, the use of curvatures in holistic 3D face recognition is to an extent limited. However, some of the most early approaches to 3D face recognition are those based on the extended Gaussian image (EGI) (Lee and Milios, 1990; Tanaka et al., 1998).

EGI facial features: An EGI is constructed by mapping (or associating) the

facial surface normals to those of a unit sphere in the same directions. The points on the unit sphere are assigned the values of the Gaussian curvatures *k* at their corresponding surface points, $G(\theta_i, \phi_i) = K_i$, but those with no correspondences are assigned a value of zero. In reality, a discretization of the unit sphere is used, in which some cells are within the continuous range of normals but are left empty due to the sampling of the surface. Interpolation is used to estimate the curvature at those cells. In the EGI variant by Tanaka *et al.* (1998), the facial surface is segmented into valleys and ridges based on *H* and *k* curvatures. EGIs are then computed from the two facial segments. In that work, the EGIs map the principal curvature directions ($\mathbf{e}_1$ and $\mathbf{e}_2$) to the normals of a unit sphere rather than from the normals of the original surface, to the normals of a unit shpere.

Surface representation using EGI has some advantages. Since the curvatures are invariant to rigid transformations and the normals are invariant to translations, EGI representations/features are also invariant to translations. However, the rotational invariance is easily handled during matching by means of spherical correlation, in which one of the spheres of the two matched EGIs, $G_1(\theta, \phi)$ and $G_2(\theta, \phi)$, is revoluted to maximize the correlation $\sum G_1(\theta, \phi) G_2(\theta, \phi) / \sqrt{\sum G_1^2(\theta, \phi) \sum G_2^2(\theta, \phi)}$.

Ideally, when a 3D surface represented by an EGI is convex, there is a one-to-one mapping between the surface and the unit sphere normals as there are no surface normals pointing in the same direction. This makes EGI representations for convex surfaces unique. However, this is not the case with facial surfaces; 3D face recognition approaches based on EGI often assume that the 3D face has (to an extent) a convex shape. Other disadvantages of EGIs include the possible information loss during mapping (the actual positions of the 3D points) and its susceptibility to noise due to the reliance on normals and curvatures.

Curvature-like method: The method proposed by Al-osaimi *et al.* (2008) extracts 11 scalar quantities pertaining to local attributes of the facial surface at each point, each called a scalar filed, and 3 other fields pertaining to the global structure of the face. Some of these local fields are (conceptually) similar, to an extent, to surface curvatures but are in contrast less affected by noise. Discriminative and transformation invariant features are extracted by constructing a 2D histogram from each combination of a local and a global field. The histograms are indexed in one direction by the local field and in the other direction by the global field. The bins of each histogram store the area of the surface with the local and the global field values corresponding to its indices.

The histograms are vectorized and concatenated. PCA is then used to compress these concatenated features.

The local fields are extracted at each vertex from a 3D facial mesh from the triangles t_i within a small and a large neighborhood, denoted by $\mathcal{N}_1$ and $\mathcal{N}_2$, respectively. Six local fields are defined as the thee eigenvalues of the covariance matrix in Equation 2.56 for each neighborhood, where $\mathbf{R}_i$ is the vector from the point to the centroid on the *i*th triangle, a_i is its area, and A is the total area of the neighborhood.

$$\sum_{a_i \in \mathcal{N}} \frac{a_i \mathbf{r}_i \mathbf{r}_i^\top}{A \|\mathbf{r}_i\|}. \quad (2.56)$$

Three other local fields are defined as the singular values of the matrix in Equation 2.57, where $\mathbf{n}$ and $\bar{\mathbf{n}}$ are the normal of the *i*th triangle and the mean normal, respectively. The fields that involve the normals were only extracted from the larger neighborhood to reduce the effect of noise because they are more sensitive to the noise than to the $\mathbf{R}$ vectors.

$$\sum_{a_i \in \mathcal{N}_2} \frac{a_i (\mathbf{n}_i - \bar{\mathbf{n}}) \mathbf{r}_i^\top}{A \|\mathbf{r}_i\|}. \quad (2.57)$$

Two other local fields are defined as (one from each neighborhood)

$$\sum_{a_i \in \mathcal{N}_2} \frac{a_i \mathbf{n}_i \cdot \mathbf{r}_i^\top}{A \|\mathbf{r}_i\|}. \quad (2.58)$$

The global fields are the dot products between the vectors from the global centroid C of the cropped facial surface to each vertex $\mathbf{c}_i$ and each of the three principal directions of the face. The principal directions of the face are the eigenvalues the matrix in Equation 2.59.

$$\sum a_i (\mathbf{c}_i - C)(\mathbf{c}_i - C)^\top. \quad (2.59)$$

Although the recognition accuracy of this method is considerably high for neutral and near neutral face recognition, its ability to handle recognition under facial expressions is limited.

2.4.2 Regional 3D Facial Features

The rigid matching of sub-facial surfaces (regions of the 3D face) or the extraction of (rigid) features from specific regions of the face has proven to mitigate the effects of facial expressions on the recognition accuracy. Chances are facial expressions leave some regions undeformed, depending on the

expression at hand. Even, deformed regions could appear locally minimal.

ICP-based approaches: Indeed, some of the most successful approaches to 3D face recognition are those based on region-matching using ICP. The reason behind their success is that ICP is very accurate when matching rigid surfaces but is, on the other hand, sensitive to deformations. As discussed earlier, region-based matching mitigates the affects of facial expressions and consequently the accuracy of ICP is retained, to an extent. Chang *et al.* (2006) performed 3D face recognition using ICP on multiple overlapping nose regions. For matching a probe to a gallery facial scans, the landmarks (e.g., the nose tip, eye pits, and the ridge of the nose) are first detected using curvatures. On the basis of the locations of the landmarks, the overlapping regions in the probe scan are extracted. ICP is then used to individually register each extracted region to the gallery scan. From the registration errors of the overlapping regions, a total dissimilarity measure between the probe and the gallery scans is computed on the basis of the product, sum, and minimum rules. The product and the sum rules have demonstrated a better recognition accuracy than the minimum rule, but they (the product and sum rules) were comparable to each other. By matching facial scans using ICP on the forehead and the nose as two separate regions, a higher recognition accuracy has been achieved (Mian et al., 2005, 2007). In order to reduce the computational complexity of ICP, a rejection classifier was used. The rejection classifier is a histogram of the number of the 3D points of the face falling in between concentric spheres centred at the tip of the nose with increasing diameters called spherical face representation (SFR).

Miscellaneous region-based approaches: The 3D face recognition approach proposed by Gunlu and Bilge (2010) divides a pose-corrected range image of the 3D face into a number of squares. The discrete cosine transform (DCT) is then applied to each region, producing a collection of DCT coefficients. The most discriminative and expression invariant DCT coefficients are selected as features for recognition, by means of a feature selection approach. Intuitively, as DCT is a template matching approach, one would expect that this approach to give a higher recognition accuracy than a holistic PCA due to the use of feature selection. However, the selected features may still be affected by pose variations as facial expressions also introduce pose correction errors.

The curvature-based approach proposed by Moreno *et al.* (2003), one of the early ones in this category, segments the 3D face into regions on the basis of the sign of the mean H and Gaussian k curvatures. A number of attributes of the segmented regions are used as features for recognition. These attributes include

their areas, distances, and angles between their centers gravity, average *H* and *k*. It is noted that unlike most other approaches in this category, the division into regions is not meant to achieve invariance to facial expressions but is rather a means for feature extraction. In fact, the segmented regions and their attributes could differ from a range image to another depending on facial expressions.

2.4.3 Point 3D Facial Features

A well-known paradigm to object recognition is to first detect points, called key points. Then, point features are extracted from the local neighborhood of the detected key points, called descriptors or point signatures. For matching a pair of images, neither all the key points need to be detected (in both images) nor all of them need to match. Nevertheless, a high rate of keypoint detection and matching translates into a high similarity measure. Approaches following this paradigm are less sensitive to clutter and occlusions and do not require the segmentation of the object from the background.

Many approaches to 3D face recognition follow this paradigm. In addition to the aforementioned advantages of keypoint based recognition, they show a degree of invariance to facial expressions. For the general case, the key points may not be at anatomically distinctive locations. On the contrary and as a special case, the key points might be the fiducial points (the landmarks) of the face.

Sphere and surface intersection method: One of the earliest approach to 3D face recognition that might be categorized in this category is the one by seng Chua *et al.* (2000). While, the notion of keypoint detection was not strongly present, the approach is largely based on point descriptors. The approach defines a point descriptor as the closed curve formed by intersecting a sphere with the 3D facial surface, the sphere center is placed at the 3D point. The curve of intersection is represented by its orthogonal distance to the tangential plane and is parametrized by an angle θ from a reference vector, $d(\theta)$. The reference vector is defined as the maximum distance from the 3D point to any point on the projected curve. Two descriptors $d_1(\theta)$ and $d_2(\theta)$ are matched (possibly) by finding the integral of their absolute difference, $\int |d_1(\theta) - d_2(\theta)| d\theta$.

During training, the approach considers only the point descriptors that can match under different facial expressions. For an efficient retrieval during matching, the selected point descriptors are stored in a 3D table along with the identities of their training scans (the gallery). The indexing dimensions of the table are the number of the local minimums of the descriptors, the number of

their local maximums, and the number of their minimums plus maximums. For matching a probe scan against the gallery identities, the point descriptors are found at every point of the probe. Then, all the point descriptors are matched against those in the 3D table and those that match then vote for the identity of the probe.

Principal directions method: The principal directions are used by Mian *et al.* (2008) to detect repeatable and descriptive (indicated by the non planarity of the local surface) key points on facial surfaces. The detections of a key point starts by cropping the local surface around a candidate key point using a sphere. A plane is then fitted to the local surface in order to uniformly sample the surface according to a regular grid, resulting in a fixed number *N* of samples. Next, the principal directions of the sampled local surface, $\mathbf{E}=[\mathbf{e}_1\ \mathbf{e}_2\ \mathbf{e}_3]$, are found in a similar fashion as described in Equation 2.59. After that, the 3D points of the sampled surface are projected on the principal directions, $\mathbf{p}_i' = \mathbf{E}^\top(\mathbf{p}_i - \bar{\mathbf{p}})$, where $i = 1 \ldots n$, and $\bar{\mathbf{p}}$ is the sampled points average. On the basis of the maximum and minimum *x*- and *y*-coordinates of the projection of all the *N* samples, a scalar measure for keypoint selection is defined, $\delta = \max(\mathbf{p}_x') - \min(\mathbf{p}_x') + \min(\mathbf{p}_y') + \max(\mathbf{p}_y')$. A 3D point on the facial surface is considered a key point if its δ exceeds a certain threshold. The majority of the detected points were located on and to the sides of the nose region, which is known to be discriminative. However, it is noticed that a small number of them are located on the forehead region, which also is very discriminative. This is because the forehead is slightly curved.

PCA features are used as descriptors of the local regions. For matching a pair of surfaces, the PCA descriptors of the key points are matched against each other. Matched key points (those with low dissimilarity measures) are used to construct two identical graphs (one on each facial surface) in which the key points are represented by the graph nodes and their positions information is stored. A total dissimilarity between the pair of surfaces is computed using the weighted-sum rule from the average of the descriptor matching errors, the number of keypoint matches, and the average of the absolute distance errors among the corresponding edges and nodes of the two graphs.

Filtering kernel methods: Some of the well-known approaches to keypoint detection in 2D images are based on filtering kernels, for example, the SIFT by Lowe (2004). A 3D variant of the SIFT was proposed by Lo and Siebert (2008) and used for face recognition from range images. For keypoint detection, a

Gaussian pyramid is constructed (following the 2D SIFT), each pyramid level is double the resolution (both vertically and horizontally) of the next level. In each level, there are multiple Gaussian range images, responses of Gaussian kernels with increasing σ parameters. From each two consecutive Gaussian images, a difference of Gaussian (DoG) range images is found. The extrema (minima and maxima) points in the DoG images in their spatial and scale (the adjacent DoG images) proximities are considered key points if their deviations from their proximities exceeds a certain threshold. It is noticed that the detected 3D SIFT key points are mostly located on the eyes regions, where the acquisition of the surface is usually unreliable and spiky due to the eyelashes and appear sparse elsewhere. This is possibly why in their system they also included key points detected on the basis of the ratio of the principal curvatures κ_1/κ_2, if the local maxima exceed a certain threshold. Parallel to the 2D SIFT, the 3D SIFT descriptors are histograms of the surface gradients around the key points. Each key point is assigned a direction, usually the dominant gradient direction, which is used as a reference for in plane rotation (as opposed to in depth rotation) invariance. A local region of a key point is divided into nine overlapping subregions by means of scaling by displaced Gaussian functions, from each a histogram is computed. The histogram bins correspond to specific ranges of directions. The nine histograms are concatenated to form the key point descriptor.

Another kernel-based approach was proposed by Al-Osaimi *et al.* (2007). Blob like images were produced by convolving range images with kernels. By taking the peaks of the blobs as key points if their difference relative to their spatial neighborhood exceeds a certain threshold, stable and repeatable key points was achieved. The underlying principle of those blob generating kernels is that a set of adjacent higher spatial frequencies (both vertically and horizontally) become constructive and destructive at certain locations and forms a pattern of peaks and bottoms. By allowing those frequencies of a range image to pass, a specific blob like image (to the range image) is produced. The kernel is found by first setting selected adjacent frequencies to a value of one in a square matrix (of the kernel size) and setting the discrete Fourier transform remaining frequencies to a value of zero. Then, the inverse of the (DFT) of that square matrix is computed. To achieve invariance to in plane rotations, the real part of the DFT inverse is rotated in small steps one complete cycle (360°). The kernels at each step are then interpolated and their average is found. Finally, the DC component is removed from the average kernel and the resulting kernel is normalized to sum

up to a value of one. The kernels have shown to detect key points in all regions of the face. A comparison with the well-known Laplacian of the Gaussian (LoG) kernel, whose response is similar to the DoG, which is also used to generate blob like images from 2D images reveals that the blobs generated by the adjacent frequencies approach gives much prominent blob like images.

As descriptors of the detected key points, the intersections of the local surface with five concentric spheres (of different radii) are found, and each is sampled at fixed angular steps. A pair of descriptors are matched by first estimating the off set roll rotation between their samples using circular correlation. The estimation of the off set roll angle is then used to vectorize the depths (*z*-coordinates) of the two sets of circular samples starting from the same point ($\mathbf{z}_1$ and $\mathbf{z}_2$, respectively, for the first and second descriptors), giving invariance to roll rotations. Before the computation of a dissimilarity measure, small out of plane rotations are adjusted for by adding a plane to one of the two depth vectors so it best fits the other depth vector, as described in Equations 2.60 and 2.61.

(2.60) $[u\ v]^\top = (\mathbf{L}^\top\mathbf{L})^{-1}\mathbf{L}^\top(\mathbf{z}_1 - (\mathbf{z}_2 - z_{c1}))$ and

(2.61) $\mathbf{z}_2' = \mathbf{z}_2 - z_{c1} + \mathbf{L}[u\ v]^\top,$

where $\mathbf{L}$ is a two-column matrix holding the *x*- and *y*-coordinates of the samples of the second descriptor, *u* and *v* are the fitting parameters, and z_{c1} is the depth at the center of the first region (at the keypoint position). The dissimilarity measure is the summation of the absolute errors between $\mathbf{z}_1$ and $\mathbf{z}_2'$, that is, $e = \sum_i |\mathbf{z}_1(i) - \mathbf{z}_2'(i)|$.

Landmark methods: Local region matching around predefined landmarks on the face has been also used in 3D face recognition. This application of landmarks is not generally as promising as the use of key points in handling facial expressions. The landmarks are very sparse in comparison to the key points and usually such approaches require the accurate and full detection of all the landmarks, in contrast to approaches based on key points that are tolerant to some failure in keypoint detection and matching.

The well-known elastic bunch graph matching (EBGM) approach to 2D face recognition (Wiskott et al., 1997), which has been adapted to perform 2D+3D and 3D face recognition (Husken et al., 2005; Wang et al., 2001), is the best known approach in this category. EBGM relies on a set of Gabor filters (of different frequencies, orientations called and scales) to detect the landmarks and also to describe their local regions. The convolution of a range image with the Gabor filters produces a vector of responses at each point called a jet. Initially,

the jets at manually localized landmarks on facial range images are computed and stored in the nodes of a graph (face bunch graph, FBG). Each node stores a bunch of jets computed from the different range images at a corresponding landmark. Once an initial FBG is obtained, it can be automatically expanded. The landmarks on an unseen range image are localized by searching for points whose jets best fit the jets in the FBG and their positions agree with the FBG nodes.

2.5 Deformation Modeling of 3D Face Surface

In the previous section, a common strategy used by the rigid approaches for the handling of expression variations for 3D face recognition is to avoid deformable and/or deformed regions. This strategy has clearly shown to improve the recognition performance. Yet, the fact remains that it does not take advantage of any discriminative information available in the non utilized regions. Another successful strategy for handling expression variations is to apply deformations to the facial surface before matching, this set of approaches fall under what is termed nonrigid 3D face recognition. The applied deformations should counteract (or neutralize) those attributed to facial expressions but should be able to retain any relevant discriminative information. In fact, the success of nonrigid approaches to 3D face recognition largely depends on the extent of achieving this criterion.

Modeling Expressions as Isomorphism

A popular class of 3D face recognition systems, which model facial expression deformations, is based on the assumption that the geodesic distances on the facial surface are invariant to facial expression. This is equivalent to assuming that the facial skin does not stretch or shrink under expression variations. This assumption is intuitive to an extent, but it is violated for expressions that involve topological changes of the facial surface, notably mouth opening. This issue, however, can be handled by the detection and closing of the open mouth in a preprocessing stage. One of the early approaches in this category is that by Bronstein *et al.* (2005). They have experimentally shown that geodesic distances on facial surfaces have much less absolute errors due to expression variations than Euclidian distances.

The approach by Bronstein *et al.* (2005) first deforms the facial surface to a canonical form (a surface of a standard shape in 3D space) subject to the

preservation of the geodesic distances. During the matching phase, the canonical forms are matched against each other using ICP or other feature-based matching. The deformation is carried out iteratively and through the optimization of an objective function. The coordinates of the surface vertices (or sample points on the surface) are varied (displaced in the 3D space) such that the objective function that incurs costs proportional to the discrepancies in geodesic distances (with respect to the original surface) is minimized. Hence, the computation of the geodesic distances is required at every iteration of deformation. The geodesic distances can be computed from every point on the facial surface to every other point on that same surface. However, this is computationally expensive. The computational load of the approach is reduced by only computing distances from certain points (the tip of the nose and the nose apex in their case). These geodesic distances are efficiently computed by the fast marching method (FMM), which emulates wave propagation on the 3D facial surface from these two selected points.

Alternative methods to the deformation of the facial surfaces have also been developed. They extract features on the basis of the geodesic distances of the surface rather than using them as constraints for deformation. In the work by Smeets *et al.* (2009), the geodesic distances among the 3D points of the facial surface are first computed and stored in a 2D matrix. The largest singular values of that matrix are then taken as features. These features are independent of the order of the surface points used during the computation of that matrix. It is worth mentioning that the isometric modeling of expression deformations only captures the intrinsic properties of the surface, that is it captures the information pertaining to the Gaussian curvature but not that pertaining to the mean curvature of the facial surface. On one hand, the approach limits the discriminative information but on the other hand, it largely removes the influence of expression variations on the retained discriminative information.

Subspace Modeling of Expressions

PCA subspaces have been used in different ways and for different purposes in 3D face recognition. PCA is mostly applied on the facial data for dimensionality reduction. When used in this manner, it simultaneously captures the facial surface data and any expression variations together in the PCA subspace/features. This blend of the discriminative information with the expression variations becomes a source of error if not addressed at a later stage in the recognition system. In the work by McCool *et al.* (2006), the variations of

facial PCA features relative to some other PCA features (differences of feature vectors) are modeled using Gaussian mixture models (GMMs). The GMMs provide a probabilistic means for computing a similarity measure between a pair of facial scans (from which a feature vector difference is formed). In matching a probe to a gallery of scans, a feature vector difference is formed between the probe and each of the gallery scans. The identity of the probe is deemed to be that of the gallery scan that produces the feature vector difference with the highest similarity measure.

In the work by Al-Osaimi *et al.* (2009), a PCA subspace is constructed from image differences between pairs of pose-corrected range images. Each range image difference is the subtraction of the neutral face from a non-neutral image belonging to the same subject but the training data includes pairs belonging to many subjects. Before the computation of the range image difference, the two range images are registered to each other using ICP and on the basis of only semirigid regions of the face (the forehead and the nose). This reduces the effects of expression deformations on the registration and therefore the accuracy of the image difference. Since each training pair belongs to the same subjects, the PCA subspace represents only expression deformations in this case. Projecting an unseen probe difference $\mathbf{d}$ (between a probe image and a gallery scan) on the PCA subspace results in the separation of the expression deformations from the discriminative information since the projection retains only the expression deformations. The projected range image difference is then reconstructed $\hat{\mathbf{d}}$ and subtracted from the original range image difference, $\mathbf{e} = \mathbf{d} - \hat{\mathbf{d}}$. The vector $\mathbf{e}$, which represents the discriminative information, is then post processed by thresholding the pixels with high absolute values and finally a dissimilarity measure is computed from the post processed vector. The approach has achieved a much higher recognition accuracy compared with the approach that models PCA features.

Elasticity-based Modeling of Expressions

In the work by Kakadiaris *et al.* (2007), a generic model of the 3D face, called the annotated facial model AFM, Kakadiaris *et al.* (2005), is elastically deformed/fitted to probe and gallery facial scans (which may be under facial expression). For matching, the features are extracted from the fitted models rather than the facial scans. The AFM is basically an average facial mesh with marked facial regions and a *u* and *v* parameterization. AFM tends to have no

person specific discriminative information and to appear under neutral expression. Before fitting, the AFM is registered to the 3D facial scan using ICP. Then, external forces are iteratively applied on the AFM to displace its vertices towards the 3D facial surface. The external forces are countered by forces proportional to the displacement (elastic force), the acceleration of displacement (inertial force) and the speed of displacement (damping force). The fitting of AFM intuitively converges when the elastic forces are in balance with the applied external forces. It is the elastic fitting that provides the fitted AFM with the discriminative information but also include some of the expression deformations. Since the AFM looks like a neutral face and the expression deformations can vary more the discriminative information, the fitting is likely to mitigate the effects of expression variations.

Exercises

1. For a cropped 3D facial surface under a near fontal pose, the principal directions are $\mathbf{p_1} = [-0.3 \;\; 0.92 \;\; 0.25219]^\top$, $\mathbf{p_1} = [0.88159 \;\; 0.16639 \;\; 0.44172]^\top$, $[0.36442 \;\; 0.35484 \;\; -0.86098]^\top$ and the tip of the nose is located at (10, −15, −35).

A. Find the rigid transformation that pose-corrects the facial surface so its principal directions become $\mathbf{p_1'} = [0 \;\; 1 \;\; 0]^\top$, $\mathbf{p_2'} = [1 \;\; 0 \;\; 0]^\top$ and $\mathbf{p_3'} = [0 \;\; 0 \;\; -1]^\top$ and the tip of the nose be at the origin of the reference frame, (0, 0, 0).

B. Explain how quaternions can be used to achieve the same transformation in part A.

2. For a facial range image and using the method of the kernels of Gaussian derivatives to find surface derivatives:

A. Write a MatLab function that computes the first and second fundamental forms at each range pixel and on the basis of the fundamental forms computes the Gaussian and the mean curvatures.

B. For different σ values of the Gaussian function and different sizes of the filtering kernel, compare estimates of the Gaussian and the mean curvatures of the same image and suggest appropriate σ value and kernel size.

3. From a publicly available data set of 3D facial scans, randomly pick a set of neutral facial scans and a set of non-neutral scans for use as probes. As a gallery set, pick one neutral scan for every individual in either of the two probe

sets. Then, compute and compare the identification rates of the following:

A. When using PCA for the identification of the scans in the neutral probe set.

B. When using PCA for the identification of the scans in the non-neutral facial scans.

C. Crop off the lower part of the facial scans, just below the nose, in both the gallery and probe sets. Then, use PCA for the identification of the scans in the neutral probe set.

D. When using PCA for the identification of the cropped non-neutral probe set.

E. When using ICP for the identification of the scans in the neutral probe set.

F. When using ICP for the identification of the scans in the non-neutral facial scans.

G. When using ICP for the identification of the cropped neutral probe set.

H. When using ICP for the identification of the cropped non-neutral probe set.

4. Select 10 facial scans of the same subject and under different facial expressions from a publicly available data set and estimate the repeatability rate of keypoint detection of the principal direction approach.

References

Agathos A, Pratikakis I, Perantonis S, Sapidis N, Azariadis P. 3D mesh segmentation methodologies for CAD applications. Computer-Aided Design and Applications 2007;4(6):827–842.

Akagunduz E, Ulusoy I. 3D object representation using transform and scale invariant 3d features. Proceedings of the 11th International Conference on Computer Vision; 2007 Oct 14–21; Rio de Janeiro, Brazil. 2007. p. 18.

Al-Osaimi FR, Bennamoun M, Mian AS. Interest-point based face recognition from range images. In: Rajpoot NM and Bhalerao AH, editors. Proceedings of the British Machine Conference; 2007 Sept 10–13; University of Warwick, UK. London: BMVA Press; 2007. p. 34.1–34.10.

Al-osaimi FR, Bennamoun M, Mian AS. Integration of local and global geometrical cues for 3d face recognition. Pattern Recognition 2008;41(2):1030–

1040.

Al-Osaimi FR, Bennamoun M, Mian AS. An expression deformation approach to nonrigid 3d face recognition. International Journal of Computer Vision 2009;81(3):302–316.

Amor BB, Ardabilian M, Chen L. New experiments on ICP-based 3D face recognition and authentication. In: Proceedings of 18th IEEE International Conference on Pattern Recognition, Volume 3; 2006 Aug 20–24, Hong Kong, China. New York: IEEE. 2006. p. 1195–1199.

Bar C. Elementary Differential Geometry. London: Cambridge University Press. 2010.

Besl P, Jain R. Segmentation through variable-order surface fitting. IEEE Transactions on Pattern Analysis and Machine Intelligence 1988;10:167–192.

Besl PJ, Mckay HD. A method for registration of 3-D shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence 1992;14(2):239–256.

Botsch M, Pauly M, Kobbelt L, Alliez P, Levy B, Bischoff S, Rossl C. Geometric modeling based on polygonal meshes. In: Proceedings of ACM SIGGRAPH Course Notes; 2007 Aug 5–9, San Diego, CA. New York: Association for Computing Machinery. 2007.

Bronstein AM, Bronstein MM, Kimmel R. Three-dimensional face recognition. International Journal of Computer Vision 2005;64(1):5–30.

Chang K, Bowyer K, Flynn P. Multiple nose region matching for 3D face recognition under varying facial expression. IEEE Transactions on Pattern Analysis and Machine Intelligence 2006;28(10):1695–1700.

Chen Y and Medioni G. Object modeling by registration of multiple range images. In: Medoni G, editor. Proceedings of IEEE International Conference on Robotics and Automation. Volume 3; 1991 Apr 9–11, Sacramento, CA. New York: IEEE. 1991. p. 2724–2729.

Dibeklioglu H, Salah A, Akarun L. 3D facial landmarking under expression, pose, and occlusion variations. In: Proceedings of the 2nd IEEE International Conference on Biometrics: Theory, Applications and Systems; 2008 Sep 29–Oct 1, Arlington, VI. New York: IEEE. 2008. pp. 1–6.

Garland M, Heckbert *PS.* Surface simplification using quadric error metrics. In: Owen GS, Whitted T, Mones-Hattal B, editors. Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '97;1997 Aug 3–8, Los Angeles, CA. New York: ACM Press/Addison-Wesley

Publishing Co. 1997. p. 209–216.

Gnanaprakasam C, Sumathi S, Malini RR. Average-half-face in 2D and 3D using wavelets for face recognition. In: Niola V, Quartieri J, Neri F, et al., editors. Proceedings of the 9th WSEAS International Conference on Signal Processing, SIP'10; 2010 May 2931. Stevens Point, WI: World Scientific and Engineering Academy and Society (WSEAS). 2010. p. 107–112.

Gokberk B, Irfanoglu M, Akarun L. 3D shape-based face representation and feature extraction for face recognition. Image and Vision Computing 2006;24(8):857–869.

Greenspan M, Yurick M. Approximate *k-d* tree search for efficient ICP. In: Proceedings of the 4th International Conference on 3-D Digital Imaging and Modeling; 2003 Oct 6–10, Banff, Canada. New York: IEEE. 2003. p. 442–448.

Gunlu G, Bilge H. 3D face decomposition and region selection against expression variations. In: Proceedings of the 20th IEEE International Conference on Pattern Recognition; 2010 Aug 23–26, Istanbul, Turkey. New York: IEEE. 2010. p. 1298–1301.

Harguess J, Gupta S, Aggarwal J. 3D face recognition with the average-half-face. In: Proceedings of the 19th IEEE International Conference on Pattern Recognition; 2008, Dec 811, Tampa, FL. New York: IEEE. 2008. p. 1–4.

Heckbert PS, Garland M. Survey of polygonal surface simplification algorithms. Technical Report, School of Computer Science, Carnegie Mellon University; 1997 and In: Proceedings of the ACM SIGGRAPH *Conference, Multiresolution Surface* Modeling, Course Note # 25; 1997 Aug 5, Los Angeles, CA. New York: ACM. 1997. p. 209–216.

Husken M, Brauckmann M, Gehlen S, Von der Malsburg C. Strategies and benefits of fusion of 2D and 3D face recognition. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition – Workshops, CVPR Workshops; 2005 Jun 20–25, San Diego, CA. New York: IEEE. 2005 p. 174.

Kakadiaris IA, Passalis G, Theoharis T, Toderici G, Konstantinidis I, Murtuza MN 2005 8d-thermo cam: Combination of geometry with physiological information for face recognition. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 2; 2005 Jun 20–25, San Diego, CA. New York: IEEE. p. 1183.

Kakadiaris IA, Passalis G, Toderici G, Murtuza MN, Lu Y, Karampatziakis N,

Theoharis T. Three-dimensional face recognition in the presence of facial expressions: An annotated deformable model approach. IEEE Transactions on Pattern Analysis and Machine Intelligence 2007;29(4):640–649.

Kettner L. Using generic programming for designing a data structure for polyhedral surfaces. Computational Geometry: Theory and Applications 1999; 13:65–90.

Lee J, Milios E. Matching range images of human faces. In: Proceedings of the 3rd IEEE International Conference on Computer Vision; 1990 Dec 4–7, Osaka, Japan. 1990. New York: IEEE. p. 722–726.

Lo TWR, Siebert JP. Sift keypoint descriptors for range image analysis. Annals of the British Machine Vision Association 2008;2008(3):1–18.

Lowe DG. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 2004;60:91–110.

Lu X, Jain AK, Colbry D. Matching 2.5D face scans to 3D models. IEEE Transactions on Pattern Analysis and Machine Intelligence 2006;28(1):31–43.

Luebke D. A developer's survey of polygonal simplification algorithms. IEEE Computer Graphics and Applications 2001;21(3):24–35.

McCool C, Cook J, Chandran V, Sridharan S. Feature modelling of PCA difference vectors for 2D and 3D face recognition. In: Proceedings of the IEEE International Conference on Video and Signal Based Surveillance, AVSS'06; 2006 Dec 11, Sydney, Australia. New York: IEEE. p. 57.

Mian A, Bennamoun M, Owens R. Region-based matching for robust 3D face recognition. In: Clocksin W, Fitzgibbon A, Torr P, editors. Proceedings of the 16th British Machine Vision Conference, Volume 1; 2005 Sept 5–8, Oxford, UK. London: BMVC. 2005. p. 199–208.

Mian A, Bennamoun M, Owens R. An efficient multimodal 2D-3D hybrid approach to automatic face recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 2007;29:1927–1943.

Mian AS, Bennamoun M, Owens R. Keypoint detection and local feature matching for textured 3D face recognition. International Journal of Computer Vision 2008;79:1–12.

Moreno AB, Sanchez A, Vlez JF, Daz FJ. Face recognition using 3D surface-extracted descriptors In: Proceedings of the Irish Machine Vision and Image Processing Conference, IMVIP'03; 2003 Sept 3–5, University of Ulster, Coleraine, Ireland. Dublin, Ireland: SIGMEDIA. 2003.

O'Neill B. Elementary Differential Geometry. Revised 2nd edition. New York/London: Elsevier/Academic Press; 2006.

Pan G, Wu Z. 3d Face Recognition from Range Data. International Journal of Image and Graphics 2005;5(3):573– 593.

Renze KJ, Oliver JH. Generalized unstructured decimation. IEEE Computer Graphics and Applications 1996;16:24–32.

Rossignac J, Borrell P. Multiresolution 3D approximations for rendering complex scenes. In: Falcidieno B, Kunii TL, editors. Proceedings of Geometric Modeling in Computer Graphics; 1993 Jun 28–Jul 2, Genova, Italy. Heidelberg: Springer Verlag. 1993. p. 455–465.

Schroeder WJ, Zarge JA, Lorensen WE. Decimation of triangle meshes. In: Proceedings of the 19th Annual Conference on Computer Graphics, SIGGRAPH '92; 1992 Jul 26–31 Chicago, IL. New York: ACM. 1992. p. 65–70.

Segundo M, Queirolo C, Bellon O, Silva L. Automatic 3D facial segmentation and landmark detection. In: Cucchiara R, editor. Proceedings of the 14th International Conference on Image Analysis and Processing, ICIAP '07; 2007 Sept 10–14, Modena, Italy. New York: IEEE Computer Society. 2007. p. 431–436.

seng Chua C, Han F, khing Ho Y. 3D human face recognition using point signature In: Proceedings of the 4th IEEE International Conference on Automatic Face and Gesture Recognition; 2000 Mar 28–30, Grenoble, France. California: IEEE Computer Society. 2000. p. 233–238.

Shamir A. A survey on mesh segmentation techniques. Computer Graphics Forum 2008;27(6):1539–1556.

Smeets D, Fabry T, Hermans J, Vandermeulen D, Suetens P. Isometric deformation modeling using singular value decomposition for 3D expression-invariant face recognition. In: Proceedings of the 3rd IEEE International Conference on Biometrics: Theory, Applications and Systems, BTAS'09; 2009 Sept 28–30, Washington, DC. Piscataway, NJ: IEEE Press. 2009. p. 68–73.

Tanaka H, Ikeda M and Chiaki H. Curvature-based face surface recognition using spherical correlation. principal directions for curved object recognition. In: Proceedings of the 3rd IEEE International Conference on Automatic Face and Gesture Recognition; 1998 Apr 14–16, Nara, Japan. New York: IEE. 1998. p. 372–377.

Wang Y, Chua CS, Ho YK, Ren Y. Integrated 2D and 3D images for face

recognition. In: Proceedings of 11th International Conference on Image analysis and Processing; 2001 Sep 2628; Palermo, Italy. California: The IEEE Computer Society, 2001. p. 0048.

Wang Y, Pan G, Wu Z, Wang Y. Exploring facial expression effects in 3d face recognition using partial ICP. In: Proceedings of the 7th Asian Conference on Computer Vision, ACCV'06. Part1; 2006 Jan 13–16, Hyderabad, India. New York: Springer. 2006. p. 581–590.

Wiskott L, Fellous JM, Kuiger N, von der Malsburg C. Face recognition by elastic bunch graph matching. IEEE Transactions on Pattern Analysis and Machine Intelligence 1997;19(7):775–779.

Zhang L, Razdan A, Farin GE, Femiani J, Bae M, Lockwood C. 3D face authentication and recognition based on bilateral symmetry analysis. The Visual Computer 2006;22:43–55.

3

3D Face Surface Analysis and Recognition Based on Facial Curves

Hassen Drira,[1] Stefano Berretti,[2] Boulbaba Ben Amor,[1] Mohamed Daoudi,[1] Anuj Srivastava,[3] Alberto del Bimbo[2] and Pietro Pala[2]

[1]Institut Mines-Télécom/Télécom Lille 1, France

[2]Dipartimento di Sistemi e Informatica, University of Firenze, Firenze, Italy

[3]Department of Statistics, Florida State University, Tallahassee, USA

3.1 Introduction

There is an increasing interest in analyzing the shapes of facial surfaces with many applications including biometrics, facial surgery, video communications, and 3D animation. This interest is fuelled by the advent of cheaper and lighter scanners that can provide high-resolution measurements of the geometry and texture of human facial surfaces. One general goal here is to develop computational tools for analyzing 3D face data, in particular, comparing the shapes of facial surfaces. Such a tool can be used to recognize human beings according to their facial shapes, to measure changes in a facial shape due to a surgery, or to study/capture the variations in facial shapes during conversations and expressions of emotions. Additionally, a subproblem may be to find an optimal deformation of one surface into another under some chosen criterion. These deformations can be useful in defining the summary statistics of a collection of faces. For example, one can compute an *average* of faces of different people, under different facial expressions. Efficient tools for understanding and studying variability in facial shapes are of great importance in our biometric-oriented society.

Major issues in surface analysis: What are the difficulties in performing standard shape analysis on the face data? The main issue is that there is no

canonical coordinate system to represent and study the geometry of a nonlinear surface such as a face. To highlight this issue, contrast the geometry of a surface with that of a curve. The points along a curve have a fixed ordering that allows us to impose a coordinate system for studying shapes of curves. Although the parametrization for a curve may not be unique, the task of analyzing the shapes of curves modulo all possible parametrizations are still tractable. More specifically, the space of all possible parametrizations of a curve is the set of all one-dimensional diffeomorphisms, and there exist efficient algorithms (such as the dynamic programming algorithm) for optimally matching curves over all parametrizations. For surfaces, there is no natural ordering of points and, hence, no natural way of parametrizing them. Thus, the goal of analyzing shapes of surfaces, modulo all possible reparametrizations, becomes quite difficult to achieve and leads to enormous computational challenges.

3.2 Facial Surface Modeling

Let S_1 and S_2 be two facial surfaces modeled as Riemannian manifolds. The two faces belong to the same person, but S_1 is a neutral face, whereas S_2 is an expressive one. Let $f : S_1 \rightarrow S_2$ be a diffeomorphism modeling the expression (see Figure 3.1).

Definition 3.2.1 Diffeomorphism *A smooth map* $f : S_1 \rightarrow S_2$ *that are bijective and whose inverse* $f^{-1} : S_2 \rightarrow S_1$ *is smooth is called diffeomorphism. If there is a diffeomorphism between them,* S_1 *and* S_2 *are said to be diffeormorphic.*

Definition 3.2.2 **Isometry** *If* S_1 *and* S_2 *are surfaces, a diffeomorphism* $f : S_1 \rightarrow S_2$ *is called an isometry if it takes curves in* S_1 *to curves of the same length in* S_2*. If an isometry* $f : S_1 \rightarrow S_2$ *exists, we say that* S_1 *and* S_2 *are isometric.*

In Bronstein *et al.* (2005), the authors assume that facial expressions can be modeled as isometries of the facial surface. This assumption reduces the problem of comparing faces in the presence of facial expressions to the problem of isometric surface matching. Indeed, Bronstein *et al.* (2005) placed 133 markers on a face and computed the distances between these points under facial expressions deformations. The distribution of the absolute change of the geodesic and the Euclidean distances shows that the geodesic distance is more stable than Euclidean one under facial expression variations.

In other words, this assumption means that *f* preserves the geodesic distance *d*

between every pair of points of facial surface, such that

(3.1) $d_{S_1}(x, y) = d_{S_2}(f(x), f(y)), \forall x, y \in S_1.$

This assumption is valid in the case of a small expression. The large expression causes some combinations of shrinking and/or stretching. Under such variations, the geodesic distance is no more stable. To quantitatively verify this assumption, we placed four markers on a face and tracked the change of the geodesic and Euclidean distances under large expression variations.

Figure 3.2 shows that a change in expression from neutral to non-neutral generally results in a shrinking or a stretching of the face shape, and both the Euclidean and geodesic distances between points on the face are changed.

Figure 3.1 Points correspondence (neutral face at left and expressive face at the right)

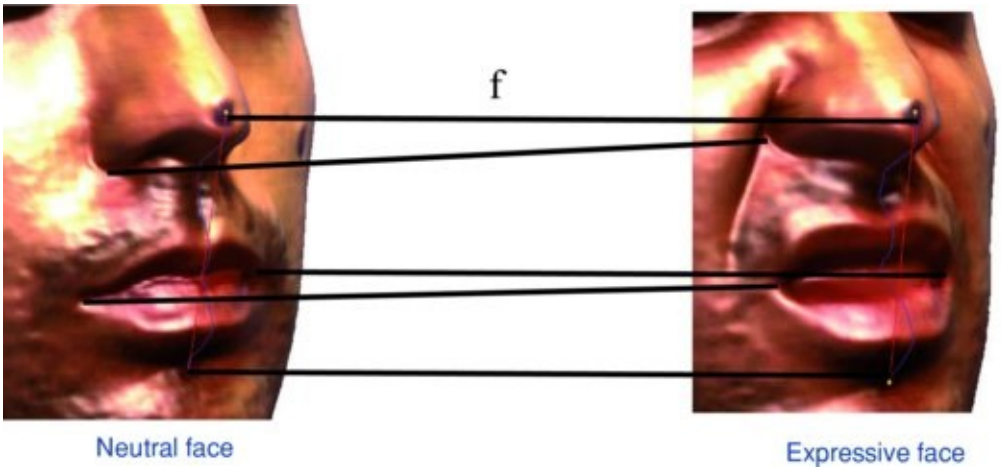
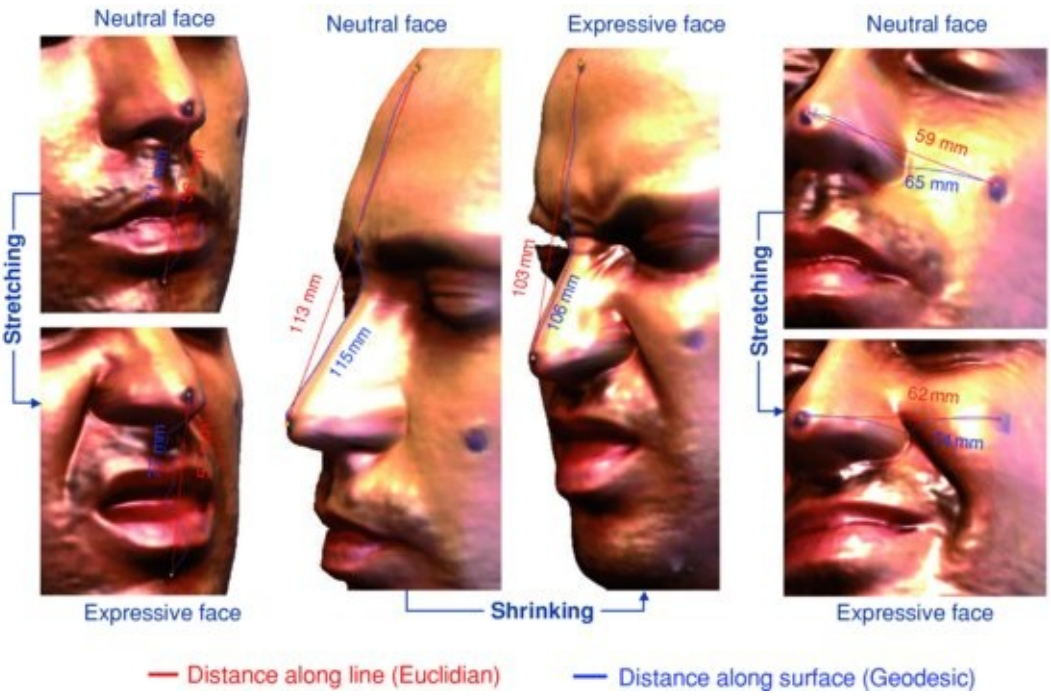


Figure 3.2 Significant changes in both Euclidean and surface distances under different face expressions. Copyright © 1969, IEEE



In one of the illustrated cases, these distances decrease (from 113 to 103 mm for Euclidean distance and from 115 to 106 mm for the geodesic distance), whereas in the other two cases they increase. This clearly shows that large expression variations cause stretching and shrinking of face shape, and under such elastic deformations, neither the Euclidean distance nor the surface distance is preserved. Hence, the assumption of isometric deformation of the shape of the face is not always valid. One way to handle such elastic deformations of faces due to changes in expressions is to use elastic shape analysis.

3.3 Parametric Representation of Curves

In the following sections, we provide a brief introduction to the elementary differential geometry of curves terms used in the manuscript. For details, please refer to some standard textbooks on differential geometry such as Pressley (2001).

Definition 3.3.1 A parametrized curve *A parametrized curve in* $\mathbb{R}^3$ *is a map* $\gamma : [a, b] \to \mathbb{R}^3$*, for some a, b with* $-\infty \leq a < b \leq +\infty$*. The symbol [a, b] denotes the open interval* $[a, b] = \{t \in \mathbb{R}, a < t < b\}$.

Definition 3.3.2 Reparametrization *A parametrized curve* $\tilde{\gamma} : [a, b] \to \mathbb{R}^3$ *is a reparametrization of a parametrized curve* $\gamma : [a, b] \to \mathbb{R}^3$ *if there is a smooth*

bijective map $\Phi : [\tilde{a}, \tilde{b}] \to [a, b]$ *(the reparametrization map) such that the inverse map* $\Phi^{-1} : [a, b] \to [\tilde{a}, \tilde{b}]$ *also smooth and*

(3.2) $\tilde{\gamma}(\tilde{t}) = \gamma(\Phi(\tilde{t}))$ for all $\tilde{t} \in [\tilde{a}, \tilde{b}]$.

Φ *is also called a diffeomorphism from [a,b] to itself.*

Note that, since Φ has a smooth inverse, $\tilde{\gamma}$ is a reparametrization of γ:

(3.3) $\tilde{\gamma}(\Phi^{-1}(t)) = \gamma(\Phi(\Phi^{-1}(t))) = \gamma(t)$ for all $t \in [a, b]$.

Two curves that are reparametrizations of each other have the same image, so they should have the same geometric properties.

Example 3.3.3 One parametrization of a circle defined by $x^2+y^2=1$ is $\gamma = (\cos(t), \sin(t))$, another parametrization of a circle is $\tilde{\gamma} = (\sin t, \cos t)$. To see that $\tilde{\gamma}$ is a reparametrization of γ, we have to find a reparametrization map such that

$(\cos(\Phi(t)), \sin(\Phi(t))) = (\sin(t), \cos(t))$.

One solution is $\Phi(t) = \pi/2 - t$.

In general the analysis of a curve is simplified when it is known to be unit-speed.

Definition 3.3.4 *Arc-Length The arc-length of a curve* γ *starting at the point* $\gamma(t_0)$ *is the functions(t) given by*

(3.4) $s(t) = \int_{t_0}^{t} \|\dot{\gamma}(u)\| \, du.$

Definition 3.3.5 *A point* $\gamma(t)$ *of a parametrized curve* γ *is called a regular point if* $\dot{\gamma(t)} \neq 0$*; otherwise* $\gamma(t)$ *is a singular point of* γ*. A curve is regular if all of its points are regular.*

Proposition 3.3.6 *A parametrized curve has a unit-speed reparametrization if and only if it is regular.*

One can show that the arc-length is the only unit-speed parameter on a regular curve.

3.4 Facial Shape Representation Using Radial Curves

Let β_α denotes the radial curve on *S*, which makes an angle α with a reference radial curve. The reference curve is chosen to be the vertical curve once the face has been rotated to the upright position. In practice, each radial curve β_α is obtained by slicing the facial surface by a plane P_α that has the nose tip as its origin and makes an angle, α, with the plane containing the reference curve, as

shown in Figure 3.3; that is, the intersection of P_α with S gives the radial curve β_α. We repeat this step to extract radial curves from the facial surface at equal angular separation. Each curve is indexed by the angle α. Figure 3.4 shows an example of some radial curves.

Figure 3.3 Procedure for extraction of radial curves, a curve β_α^γ is obtained by slicing the facial surface by P_α defined by the angle α with the vertical plane and having as origin the nose tip. Copyright © 2012, IEEE

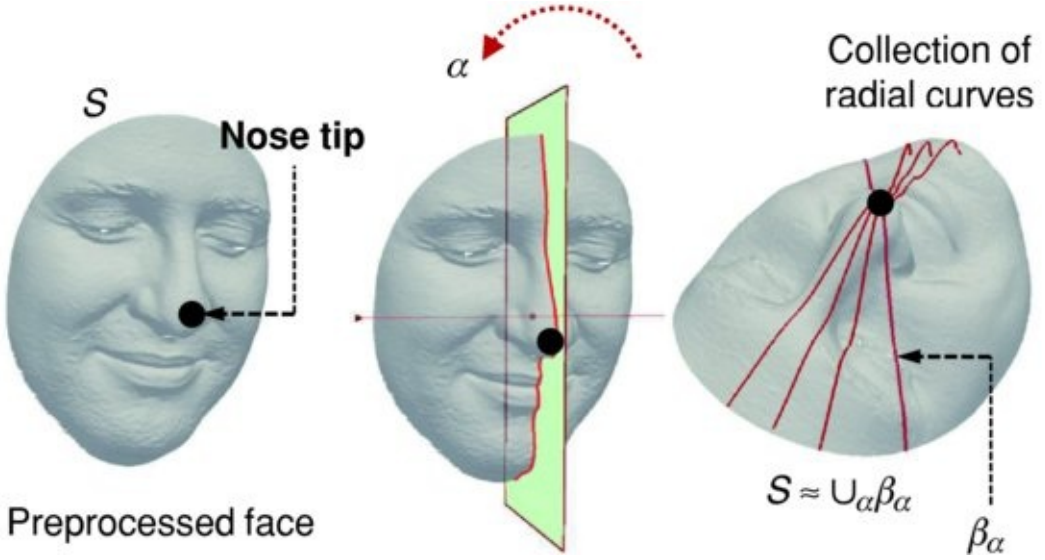
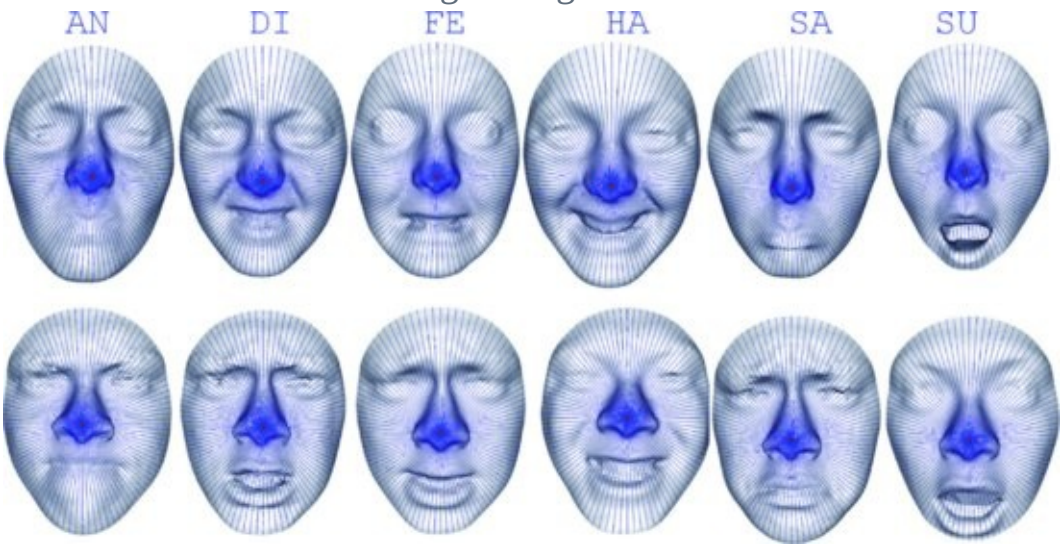


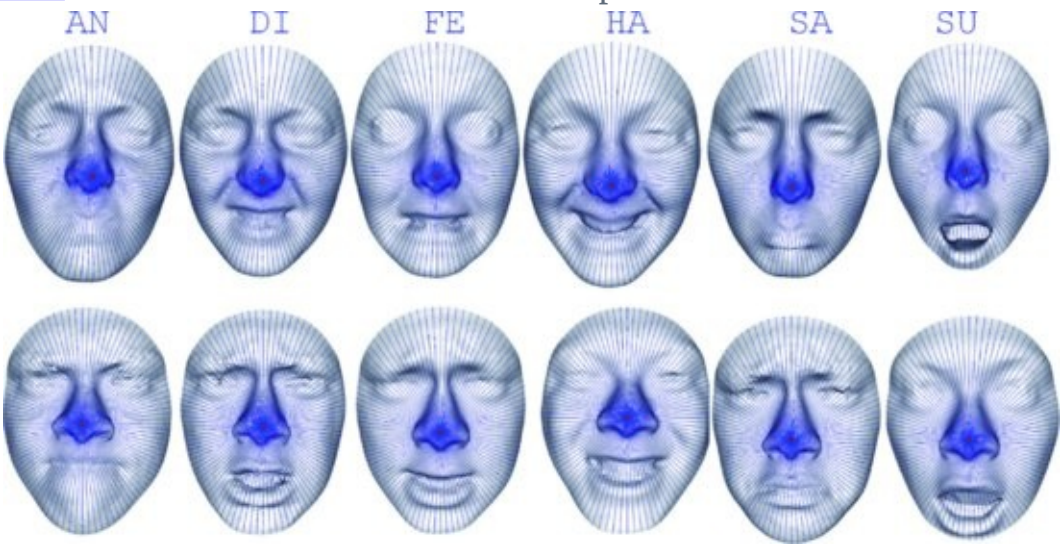
Figure 3.4 Radial curves extraction: left image illustrates the intersection between the face surface and a plan to form two radial curves. The collection of radial curves is illustrated on the right image



If needed, we can approximately reconstruct S from these radial curves according to $S \approx \cup_\alpha \beta_\alpha = \cup_\alpha \{S \cap P_\alpha\}$ as illustrated in Figures 3.4 and 3.5. This

indexed collection of radial curves captures the shape of a facial surface and forms its mathematical representation. In practice, we used 80 curves.

Figure 3.5 Radial curves extraction: facial expression



We have chosen to represent a surface with a collection of curves because we have better tools for elastically analyzing shapes of curves than we have for surfaces. More specifically, we are going to utilize an elastic method for studying shapes of curves that is especially suited for modelling deformations associated with changes in facial expressions.

3.5 Shape Space of Open Curves

In the last few years, many approaches have been developed to analyze shapes of 2D curves. We can cite approaches based on Fourier descriptors, moments, or the median axis. More recent works in this area consider a formal definition of shape spaces as a Riemannian manifold of infinite dimension on which they can use the classic tools for statistical analysis. The recent results of Michor and Mumford (2006) and Yezzi and Mennucci (2005) show the efficiency of this approach for 2D curves. Joshi *et al.* (2007) have recently proposed a generalization of this work to the case of curves defined in $\mathbb{R}^n$. We will adapt this work to our problem since our 3D curves are defined in $\mathbb{R}^3$.

3.5.1 Shape Representation

We start by considering a curve β in $\mathbb{R}^3$. Although there are several ways to analyze shapes of curves, an elastic analysis of the parametrized curves is

particularly appropriate in our application – face analysis under facial expression variations. This is because of the following reasons: (1) Such analysis uses the square-root velocity function representation, which allows comparison local facial shapes in the presence of elastic deformations. (2) This method uses a square-root representation under which the elastic metric reduces to the standard $\mathbb{L}^2$ metric and thus simplifies the analysis (3) Under this metric, the Riemannian distance between curves is invariant to the reparametrization. To analyze the shape of β, we shall represent it mathematically using a square-root representation of β as follows: For an interval *I*=[0, 1], let $\beta : I \longrightarrow \mathbb{R}^3$ be a curve and define $q : I \longrightarrow \mathbb{R}^3$ to be its square-root velocity function (SRVF), given by

$$q(t) \doteq \frac{\dot{\beta}(t)}{\sqrt{|\dot{\beta}(t)|}}. \quad (3.5)$$

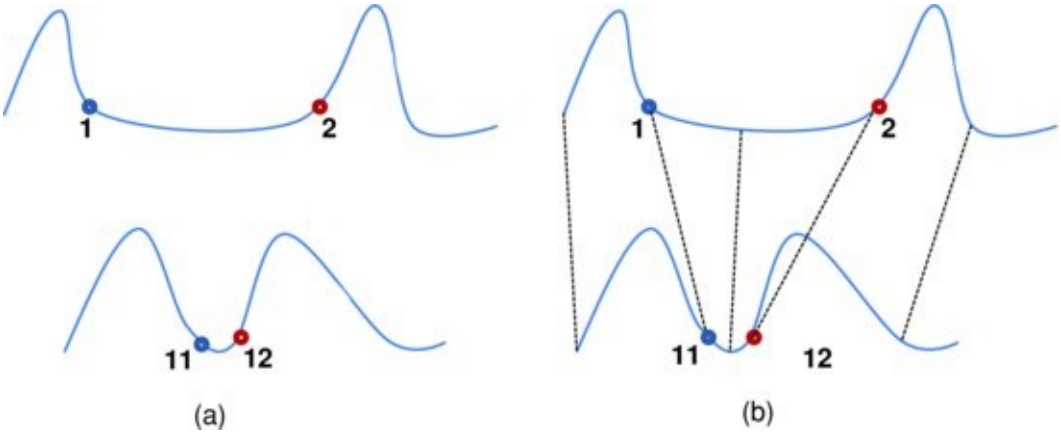
Here *t* is a parameter $\in I$ and $|\cdot|$ is the Euclidean norm in $\mathbb{R}^3$. We note that *q*(*t*) is a special function that captures the shape of β and is particularly convenient for shape analysis, as we describe next. The classical elastic metric for comparing shapes of curves becomes the $\mathbb{L}^2$-metric under the SRVF representation (Srivastava et al., 2011). This point is very important as it simplifies the calculus of elastic metric to the well-known calculus of functional analysis under the $\mathbb{L}^2$-metric. Also, the squared $\mathbb{L}^2$-norm of *q*, given by $\|q\|^2 = \int_I \langle q(t), q(t) \rangle \, \mathrm{d}t = \int \|\dot{\beta}(t)\| \mathrm{d}t,$ is the length of β. If we set $\|q\| = 1$, implying all curves are rescaled to unit length, then translation and scaling variability have been removed by this mathematical representation of curves.

$$\|\beta_1 - \beta_2\| \neq \|\beta_1 \circ \gamma - \beta_2 \circ \gamma\|. \quad (3.6)$$

Consider the two curves in Figure 3.6**a**. Let us fix the parametrization of the top curve to be arc-length, that is, we are going to traverse that curve with speed equal to one. To have the best matching of the curves, we should know at what rate we must move along the two curves so that points reached at the same time on two curves are as close as possible under some geometric criterion. In other words, peaks and valleys should be reached at the same time. Figure 3.6**b** illustrates the matching, where point 1 on the top curve matches point 11 on the down curve. The part between the point 1 and 2 on the top curve shrinks on the curve 2. Therefore, the point 2 matches the point 12 on the second curve. An elastic metric is the measure of that shrinking.

Figure 3.6 Illustration of elastic metric. In order to compare the two curves in (a), some combination of stretching and bending are needed. The elastic metric

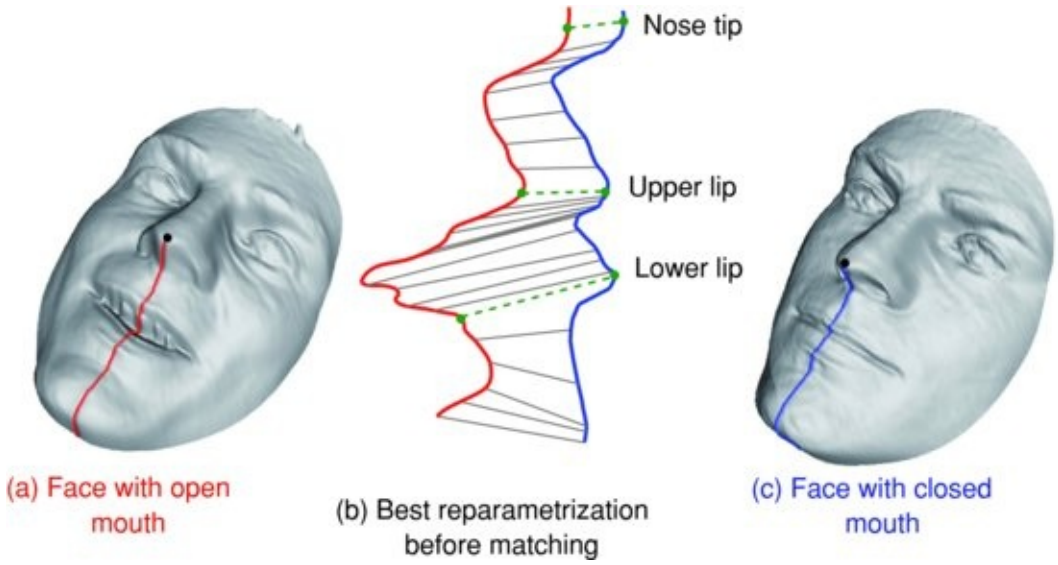
measures the amounts of these deformations. The optimal matching between the two curves is illustrated in (b). Copyright © 2012, IEEE



The use of SRV representation allows the reparametrization group to act by isometry on the manifold of SRV representations. This point is very important as the curve matching could be done after reparametrization. The change of parametrization before the matching is able to reduce the effect of stretching and/or stretching of the curve.

This idea is illustrated in Figure 3.7. The task is to match two radial curves on two faces with two different expressions. The expression in the face at the left induces open mouth in contrast with the expression in face at the right. As shown in the middle panel, the anatomic points on the curves (upper and down lips) match together after reparametrizing one curve. More formally, the elastic matching of the curves allows better matching of anatomical points on them.

Figure 3.7 An example of matching of two radial curves extracted from two faces. (a) A curve on an open mouth, (c) a curve on closed mouth, (b) change of parametrization before matching. Copyright © 1969, IEEE



3.5.2 Geometry of Preshape Space

We denote $\mathcal{C} = \{q : I \to \mathbb{R}^3, \|q\| = 1\} \subset \mathbb{L}^2(I, \mathbb{R}^3)$ as the space of all unit-length, elastics curve. The space $\mathcal{C}$ is in fact an infinite-dimensional unit-sphere and represents the preshape space of all open elastic curves invariant to translation and uniform-scaling.

With the $\mathbb{L}^2$ metric on its tangent spaces, $\mathcal{C}$ becomes a Riemannian manifold. In particular, since the elements of $\mathcal{C}$ have a unit $\mathbb{L}^2$ norm, $\mathcal{C}$ is a hypersphere in the Hilbert space $\mathbb{L}^2(I, \mathbb{R}^3)$. To compare the shapes of two radial curves, we can compute the distance between them in $\mathcal{C}$ under the chosen metric. This distance is defined as the length of a geodesic connecting the two points in $\mathcal{C}$. Since $\mathcal{C}$ is a sphere, the geodesic length between any two points $q_1, q_2 \in \mathcal{C}$ is given by

(3.7) $$d_c(q_1, q_2) = \cos^{-1}(\langle q_1, q_2 \rangle),$$

and the geodesic path $\psi : [0, 1] \to \mathcal{C}$, is given by

$$\psi(\tau) = \frac{1}{\sin(\theta)}(\sin((1-\tau)\theta)q_1 + \sin(\theta\tau)q_2),$$

where $\theta = d_c(q_1, q_2)$. Figure 3.8 illustrates the space $\mathcal{C}$ and geodesic path between two elements of that space. As illustrated in Figure 3.8, the space of all curves is a sphere in Hilbert space. Thus, the geodesic on the space of curves is the arc of the great circle connecting the two curves seen as elements of this sphere.

Figure 3.8 Illustration of shape space and geodesic between its elements

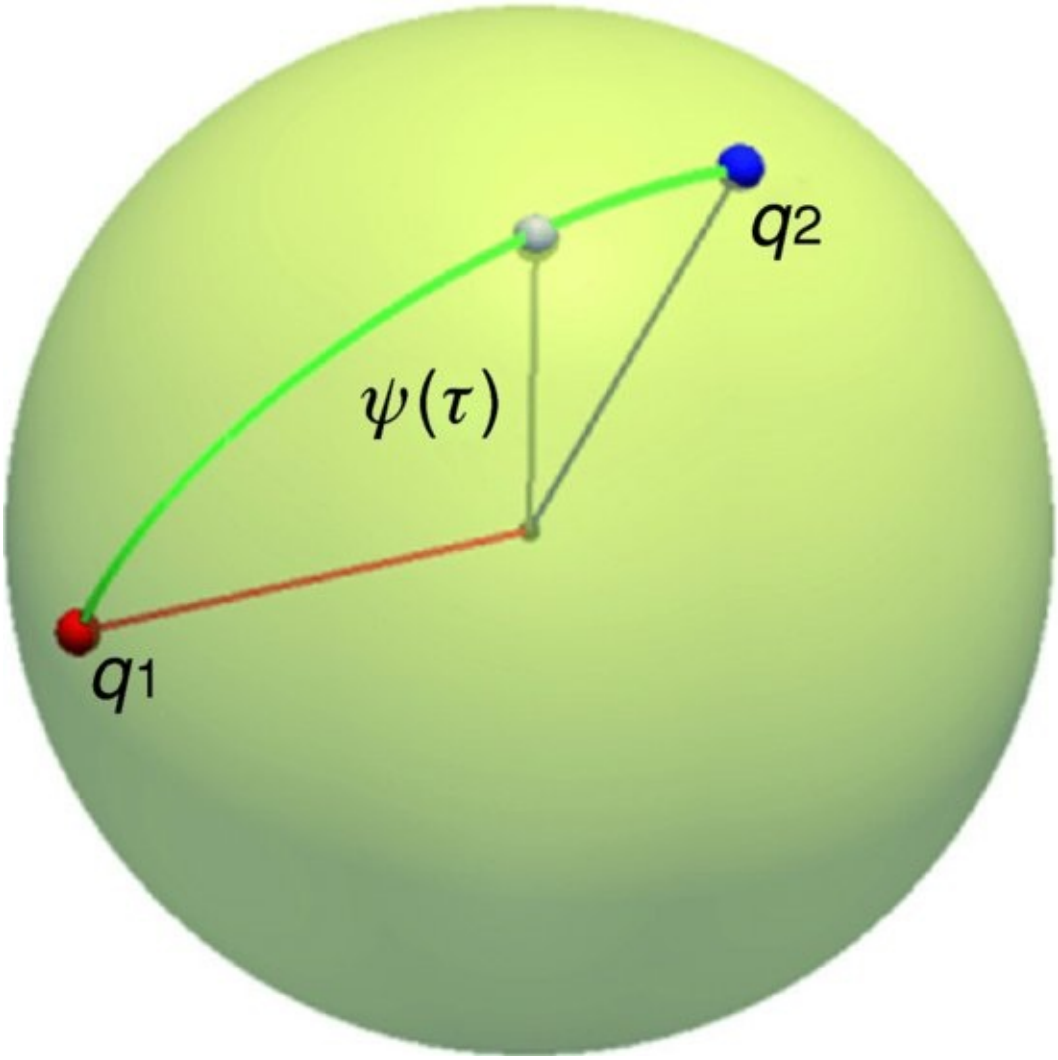
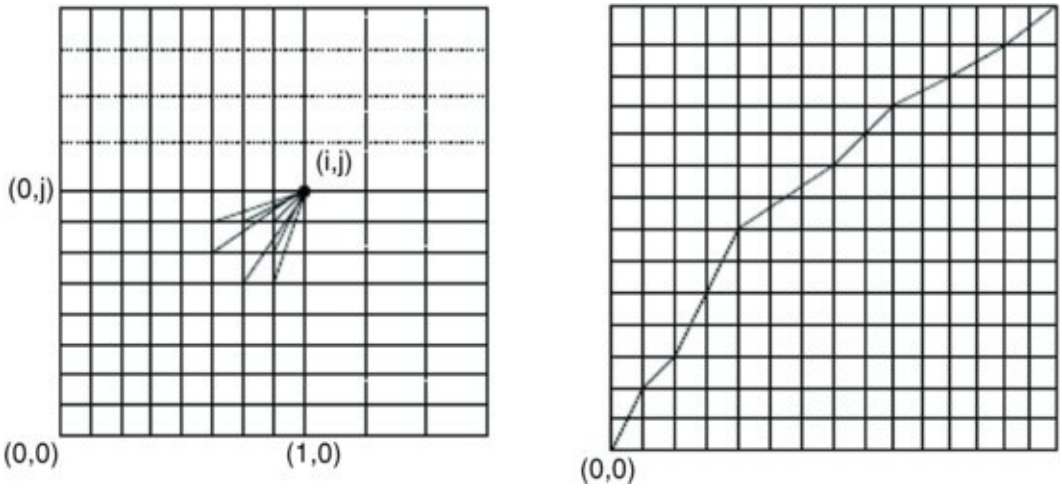


Figure 3.9 Right: an illustration of some nodes that are allowed to go to (i/n, j/n) point on the graph. Left: an example of a ϕ_1 function restricted to a finite graph



It is easy to see that several elements of C can represent curves with the same

shape. For example, if we rotate a face in $\mathbb{R}^3$, and thus its facial curves, we get different SRVFs for the curves but their shapes remain unchanged. Another similar situation arises when a curve is reparametrized; a reparametrization changes the SRVF of curve but not its shape. In order to handle this variability, we define orbits of the rotation group $SO(3)$ and the reparametrization group Γ as equivalence classes in $\mathcal{C}$. Here, Γ is the set of all orientation-preserving diffeomorphisms of I (to itself) and the elements of Γ are viewed as reparametrization functions. For example, for a curve $\beta : I \to \mathbb{R}^3$ and a function $\gamma \in \Gamma$, the curve $\beta \circ \gamma$ is a reparametrization of β. The corresponding SRVF changes according to $q(t) \mapsto \sqrt{\dot{\gamma}(t)}q(\gamma(t))$. We define the equivalent class containing q as

$$[q] = \{\sqrt{\dot{\gamma}(t)}Oq(\gamma(t)) | O \in SO(3), \ \gamma \in \Gamma\}.$$

The set of such equivalence class is called *shape space* of open curves in $\mathbb{R}^3$ denoted by $\mathcal{S} \doteq \mathcal{C}/(SO(3) \times \Gamma)$. Thanks to SRV representation, the groups $\Gamma \times SO(3)$ act by isometries. This is a necessary condition to let the quotient space $\mathcal{S}$ inherits the Riemannian metric from the preshape space $\mathcal{C}$. To obtain geodesics and geodesic distances between elements of $\mathcal{S}$, one needs to solve the optimization problem, which is typically done using dynamic programming.

3.5.3 Reparametrization Estimation by Using Dynamic Programming

Given two shapes q_1 and q_2, the idea is to estimate a nonlinear function γ that reparametrizes q_2 so as to match both shapes closely, we want to solve for

$$\hat{\gamma} = \arg\min_{\gamma \in \Gamma} \int_0^1 \|q_1(t) - \sqrt{\dot{\gamma}(t)}\hat{q}_2(\gamma(t)\|^2 \, dt \tag{3.8}$$

Γ is the set of all reparametrizations. We can solve a discrete approximation of this problem using dynamic programming (DP). A necessary condition for applying DP to such problems is that the cost function is additive in time t. To decompose the problem into several subproblems, define a partial cost function:

$$E(s, t; \gamma) = \int_s^t \|q_1(\tau) - \sqrt{\dot{\gamma}(\tau)}\hat{q}_2(\gamma(\tau)\|^2 \, d\tau \tag{3.9}$$

so the original cost function is simply $E(0, 1; \gamma)$. γ is seen as a graph from (0, 0) to (1, 1) in $\mathbb{R}^2$ such that the slope of this graph is always strictly between 0 and 90 degrees.

Our goal is to find an optimal path from (0, 0) to (1, 1) in $\mathbb{R}^2$, corresponding to

$(t, \gamma(t))$ that minimizes the cost function.

To use a numerical approach, the domain $[0,1] \times [0,1]$ is replaced with a finite grid and we restrict over search to that grid. The grid $G_n \times G_n$ is formed by uniform partition of G_n as $G_n = \{1/n, 2/n, \ldots, (n-1)/n, 1\}$. The search will be done over the set of all restrictions of γ to this grid. The total cost associated with the path is the sum of the costs associated with its linear segments. On an $n \times n$ grid, there are only a finite number of paths, even less when we impose the slope constraint. Actually the path is never vertical or horizontal. However, this number of paths grows exponentially with *n* and we can not possibly search over all possible paths in an exhaustive fashion. Instead, the DP finds the optimal path in $O(n^2)$ time.

Denote a point on the grid (*i*/*n*, *j*/*n*) by (*i*, *j*). Certain nodes are not allowed to go to (*i*, *j*) because of the slope constraint. Denote by N_{ij} the set of nodes that are allowed to go to (*i*, *j*). For instance $N(i,j) = \{(k,l)/0 \le k < i; l < j \le n\}$ is a valid set. Let *L*(*k*, *l*; *i*, *j*) denote a straight line joining the nodes (*k*, *l*) and (*i*, *j*); for $(k,l) \in N_{ij}$ this is a line with slope strictly between 0 and 90 degrees. This sets up the iterative optimization problem:

(3.10) $(\hat{k}, \hat{l}) = \arg\min_{(k,l)\in N_{ij}} E(k/n, l/n; L(k,l;i,j)),$

with E as defined in Equation 3.9. Define the minimum energy of reaching the point (*i*, *j*), in an iterative fashion as

(3.11) $H(i,j) = E(\hat{k}/n, \hat{l}/n; L(\hat{k}, \hat{l}; i, j)) + \hat{H}(\hat{k}, \hat{l}), \text{ with } H(0,0) = 0.$

This subproblem is solved sequentially for each node (*i*, *j*), starting from (1, 1) and increasing *i*, *j* till one reaches the node (*n*, *n*). Tracing the path that results in the energy *H*(*n*, *n*) provides a discrete version of the optimal γ.

Algorithm 1 Dynamic programming algorithm for optimal reparametrization estimation

Input: Discrete version of q_1 and q_2, N_{ij}: set of nodes that are allowed to go to (i, j)
Output: Discrete version of γ

E n by n matrix initialized by zeros;
for $i \leftarrow 1$ **to** n **do**
  E(1,i)=1;
end
for $i \leftarrow 1$ **to** n **do**
  E(i,1)=1;
end
E(1,1)=0;
for $i \leftarrow 2$ **to** n **do**
  for $j \leftarrow 2$ **to** n **do**
    for $Num \leftarrow 1$ **to** $length(N_{ij})$ **do**
      k = i - Nbrs(Num,1);
      l = j - Nbrs(Num,2);
      if $((l > 0)\&(k > 0))$ **then**
        $H_c(Num) = H(k, l) + FunctionE(q_1, q_2, \frac{k}{n}, \frac{i}{n}, \frac{l}{n}, \frac{j}{n})$;
      else
        $H_c(Num) = C$;
      end
      $H(i, j) = min(H_c)$
    end
  end
end

Algorithm 1 summarizes the dynamic pogramming algorithm for optimal reparametrization estimation. In this algorithm, C is a large positive number, function E is a subroutine that computes $E(\hat{k}/n, \hat{l}/n; L(\hat{k}, \hat{l}; i, j))$, and Nbrs is a list of sites used to define N_{ij}. In practice, one often restricts to a smaller subset to seek a computational step sped up. The next effect is that the number of possible values for the slope along the path are further restricted (see Figure 3.9).

An example of this idea is shown in Figures 3.10 and 3.11. The optimal matching using dynamic programming for the two curves corresponding to the open and the closed mouth is illustrated in Figure 3.11**b** and it highlights the elastic nature of this framework. For the left curve, the mouth is open and for the right curve, it is closed. Still the feature points (upper and bottom lips) match each other well. Figure 3.11**d** shows the geodesic path between the two curves in the shape space $\mathcal{S}$ and this evolution looks very natural under the elastic matching.

Figure 3.10 Examples of matching result of matching using dynamic programming

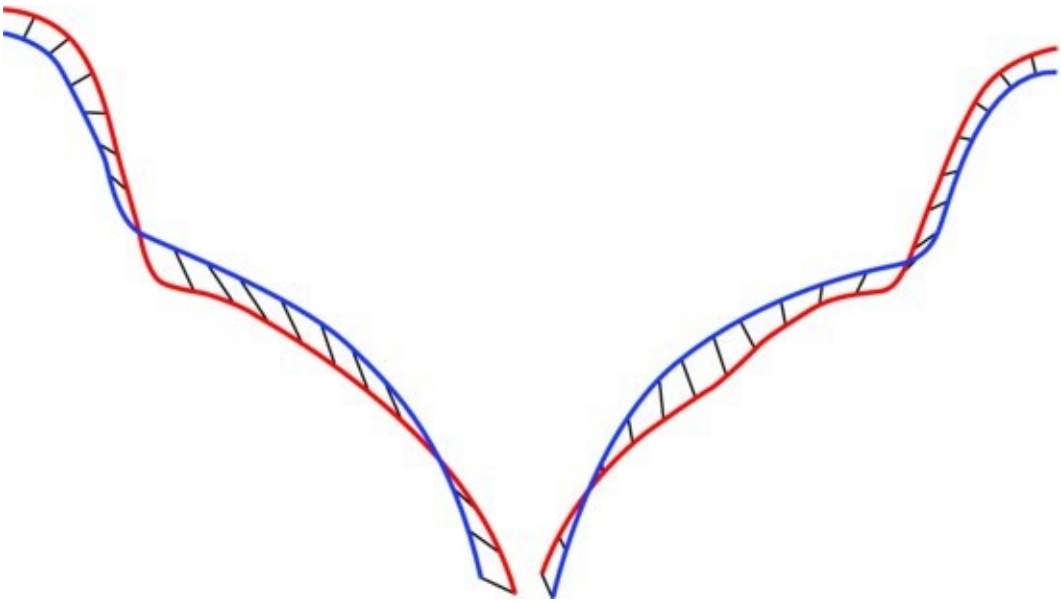
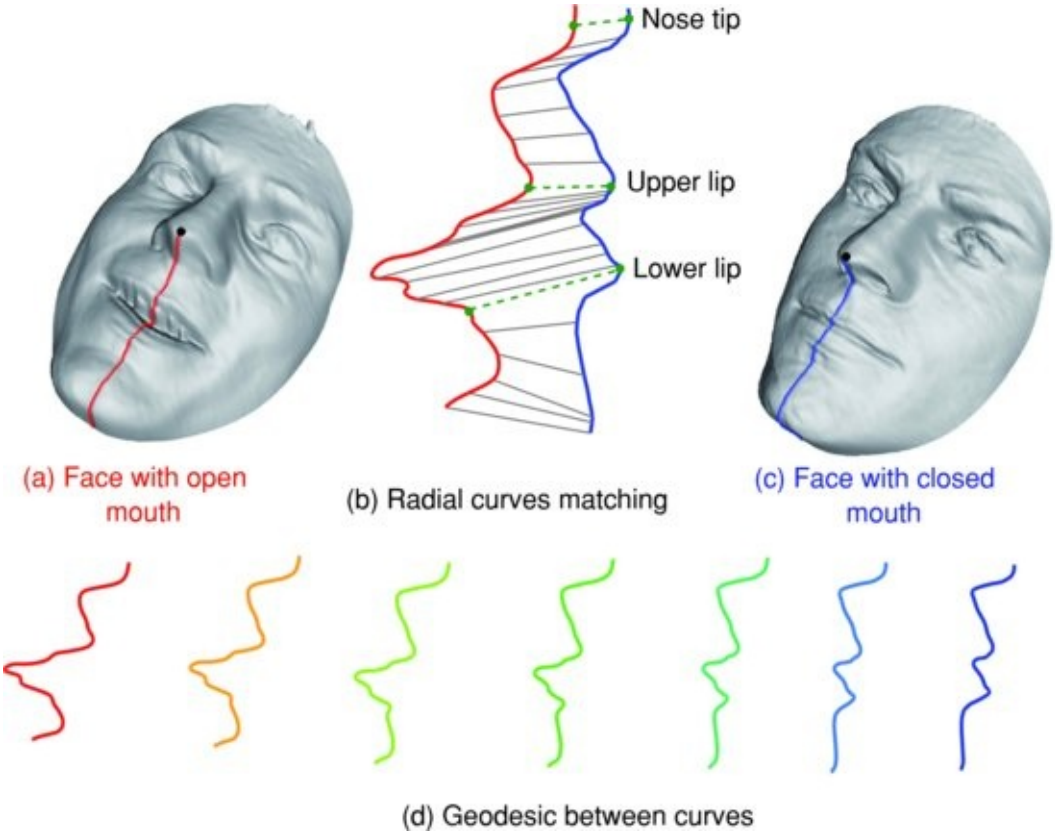


Figure 3.11 Examples of shape matching and geodesic deforming radial curves extracted from two faces: (a) a curve on an open mouth, (b) a result of matching using dynamic programming, (c) a curve on closed mouth, and (d) a geodesic path between curves (a) and (c)



The middle panel in the top row shows the optimal matching for the two

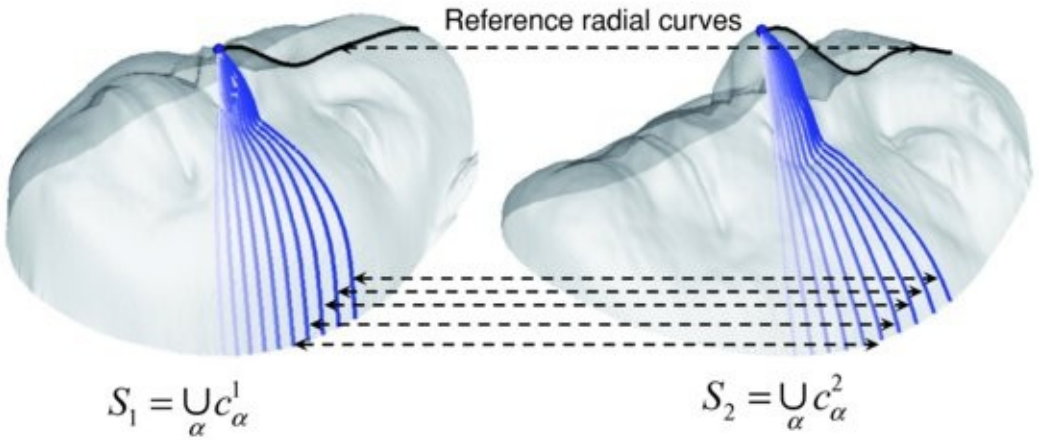
curves obtained using the dynamic programming, and this highlights the elastic nature of this framework. For the left curve, the mouth is open and for the right curve, it is closed. Still the feature points (upper and bottom lips) match each other very well. The bottom row shows the geodesic path between the two curves in the shape space $\mathcal{S}$ and this evolution looks very natural under the elastic matching. Since we have geodesic paths denoting optimal deformations between individual curves, we can combine these deformations to obtain full deformations between faces.

3.5.4 Extension to Facial Surfaces Shape Analysis

Now we extend the framework from radial curves to full facial surfaces. As mentioned earlier, we are going to represent a face surface *S* with an indexed collection of radial curves. That is, $S \leftrightarrow \{\beta_\alpha, \alpha \in [0, \alpha_0]\}$. Through this relation, each facial surface has been represented as an element of the set $\mathcal{S}^{[0,\alpha_0]}$. The indexing provides a correspondence between curves across faces. For example, a curve at an angle α on the probe face is matched with the curve at the same angle on the gallery face. Figure 3.12 illustrates an example of this correspondence. With this correspondence, we can compute pairwise geodesic paths and geodesic distances between the matched curves across faces. This computation has several interesting properties. Firstly, it provides a Riemannian distance between shapes of facial surfaces by combining distances between the corresponding radial curves. The distance between two facial surfaces is given by

$$\mathrm{d} : \mathcal{S}^{[0,\alpha_0]} \times \mathcal{S}^{[0,\alpha_0]} \to \mathbb{R}_{\geq 0}, \mathrm{d}(S^1, S^2) = \frac{1}{\alpha_0} \sum_{\alpha=1}^{\alpha_0} \mathrm{d}_s([q_\alpha^1], [q_\alpha^2]).$$

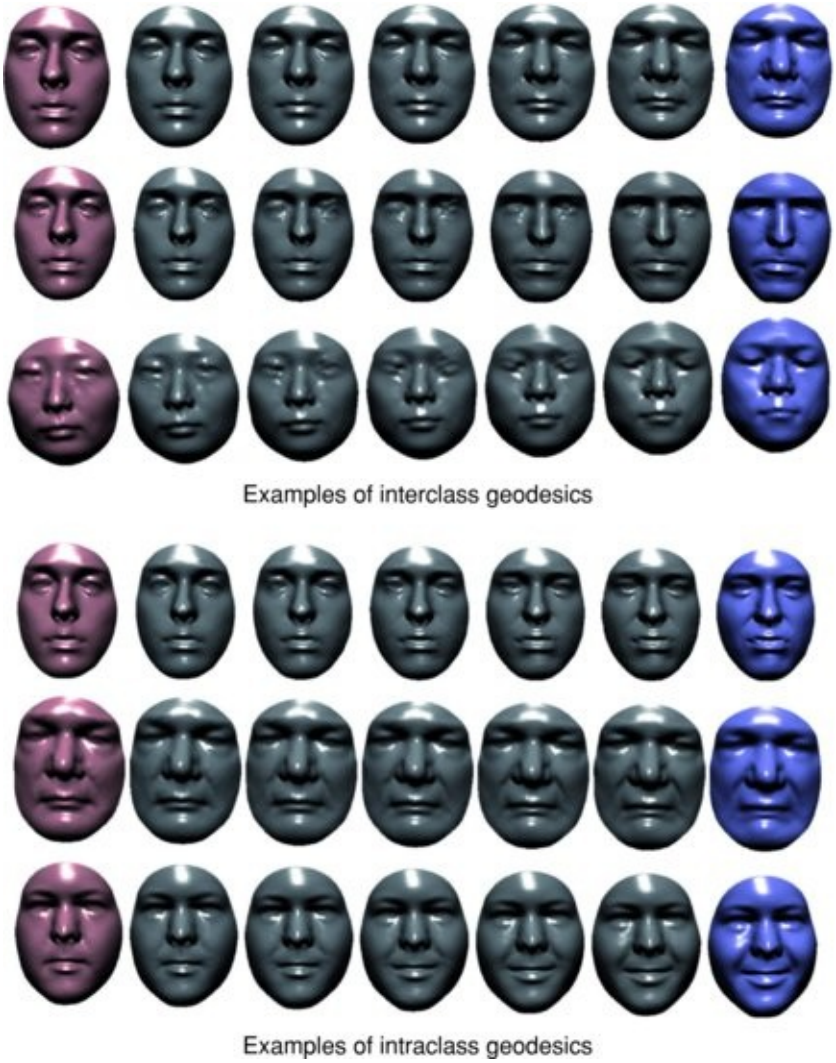
Figure 3.12 Faces comparison by pairwise curves comparisons



Here, q_α^i denotes the SRVF of the radial curve β_α^i on the *I*th, *i*=1, 2 facial surface, and d_s is the geodesic distance between the curves represented by q_α^i.

Secondly, since we have geodesic paths denoting optimal deformations between individual curves, we can combine these deformations to obtain full deformations between faces. In fact, these full deformations are geodesic paths between faces when represented as elements of $\mathcal{S}^{[0,\alpha_0]}$. Shown in Figure 3.13 are examples of such geodesic paths between source and target faces. The three top rows illustrate paths between different subjects called interclass geodesics, whereas the remaining rows illustrate paths between the same person under different expressions called intraclass geodesics.

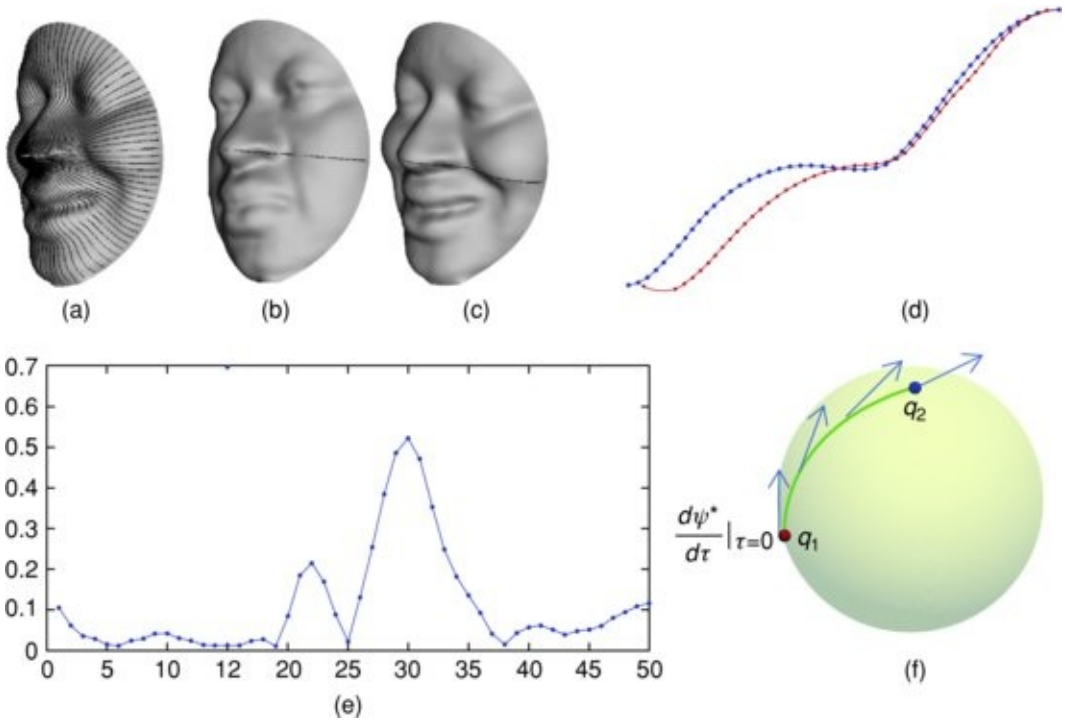
Figure 3.13 Examples of intra-and interclass geodesics in the shape space.



3.6 The Dense Scalar Field (DSF)

With the proposed representation, a facial surface is approximated by an indexed collection of radial curves β_α, where the index α denotes the angle formed by the curve with respect to a *reference* radial curve. In particular, the reference radial curve (i.e., the curve with α = 0) is chosen as oriented along the vertical axis, whereas the other radial curves are separated each other by a fixed angle and are ordered in a clockwise manner. As an example, Figure 3.14**a** shows the radial curves extracted for a sample face with happy expression. To extract the radial curves, the nose tip is accurately detected, and each face scan is rotated to the upright position so as to establish a direct correspondence between radial curves that have the same index in different facial scans (the preprocessing steps, including nose tip detection and pose normalization are discussed in more detail in Sect. 5.5.1). In Figure 3.14**b**,**c**, two radial curves at $\alpha = 90°$ in the neutral and happy scans of a same subject are shown. As shown in the plot Figure 3.14**d**, facial expressions can induce consistent variations in the shape of corresponding curves. These variations are not the same in strength from expression to expression and for different parts of the face. To effectively capture these variations, a dense scalar field is proposed, which relies on a Riemannian analysis of facial shapes.

Figure 3.14 The figure illustrates: (a) the extracted radial curves; (b)-(c) A radial curve on a neutral face, and the correspondent radial curve on the same face with happy expression, respectively; (d) the two radial curves are plotted together; (e) the values of the magnitude of $\frac{d\psi^*}{d\tau}|_{\tau=0}(k)$ computed between the curves in (d) are reported for each point *k* of the curves; (f) the parallel vector field across the geodesic between q_1 and q_2 in the space of curves $\mathcal{C}$



Each radial curve is represented on the manifold $\mathcal{C}$ by its SRVF. According to this, given the SRVFs q_1 and q_2 of two radial curves, the *geodesic path* ψ^* on the manifold $\mathcal{C}$ between q_1 and q_2 is a critical point of the following energy function:

$$E(\psi) = \frac{1}{2}\int < \dot{\psi}(\tau), \dot{\psi}(\tau) > d\tau, \quad (3.12)$$

where ψ denotes a path on the manifold $\mathcal{C}$ between q_1 and q_2, $\dot{\psi} \in T_\psi(\mathcal{C})$ is the tangent vector field on the curve $\psi \in \mathcal{C}$, and <.> denotes the $\mathbb{L}^2$ inner product on the tangent space.

Because elements of $\mathcal{C}$ have an unit $\mathbb{L}^2$ norm, $\mathcal{C}$ is a hypersphere in the Hilbert space $\mathbb{L}^2(I, \mathbb{R}^3)$. As a consequence, the geodesic path between any two points $q_1, q_2 \in \mathcal{C}$ is simply given by the minor arc of the great circle connecting them on this hypersphere, $\psi^* : [0,1] \to \mathcal{C}$. This is given by the following expression:

$$\psi^*(\tau) = \frac{1}{\sin(\theta)}(\sin((1-\tau)\theta)q_1 + \sin(\theta\tau)q_2), \quad (3.13)$$

where $\theta = d_{\mathcal{C}}(q_1, q_2) = \cos^{-1}(\langle q_1, q_2 \rangle)$. We point out that $\sin(\theta) = 0$, if the distance between the two curves is zero, in other words q_1=q_2. In this case, for each τ, $\psi^*(\tau) = q_1 = q_2$. The tangent vector field on this geodesic is then written as $\frac{d\psi^*}{d\tau} : [0,1] \to T_\psi(\mathcal{C})$, and is obtained by the following equation:

$$\frac{d\psi^*}{d\tau} = -\frac{\theta}{\sin(\theta)}(\cos((1-\tau)\theta)q_1 - \cos(\theta\tau)q_2). \quad (3.14)$$

Knowing that on geodesic path, the covariant derivative of its tangent vector field is equal to 0, $\frac{d\psi^*}{d\tau}$ is parallel along the geodesic ψ^*, and it can be represented with $\frac{d\psi^*}{d\tau}|_{\tau=0}$ without any loss of information. Accordingly, Equation 3.14 becomes

$$\frac{d\psi^*}{d\tau}|_{\tau=0} = \frac{\theta}{\sin(\theta)}(q_2 - \cos(\theta)q_1) \ (\theta \neq 0). \quad (3.15)$$

A graphical interpretation of this mathematical representation is shown in Figure 3.14. In Figure 3.14**a**, we show a sample face with the happy expression and all the extracted radial curves. In Figures 3.14**b** and 3.14**c** two corresponding radial curves (i.e., radial curves at the same angle α), respectively, on neutral and happy faces of the same person are highlighted. These curves are reported together in Figure 3.14**d**, where the amount of the deformation between them can be appreciated, although the two curves lie at the same angle α and belong to the same person. The amount of deformation between the two curves is calculated using Equation 3.15, and the plot of the magnitude of this vector at each point of the curve is reported in Figure 3.14**e** (i.e., 50 points are used to sample each of the two radial curves and reported in *x* axis, the magnitude of DSF is reported in *y* axis).

Finally, Figure 3.14**f** illustrates the idea to map the two radial curves on the hypersphere $\mathcal{C}$ in the Hilbert space through their SRVFs q_1 and q_2 and shows the geodesic path connecting these two points on the hypersphere. The tangent vectors of this geodesic path represent a vector field whose covariant derivative is zero. According to this, $\frac{d\psi^*}{d\tau}|_{\tau=0}$ becomes sufficient to represent this vector field, with the remaining vectors generatable by parallel transport of $\frac{d\psi^*}{d\tau}|_{\tau=0}$ along the geodesic ψ^*.

On the basis of the preceding representation, we define a DSF capable of capturing deformations between two corresponding radial curves β_α^1 and β_α^2 of two faces approximated by a collection of radial curves.

Definition 3.6.1 Dense Scalar Field (DSF) *Let* $x_\alpha(t) = ||\frac{d\psi_\alpha^*}{d\tau}|_{\tau=0}(t)||$, *which corresponds to the values of the magnitude computed between the curves* q_1^α *and* q_2^α *for each point t of the curves. Let t be the number of sampled points per curve and* $|\Lambda|$ *be the number of curves used per face. So, we define the function f by:*

$$f : \mathcal{C} \times \mathcal{C} \longrightarrow (\mathbb{R}^+)^T,$$
$$f(q_1^\alpha, q_2^\alpha) = (x_\alpha^1, \ldots, x_\alpha^k, \ldots, x_\alpha^T).$$

Assuming that $\{\beta_\alpha^1 | \alpha \in \Lambda\}$ *and* $\{\beta_\alpha^2 | \alpha \in \Lambda\}$ *be the collections of radial curves associated with the two faces* F_1 *and* F_2 *and let* $\{q_\alpha^1$ *and* $\{q_\alpha^2$ *be their SRVFS, the DSF vector is defined by:*

(3.16) $$\text{DSF}(F_1, F_2) = \left(f(q_1^0, q_2^0), \ldots, f(q_1^\alpha, q_2^\alpha), \ldots, f(q_1^{|\Lambda|}, q_2^{|\Lambda|})\right).$$

The dimension of the vector DSF vector is $|\Lambda| \times T$.

Algorithm 2 summarizes the proposed approach.

Algorithm 2 Computation of the dense scalar field

Input: Facial surfaces F_1 and F_2; T: number of points on a curve; α_0: angle between successive radial curves; $|\Lambda|$: number of curves per face.
Output: DSF(F_1, F_2): the Dense Scalar Field between the two faces.

$\alpha = 0$;
while $\alpha < |\Lambda|$ **do**
  for $i \leftarrow 1$ **to** 2 **do**
    Extract the curve β_α^i;
    Compute corresponding square-root velocity function $q_\alpha^i(t) \doteq \frac{\dot{\beta}_\alpha^i(t)}{\sqrt{\|\dot{\beta}_\alpha^i(t)\|}} \in \mathcal{C}$;
    $t = 1, 2 \ldots T$.
  end
  Compute θ the distance between q_α^1 and q_α^2 as: $\theta = d_{\mathcal{C}}(q_\alpha^1, q_\alpha^2) = \cos^{-1}(\langle q_\alpha^1 q_\alpha^2 \rangle)$
  Compute the deformation vector $\frac{d\psi^*}{d\tau}|_{\tau=0}$ using Equation 3.15 as:

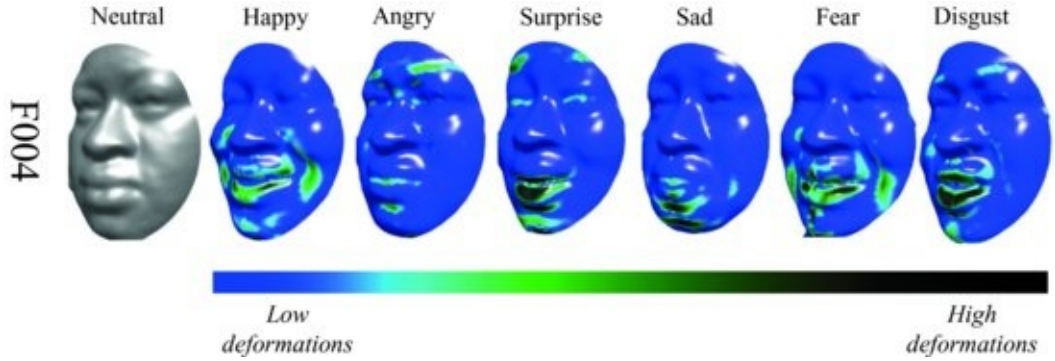
$$f(q_1^\alpha, q_2^\alpha) = (x_\alpha(1), x_\alpha(2), \ldots, x_\alpha(T)) \in \mathbb{R}_+^T;$$
$$x_\alpha(t) = \left| \frac{\theta}{\sin(\theta)} \left(q_\alpha^2 - \cos(\theta) q_\alpha^1\right) \right|, t = 1, 2 \ldots T;$$

end
Compute the local deformation DSF(F_1, F_2) as the magnitude of $\frac{d\psi^*}{d\tau}|_{\tau=0}(k)$;
DSF(F_1, F_2) = $(f(q_1^0, q_2^0), \ldots, f(q_1^\alpha, q_2^\alpha), \ldots, f(q_1^{|\Lambda|}, q_2^{|\Lambda|}))$

In Figure 3.15, an example of the deformation field computed on the 3D frames of a sample subject is reported. In particular, a neutral mesh is reported on the left, and the vector field is computed between the 3D neutral face and the 3D apex frames of each expression of the same subject. The values of the vector field needed to be applied on the neutral face to convey the six different universal expressions reported using a color scale. In particular, colors from green to black represent the highest deformations, whereas blue represents the lower values of the vector field. It can be observed, as the regions with high deformation lie in different parts of the face for different expressions. For

example, as intuitively expected, the corners of the mouth and the cheeks are strongly deformed for happiness expression, whereas the eyebrows are strongly deformed for the angry expression.

Figure 3.15 Deformation maps computed between the neutral face of a sample subject and the apex frame of the six prototypical expressions sequences of the same subject



3.7 Statistical Shape Analysis

As mentioned earlier, an important advantage of our Riemannian approach over many past papers is its ability to compute summary statistics of a set of faces.

3.7.1 Statistics on Manifolds: Karcher Mean

What are the challenges in applying classical statistics if the underlying domain is nonlinear? Take the case of the simplest statistic, the sample mean, for a sample set $\{x_1, x_2, \ldots, x_k\}$ on $\mathbb{R}^n$:

$$\bar{x_k} = \frac{1}{k}\sum_{i=1}^{k} x_i, x_i \in \mathbb{R}^n \quad (3.17)$$

Now what if underlying space is not $\mathbb{R}^n$ but nonlinear manifold? In this situation, the summation in Equation 3.17 is not valid operation, and the equation is not useful. So, how do we define the sample mean in this case?

For example, one can use the notion of Karcher mean (Karcher, 1977) to define an average face that can serve as a representative face of a group of faces.

We recall that $\mathcal{S} \doteq \mathcal{C}/(SO(3) \times \Gamma)$ is the *shape space* of open curves in $\mathbb{R}^3$. As described in Section 3.5.4, the distance between two facial surfaces is given by

$$\mathrm{d}_s : \mathcal{S}^{[0,\alpha_0]} \times \mathcal{S}^{[0,\alpha_0]} \to \mathbb{R}_{\geq 0}$$

$$\mathrm{d}_s(S^1, S^2) = \frac{1}{\alpha_0}\sum_{\alpha=1}^{\alpha_0} \mathrm{d}_s([q_\alpha^1], [q_\alpha^2]).$$

Here, q_α^i denotes the SRVF of the radial curve β_α^i on the *I*th, *i*=1, 2 facial surface.

To calculate the Karcher mean of facial surfaces $\{S^1, \ldots, S^n\}$ in $\mathcal{S}^{[0,\alpha_0]}$, we define the variance function as

$$\mathcal{V} : \mathcal{S}^{[0,\alpha_0]} \to \mathbb{R}, \mathcal{V}(S) = \sum_{i=1}^{n} d_s(S, S^i)^2. \quad (3.18)$$

The Karcher mean is then defined by

$$\overline{S} = \arg \min_{\mu \in \mathcal{S}^{[0,\alpha_0]}} \mathcal{V}(\mu). \quad (3.19)$$

To calculate a Karcher mean of facial surfaces $\{S^1, \ldots, S^n\}$ in $\mathcal{S}^n$, we define an objective function: $\mathcal{V} : \mathcal{S}^n \to \mathbb{R}, \mathcal{V}(S) = \sum_{i=1}^{k} d_S(S, S^i)^2$. The Karcher mean is then defined by $\overline{S} = \arg\min_{S \in \mathcal{S}^n} \mathcal{V}(S)$. This minimizer may not be unique and, in practice, any one of those solutions may be picked as the mean face. This mean has a nice geometrical interpretation: $\overline{S}$ is an element of $\mathcal{S}^n$ that has the smallest total (squared) deformation from all given facial surfaces $\{S^1, \ldots, S^n\}$.

We present a commonly used algorithm for finding Karcher mean for a given set of facial surfaces. This approach, presented in Algorithm 1, uses the gradient of $\mathcal{V}$ to iteratively update the current mean μ. An iterative algorithm for computing the sample Karcher mean is defined by Algorithm 3.

Algorithm 3 Karcher mean algorithm

Gradient search

Set $k = 0$. Choose some time increment $\epsilon \leq \frac{1}{n}$. Choose a point $\mu_0 \in \mathcal{S}^{[0,\alpha_0]}$ as an initial guess of the mean. (For example, one could just take $\mu_0 = S^1$.)

1. For each $i = 1, \ldots, n$ choose the tangent vector $f_i \in T_{\mu_k}(\mathcal{S}^{[0,\alpha_0]})$, which is tangent to the geodesic from μ_k to S^i. The vector $g = \sum_{i=1}^{i=n} f_i$ is proportional to the gradient at μ_k of the function $\mathcal{V}$.
2. Flow for time ϵ along the geodesic that starts at μ_k and has velocity vector g. Call the point where you end up μ_{k+1}.
3. Set $k = k + 1$ and go to step 1.

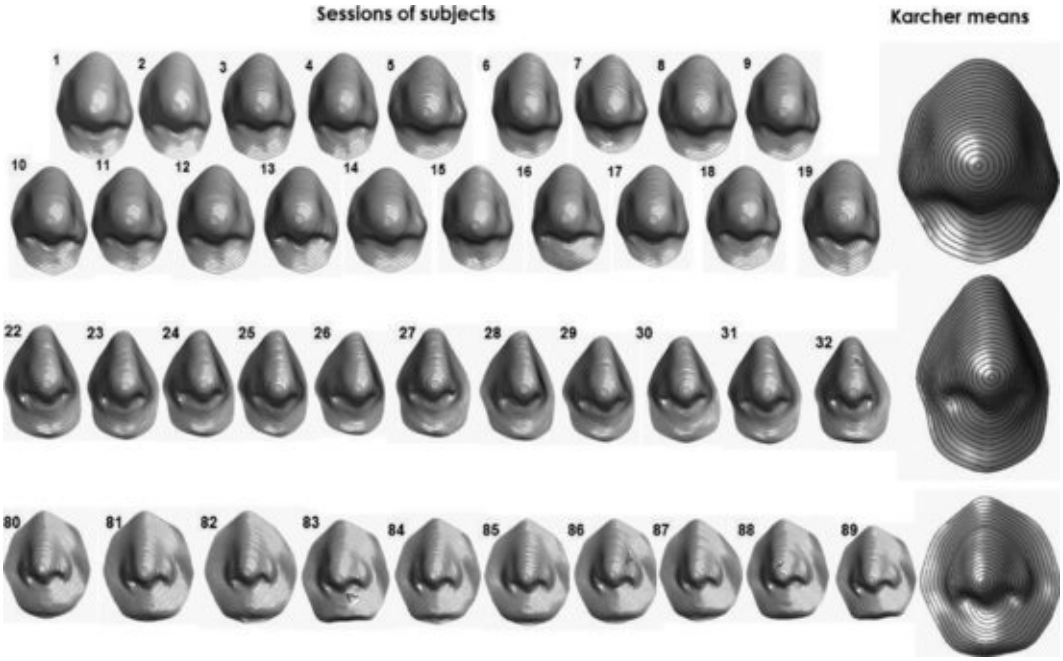
Since this is a gradient approach, it only ensures a local minimizer of the variance function $\mathcal{V}$.

An example of a Karcher mean face for eight faces belonging to different people is shown in Figure 3.16. This figure illustrates the mean face of faces belonging to different persons. Several examples of using the Karcher mean to compute average noses are shown in Figure 3.17.

Figure 3.16 An example of Karcher mean of faces, the face at the right is the

karcher mean of the eight faces in the left. Copyright © 1969, IEEE

Figure 3.17 Examples of nasal shapes and their means. Copyright © 2009, IEEE

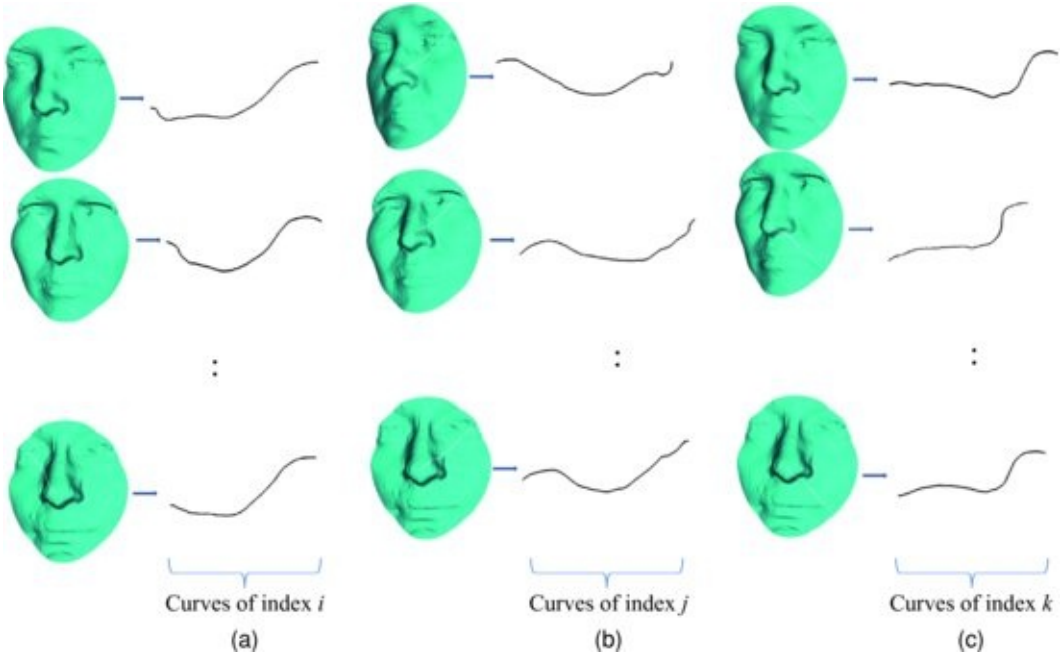


3.7.2 Learning Statistical Models in Shape Space

We have a way of estimating the Karcher mean of a sample set on a nonlinear manifold. What about principal component analysis? The core of this problem, in our representation of facial surfaces by curves, is to take a partial facial curve and predict its completion. The sources of information available for this prediction are (1) the current (partially observed) curve and (2) several (complete) training curves at the same angle that are extracted from full faces. The basic idea is to develop a sparse model for the curve from the training curves and use that to complete the observed curve. To keep the model simple,

we use the PCA of the training data (in an appropriate vector space) to form an orthogonal basis representing training shapes. Then, this basis is used to estimate coefficients of the observed curve, and the coefficients help us to reconstruct the full curve. Because the shape space of curve $\mathcal{S}$ is a nonlinear space, we use the tangent space $T_\mu(\mathcal{S})$, where μ is the mean of the training shapes, to perform PCA. Some examples of curves used during training step are shown in Figure 3.18.

Figure 3.18 Radial curves collections for training step



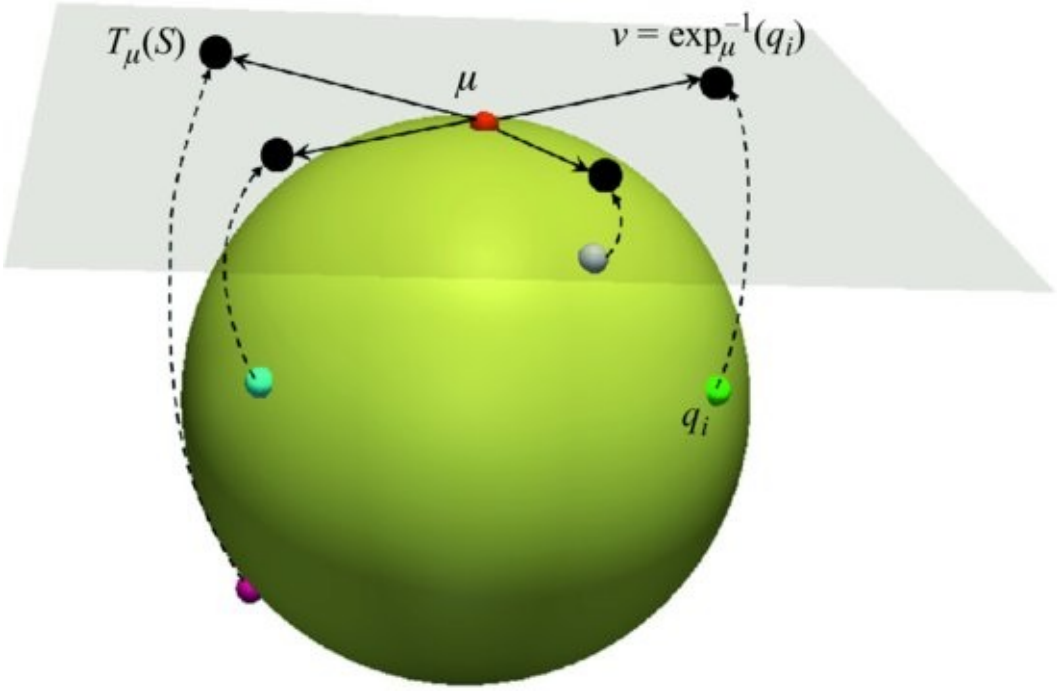
Let α denote the angular index of the observed curve, and let $q_\alpha^1, q_\alpha^2, \ldots, q_\alpha^k$ be the SRVFs of the curves taken from training faces at that angle. As described earlier, we can compute the sample Karcher mean of their shapes $\{[q_\alpha^i] \in \mathcal{S}\}$, denoted by μ_α. Then, using the geometry of $\mathcal{S}$ we can map these training shapes in the tangent space using the inverse exponential map, that is, obtain $v_{i,\alpha} = \exp_{\mu_\alpha}^{-1}(q_\alpha^i)$, where

$$\exp_{q_1}^{-1}(q_2) = \frac{\theta}{\sin(\theta)}(q_2^* - \cos(\theta)q_1), \quad \theta = \cos^{-1}(\langle q_1, q_2^* \rangle),$$

and where q_2^* is the optimal rotation and reparametrization of q_2 to be aligned with q_1, as discussed earlier. A PCA of the tangent vectors $\{v_i\}$ leads to the principal basis vectors $u_{1,\alpha}$, $u_{2,\alpha}, \ldots, u_{J,\alpha}$, where J represents the number of significant basis elements.

Figure 3.19 illustrates a mapping of elements of the shape space onto the tangent space on the mean shape μ.

Figure 3.19 Illustration of mapping shapes onto the tangent space of μ, $T_\mu(\mathcal{S})$



We summarize the calculation of eigencurves in Algorithm 4.

Algorithm 4 Eigencurves computation

Input: Training faces (without missing data) $G = \{y_i\}_{1 \le i \le N_G}$
Output: $B = \{v_{k_j}\}$: eigenvectors

K : number of curves in each face;
for $k \leftarrow 1$ **to** K **do**
  μ_k = intrinsic mean of $SRVF(y_{k_i})$ (Karcher mean).
  for $i \leftarrow 1$ **to** N_G **do**
    $\beta_{k_i} = \exp_{\mu_k}^{-1}(SRVF(y_{k_i}))$
  end
  $S_k = \sum_{i=1}^{N_G} \beta_{k_i} \beta_{k_i}^T$
  $\{v_{k_j}\}$ = eigenvectors of S_k
end

3.8 Applications of Statistical Shape Analysis

3.8.1 3D Face Restoration

Now returning to the problem of completing a partially occluded curve, let us assume that it is observed for parameter value t in $[0, \tau] \subset [0, 1]$. In other words, the SRVF of this curve $q(t)$ is known for $t \in [0, \tau]$ and unknown for $t > \tau$. Then,

we can estimate the coefficients of q under the chosen basis according to $c_{j,\alpha} = \langle q, u_{j,\alpha} \rangle \approx \int_0^\tau \langle q(t), u_{j,\alpha}(t) \rangle \mathrm{d}t$, and estimate the SRVF of the full curve according to

$$\hat{q}_\alpha(t) = \sum_{j=1}^{J} c_{j,\alpha} u_{j,\alpha}(t), \quad t \in [0, 1].$$

In our case, the exponential mapping exp maps a vector $v \in T_\mu(\mathcal{S})$ to a point of $\mathcal{S}$. In other words, to reach the point $\exp_\mu(v)$, one starts at μ, and then moves for time 1 along the unique constant speed geodesic whose velocity vector at q is v. The inverse of an exponential map takes a point q_i on the manifold $\mathcal{S}$ and maps it to an element (or multiple elements) of the tangent space $T_\mu(\mathcal{S})$. A vector v is said to be the inverse exponential map of a point $q_i \in \mathcal{S}$, at the point $\mu \in \mathcal{S}$, if $\exp_\mu(v) = q_i$. It is denoted by $v = \exp_\mu^{-1}(q_i)$.

As a result, the exponential map, $\exp : T_\mu(\mathcal{S}) \to \mathcal{S}$, has also a simple expression. Let v be a vector $v \in T\mu(\mathcal{S})$, the exponential mapping of v gives an element of the manifold $\mathcal{S}$ as:

$$\exp_\mu(v) = \cos(\|v\|)\mu + \sin(\|v\|)\frac{v}{\|v\|},$$

The exponential map is a bijection if we restrict $\|v\|$ so that $\|v\| \in [0, \pi)$. For a point $q_i \in \mathcal{S}$, such that ($q_i \neq \mu$), the inverse exponential map $v = \exp_\mu^{-1}(q_i)$ projects q_i on the tangent space of μ as u:

$$u = \frac{\theta}{\sin(\theta)}(q_i - \cos(\theta)\mu),$$

where $\theta = \cos^{-1}(< \mu, q >)$. The result of this projection is elements on the tangent space $\{v_1, v_2, \ldots, v_N\}$. It is possible to perform traditional operations and work as in an Euclidean space using the projected elements. We summarize this procedure in Algorithm 5.

Algorithm 5 Projection of facial curves on eigencurves space

Input: P: face without missing data, eigenvectors $B_{k_j}, k = 1 \ldots K$
Output: *P_restored*

K : number of curves in each face;
for $k \leftarrow 1$ **to** K **do**
  $q_k = SRVF(c_k)$;
  $v_k = \exp_{\mu_k}^{-1}(q_k)$;
  $q_k = \exp_{\mu_k}(\sum_{j=1}^{N_{B_k}} \langle c_k v_{k_j} \rangle v_{k_j})$;
  $c_k = SRVF(q_k)$;
end

Figure 3.20 illustrates the restoration of three different curves. As illustrated respectively in Figure 3.20**a**–**c**, the restoration of curve generates the same curve. This validates our model, which represents curves in a new basis while keeping 90% of the information. The same idea is followed to recover 3D faces: Curves of different index are restored. The collection of the restored curves represents the 3D face. Notice that the number of eigenvectors differs from one level to another. Figure 3.21 illustrates the face projection. The first row illustrates the original face (to the left) and the restored one (to the right). In the second row, we see in the middle the original and restored face together. The left and right images illustrate, respectively, the signed and absolute deviation between the original and the restored face. The average absolute deviation is 0.21 mm with 0.28 mm as standard deviation. These observations demonstrate on the closeness of the restored face to the original one. Therefore, we propose next to restore curves with missing data and keep complete ones.

Figure 3.20 Restoration of curves of different index

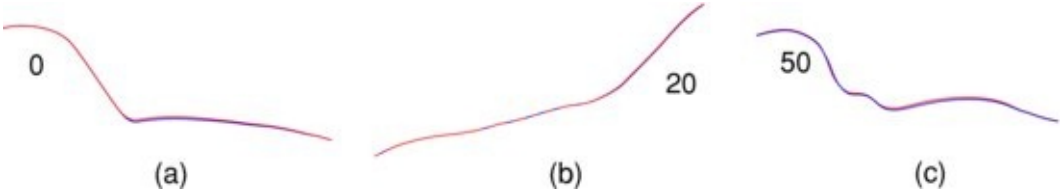
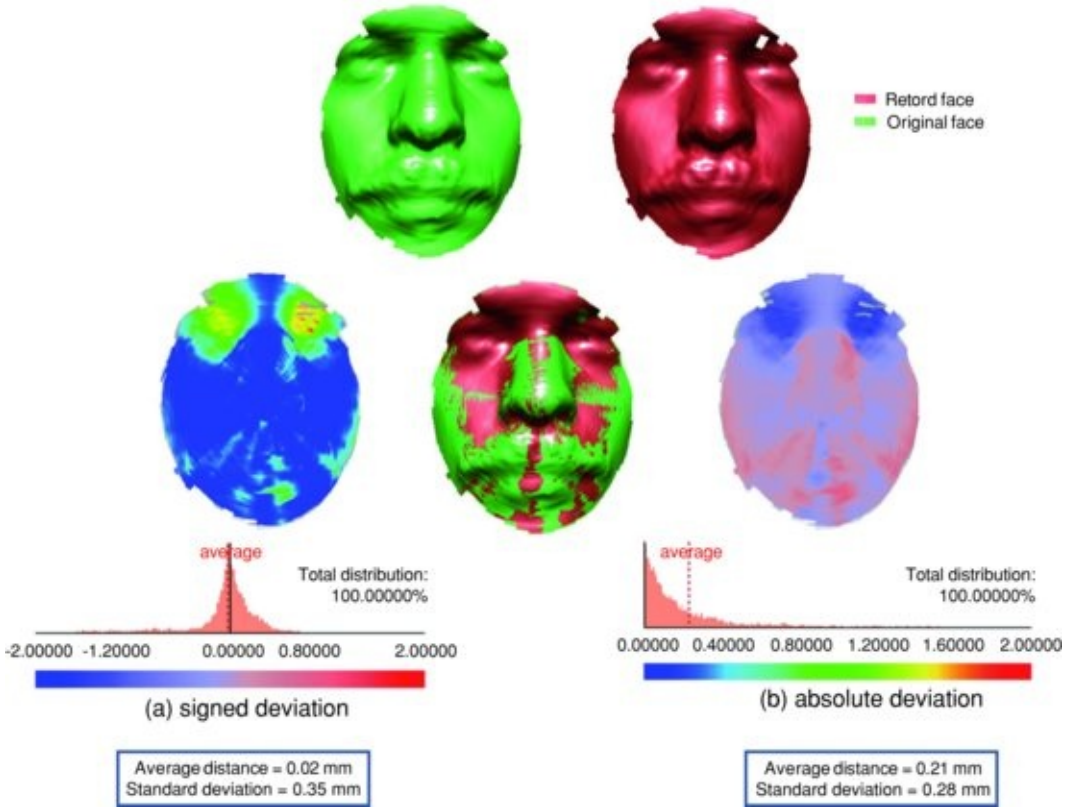


Figure 3.21 Restoration of full face: the absolute and signed deviations



In order to evaluate the face recovery procedure, we compare the restored facial surface (shown in the top row of Figure 3.22) with the complete neutral face of that class, as shown in Figure 3.22. Small values of both absolute deviation and signed deviation between the restored face and the corresponding face in the gallery demonstrate the success of the restoration process.

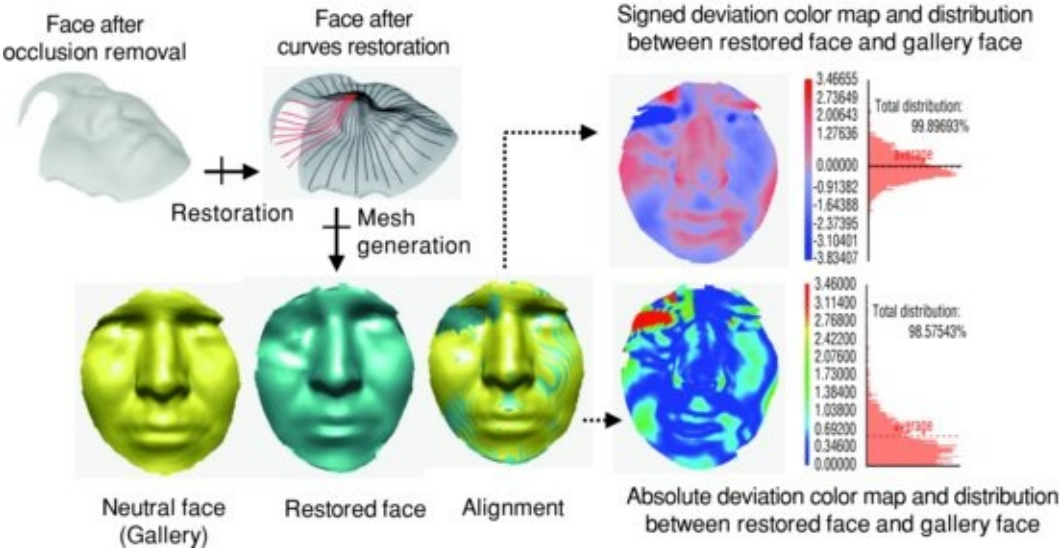
Algorithm 6 Curves restoration

Input: P: face with missing data, eigenvectors $B_{k_j}, k = 1 \ldots K$
Output: $P\_restored$

K : number of curves in each face; n: number of points in a complete curve.

for $k \leftarrow 1$ **to** K **do**
  if *quality*$(c_k = 0)$ **then**
    for $i \leftarrow 1$ **to** n **do**
      $flag_k[i] = 1$;
      if *i corresponds to missing part* **then**
        $flag[i] = 0$;
      end
    end
    $q_k = SRVF(c_k)$;
    $v_k = \exp^{-1}_{\mu_k}(q_k)$; if $flag[i] = 1$
    $q_k = \exp_{\mu_k}(\sum_{j=1}^{N_{B_k}} \langle c_k v_{k_j} \rangle v_{k_j})$; if $flag[i] = 1$
  end
end

Figure 3.22 Illustration of a face with missing data (after occlusion removal) and its restoration. The deviation between the restored face and the corresponding neutral face is also illustrated. Copyright © 1969, IEEE



3.8.2 Hierarchical Organization of Facial Shapes

One of the main goals for studying shapes of the nose/face region is to conduct biometric identification where query is often compared to a set of gallery shapes. This comparison can be made more efficient if we organize the gallery elements in the form of a hierarchical database, that is, a tree, where the comparisons are

performed only at the nodes. To construct such a shape tree, we need to be able to cluster similar shapes, and that is the problem we study next.

Let $\mathcal{S}^{[0,\alpha_0]}$ denotes the set of facial shapes. Consider the problem of clustering *n* shapes (in $\mathcal{S}^{[0,\alpha_0]}$) into *k* clusters. A general approach is to form clusters in such a way that they minimize total "within-cluster" variance. Let a configuration *C* consists of clusters denoted by $C_1, C_2, \ldots, C_k$, and let μ_i s be the mean shapes in C_is and n_is be the sizes of C_is. There are several cost functions used for clustering, for example, the sum of traces of covariances within clusters. However, the computation of means μ_i s of large shape clusters, and therefore their variances, is computationally expensive, especially when they are updated at every iteration. As a solution, a variation called *pairwise clustering* is used. (Hofmann and Buhmann, 1997), where the variance of a cluster is replaced by a scaled sum of distances (squared) between its elements as follows:

$$Q(C) = \sum_{i=1}^{k} \frac{2}{n_i} \left(\sum_{S^a \in C_i} \sum_{b<a, \theta_b \in C_i} d_s(S^a, S^b)^2 \right). \quad (3.20)$$

We seek configurations that minimize *q*, i.e., $C^* = \operatorname{argmin} Q(C)$. Notice that the metric used is the arithmetic mean d_a. We will minimize the clustering cost using a Markov chain search process on the configuration space. The basic idea is to start with a configuration of *k* clusters and to reduce *q* by rearranging shapes amongst the clusters. The rearrangement is performed in a stochastic fashion using two kinds of moves. These moves are performed with probability proportional to the negative exponential of the *q*-value of the resulting configuration. The two types of moves are as follows: (1) **Move a shape**: Here we select a shape randomly and reassign it to another cluster. Let $Q^{(i)}_j$ be the clustering cost when a shape θ_j is reassigned to the cluster C_i keeping all other clusters fixed. If θ_j is not a singleton, that is, not the only element in its cluster, then the transfer of θ_j to cluster C_i is performed with probability: $P_M(j, i; T) = \frac{\exp(-Q_j^{(i)}/T)}{\sum_{i=1}^{k} \exp(-Q_j^{(i)}/T)} \quad i = 1, 2, \ldots, k$. Here *t* plays a role similar to temperature in simulated annealing. If θ_j is a singleton, then moving it is not allowed in order to fix the number of clusters at *k*. (2). **Swap two shapes**: Here we select two shapes randomly from two different clusters and swap them. Let $Q^{(1)}$ and $Q^{(2)}$ be the *q*-values of the original configuration (before swapping) and the new configuration (after swapping), respectively. Then, swapping is performed with probability: $P_S(T) = \frac{\exp(-Q^{(2)}/T)}{\sum_{i=1}^{2} \exp(-Q^{(i)}/T)}$.

To seek global optimization, we have adopted a simulated annealing approach. Although simulated annealing and the random nature of the search help in avoiding local minima, the convergence to a global minimum is difficult to establish. Algorithm 7 illustrates the steps of the clustering algorithm.

It is important to note that once the pairwise distances are computed, they are not computed again in the iterations. Secondly, unlike *k*-mean clustering, the mean shapes are never calculated in this clustering. The algorithms for computing Karcher mean and clustering can be applied repeatedly for organizing a large database of human faces into a hierarchy that allows efficient searches during biometrics. Among the 466 faces used as gallery in the FRGCv2 experimentation, we propose to organize the 410 faces that have corresponding face in the probe.

Algorithm 7 Statistical shape clustering

For n shapes and k clusters, initialize by randomly distributing n shapes among k clusters. Set a high initial temperature T.

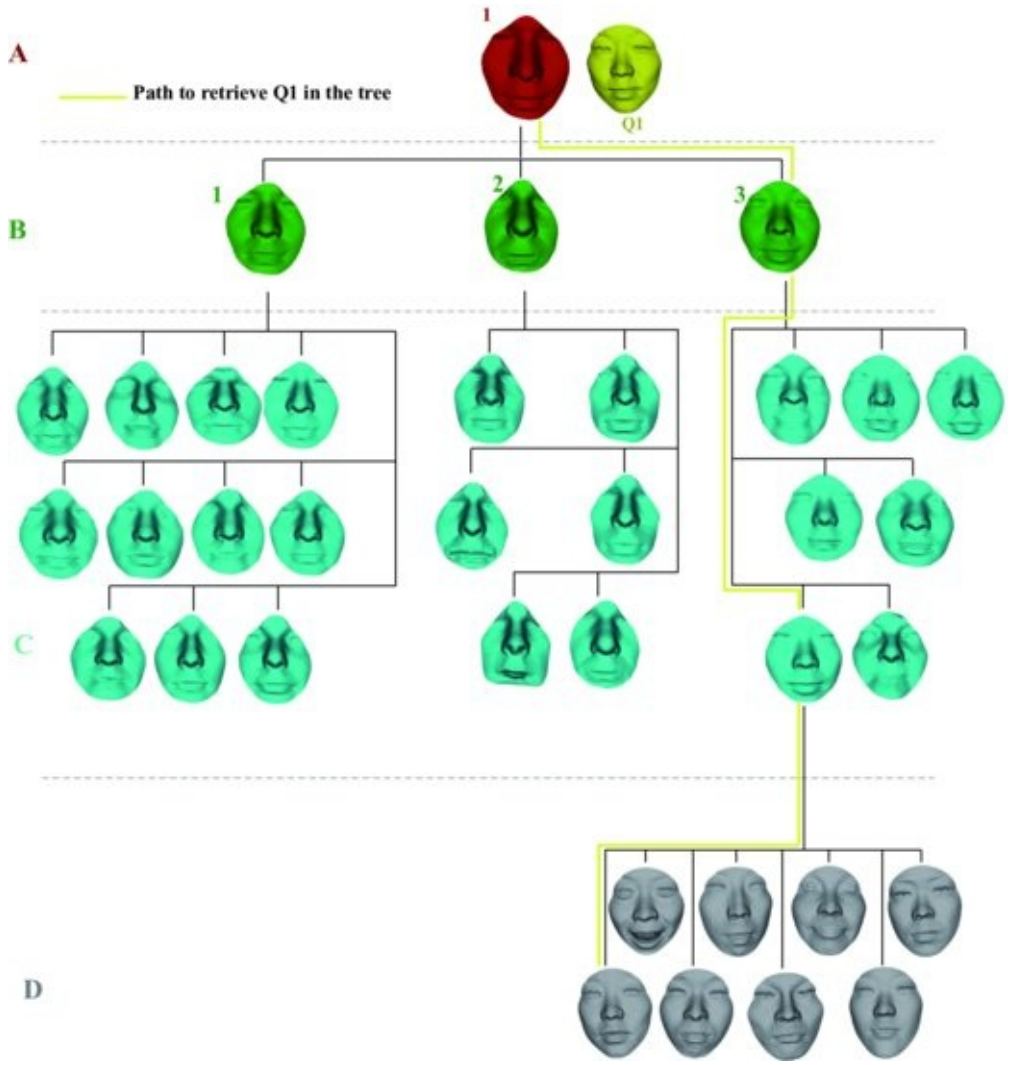
1. Compute pairwise geodesic distances between all n shapes. This requires $n(n-1)/2$ geodesic computations.
2. With equal probabilities pick one of the two moves:

- **Move a shape**: Pick a shape θ_j randomly. If it is not a singleton in its cluster, then compute $Q_j^{(i)}$ for all $i = 1, 2, \ldots, k$. Compute the probability $P_M(j, i; T)$ for all $i = 1, \ldots, k$ and re-assign θ_j to a cluster chosen according to the probability P_M.
- **Swap two shapes**: Select two clusters randomly, and select a shape from each. Compute the probability $P_S(T)$ and swap the two shapes according to that probability.

3. Update the temperature using $T = T/\beta$ and return to Step 2. We have used $\beta = 1.0001$.

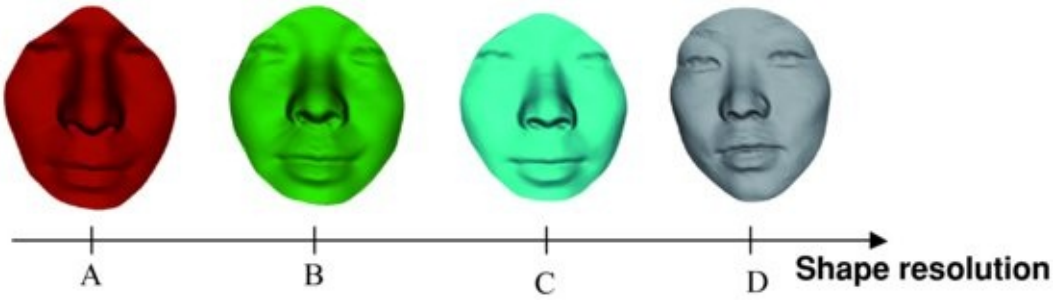
Figure 3.23 shows a hierarchical organization of these shapes all the way up to the top. At the bottom level (level D in the figure), these 410 are gathered into 29 clusters. Computing the means of each of these clusters, we obtain faces that are to be clustered at the next level (level C in the Figure 3.23). Repeating the clustering on the mean faces at level C, we obtain the next level (level B) containing 3 mean faces representing the clusters of the previous level. In level A, we just calculate the mean of the three mean faces in level B. This face represents the coarsest face along the tree.

Figure 3.23 The result of hierarchical organization of gallery faces from FRGCv2 data set and an example of path parsed by a query across the tree



If we follow a path from top to bottom of the tree, we can see the shapes getting more detailed structurally and leading up to individual faces, as illustrated in Figure 3.24.

Figure 3.24 Path from top to bottom in the tree show increasing shape resolution



Another example, we start with approximately 500 nose scans corresponding to 50 distinct subjects. These noses form the bottom layer of the hierarchy,

called level E in Figure 3.25. Then, we compute Karcher mean shapes for each person to obtain shapes at level D. These shapes are further clustered together and a Karcher mean is computed for each cluster. These mean shapes form the level C of the hierarchy. Repeating this idea a few times, we reach the top of the tree. If we follow a path from top to bottom of the tree, we can see the shapes getting more detailed structurally and leading up to individual faces, as illustrated in Figure 3.26.

Figure 3.25 The tree resulting on hierarchical clustering. Copyright © 2009, IEEE

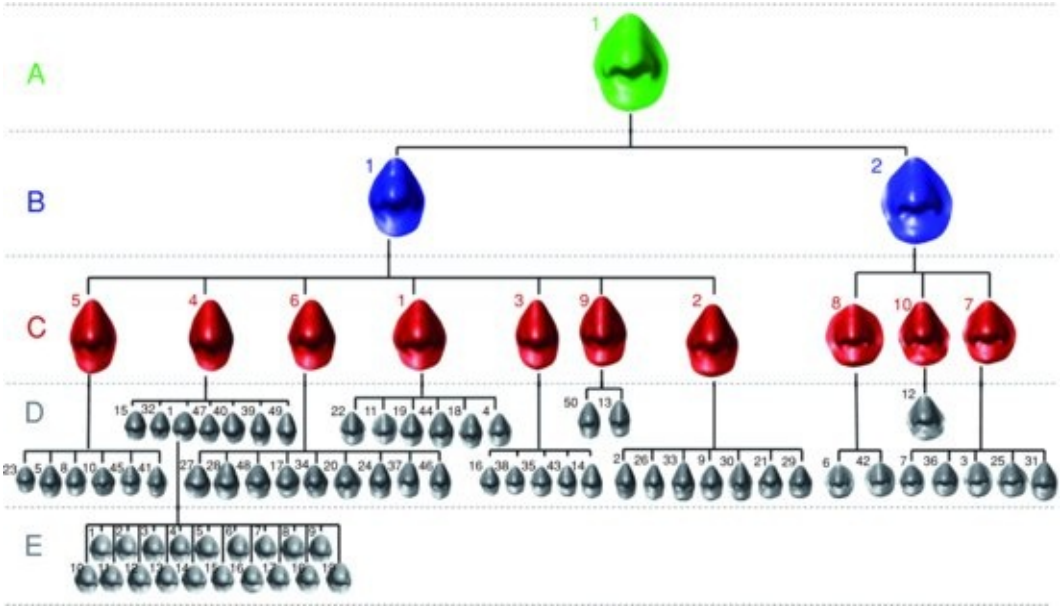
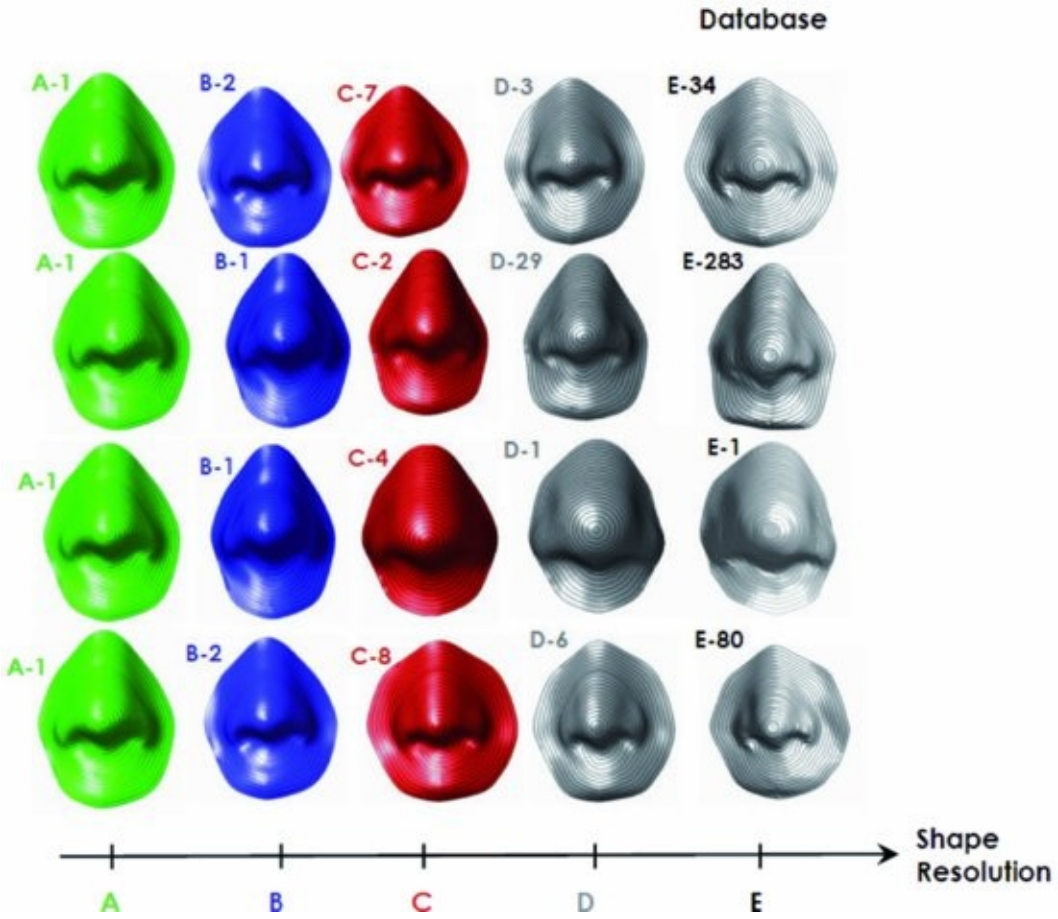


Figure 3.26 Paths from top to buttom in the tree show increasing shape resolutions. Copyright © 2009, IEEE



3.9 The Iso-geodesic Stripes

One main difficulty in extracting *facial curves* from the surface of 3D face scans is related to the presence of noise. In fact, in addition to the intrinsic noise components as a result of acquisition devices and surface characteristics that can be smoothed through some preprocessing, other sources of noise can contribute to alter the extraction of facial curves from the acquired data. For example, in the approach proposed by Samir *et al.* (2006), the *iso-depth curves* extracted from the face are greatly influenced by the misalignment of 3D scans with respect to a common 3D cartesian reference system, whereas in Samir *et al.* (2009) the *iso-geodesic curves* are sensitive to the noise in computing the geodesic distances between points of the surface and a reference point of the face. As a consequence, the shape of these facial curves can be largely influenced by the noise, and the effect of these variations can be amplified by the fact that the shape of facial curves is used to match each other to compute the similarity between faces.

A possible way to smooth the effect of the noise without losing the

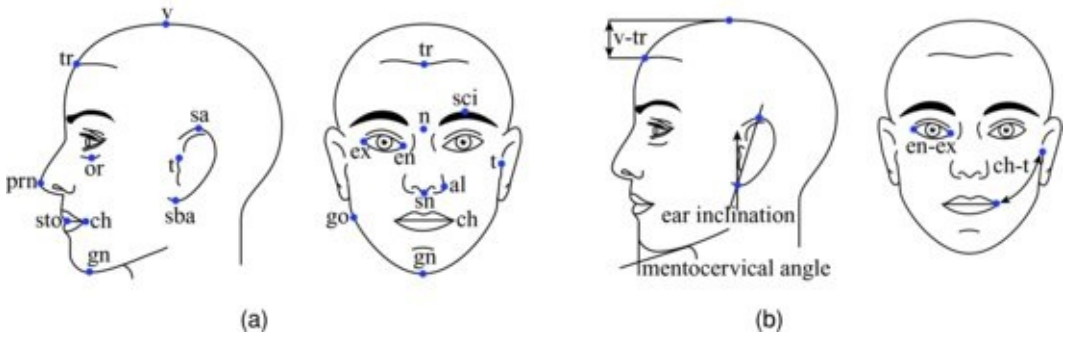
effectiveness of representations based on facial curves is to consider aggregations of facial curves instead of individual ones. In practice, this corresponds to extend *facial curves* to *facial surfaces*. In fact, the use of extended surfaces of the face should permit the punctual influence of noisy data to be reduced, thus making the representation extracted from the facial surfaces more descriptive and reliable. Following this intuition, the approaches in Samir *et al.* (2006) and Samir *et al.* (2009) can be extended to the case of *iso-depth surfaces* and *iso-geodesic surfaces*, respectively. In particular, the idea to describe the face using iso-geodesic surfaces has been originally proposed in Berretti *et al.* (2006), and then developed in Berretti *et al.* (2010), and Berretti *et al.* (2012).

In the approach of Berretti *et al.* (2010), the structural information of a face scan is captured through the 3D shape and relative arrangement of *iso-geodesic stripes* identified on the 3D surface. Iso-geodesic stripes are defined by computing, for every surface point, the normalized geodesic distance between the point and a reference point located at the *nose tip* of the face. Normalized values of the geodesic distance are obtained dividing the geodesic distance by the Euclidean *eye-to-nose* distance, that is the sum of the distances between the nose tip and the two points located at the inner commissure of the left and right eye fissure, and the distance between the two points at the inner eyes. The algorithm reported in Mian *et al.* (2007) is used for the identification of the nose tip and of the two inner eye points. This normalization guarantees invariance of the distance values with respect to scaling of the face scan. Furthermore, since the Euclidean eye-to-nose distance is invariant to face expressions, this normalization factor does not bias values of the distance under expression changes.

The fact that geodesic and Euclidean distances of the face are capable to characterize individual morphological traits of the face is strongly supported by studies of face anthropometry conducted by Farkas (Farkas, 1994). These studies have evidenced that relevant facial information is conveyed by measuring the Euclidean and geodesic distances, and the angles between 47 fiducial points. Figure 3.27**a**,**b**, illustrate some of the fiducial points and facial measurements.

Figure 3.27 (a) Some of the fiducial points proposed by Farkas: *alare* (al); *cheilion* (ch); *endocanthion* (en); *exocanthion* (ex); *gnathion* (gn); *nasion* (n); *orbitale* (or); *pronasale* (prn); *subnasale* (sn); *tragion* (t); *trichion* (tr); *vertex* (v). (b) Some facial measurements (Farkas, 1994): geodesic distance (ch-t);

Euclidean distances (v-tr and en-ex); angular measures (mentocervical angle and ear inclination)



Stability of facial fiducial points and measurements under changes following facial expressions is a fundamental issue for recognition. It has been demonstrated that geodesic distances are almost preserved under many expressions (Bronstein et al., 2005). Experimental evidence of this fact has been reported in Mpiperis *et al.* (2006), where the maximum change of 5% is measured for the geodesic distances computed between the nose and the cheek of the same subject under different expressions. Similar results are reported in Bronstein *et al.* (2007), where the average standard deviation of the absolute distance error due to facial expressions was measured in 5.89 mm and 12.03 mm, respectively, for the geodesic and Euclidean distance. Moreover, the pronasale (i.e., the nose tip) and the left and right endocanthion (i.e., the points at the inner commissure of the left and right eye fissure) have been verified to be stable with respect to face variations (Bronstein et al., 2005; Chang et al., 2005).

3.9.1 Extraction of Facial Stripes

In the proposed approach, computation of the geodesic distance on the piecewise planar mesh is accomplished through the Dijkstra's algorithm (Cormen et al., 2001), and approximates the actual geodesic distance between two surface points with the length of the shortest piecewise linear path on mesh edges. In particular, considering a mesh as a graph $G=(V, E)$ with the edge weights $w(e)$, $w(e)>0$, for each edge $e \in E$, the Dijkstra's algorithm solves the problem to find the *shortest path* from a *source vertex* $v_s \in V$ to a *target vertex* $v_t \in V$. In our specific case, the weight $w(e_{ij})$ of an edge $e_{ij}=(v_i, v_j)$ connecting vertices v_i and v_j, is given by the linear length of the edge itself, that is, $w(e_{ij})=|v_i-v_j|$. It is worth noting that the computation of geodesic distances on the mesh can be affected by the regularity of the mesh. In fact, since the Djikstra's algorithm approximates the actual geodesic distances through edge lengths, a nontriangular mesh (i.e., a mesh

composed of general polygons) or a nonregular mesh (i.e., a mesh composed of triangles of different sizes) makes the estimate less accurate. So, a mesh should be preprocessed to triangularize and regularize its polygons, thus making sufficiently accurate the computation of the geodesic distance using the Dijkstra's approximation (Antini et al., 2005).

Computer Implementation

A pseudo-code description of the Dijkstra's algorithm is reported in Figure 3.28. In this simple implementation, vertices of the set *v* are stored in an ordinary *linked list* (or array), and extract minimum from *v* is simply a linear search through all vertices in *v*. In this case, it can be easily shown that if $|V|$ and $|E|$ are the number of vertices and edges in *v* and *E*, respectively, the algorithm runs with a worst case time complexity of $O(|V|^2+|E|)=O(|V^2|)$. For *sparse graphs*, that is, graphs with far fewer than $O(|V|^2)$ edges, Dijkstra's algorithm can be implemented more efficiently by storing the graph in the form of *adjacency lists* and using a *binary heap* as a priority queue to implement extracting minimum efficiently. With a binary heap, the algorithm requires $O((|E|+|V|)\log |V|)$ time, which is dominated by $O(|E|\log |V|)$, assuming the graph is connected. Since, meshes representing 3D faces are sparse and connected, this latter optimized implementation of the algorithm can be used in our case, thus reducing the overall time complexity in computing the geodesic distances to the nose tip. A simplified description of the algorithm using a binary tree *v*-TREE, where the vertices are sorted in ascendant order of distance attribute from the source vertex is reported in Figure 3.29.

Figure 3.28 A pseudo-code description of Dijkstra's shortest path algorithm. This algorithm actually only computes the *length* of the shortest path rather than the shortest path itself

Algorithm: The shortest path

```
struct vertex {
   ...
   int distance;
};

void dijkstra_shortest_path( set of struct vertex V, struct vertex v_s, struct vertex v_t )
/*************************************************************************
receives a set of vertices V, a source vertex v_s and a target vertex v_t,
and computes the shortest path between v_s and v_t.
*************************************************************************/
{
   set of struct vertex T;
   struct vertex u, v;
   V ← V \ {v_s};
   T ← {v_s};
   v_s.distance ← 0;
   // initialization: the distance attribute of a vertex u ∈ V is equal to the edge weight w((v_s, u))
   // for all vertices incident from v_s
   for each u ∈ V
      if ( (v_s, u) ∈ E )
         u.distance ← w((v_s, u));
      else u.distance ← +∞;
   end
   // main loop: at each iteration the vertex u in V with minimum distance attribute is moved to T.
   // Distance attributes of the vertices in V are updated in the case that a path via u is shorter
   // than the direct connection from v_s
   while ( v_t ∉ T ) {
      u ← "u ∈ V, such that ∀v ∈ V: u.distance ≤ v.distance";
      T ← T ∪ {u};
      V ← V \ {u};
      for each v "such that (u, v) ∈ E"
         if ( v.distance > w((u, v)) + u.distance )
            v.distance ← w((u, v)) + u.distance;
      end
   }
}
```

Figure 3.29 Algorithm to operatively compute the geodesic distance from a source vertex of the mesh v_s to all the other vertices v of the mesh. The use of a binary tree v-TREE, where the vertices are sorted in ascendant order of distance attribute from the source vertex permits to optimize the time complexity

Algorithm: Geodesic distance computation

Given a set of vertices V and a source vertex v_s:

1. Set v.distance $= +\infty$ for every vertex $v \in V$;
2. Set v_s.distance $= 0$ and insert v_s in V-TREE;
3. Take the vertex v, which has smallest v.distance in V-TREE, and remove it from V-TREE;
4. For each vertex u adjacent to v, if v.distance $+ |(v, u)| < u$.distance, update u.distance $= v$.distance $+ |(v, u)|$ and insert (or reinsert) u to V-TREE;
5. Repeat Step 3 and 4 until V-TREE is empty.

The Exercise 6 proposes the computation of the geodesic distance of the shortest path between two vertices of a simple graph. In Exercise 7, the shortest path itself connecting two vertices instead of merely its length is required to be computed.

3.9.2 Computing Relationships between Facial Stripes

Once values of the normalized geodesic distance are computed for every surface point, iso-geodesic stripes can be identified. For this purpose, the range of the normalized geodesic distance values is quantized into *n* intervals $c_1, \ldots, c_n$. Accordingly, *n* stripes concentric with respect to the nose tip, are identified on the 3D surface, the *I*th stripe corresponding to the set of surface points for which the value of the normalized geodesic distance falls within the limits of interval C_i.

Figure 3.30 An edge-weighted directed graph

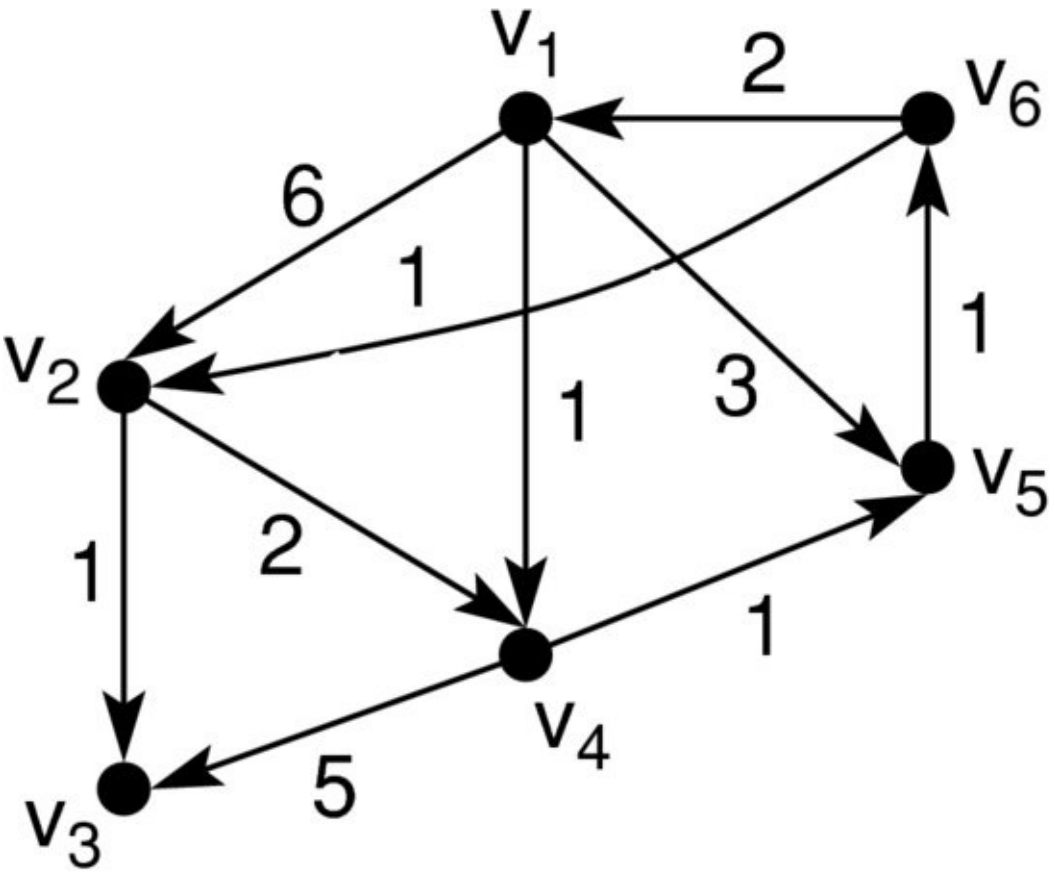


Table 3.1 The evolution of the `distance` attributes in Dijkstra's algorithm when applied to the graph of Figure 3.30

iteration	V	T	v_i distance for i = 1	2	3	4	5	6
1	$\{v_2, v_3, v_4, v_5, v_6\}$	$\{v_1\}$	0	6	$+\infty$	**1**	3	$+\infty$
2	$\{v_2, v_3, v_5, v_6\}$	$\{v_1, v_4\}$		6	6		**2**	$+\infty$
3	$\{v_2, v_3, v_6\}$	$\{v_1, v_4, v_5\}$		6	6			**3**
4	$\{v_2, v_3\}$	$\{v_1, v_4, v_5, v_6\}$		**4**	6			
5	$\{v_3\}$	$\{v_1, v_4, v_5, v_6, v_2\}$			**5**			
6	$\{\}$	$\{v_1, v_4, v_5, v_6, v_2, v_3\}$						

In this example $v_s = v_1$ and $v_t = v_2$. At each iteration, the distance of the vertex $u \in V$, such that $\forall v \in V : u.\text{distance} \leq v.\text{distance}$ is marked in bold in the table. This highlighted distance value indicates that the associated vertex is transferred from V to T in that iteration. After 5 iterations the $v_t = v_2$ condition is reached, and continuing for one iteration more the lengths of the shortest paths for all vertices in the graph are computed.

Figure 3.31**b** shows the projection on the XY plane of the pairs of iso-geodesic stripes of the three subjects in Figure 3.31**a**, thus evidencing the shape variations of the stripes. As an example, Figure 3.35 shows the first nine iso-geodesic stripes identified on the face scans of two individuals.

Figure 3.31 (a) Sample face models, where the fourth and seventh iso-geodesic stripes are evidenced. The 3DWW relationship descriptors are computed for the UL, UR, and L parts of the stripe pair; (b) Projection of the pairs of iso-geodesic stripes in (a) on the XY plane, with the partitioning of the iso-geodesic stripes into three parts

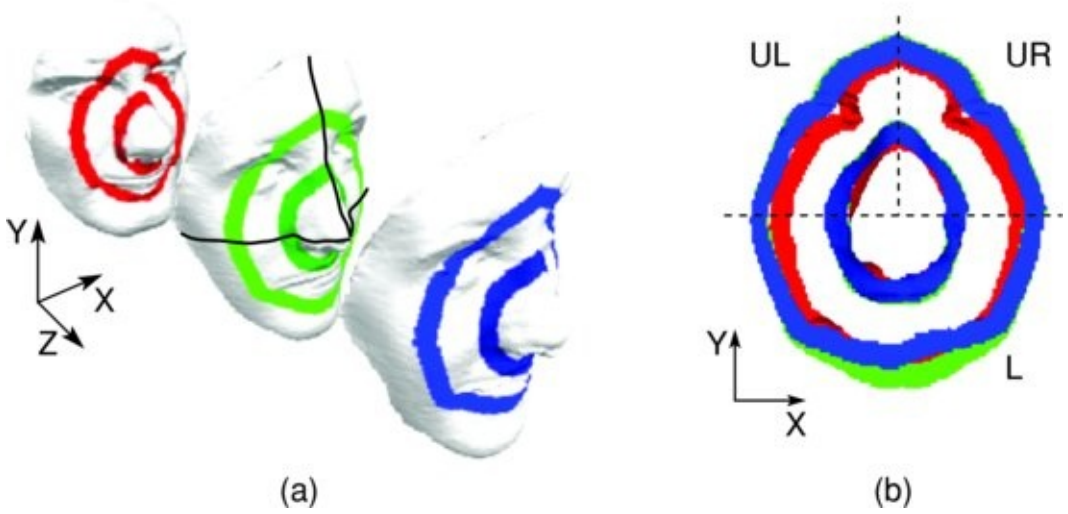
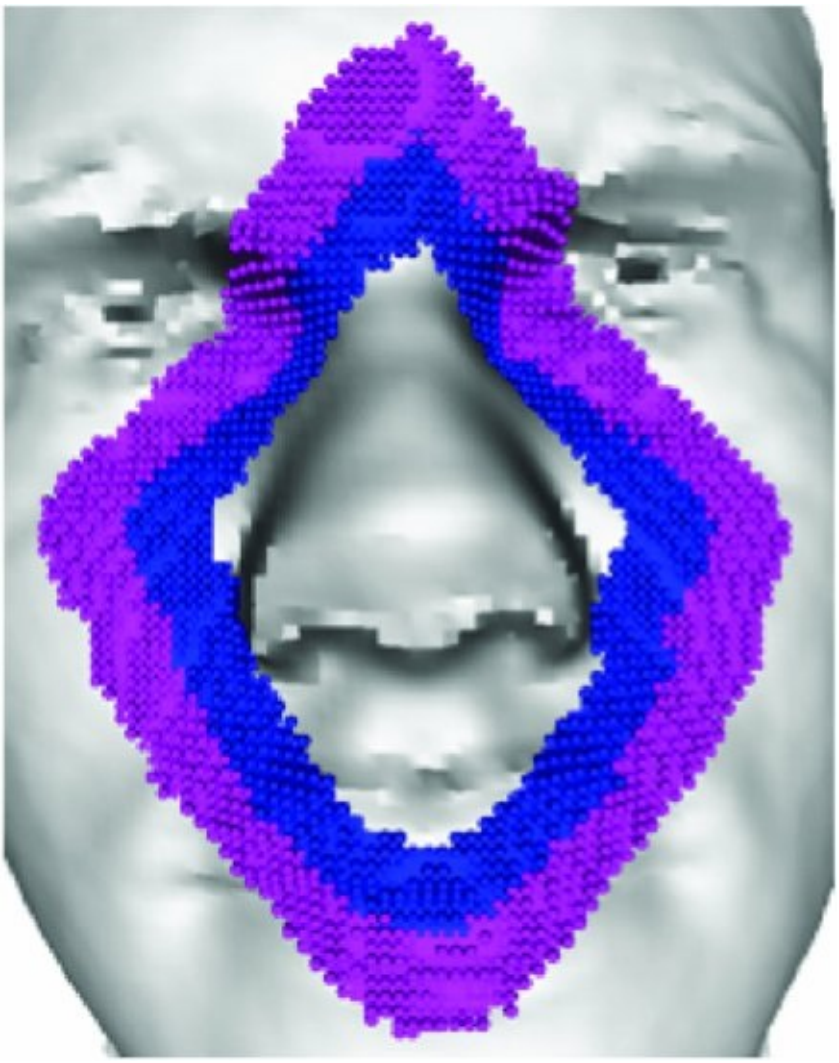


Figure 3.32 The 3D voxelization of a pair of facial stripes



Results of the analysis of the deformation that non-neutral facial expressions induce in the shape of the iso-geodesic stripes, as is detailed in Berretti *et al.* (2010), motivate the decomposition of the facial stripes into three parts, *upper left* (UL), *upper right* (UR) and *lower* (L), with respect to the coordinates of the nose tip (see Figure 3.31**b**). In general, under the effect of non-neutral facial expressions, the region around the mouth is subject to larger deformations than the other regions of the face. Furthermore, decomposition of the upper part into the upper left and upper right allows the face representation model to better deal with slight asymmetries of the face that constitute a characterizing trait of some individuals. This subdivision resulted necessary in improving the performance of the approach in the case of faces with expression variations (results of the iso-geodesic stripes approach without face partitioning were first reported in Berretti *et al.* (2006)).

Once facial stripes are extracted, distinctive structural features of 3D face

scans are captured by describing the pointwise 3D spatial relationships between homologous parts of pairs of iso-geodesic stripes. To this end, the *3D weighted walkthroughs* (3DWWs) descriptor has been used. The 3DWW was first introduced in Berretti *et al.* (2006), and their use and properties in the context of 3D face recognition have been extensively discussed in Berretti *et al.* (2010). It defines a set of integral measures over the points of two regions, *A* and *B*, in the 3D domain. These measures are captured through weights $w_{i,j,k}(A, B)$ that encode the number of pairs of points belonging to *A* and *B*, whose displacement is captured by the walkthrough $\langle i, j, k \rangle$ (with *i*, *j*, *k* taking values in $\{-1, 0, +1\}$):

$$w_{i,j,k}(A, B) = \frac{1}{K_{i,j,k}} \cdot \int_A \int_B C_i(x_b - x_a) C_j(y_b - y_a) C_k(z_b - z_a) \, \mathrm{d}\vec{b} \, \mathrm{d}\vec{a}, \tag{3.21}$$

where $\mathrm{d}\vec{b} = dx_b dy_b dz_b$ and $\mathrm{d}\vec{a} = dx_a dy_a dz_a$; $K_{i,j,k}$ acts as a normalization factor to guarantee that $w_{i,j,k}$ takes value in [0, 1]; $C_{\pm 1}(\cdot)$ are the characteristic functions of the positive and negative real semi-axis $(0, +\infty)$ and $(-\infty, 0)$, respectively; $C_0(\cdot)$ denotes the Dirac's function that is used to reduce the dimensionality of the integration domain to enable a finite non-null measure. In particular, $C_{\pm 1}(t)$ are defined in the following way:

$$C_{+1}(t) = \begin{cases} 1 & \text{if } t > 0 \\ 0 & \text{otherwise} \end{cases} \qquad C_{-1}(t) = \begin{cases} 1 & \text{if } t < 0 \\ 0 & \text{otherwise,} \end{cases} \tag{3.22}$$

being the Dirac's function, by definition:

$$C_0(t) = \begin{cases} 1 & \text{if } t == 0, \\ 0 & \text{otherwise.} \end{cases} \tag{3.23}$$

3DWWs are capable to quantitatively measure the relative spatial arrangement of two extended sets of 3D points by computing 27 weights and organizing them in a $3 \times 3 \times 3$ matrix. As a particular case, the 3DWWs computed between an extended 3D entity and itself also account for intrinsic shape information.

By developing on the properties of integrals, it can be easily proved that weights $w_{i,j,k}(A, B)$ are reflexive (i.e., $w_{i,j,k}(A, B) = w_{-i,-j,-k}(B, A)$) and invariant with respect to shifting and scaling. In addition, 3DWWs are compositional, in that the walkthroughs between *A* and the union $B_1 \cup B_2$ can be derived by the linear combination of the 3DWWs between *A* and B_1 and *A* and B_2. The demonstrations for the 2D case can be found in Berretti *et al.* (2003). According to this, the integral of Equation 3.21 can be reduced to the linear combination of subintegrals computed on any partition of *A* and *B*.

Computer Implementation

Considering the uniform voxelization of two 3D entities A and B ($A = \bigcup_n A_n$, and $B = \bigcup_m B_m$), the integral of Eq. 3.21 can be replaced by a linear combination of terms $w_{i,j,k}(A_n, B_m)$ computed on the voxels pairs $\langle A_n, B_m \rangle$:

$$w_{i,j,k}\left(\bigcup_n A_n, \bigcup_m B_m\right) = \frac{1}{K_{i,j,k}(A,B)} \sum_n \sum_m K_{i,j,k}(A_n, B_m) \cdot w_{i,j,k}(A_n, B_m) \quad (3.24)$$

Two 3D voxels in the 3D space can be posed, one with respect to the other, in a set of 27 different mutual arrangements (*basic* arrangements). Because the $w_{i,j,k}$ coefficients for these basic arrangements can be computed in closed form using Equation 3.21, the relationships between two extended entities can be reduced to the combination through Equation 3.24 of the coefficients computed in the basic cases, thus avoiding the numerical evaluation of the integral measure.

As an example, Figure 3.32 shows a 3D face with the voxelization of two facial stripes (for the clarity of the visualization, the cube shape of a 3D voxel is approximated with a sphere).

Suggestion: *As an example and suggestion for the derivation of the other cases, in the following the closed form computation of the* $w_{+1,-1,+1}$ *coefficient is detailed. To make easier the analytical derivation, we assume the two voxels are cubes with side equal to t. The voxels have also the 3D coordinates in the reference system shown in .* Figure 3.34.

According to Equation 3.21, *the relationship between voxels* a_n *and* b_m *can be written as follows:*

$$w_{+1,-1,+1}(a_n, b_m) = \frac{1}{K_{+1,-1,+1}} \int_{a_n} \int_{b_m} C_{+1}(x_b - x_a) C_{-1}(y_b - y_a) C_{+1}(z_b - z_a)\, d\vec{b}\, d\vec{a}. \quad (3.25)$$

According to the coordinates of . Figure 3.34 *for the two voxels, the integral of Equation* 3.25 *can be split into*

$$= \frac{1}{K_{+1,-1,+1}} \int_{V_b}^{V_b+T} dy_b \int_{V_a}^{V_a+T} dy_a \int_{D}^{D+T} \int_{z_a}^{D+T} \int_{L}^{L+T} \int_{x_a}^{L+T} dx_b\, dx_a\, dz_b\, dz_a, \quad (3.26)$$

being the integration along y independent from x and z. Then, by applying the usual integration rules, it results in the following:

$$w_{+1,-1,+1}(a_n, b_m) = \frac{T^2}{K_{+1,-1,+1}} \int_D^{D+T} \int_{z_a}^{D+T} \int_L^{L+T} (L + T - x_a)\, dx_a\, dz_b\, dz_a =$$
$$= \frac{T^2}{K_{+1,-1,+1}} \int_D^{D+T} \int_{z_a}^{D+T} \left[(L + T)x_a - \frac{x_a}{2} \right]_L^{L+T} dz_b\, dz_a =$$
$$= \frac{T^2}{K_{+1,-1,+1}} \int_D^{D+T} \int_{z_a}^{D+T} \frac{T^2}{2}\, dz_b\, dz_a =$$
$$= \frac{T^2}{K_{+1,-1,+1}} \int_D^{D+T} \frac{T^2}{2}(D + T - z_a)\, dz_a = \frac{1}{K_{+1,-1,+1}} \frac{T^6}{4}.$$

To obtain an adimensional value, the normalization factor $K_{+1,-1,+1}$ is posed equal to the products of the volumes of the two voxels, that is: $|a_n||b_m|=T^6$. This results in a final value for the coefficient $w_{+1,-1,+1}(a_n, b_m)=1/4$. A similar approach can be used in the derivation of the other coefficients.

The properties of 3DWW, joined with the geodesic distance computation and the face partitioning provide the method with robustness to expression variations. In fact, geodesic distances between two facial points keep sufficiently stable under expression changes resulting into the fact that the large majority of the points of each stripe still remain within the same stripe, even under facial expression changes. In addition, because of the constrained elasticity of the skin tissue, neighbor points can be assumed to feature very similar motion for moderate facial expressions in most parts of the face. For all these points the mutual displacement between the two points is mainly determined by the geometry of the neutral face. This property is preserved by 3DWWs that provide an integral measure of displacements between pairs of points.

3.9.3 Face Representation and Matching Using Iso-geodesic Stripes

A generic face model *f*, is represented through a set of N_F stripes. In that 3DWWs are computed for every pair of iso-geodesic stripes (including the pair composed by a stripe and itself), a face is represented by a set of $N_F \cdot (N_F + 1)/2$ relationship matrixes. According to the proposed representation, iso-geodesic stripes and 3DWW computed between pairs of stripes (*interstripe* 3DWW) and between each stripe and itself (*intrastripe* 3DWW), have been cast to a graph representation where *intrastripe* 3DWW are used to label the graph nodes and *interstripe* 3DWWs to label the graph edges (see Figure 3.35).

To compare graph representations, distance measures for node labels and for edge labels have been defined. Both of them rely on the L_1 distance measure $\mathcal{D}$

defined between 3DWWs (Berretti et al., 2006). The similarity measure between two face models represented through the graphs P and G with nodes p_k and g_k, is then derived as follows:

$$\mu(P,G) = \frac{\alpha}{N_P} \cdot \sum_{k=1}^{N_P} \mathcal{D}(w(p_k, p_k), w(g_k, g_k)) + \frac{2(1-\alpha)}{N_P(N_P-1)} \cdot \sum_{k=1}^{N_P} \sum_{h=1}^{k-1} \mathcal{D}(w(p_k, p_h), w(g_k, g_h)), \quad (3.27)$$

where the first summation in Equation 3.27 accounts for the *intrastripe* 3DWWs similarity measure, and the second summation evaluates the *interstripe* 3DWWs similarity measure. The α parameter permits to weight differently the two distance components and its value has been set to 0.3 (Berretti et al., 2010). This value has been tuned in a pilot set of experiments carried out on the *Face Recognition Grand Challenge* version 1.0 (FRGC v1.0) (Phillips et al., 2005) database and shows that to support face recognition, *interstripe* spatial relationships are more discriminant than *intrastripe* spatial relationships.

Implicitly, Equation 3.27 assumes that the number of nodes N_P in the graph P, is not greater than the number of nodes N_G of the graph G. This can be assumed with no loss of generality, in that if $N_P > N_G$, graphs P and G can be exchanged.

Following the considerations discussed earlier, distances between faces of two individuals are measured by computing the 3DWW for each pair of iso-geodesic stripes, separately in the three face parts, and then comparing the 3DWWs of homologous pairs of the two faces. The final dissimilarity measure is obtained by averaging distances in the three parts.

According to Equation 3.27, the overall runtime complexity can be estimated as $O(N_P^2 T_D)$, being T_D the complexity in computing $\mathcal{D}$, that is estimated to be a constant value (Berretti et al., 2010). This permits efficient implementation for face identification in large datasets, also with the use of appropriate index structures, with great savings in performance. More details on the index structure and its performance are discussed in Berretti *et al.* (2001).

Exercises

1. Let S_1, S_2, and S_3 be regular surfaces, Prove that

A. If $\phi : S_1 \to S_2$ is an isometry, then $\phi^{-1} : S_2 \to S_1$ is also an isometry.

B. If $\phi : S_1 \to S_2$ is an isometry, $\phi : S_1 \to S_3$ are isometries, then,

$\psi \circ \phi : S_1 \rightarrow S_3$ is an isometry.

C. Prove that the isometry of regular surface *S* constitutes a group. This group is called *the group of isometries*.

2. Compute *square-root velocity function* of the circle
$x = \cos 2t, y = \sin 2t, t \in [0, 2\pi]$

3. Derive the relation 3.6

4. Given two points *P* and *q* in $\mathbb{R}^n$, compute the geodesic path between them.

5. Derive the relation 3.7 and the formula of geodesic path $\psi(\tau)$

6. Write a pseudo-algorithm of the geodesic distance between two curves represented by q_α^i , *i*=1, 2

7. Given the edge-weighted directed graph of Figure 3.30, compute the Dijkstra's shortest path between v_1 and v_2. The growth of the set *t*, the corresponding variation of the set *v*, and the subsequent values of the `distance` attributes of the vertices in the graph have been listed in Table 3.1.

8. Modify the description of Dijkstra's algorithm of Figure 3.28 in order to find the shortest path itself instead of merely the length of the shortest path. *Suggestion: Pay special attention to the data structure for the representation of the path.*

9. Using the Equation 3.21 derive the $w_{i,j,k}$ between the two 3D voxels illustrated on the left of Figure 3.33.

Figure 3.33 Two 3D voxels in one of the *basic* spatial arrangements (b_n is lower positioned with respect to a_n, and aligned with a_n along the *X* and *Z* axes) and the resulting values of the $w_{i,j,k}$ coefficients

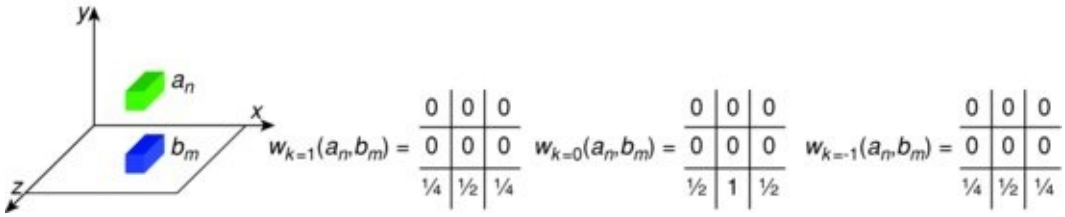


Figure 3.34 The two voxels are cubes with side equal to *t* and the coordinates given in the figure

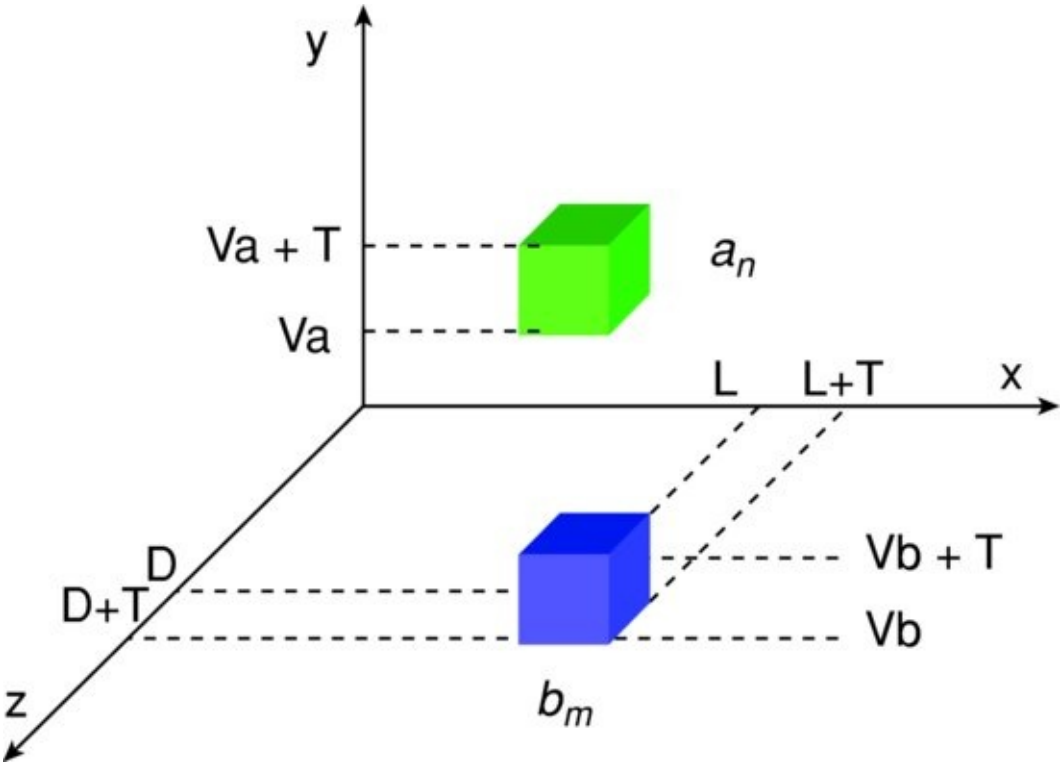
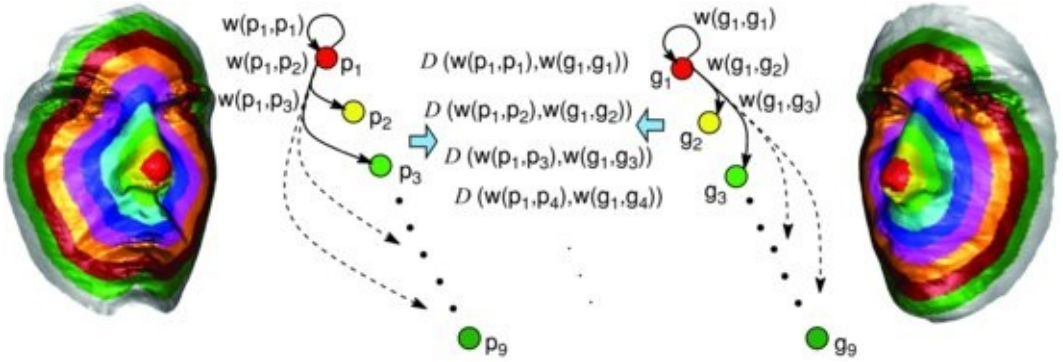
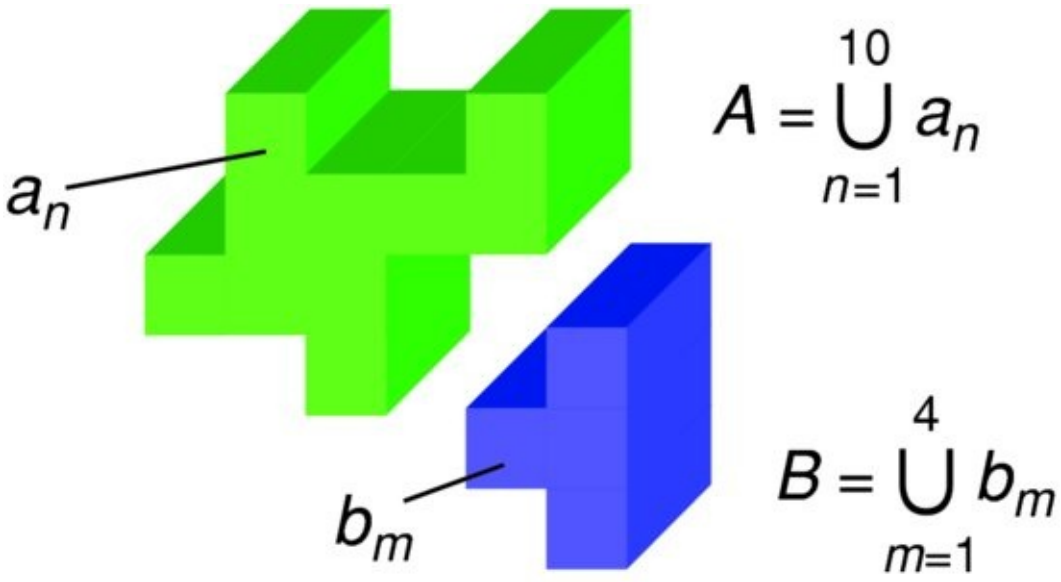


Figure 3.35 The first nine iso-geodesic stripes for two sample face scans. The graphs constructed on a subset of the stripes and their matching are also shown



10. Using Equation 3.24 and the relationship between 3D voxels in the basic arrangements, compute the relationships between the entities A and B of Figure 3.36.

Figure 3.36 Two extended 3D entities comprising, respectively, 10 voxels (*A*) and 4 voxels (*B*)



Glossary

In the following we provide a brief introduction to the geometrical terms used in the book. For details, please refer to some standard textbooks on differential geometry such as Boothby (1986) or Spivak (1979).

- **Geodesic Path**: For any two points $p_1, p_2 \in M$, a geodesic is a path α connecting p_1 and p_2 on M that is locally length-minimizing, that is, this path cannot be changed or perturbed in any way and its length cannot be reduced. If is the shortest path connecting p_1 and p_2 then it is called the minimizing geodesic. Note that the definition of a geodesic is dependent upon the Riemmanian structure of M, that is,

  $$\hat{\alpha} = \underset{\alpha:[0,1]\to M,\ \alpha(0)=p_1,\alpha(1)=p_2}{\operatorname{argmin}} L[\alpha].$$

  The length of this path is called the geodesic length $\mathrm{d}(p_1, p_2) = L[\hat{\alpha}]$.
- **Group Action by Isometries**: If a group G acts on a Riemannian manifold M, then the group action is said to be by isometries if $\mathrm{d}(p_1, p_2) = \mathrm{d}((g \cdot p_1), (g \cdot p_2))$ for all $g \in G$ and $p_1, p_2 \in M$.
- **Group Action on a Manifold**: If G is an algebraic group and M is a manifold, then the mapping $G \times M \to M$, given by $(g, p) \mapsto g \cdot p$ is called a group action if: (1) $(g_1 (g_2, p)) = (g_1 * g_2) \cdot p$, (2) $(e, p)=p$, for all $g_1, g_2 \in G$ and $p \in M$ and where E denotes the identity element of G. The set $\{g \cdot p | g \in G\}$ is called the orbit of P and is denoted by $[p]$.
- **Inherited Distance on** M/G: As in the previous item, let G acts on M by

isometries and, additionally, let the orbits of *M* under *G* be closed. Then, the inherited distance between any two elements of *M*/*G* is defined to be

$$d_{M/G}([p_1],[p_2]) = \inf_{g \in G} d_M(p_1,(g \cdot p_2)).$$

- **Inherited Metric**: If the action of *G* on a Riemannian manifold *M* is by isometries, and the quotient space *M*/*G* is a differentiable manifold, then *M*/*G* inherits the Riemannian metric from *M* and this metric can be used to define geodesics and geodesic distances between elements of *M*/*G*.
- **Length of Curves**: For any curve $\alpha : [0,1] \to M$, the length of α can be defined as:

  $$L[\alpha] = \int_0^1 \left\langle \frac{d\alpha}{dt}, \frac{d\alpha}{dt} \right\rangle^{(1/2)} dt,$$

  where the inner product inside the integral is defined using the Riemannian metric of *M*.
- **Manifold**: A manifold is a topological space that is *locally Euclidean*, that is, for each point on this set, an open neighborhood can be locally identified with an open set of a Euclidean space using a homemorphism. If these mappings are smooth (diffeomorphsisms) and compatible with each other (their concatenations are also smooth), then the manifold is called adifferentiable manifold. In this chapter, all the manifolds we deal with are differentiable manifolds and, hence, we often drop the work differentiable.
- **Path-Straightening Approach**: This is an approach for finding geodesic paths between points on a manifold by minimizing the energy functional:

  $$E[\alpha] = \int_0^1 \left\langle \frac{d\alpha}{dt}, \frac{d\alpha}{dt} \right\rangle dt.$$

  Klassen & Srivastava (2006) derived a particularly convenient form of the gradient of *E* in cases where *M* is a submanifold of a larger vector space *v* and the Riemannian metric on *M* is a restriction of the metric on *v*. This is the case for all of the manifolds studied in this book.
- **Quotient Space of a Manifold**: Let a group *G* act on the manifold *M* as defined earlier. Then, the set of all orbits of all elements of *M* is called the quotient of *M* under *G*. IT is denoted by *M*/*G*. We have:

  $$M/G = \{[p] | p \in M\}, \text{ where } [p] = \{(g \cdot p) | g \in G\}.$$

- **Reparametrization Action on Curves**: Let $\mathcal{B}$ be an appropriate space of curves of the type $\beta : [0,1] \to \mathbb{R}^n$ and Γ be the reparametrization group. Then, Γ acts on $\mathcal{B}$ according to the action $(\gamma, \beta) = \beta \circ \gamma$. It is interesting to note that if we assume the standard $\mathbb{L}^2$ metric on $\mathcal{B}$, then this action is not

by isometries. However, if one assumes an elastic metric, as defined by Mio *et al.* [2007], then this action is by isometries and one can define the inherited distance on the quotient space $\mathcal{B}/\Gamma$.

- **Reparametrization Group**: Let Γ be the set of all corner-preserving diffeomorphisms from [0, 1] to itself. That is, Γ is the set of all γ such that γ is a diffeomorphisms and $\gamma(0) = 0$ and $\gamma(1) = 1$. Γ is a group under concatenation: for any $\gamma_1, \gamma_2 \in \Gamma$ the group operation is $\gamma_1 \circ \gamma_2$. The identity element of the group is given by $\gamma_{id}(t) = t$.
  If $\beta : [0, 1] \to \mathbb{R}^n$ is a parametrized curve in $\mathbb{R}^n$, then for a $\gamma \in \Gamma$, the curve $\tilde{\beta}(t) = \beta(\gamma(t))$ is called a reparametrization of β; therefore, Γ is also called the reparametrization group.
- **Riemannian Metric**: A Riemannian metric is map that smoothly associates to each point $p \in M$ a form for computing inner product between elements of $T_p(M)$. A manifold with a Riemannian metric on it is called a Riemannian manifold.
- **Tangent Space**: For a point $p \in M$, $T_p(M)$ is the set of all vectors that are tangent to *M* at point *P*. A tangent vector can be defined by constructing a differentiable curve on *M* passing through *P* and evaluating the velocity to the curve at that point. $T_p(M)$ is a vector space of the same dimension as the manifold *M*.

References

Antini G, Berretti S, del Bimbo A, Pala P. 3D mesh partitioning for retrieval by parts applications. In: Proceeding of the IEEE International Conference on Multimedia and Expo; 2005 Jul 6–9 Amsterdam, The Netherlands. New York: IEEE. 2005. p. 1210–1213.

Berretti S, Bimbo AD, Pala P. Distinguishing facial features for ethnicity-based 3d face recognition. ACM Transactions on Intelligent Systems and Technology 2012;3(3):45.

Berretti S, del Bimbo A, Pala P. Description and retrieval of 3D face models using iso-geodesic stripes. In: Proceeding of the 8th ACM International Workshop on Multimedia Information Retrieval; 2006 Oct 26–27, Santa Barbara, CA. New York: ACM Press. 2006. p. 13–22.

Berretti S, del Bimbo A, Pala P. 3D face recognition using iso-geodesic stripes. IEEE Transactions on Pattern Analysis and Machine Intelligence

2010;32(12):2162–2177.

Berretti S, del Bimbo A, Vicario E. Efficient matching and indexing of graph models in content based retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence 2001;23(10):1089–1105.

Berretti S, del Bimbo A, Vicario E. Weighted walkthroughs between extended entities for retrieval by spatial arrangement. IEEE Transactions on Multimedia 2003;5(1):52–70.

Boothby WM. *An Introduction to Differentiable Manifolds and Riemannian Geometry*. 2nd ed. London: Academic Press; 1986.

Bronstein AM, Bronstein MM, Kimmel R. Three-dimensional face recognition. International Journal of Computer Vision 2005;64(1):5–30.

Bronstein AM, Bronstein MM, Kimmel R. Expression-invariant representations of faces. IEEE Transactions on Image Processing 2007;16(1):188–197.

Chang KI, Bowyer KW, Flynn PJ. An evaluation of multimodal 2D+3D face biometrics. IEEE Transactions on Pattern Analysis and Machine Intelligence 2005;27(4):619–624.

Cormen TH, Leiserson CE, Rivest RL, Stein C. *Introduction to Algorithms*. Cambridge: MIT Press and McGraw-Hill; 2001.

Farkas LG. *Anthropometry of the Head and Face*. New York: Raven Press; 1994.

Hofmann T, Buhmann JM. Pairwise data clustering by deterministic annealing. IEEE Transactions on Pattern Analysis and Machine Intelligence 1997;19(1):1–14.

Joshi SH, Klassen E, Srivastava A, Jermyn I. A novel representation for Riemannian analysis of elastic curves in RN. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition; 2007 Jun 17–22, Minneapolis, MN. New York: IEEE. 2007. p. 1–7.

Karcher H. Riemannian center of mass and mollifier smoothing. Communications on Pure and Applied Mathematics 1977;30:509–541.

Mian AS, Bennamoun M, Owens R. An efficient multimodal 2D-3D hybrid approach to automatic face recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 2007;29(11):1927–1943.

Michor PW, Mumford D. Riemannian geometries on spaces of plane curves. Journal of European Mathematical Society 2006;8:1–48.

Mpiperis I, Malassiotis S, Strintzis M. Expression compensation for face recognition using a polar geodesic representation. In: Proceedings of the 3rd International Symposium on 3D Data Processing, Visualization, and Transmission; 2006 Jun 14–16, University of North Carolina, Chapel Hill, NC. New York: IEEE. 2006. p. 224–231, Chapel Hill, NC. 2006.

Phillips PJ, Flynn PJ, Scruggs T, Bowyer KW, Chang J, Hoffman K, Marques J, Min J, Worek W. Overview of the face recognition grand challenge. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshop on Face Recognition Grand Challenge Experiments. Volume 1; 2005 Jun 21, San Deigo, CA. Los Alamitos: IEEE. Computer Society Press. 2005. p. 947–954.

Pressley A. *Elementary Differential Geometry*. New York: Springer-Verlag; 2001.

Samir C, Srivastava A, Daoudi M. 3D face recognition using shapes of facial curves. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, Volume 5; 2006 May 14–19, Toulouse, France. New York: IEEE. 2006. p. 933–936.

Samir C, Srivastava A, Daoudi M, Klassen E. An intrinsic framework for analysis of facial surfaces. International Journal of Computer Vision 2009;82(1):80–95.

Spivak M. *A Comprehensive Introduction to Differential Geometry*. Vol. I. 2nd ed., Wilmington, Del: Publish or Perish Inc.; 1979.

Srivastava A, Klassen E, Joshi SH, Jermyn IH. Shape analysis of elastic curves in euclidean spaces. IEEE Transactions on Pattern Analysis and Machine Intelligence 2011;33(7):1415–1428.

Yezzi AJ, Mennucci A. Conformal metrics and true “gradient flows” for curves. In: Proceedings of the 10th IEEE International Conference on Computer Vision ICCV'05, Volume. 1; 2005, Oct 17–20, Beijing, China. New York: IEEE. 2005. p. 913–919.

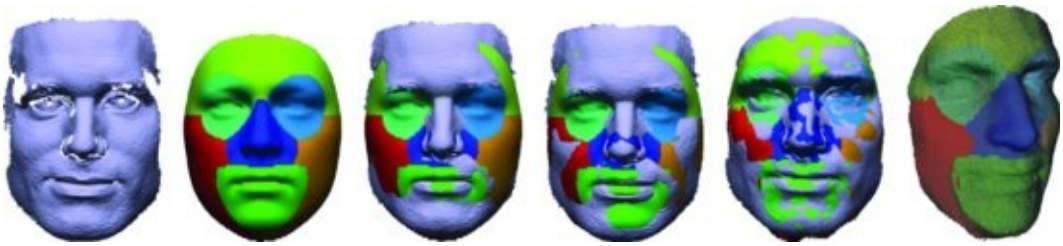
4

3D Morphable Models for Face Surface Analysis and Recognition

Frank B. ter Haar[1] and Remco Veltkamp[2]

[1]TNO – Intelligent Imaging, The Hague, The Netherlands

[2]Department of Information and Computing Sciences, Utrecht University, The Netherlands



In this chapter, we present an automatic and efficient method to fit a statistical deformation model of the human face to 3D scan data. In a global-to-local fitting scheme, the shape parameters of this model are optimized such that the produced instance of the model accurately fits the 3D scan data of the input face. To increase the expressiveness of the model and to produce a tighter fit of the model, the method fits a set of predefined face components and blends these components afterwards. Quantitative evaluation shows an improvement of the fitting results when multiple components are used instead of one.

For a 3D face recognition system based on model coefficients, it is of utmost importance that the statistics of many realistic faces are captured in the morphable model. In case a face cannot be modeled, the automatically acquired model coefficients are unreliable, which hinders the automatic recognition. We present a new bootstrapping algorithm to automatically enhance a 3D morphable face model with new face data. The algorithm is based on a morphable model-fitting method that uses a set of predefined face components. This fitting method produces accurate model fits to 3D face data with noise and holes. In the fitting

process, the dense point-to-point correspondences between the scan data and the face model can become less reliable at the border of components. This is solved by introducing a blending technique that improves on the distorted correspondences close to the borders. Afterwards, a new face instance is acquired similar to the 3D scan data and in full correspondence with the face model. These newly generated face instances, which go beyond the statistics of the initial morphable face model, can then be added to the morphable face model to build a more descriptive one. To avoid our bootstrapping algorithm from needlessly adding redundant face data, we incorporate a redundancy estimation algorithm. Quantitative and qualitative evaluation shows that this algorithm successfully enhances an initial morphable face model with new face data, in a fully automatic manner.

The accurately generated face instances are manifold meshes without noise and holes and can be effectively used for 3D face recognition. The results show that model coefficient–based face matching outperforms contour curve and landmark-based face matching, and is more time efficient than contour curve matching.

4.1 Introduction

There are numerous methods to perform 3D face analysis and recognition. Some of these techniques are based on 3D geodesic surface information, such as the methods of Bronstein *et al.* (2005) and Berretti *et al.* (2007). The geodesic distance between two points on a surface is the length of the shortest path between two points. To compute accurate 3D geodesic distances for face recognition purposes, a 3D face without noise and without holes is desired. Because this is typically not the case with laser range scans, the noise has to be removed and the holes in the 3D surface interpolated. However, the success of basic noise removal techniques, such as Laplacian smoothing, is very much dependent on the resolution and density of the scan data. Straightforward techniques to interpolate holes using curvature information or flat triangles often fail in case of complex holes, as pointed out by Davis *et al.* (2002). The use of a deformation model to approximate new scan data and interpolate missing data is a gentle way to regulate flaws in scan data.

A well known *statistical deformation model* specifically designed for surface meshes of 3D faces, is the 3D morphable face model of Blanz and Vetter (1999). This statistical model was built from 3D face scans with dense correspondences

to which principal component analysis (PCA) was applied. In their early work, Blanz and Vetter fit this 3D morphable face model to 2D color images and cylindrical depth images from the $Cyberware^{TM}$ (Del Monte AvenueMonterey, CA) scanner. In each iteration of their fitting procedure, the model parameters are adjusted to obtain a new 3D face instance, which is projected to 2D cylindrical image space allowing the comparison of its color values (or depth values) to the input image. The parameters are optimized using a stochastic Newton algorithm. More recently, Blanz *et al.* (2007) proposed a method to fit their 3D morphable face model to more common textured depth images. In the fitting process, a cost function is minimized using both color and depth values after the projection of the 3D model to 2D image space. To initialize their fitting method, they manually select seven corresponding face features on their model and in the depth scan. A morphable model of expressions was proposed by Lu and Jain (2008). Starting from an existing neutral scan, they use their expression model to adjust the vertices in a small region around the nose to obtain a better fit of the neutral scan to a scan with a certain expression. Amberg *et al.* (2008) built a PCA model from 270 identity vectors and a PCA model from 135 expression vectors and combined the two into a single morphable face model. They fitted this model to 3D scans of both the UND (University of Notre Dame) and GAVAB (Grupo de Algorítmica para la Visión Artificial y la Biometría) face sets (see next section), and use the acquired model coefficients for expression invariant face matching with considerable success.

Non statistical deformation models were proposed as well. Huang *et al.* (2006) proposed a global-to-local deformation framework to deform a shape with an arbitrary dimension (2D, 3D or higher) to a new shape of the same class. They show their framework's applicability to 3D faces, for which they deform an incomplete source face to a target face. Kakadiaris *et al.* (2006) deform an annotated face model to scan data. Their deformation is driven by triangles of the scan data attracting the vertices of the model. The deformation is restrained by stiffness, mass, and damping matrices that control the resistance, velocity, and acceleration of the models vertices. Whitmarsh *et al.* (2006) fit a parameterized CANDIDE face model to scan data by optimizing shape and action parameters. The advantage of such deformable faces is that they are not limited to the statistical changes of the example shapes, so the deformation has less restrictions. However, this is also their disadvantage, because these models cannot rely on statistics in case of noise and missing data.

4.2 Range Sc

4.2 Data Sets

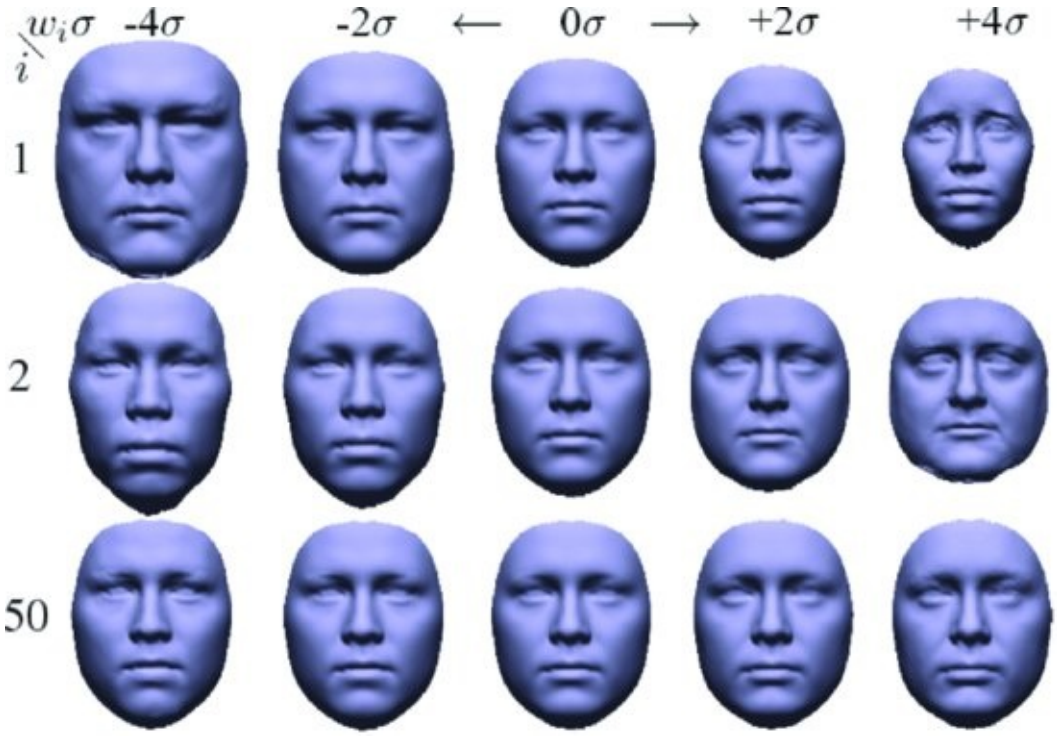
In this chapter, we fit a morphable face model that is defined as $S_{\text{inst}} = \bar{S} + \sum_{i=1}^{m} w_i \sigma_i s_i$ to 3D scan data. By doing this, we obtain a clean model of the face scan, that we can use to identify 3D faces. The scans that we fit the morphable face model to are the 3D face scans of the UND (Chang et al., 2005), a subset of the GAVAB (Moreno and Sanchez, 2004; ter Haar et al., 2008) and a subset of the Binghamton University 3D Facial Expression (BU-3DFE) (Yin et al., 2006) databases. The UND set contains 953 frontal range scans of 277 different subjects with mostly neutral expression. The GAVAB set consists of nine low quality scans for each of its 61 subject, including scans for different poses and expressions. From this set, we selected, per subject, four neutral scans, namely the two frontal scans and the scans in which subjects look up and down. Acquired scan data from these poses differ in point cloud density, completeness and relatively small facial changes. The BU-3DFE set was developed for facial expression classification. This set contains one neutral scan and 24 expression scans having different intensity levels, for each of its 100 subjects. From this set, we selected the neutral scans and the low level expression scans (anger, disgust, fear, happiness, sadness, surprise at level 1).

Although the currently used morphable model is based on faces with neutral expressions only, it makes sense to investigate the performance of our face model fitting in case of changes in pose and expressions. These variations in 3D scan data, which are typical for a non cooperative scan environment, allow us to evaluate our 3D face recognition methods.

We aim at 3D face recognition, so we need to segment the face from each scan. For that, we employ our pose normalization method (ter Haar and Veltkamp, 2008) that normalizes the pose of the face and localizes the tip of the nose. Before pose normalization, we applied a few basic preprocessing steps to the scan data: The 2D depth images were converted to triangle meshes by connecting the adjacent depth samples with triangles, slender triangles and singularities were removed, and only considerably large connected components were retained. Afterwards, the face is segmented by removing the scan data with a Euclidean distance larger than 110 mm from the nose tip. The face segmentation is visualized in Figure 4.1.

Figure 4.1 Face segmentation. The depth image (left) is converted to a surface mesh (middle). The surface mesh is cleaned, the tip of the nose is detected and the face segmented (right, in pink)

Figure 4.2 Face morphing along eigenvectors starting from the mean face (center column). Different weights for the principal eigenvectors (e.g., i = 1, 2) changes the global face shape. For latter eigenvectors the shape changes locally (e.g., i = 50)



4.3 Face Model Fitting

In general, 3D range scans suffer from noise, outliers, and missing data, and their resolution may vary. The problem with single face scans, the GAVAB scans in particular, is that large areas of the face are missing, which are hard to fill using simple hole filling techniques. When the morphable face model is fitted to a 3D face scan, a model is obtained that has no holes, has a proper topology, and has an assured resolution. By adjusting the m=99 weights w_i for the eigenvectors, the morphable model creates a new face instance. To fit the

morphable model to 3D scan data, we need to find the optimal set of m weights w_i. This section describes a fully automatic method that efficiently finds a proper model of the face scan in the m-dimensional space.

4.3.1 Distance Measure

To evaluate if an instance of the morphable face model is a good approximation of the 3D face scan, we use the root mean square (RMS) distance of the instance's vertices to their closest points in the face scan. For each vertex point (p) from the instance (M_1), we find the vertex point (p') in the scan data (M_2) with the minimal Euclidean distance

(4.1) $$e_{\min}(p, M_2) = \min_{p' \in M_2} d(p, p')$$

using a kD-tree. The RMS distance is then measured between M_1 and M_2 as:

(4.2) $$d_{\text{rms}}(M_1, M_2) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} e_{\min}(p_i, M_2)^2}$$

using n vertices from M_1. Closest point pairs (p, p') for which p' belongs to the boundary of the face scan, are not used in the distance measure.

The morphable face model has n=75, 972 vertices. These vertices cover the face, neck and ear regions and its resolution in the upward direction is three times higher than in its sideways direction. Because the running time of our measure is dependent on the number of vertices, we use for the model fitting only those vertices that cover the face (data within 110 mm from the tip of the nose) and not the neck and ears. To obtain a more uniform resolution for the model, we reduced the upward resolution to one third of the original model. The number of vertices used in the fitting process is now reduced to n=12, 964 vertices, a sample every ≈ 2.6 mm^2 of the face area. Note that we do not change or recompute the PCA model, we simply select a set of vertex indices.

4.3.2 Iterative Face Fitting

With the defined distance measure for an instance of our compressed morphable face model, the m-dimensional space can be searched for the optimal instance. The fitting is done by choosing a set of m weights w_i, adjusting the position of the instance's vertices according to $S_{\text{inst}} = \bar{S} + \sum_{i=1}^{m} w_i \sigma_i s_i$, measuring the RMS-distance of the new instance to the scan data, selecting new weights and continue until the optimal instance is found. Knowing that each instance is evaluated

using a large number of vertices, an exhaustive search for the optimal set of *m* weights is too computationally expensive.

A common method to solve large combinatorial optimization problems is *simulated annealing* (SA) (Kirkpatrick et al., 1983). In our case, random *m*-dimensional vectors could be generated which represent different *morphs* for a current face instance. A morph that brings the current instance closer to the scan data is accepted (downhill), and otherwise it is either accepted (uphill to avoid local minima) or rejected with a certain probability. In each iteration, the length of the *m*-dimensional morph vector can be reduced as implementation of the "temperature" scheme. The problem with such a naïve SA approach is that most random *m*-dimensional morph vectors are uphill. In particular, close to the optimal solution, a morph vector is often rejected, which makes it hard to produce an accurate fit. Besides this inefficiency, it doesn't take the eigensystem of the morphable face model into account.

Instead, we use an *iterative downhill walk* along the consecutive eigenvectors from a current instance toward the optimal solution. Starting from the mean face $\bar{S}$ ($\forall_{i=1}^{m} w_i = 0$), try new values for w_1 and keep the best fit, then try new values for w_2 and keep the best fit, and continue until the face is morphed downhill along all *m* eigenvectors. Then iterate this process with a *smaller search space* for w_i. The advantage in computation costs of this method is twofold. First, the discrete number of morphs in the selected search space directly defines the number of rejected morphs per iteration. Second, optimizing one w_i at a time means only a one (instead of *m*) dimensional modification of the current face instance $S_{\text{new}} = S_{\text{prev}} + (w_{\text{new}} - w_{\text{prev}})\sigma_i \mathbf{s}_i$.

Because the first eigenvectors induce the fitting of global face properties (e.g., face height and width) and the last eigenvectors change local face properties (e.g., nose length and width), each iteration follows a global-to-local fitting scheme (see Fig. 4.2). To avoid local minima, two strategies are applied. (1) The selected w_i in one iteration is not evaluated in the next iteration, forcing a new (similar) path through the *m*-dimensional space. (2) The vertices of the morphable face model are uniformly divided over three sets and in each iteration a different set is modified and evaluated. Only in the first and last iteration are all vertices evaluated. Notice that this also reduces the number of vertices to fit and therefore the computation costs.

The fitting process starts with the mean face and morphs in place toward the scan data, which means that the scan data should be well aligned to the mean

face. To do so, the segmented and pose normalized face is placed with its center of mass on the center of mass of the mean face, and finely aligned using the iterative closest point (ICP) algorithm (Besl and McKay, 1992). The ICP algorithm iteratively minimizes the RMS distance between vertices. To further improve the effectiveness of the fitting process, our approach is applied in a *coarse-fitting* and a *fine-fitting* step, see the next subsections.

4.3.3 Coarse Fitting

The more the face scan differs from the mean face $\bar{S}$, the less reliable the initial alignment of the scan data to the mean face is. Therefore, the mean face is coarsely fitted to the scan data by adjusting the weights of the first 10 principal eigenvectors (m_{max}=10) in a single iteration (k_{max}=1) with 10 different values for w_{new}=[-1.35,-1.05,...,1.05,1.35], see Algorithm `model fitting`($\bar{S}$, scan). Fitting the model by optimizing the first 10 eigenvectors results in the face instance S_{coarse}, with global face properties similar to those of the scan data. After that, the alignment of the scan to S_{coarse} is further improved with the ICP algorithm.

4.3.4 Fine Fitting

Starting with the improved alignment, we again fit the model to the scan data. This time the model-fitting algorithm is applied using all eigenvectors (m_{max}=m) and multiple iterations (k_{max}=9). In the first iteration of Algorithm `model fitting` ($\bar{S}$, scan), 10 new weight values w_{new} are tried for each eigenvector, to cover a large range of facial variety. The best w_{new} for every sequential eigenvector is used to morph the instance closer to the face scan. In the following k_{max} -1 iterations only four new weight values w_{new} are tried around w_i with a range w_{range} equal to w_{incr} of the previous iteration. By iteratively searching for a better w_i in a smaller range, the weights are continuously optimized. Local minima are avoided as described in Section 4.3.5. The range of the first iteration and the number of new weights tried in each next iteration were empirically selected as good settings.

4.3.5 Multiple Components

Knowing that the morphable model was generated from 100 3D face scans, an increase of its expressiveness is most likely necessary to cover a large population. To increase the expressiveness, also Blanz and Vetter (1999)

proposed to independently fit different components of the face, namely the eyes, nose, mouth, and the surrounding region. Because each component is defined by its own linear combination of shape parameters, a larger variety of faces can be generated with the same model. The fine fitting scheme from the previous section was developed to be applicable to either the morphable face model as a whole, but also to individual components of this model.

Algorithm 8 Model fitting (S_{inst}, scan)

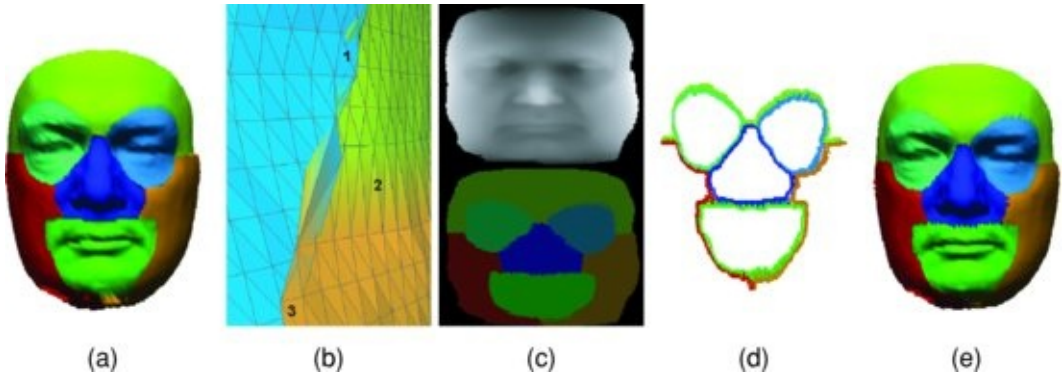
$w_{\text{range}} = 1.5, w_{\text{incr}} = 0.3$
for $k \leftarrow 1$ **to** $k_{\max}$ **do**
  select vertices (uniform subset of component)
  for $i \leftarrow 1$ **to** $m_{\max}$ **do**
    $w_{\min} = w_i - w_{\text{range}} + \frac{1}{2} w_{\text{incr}}$
    $w_{\max} = w_i + w_{\text{range}} - \frac{1}{2} w_{\text{incr}}$
    for $w_{\text{new}} \leftarrow w_{\min}$ **to** $w_{\max}$ **do**
      morph S_{inst} with w_{new}
      $d_{\text{rms}}(S_{\text{inst}}, \text{scan})$ smaller $\rightarrow$ keep w_{new}
      undo morph
      $w_{\text{new}} = w_{\text{new}} + w_{\text{incr}}$
    morph S_{inst} with $w_i \leftarrow$ best w_{new}
  $w_{\text{range}} = w_{\text{incr}}, w_{\text{incr}} = \frac{1}{2} w_{\text{incr}}$
return S_{inst}

Component selection. All face instances generated with the morphable model are assumed to be in correspondence, so a component is simply a subset of vertices in the mean shape $\bar{S}$ (or any other instance). We define seven components in our adjusted morphable face model (see Fig. 4.3). Starting with the improved alignment, we can individually fit each of the components to the scan data using the fine-fitting scheme, obtaining a higher precision of the fitting process (as shown in Sect. 9). Individual components for the left and right eyes and cheeks were selected, so that our method applies to non symmetric faces as well. For comparison, we also used a face model with four components, one for the eyes, one for the nose, one for the mouth, and one for the forehead and cheeks. The use of multiple components has no influence on the fitting time, because the total number of vertices remains the same and only the selected vertices are modified and evaluated.

Component blending. A drawback of fitting each component separately is that inconsistencies may appear at the borders of the components. During the fine fitting, the border triangles of two components may start to intersect, move apart, or move across (Fig. 4.3). The connectivity of the complete mesh remains the same, so two components moving apart remain connected with elongated

triangles at their borders. We solve these inconsistencies by means of a post processing step, as described in more detail below.

Figure 4.3 Multiple components (a) may intersect (b1), move apart (b2), or move across (b3). (c) Simulating a cylindrical scan and (d) smoothing the new border vertices (e) solves these problems



Knowing that the morphable face model is created from cylindrical range scans and that the position of the face instance does not change, it is easy to synthetically rescan the generated face instance. Each triangle of the generated face instance S_{fine} is assigned to a component (Fig. 4.3**a**). A cylindrical scanner is simulated, obtaining a cylindrical depth image $d(\theta, y)$ with a surface sample for angle θ, height *y* with radius distance *d* from the y-axis through the center of mass of $\bar{S}$ (Fig. 4.3**c**). Basically, each sample is the intersection point of a horizontal ray with its closest triangle, so we still know to which component it belongs. The cylindrical depth image is converted to a 3D triangle mesh by connecting the adjacent samples and projecting the cylindrical coordinates to 3D. This new mesh S'_{fine} has a guaranteed resolution depending on the step sizes of θ and *y*, and the sampling solves the problem of intersecting and stretching triangles. However, ridges may still appear at borders where components moved across. Therefore, Laplacian smoothing is applied to the border vertices and their neighbors (Fig. 4.3**d**). Laplacian smoothing moves each vertex toward the center of mass of its connected vertices. Finally, data further then 110 mm from the tip of the nose is removed to have the final model S_{final} (Fig. 4.3**e**) correspond to the segmented face. In Section 4.3.6, we evaluate both the single and multiple component fits.

4.3.6 Results

In this section, we evaluate our fitting results for the UND, GAVAB, and BU-

3DFE data sets. We perform a qualitative and quantitative evaluation of the acquired model fits and compare the results with other model-fitting methods. To prove that the use of multiple components improves the fitting accuracy over a single component, we compare the quantitative measures and relate the fitting accuracy to face recognition by applying different face-matching algorithms to the produced fits. By applying and comparing different face-matching methods, we end up with a complete 3D face recognition system with high recognition rates for all three data sets.

Starting with the 3D face scans from a data set, we apply our face segmentation method (Section 4.2). The presented face segmentation method correctly normalized the pose of all face scans and adequately extracted the tip of the nose in each of them. For the 953 scans of the UND face set, we evaluated the tip of the nose extraction by computing the average distance and standard deviation of the 953 automatically selected nose tips to our manually selected nose tips, which was 2.3 $\pm$ 1.2 mm. Because our model-fitting method aligns the face scan to the mean face, and at a later stage to the coarsely fitted face instance, these results are good enough.

We evaluated the face model fitting as follows. Each segmented face was aligned to $\bar{S}$ and the coarse-fitting method of Section 4.3.3 was applied. After the improved alignment of the scan data to S_{coarse}, the fine-fitting method of Section 4.3.4 was applied to either the entire face or to each of the individual components. For a fair comparison the same post-processing steps (Sect. 8) were applied to the final S_{fine} instances. Figure 4.4 shows qualitative better fits when multiple components are used instead of a single component. Globally, by looking at the more frequent surface interpenetration of the fitted model and face scan, which means a tighter fit. Locally, by looking at facial features, such as the nose, lips, and eyes. Note that our fitting method correctly neglects facial hair and interpolates holes, which is often a problem for 3D face recognition methods.

Figure 4.4 Fitted face models S_{final} based on a single component (1st and 3rd column) and multiple components (2nd and 4th column) to scan data in blue. Results from the front and side view, show a qualitative better fit of the multiple components to the scan data

To quantitatively evaluate the produced fits, we determined the RMS distance (Eq. 4.2) for each of the fitted models to their face scan $d_{\mathrm{rms}}(S_{\mathrm{final}}, \mathrm{scan})$. To report merely the measurements in overlapping face regions, the points paired with boundary points are not included. Also outliers, point-pairs with a distance larger than 10 mm, are not taken into account. The RMS errors are shown in Table 4.1. Note that the UND scans have a higher resolution and thus smaller point-to-point distances, the RMS distances are therefore lower for the UND set than for the GAVAB and BU-3DFE sets.

Table 4.1 RMS errors (mm) of output models to input scans

Data sets	model	min	max	mean	sd
UND	1 comp	0.51	1.73	0.79	0.13
	4 comp	0.39	1.57	0.66	0.11
	7 comp	0.39	1.50	0.64	0.10
GAVAB	1 comp	0.94	3.45	1.22	0.19
	4 comp	0.80	2.30	1.06	0.14
	7 comp	0.80	2.24	1.05	0.14
BU-3DFE	1 comp	0.92	1.97	1.20	0.18
	4 comp	0.87	1.77	1.09	0.15
	7 comp	0.87	1.78	1.08	0.16

Blanz *et al.* (2007) reported the accuracy of their model-fitting method using the average depth error between the depth images of the input scan and the output model, neglecting point-pairs with a distance larger than 10 mm. To compare the accuracy of our method with their method, we produced cylindrical depth images (as in Fig. 4.3**c**) for both the segmented face scan and the fitted model and computed the average depth error $|d_{\mathrm{scan}}(\theta, y) - d_{\mathrm{final}}(\theta, y)|$, excluding the outliers. Because of the surface mesh resampling, these projection errors (Table 4.2) are resolution independent. The GAVAB set has more acquisition artifacts causing higher projection errors, with high maximum projection errors in particular. The available BU-3DFE scans were heavily smoothed right after the acquisition process, causing lower projection errors than the high resolution UND scans.

Table 4.2 Projection errors (mm) of output models to input scans

Data sets	model	min	max	mean	sd
UND	1 comp	0.40	1.65	0.65	0.15
	4 comp	0.28	1.33	0.47	0.11
	7 comp	0.26	1.20	0.43	0.10
GAVAB	1 comp	0.49	3.03	0.79	0.20
	4 comp	0.35	1.91	0.55	0.14
	7 comp	0.35	1.69	0.53	0.12
BU-3DFE	1 comp	0.37	1.58	0.63	0.17
	4 comp	0.25	0.99	0.43	0.10
	7 comp	0.23	0.92	0.39	0.11

The errors reported in Tables 4.1 and 4.2 are all in favor of the multiple component fits with an average RMS gain of 0.2 mm per point pair. However, only a marginal gain in accuracy is acquired when seven components are used instead of four. So, with the use of multiple components we can increase the model's expressiveness to some extend.

Comparison. Blanz *et al.* (2007) reported a mean depth error over 300 UND

scans of 1.02 mm when they neglected outliers. For our fitted single component to UND scans the error $d_{\text{avr.depth}}$ is 0.65 mm, which is already more accurate. For the fitted multiple components these errors are 0.47 and 0.43, for four and seven components, respectively.

Our time to process a raw scan requires ca. 3 seconds for the face segmentation, ca. 1 second for the coarse fitting, and ca. 30 seconds for the fine fitting on a Pentium IV 2.8 GHz. Blanz method reported ca. 4 minutes on a 3.4 GHz Xeon processor, but includes texture fitting as well. Huang *et al.* (2006) report for their deformation model a matching error of 1.2 mm after a processing time of 4.6 minutes. Recently, Amberg *et al.* (2008) proposed a competitive fitting time of 40 to 90 seconds for their face model with 310 model coefficients and 11.000 vertices.

4.4 Dynamic Model Expansion

For a statistical face model to be applicable in face recognition systems all over the world, it is important to include example data on all possible face variations. Because this is an intractable task, a flexible model is required that updates itself in case of new example faces. For that, a system should automatically fit the face model to face data, estimate dense and accurate correspondences beyond the linear combinations of current example data, and measure the redundancy of the new example faces.

When full correspondence between the face model and the scan data is established and the new face instance is not redundant, it can be added as a new example to the statistical face model, increasing the descriptiveness of the model. The process of using a statistical model to enhance itself automatically, is referred to as *bootstrapping* the synthesis of the model. The difficulty of bootstrapping is that (1) if the model (as is) fits a new example well, there is no use of adding the new example to the model. This must be automatically verified. (2) If the model doesn’t fit the new example, the correspondences are incorrect and the example cannot be added to the model. (3) It should be fully

automatic. Nowadays, several statistical models are available, ready to be used and reused. In this chapter, we present a bootstrapping algorithm on the basis of an initial statistical model, which automatically fits to new scan data with noise and holes, and which is capable of measuring the redundancy of new example faces.

The need for bootstrapping statistical models was posed by Vetter *et al.* (1997). They introduced a bootstrapping algorithm for statistical models, and showed that the use of merely an optic flow algorithm was not enough to establish full correspondence between example faces and a reference face. Instead, they attain an effective bootstrapping algorithm by iteratively fitting the face model, applying the optic flow algorithm, and updating the face model. They also used this bootstrapping algorithm to build a 3D morphable face model (Blanz and Vetter, 1999). Their bootstrapping algorithm works well in case of input data with constant properties, but fails when input data is incomplete and when the optic flow algorithm fails. To bootstrap the 3D morphable face model with more general face data, (Basso et al., 2006) added a smoothness term to regularize the positions of the vertices where the optic flow correspondence is unreliable. In case a 3D morphable face model is not yet available, a reference face can be used as an approximation instead, which is a major advantage.

Amberg *et al.* (2007) proposed a non rigid ICP algorithm to establish dense correspondences between a reference face and face scans, but they need an initial rigid transformation for the reference face on the basis of 14 manually selected landmarks. Afterwards, the reference face and the fitted face instances can be used to construct a new morphable face model.

Basso and Verri (2007) fit the morphable face model to scan data using implicit representations. They also use multiple components and blend the implicit functions at the borders of components, but they lose the full point-to-point correspondence in the process. So the fitted examples cannot be added to the morphable model.

Huang *et al.* (2006) proposed a global-to-local deformation framework to deform a shape with an arbitrary dimension (2D, 3D or higher) to a new shape of the same class. Their method also operates in the space of implicit surfaces, but uses a non statistical deformation model. They show their framework’s applicability to 3D faces, for which they deform an incomplete source face to a target face.

The use of multiple components has been used by to improve the face model fitting by Blanz and Vetter (1999) and for face recognition purposes by Blanz

and Vetter (2003), but, so far, the resulting face instances were not accurate enough to be incorporated in the statistic model. The explicit point-to-point correspondences of the fitted face instance and the statistical model had to be established by techniques on the basis of optic flow or non rigid ICP.

In this chapter, a set of predefined face components was used to increase the descriptiveness of a 3D morphable face model. With the use of multiple components, a tighter fit of the face model was obtained and higher recognition rates were achieved. However, by fitting each component individually, components started to intersect, move apart, or move across. So, afterwards the full point-to-point correspondences between the morphable model and the fitted instance were distorted. The post processing method to blend the borders of the components introduces a new set of surface samples without correspondence to the model either.

We present a new bootstrapping algorithm that can be applied to general 3D face data. Our algorithm automatically detects if a new face scan cannot be sufficiently modeled, establishes full correspondence between the face scan and the model, and enhances the model with this new face data. Without the use of unreliable optic flow (Basso et al., 2006) or semi-automatic non rigid ICP (Amberg et al., 2007), we are able to bootstrap the 3D morphable face model with highly accurate face instances. As a proof of concept, we (1) fit the initial morphable face model to several 3D face scans using multiple components, (2) blend the components at the borders such that accurate point-to-point correspondences with the model are established, (3) add the fitted face instances to the morphable model, and (4) fit the enhanced morphable model to the scan data as one single component. In the end, we compare each single component fit obtained with the enhanced morphable model to the single component fit obtained with the initial morphable model. Qualitative and quantitative evaluation shows that the new face instances have accurate point-to-point correspondences that can be added to the initial morphable face model. By comparing the multiple and single component fit, our bootstrapping algorithm automatically distinguishes between new face data to add and redundant data to reject. This is important to keep both the model fitting and the face recognition with model coefficients time-efficient.

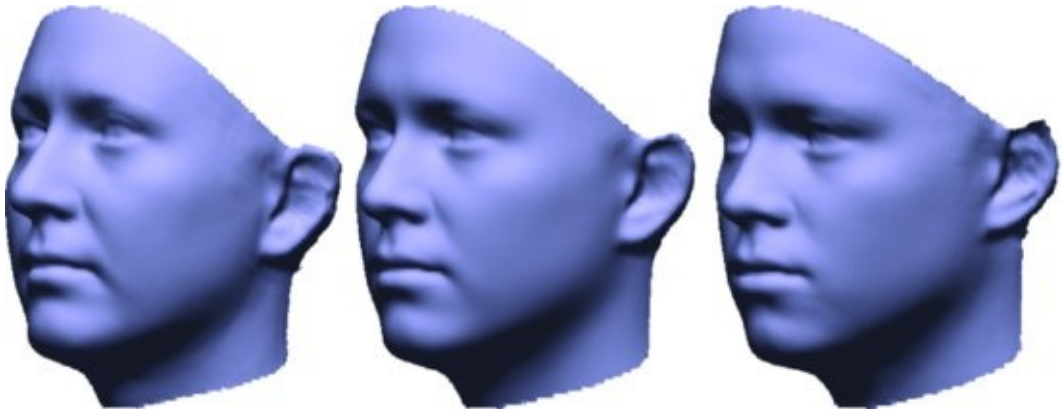
4.4.1 Bootstrapping Algorithm

We fit the morphable face model to 3D scan data to acquire full correspondence

between the scan and the model. We crop the morphable face model and lower its resolution so that n=12, 964 vertices remain for the fitting. We use the new set of correspondences $S = (x_1, y_1, z_1, \ldots, x_n, y_n, z_n)^T \in \Re^{3n}$ to automatically bootstrap the model, in order to increase its expressiveness.

Figure 4.5 shows the changes of the original face model when the weight of the third eigenvector (w_3) is varied. It can be noticed that the model is tilted upwards and downwards. This variation in one of the first eigenvectors means that the alignment of the 100 sets S_i is not optimal for face identification using model coefficients. To adjust the original model, we realigned each reduced face shape S_i to the mean face $\bar{S}$ of the morphable model using the ICP algorithm, and recomputed the PCA model. Visual inspection of our newly constructed PCA model showed no signs of pose variations.

Figure 4.5 Changing weight w_3 {−2,0,+2} causes an unwanted change in the gaze direction. Copyright © 2009, IEEE



The main problem in bootstrapping the 3D morphable face model, is that (1) we only want to add example faces that are not covered by the current model, (2) new example faces suffer from noise and missing data, which makes it hard to establish the point-to-point correspondences, and (3) it should be fully automatic. To establish point-to-point correspondences between the 3D morphable face model and new face data with noise and missing data, we apply our model-fitting method described in Section 4.3. This method either fits the model as a single component or as a set of predefined components. In case the model is fitted as a single component, the final model fit is in full point-to-point correspondence with the face model, but adds no additional information to the current face model. In case the model is fitted as a set of predefined components, this method produces model fits that go beyond the current statistics of the face model, but the point-to-point correspondence are inaccurate or lost. In this

section, we briefly describe the used model-fitting method, then we explain our algorithm to establish dense point-to-point correspondences between the multiple component fits and the morphable face model, and finally, we explain how the bootstrapping algorithm can distinguish between new face data to add to the model and redundant data to reject.

Model Fitting

The morphable face model that we use (see Section 4.4) has m=99 coefficients that can be varied to fit the face model to new scan data. To fit the model, we bring a new 3D face scan into alignment with the 3D face model automatically. First, we automatically normalize the pose of the face, detect the tip of the nose, and segment the face as described in Section 4.2. Secondly, we align the face scan to the face model, coarsely by using their nose tips and more accurately using the ICP algorithm. Then we apply the model-fitting algorithm as described in Section 4.3, using a set of four face components. This model-fitting algorithm iteratively adjusts the m coefficients w_i for each component, such that the vertices move closer to the vertices of the scan data. After all components are fitted to the scan data individually, an accurate representation of the new face S_{fine} is acquired. Each component can then be described using the set of m coefficients w_i for the eigenvectors of the face model. However, the fitted face instance S_{fine} may show artifacts at the borders of these fitted component. Note that we use the same PCA model for each component, but with a subset of the model's vertices only.

Correspondence Estimation

After the application of the model-fitting method, most of the face model's vertices are brought into correspondence with the face scan, but at the component's borders these point-to-point correspondences are less accurate. Only in highly exceptional cases, borders are good enough to bootstrap the face model with S_{fine} directly. To resolve the artifacts at the borders of the individually fitted components, we use their sets of m coefficients w_i that were used to obtain each component. In fact, each set of coefficients can be used to acquire a full face instance of which the component is simply a predefined subset of vertices. We refer to such a full face instance as S_{comp}. Because we fitted a set of c=4 components, we have c=4 face instances S_{comp}. So the face instance S_{fine} is basically a composition of the c intermediate face instances (Fig. 4.6). To

blend the *c* components, we blend the vertices of the *c* face instances. In this process, the goal is to determine for each vertex in S_{fine} a new position, such that it has a smooth transition to its neighboring vertices. Once we reach this state, we refer to this final face instance as S_{final}.

Figure 4.6 Model Fitting. Four face instances S_{comp}, are constructed during the model-fitting process each with the focus on a smaller subset of vertices. The composition of these components provides an accurate fit. Copyright © 2009, IEEE

Because components can overlap more than several millimeters, Laplacian smoothing of the vertices at the borders would not suffice. The selection of a larger region of triangles and vertices to smooth causes a non statistical shape deformation and local features may not be preserved. In particular, the places were three or more components overlap, it is hard to regularize the vertex positions using smoothing techniques. Surface stitching techniques as by Turk and Levoy (1994) could be applied to stitch the components, but this would distort the point-to-point correspondences. Mesh fairing using Gaussian curvatures, as done by Zhao and Xu (2006) for instance, could smooth some of the border triangles properly. However, these techniques focus on the preservation of sharp features, whereas we want to remove sharp creases caused by overlapping components.

To repair the discontinuities at the borders, we have developed an algorithm that uses the vertex positions of the intermediate face instances S_{comp}, local neighborhood properties and the PCA model. Because all instances S_{comp} were acquired with the same PCA model, their vertices are all in full correspondence. In case two connected vertices cause a discontinuity in S_{fine}, one can assume that moving each vertex toward the center of mass of its adjacent vertices (as in Laplacian smoothing) slightly improves on the situation. However, propagating this smoothing locally causes the border discontinuities to attract well positioned vertices instead of solving the discontinuity. Propagating such smoothing globally, changes the statistical shape of the face (Fig. 4.8 **h**). Instead, we morph

each vertex $S_{\text{fine}}[v_i]$ toward its c corresponding positions $S_{\text{comp}}[v_i]$. For that we need to (1) detect the local distortions, (2) know which components lie close to which vertices, (3) have for each vertex a weight for each close by component, and (4) recompute the vertex positions.

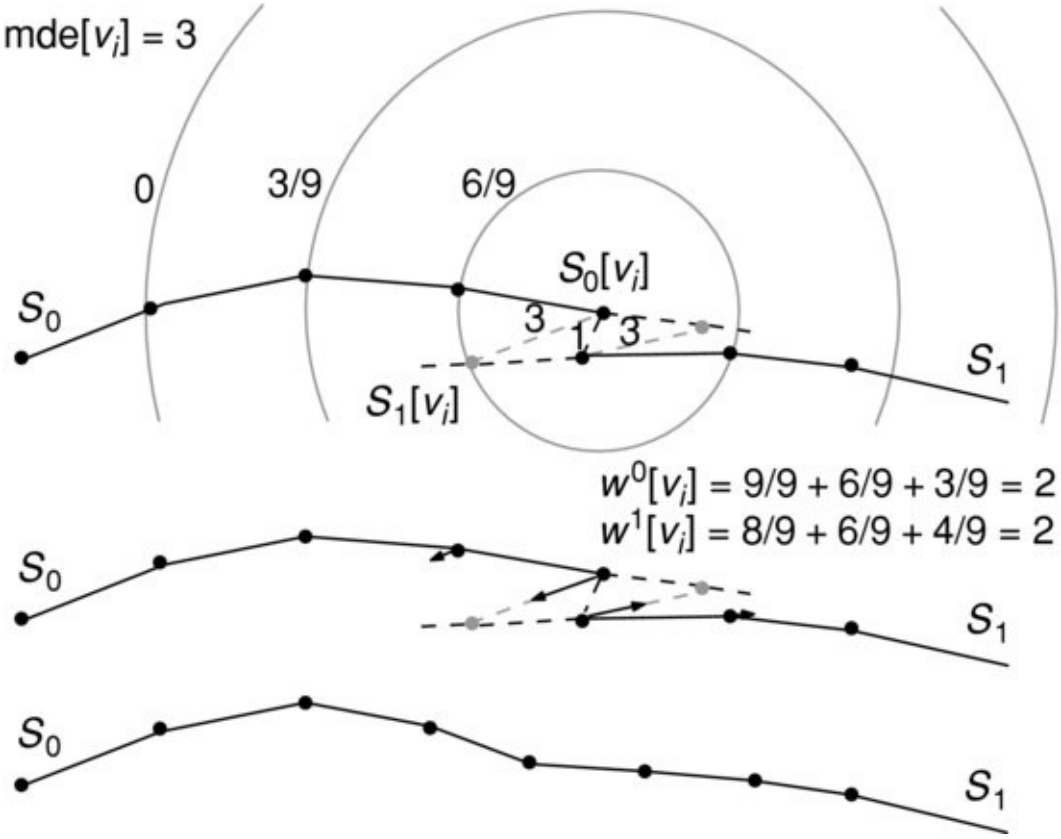
First, to detect the local distortion, we determine for each vertex $S_{\text{fine}}[v_i]$ its *maximum displacement error* to one of its c corresponding positions $S_{\text{comp}}[v_i]$, using

$$\text{mde}[v_i] = \max_{c \in comp} (e(S_{\text{fine}}[v_i], S_c[v_i])),$$

where $e(p, q)$ is the Euclidean distance between two 3D coordinates p and q. When a vertex has approximately the same 3D position in all face instances S_{comp}, its displacement error is small. In that case, we have consensus on its position and thus less need for a repositioning. Second, close by vertices are selected ($S_{\text{fine}}[v_j]$) using a sphere of radius $r[v_i]$ around $S_{\text{fine}}[v_i]$, where radius $r[v_i]$ equals the displacement error times three, $r[v_i] = 3 \cdot \text{mde}[v_i]$. For a vertex $S_{\text{fine}}[v_i]$ on a component border, this set of close-by vertices include vertices from different components. If all close by vertices belong to the same component as the current vertex nothing will change. Third, each close by vertex $S_{\text{fine}}[v_j]$ adds a weighted vote for the component it belongs to. This weight decreases linearly with the distance of vertex $S_{\text{fine}}[v_j]$ to $S_{\text{fine}}[v_i]$. The maximum weight of one is for the vertex $S_{\text{fine}}[v_i]$ itself, and decreases linearly to zero for radius $r[v_i]$. Because components can move away from each other, radius $r[v_i]$ must be at least two times larger than the maximum displacement error, otherwise nothing changes. In the end, a vertex next to the border of the two components, has a number of close by vertices that vote for its own component and a number of close by vertices that vote for the other component. Then, vertex $S_{\text{fine}}[v_i]$ is morphed toward its corresponding position $S_{\text{comp}}[v_i]$ in the other component according to these weighted votes. For example, if a vertex (weighted vote of 1) has four close by vertices with weighted votes of 0.25 of which two vertices lie on an other component S_{comp}, then we have a 3-to-1 vote, and vertex $S_{\text{fine}}[v_i]$ if morphed for 25% toward $S_{\text{comp}}[v_i]$. Another example in 2D is shown in Figure 4.7. In case close by vertices belong to three or more components, the vertex $S_{\text{fine}}[v_i]$ is morphed to three or more different positions $S_{\text{comp}}[v_i]$.

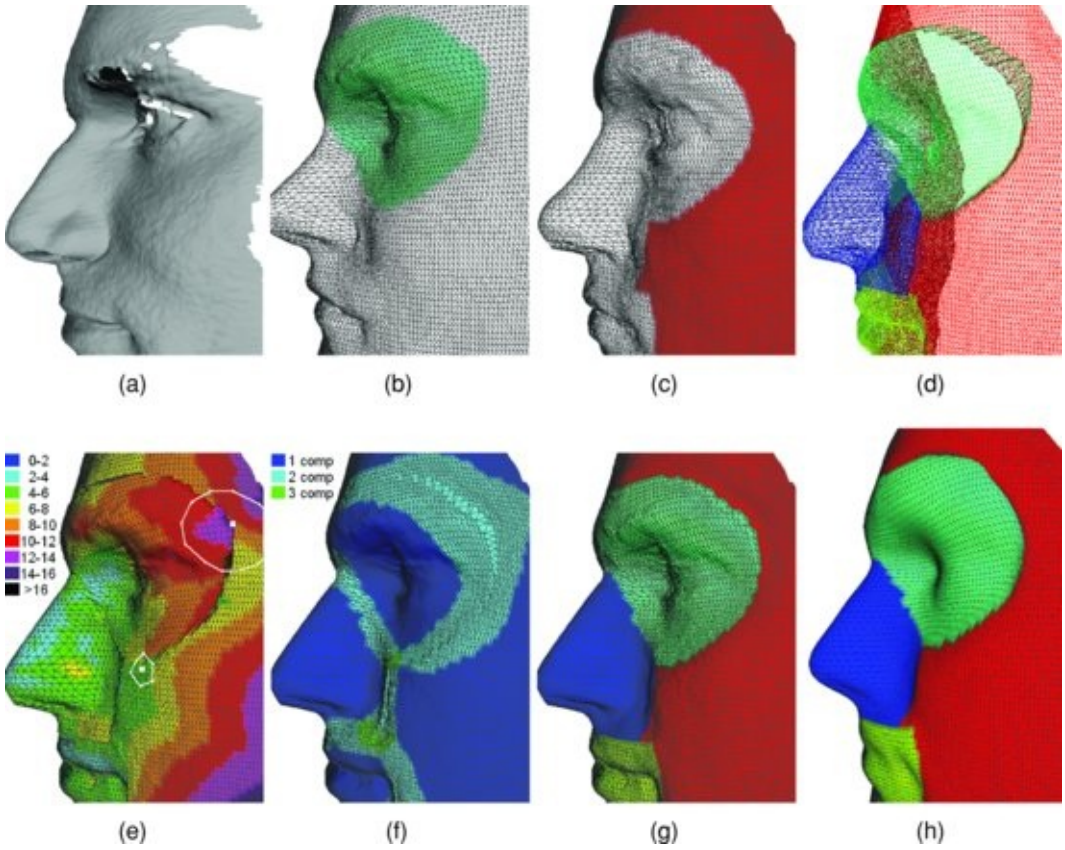
Figure 4.7 Repairing inconsistencies at the borders of components. Vertex $S_0[v_i]$ is moved toward its corresponding point $S_1[v_i]$ using a weighted voting scheme. Close by vertices in both components S_0 and S_1 are used to compute the

appropriate weight. Copyright © 2009, IEEE



This *weighted voting scheme* for the local geometry of a face instance S_{fine}, results in a proper blending of the components. Because the overall topology of the face instance is retained, and the problematic vertices and their local neighborhood are assigned to new positions, the adjusted face instance S_{fine} is now in full correspondence with the face model. Figure 4.8, shows the multiple component fit S_{fine} on the basis of four components S_{comp} (two are shown), the displacement errors, the blending, and the final result S_{final}.

Figure 4.8 Correspondence estimation. By individually fitting components (b,c) to scan data (a), artifacts may occur at the borders of S_{fine} (d). The maximum displacement error (r in mm) for a vertex is an indication of the local mesh distortion (e), toward purple/black means a higher distortion. Using this error, surrounding vertices are selected that all contribute to the voting for a new position of the vertex. After the voting, each vertex is repositioned on the basis of the weighted votes for close by components (f). The final model fit S_{final} (g) presents smooth transitions between the different components. Five iterations of Laplacian smoothing (h) was not enough to repair the artifact close to the eyebrow whereas the face already lost most of its detail. Copyright © 2009,



Redundancy Estimation

After the regularization of the vertex positions, the repaired face instance S_{final} can be added to the morphable face model. To do so, we can apply the ICP algorithm to finely align S_{final} to the mean face $\bar{S}$ of the morphable model and include it in the example set of 3D faces from which the morphable face model was built. Then, we can recompute the PCA model using 100+*k* example faces, and keep the *m*+*k* principal eigenvectors and eigenvalues of the face (see Section 4.4). This way the face properties of a new example face can be added to the statistical face model. With this enhanced model, we should be able to produce accurate model fits with only a single component to face scans similar to those *k* example scans. In the end, each face scan can be described using *m*+*k* model coefficients, and face identification can be performed on the basis of such a *m*+*k* feature vector.

The addition of *k* extra example faces to the current face model, causes an increase of computation costs for both the model-fitting and face identification method. So, it is important not to add example faces that are already covered in

the current morphable face model. Therefore, we estimate the redundancy of encountered example data. First, the morphable face model is fitted as a single face component to the 3D scan data, which we refer to as S_{single}. Second, we fit the morphable face model using multiple face components with the improved correspondence estimation described earlier, S_{mult}. After the model-fitting process, a residual error between the vertices of the model fit and the scan data remains. The difference of the two residual errors of S_{single} and S_{mult}, can be used to estimate the redundancy of the new face scan. In case the residual error of S_{single} is significantly larger than that of S_{mult}, then the face scan is most likely not contained in the current morphable face model, and we should add S_{mult} to the model.

To compute the residual error of these model fits we use the RMS distance of closest point pairs,

$$d_{\text{rms}}(S, \text{scan}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_{\text{min}}(p_i, \text{scan})^2} \quad (4.3)$$

using all n vertices of S_{single} and S_{mult}. Closest point pairs (p, p') for which p' belongs to the boundary (including holes) of the face scan, are not used in the distance measure. Figure 4.9 shows for one face scan, the two model fits and their residual error maps. For this example the RMS error for S_{single} is 0.89 mm and for S_{mult} 0.68 mm.

Figure 4.9 Redundancy estimation. From left to right, the face scan, S_{single}, the residual errors of S_{single}, S_{mult}, the residual errors of S_{mult}, and the color map for the point-to-point distances (in mm). Copyright © 2009, IEEE

4.4.2 Results

We fit the morphable face model to 3D scan data from the UND (Chang et al., 2005), GAVAB (Moreno and Sanchez, 2004), BU-3DFE (Yin et al., 2006), Dutch CAESAR-survey (2008), and our local data set. From all except the UND set, we randomly select four scans, giving a first test set of 16 scans. These scans

vary in pose, facial expression, resolution, accuracy, and coverage. This set of 16 face scans is used to test our bootstrapping algorithm. To test the automatic redundancy check, we use a subset of 277 face scans from the UND data set, namely the first scan of each new subject. To acquire the 3D face data from the scans, we apply the face pose normalization and face segmentation methods described in Section 4.2).

To elaborate on the performance of our bootstrapping algorithm, we applied it to the data set of 16 different face scans and to the subset of 277 UND scans. The small set is used to evaluate the model-fitting and correspondence estimation algorithm. The UND set is used to test the redundancy estimation.

Correspondence Estimation

To evaluate the correspondence estimation, we compare the residual errors of fitted face instances. First, we fit the initial morphable face model as a single component to the segmented face data S_{single}. Second, we fit the initial model to the segmented face data using the four components and blend their borders. Third, we add these 16 face instances S_{mult} to the example set and recompute the PCA model, keeping the m=99 principal components of the face. Finally, we fit the enhanced morphable face model to the same segmented face data using a single component S^{+}_{single}.

In Table 4.3, we report the residual errors of the fitted face instances. We use these errors for quantitative evaluation of the produced fits. For all face scans, S_{mult} has a lower residual error than S_{single}, which means that a higher fitting accuracy is achieved with the use of multiple components in combination with the improved correspondence estimation. After the model was enhanced with the 16 face instances S_{mult}, the model was again fitted as a single component to the 16 face scans. Now, all residual errors are lower for S^{+}_{single} than they were for S_{single}, which means that our bootstrapping algorithm successfully enhanced the morphable face model. For one face scan, the face instance S^{+}_{single} is even more accurate than S_{mult}. This is possible, because we enhanced the model with the facial variety of 16 faces at once. We expect that the residual errors of S^{+}_{single} can be lowered further, by iterating the process of (1) fitting the enhanced model using multiple components, (2) replacing the 16 face instances S^{+}_{mult}, and (3) building a new PCA model. In fact, we tried it for the face scan in Figure 4.9 and

lowered its RMS distance for S_{mult} (0.68 mm) and S^{+}_{single} (0.69 mm) to 0.64 mm for S^{+}_{mult}. So, iteratively replacing the 16 instances of the enhanced model with their improved instances S^{+}_{mult}, will probably help to some extend. Note that the residual errors are lowest for our local set, which contains the highest resolution scans, and the highest errors for the low resolution CAESAR faces. This is as a result of the RMS distance that measures point-to-point distances. Because we are interested in the difference between residual errors, this works fine, otherwise, one could use a surface mesh comparison tool instead. In previous experiments, we used *metro* Cignoni *et al.* (1998) for that.

Table 4.3 RMS errors (mm) of output models to input scans. The model fits with the smallest and largest difference in residual errors (bold) are show in Figure 4.10

Data set	S_{single}	S_{mult}	S^{+}_{single}	$S_{\text{single}} - S_{\text{mult}}$	$S^{+}_{\text{single}} - S_{\text{mult}}$
GAVAB	1.36	1.24	1.27	**0.13**	0.04
GAVAB	1.34	1.16	1.19	0.18	0.03
GAVAB	2.04	1.59	1.56	**0.45**	−0.03
GAVAB	1.32	1.16	1.19	0.17	0.03
BU-3DFE	1.18	1.01	1.02	0.18	0.02
BU-3DFE	1.28	1.12	1.15	0.16	0.03
BU-3DFE	1.06	0.96	0.96	**0.10**	0.00
BU-3DFE	1.61	1.40	1.49	**0.21**	0.09
local	0.70	0.55	0.58	0.15	0.03
local	0.89	0.68	0.69	**0.21**	0.01
local	0.79	0.62	0.67	0.17	0.05
local	0.69	0.55	0.58	**0.14**	0.04
CAESAR	1.86	1.77	1.80	0.09	0.03
CAESAR	1.88	1.77	1.78	**0.11**	0.01
CAESAR	1.81	1.74	1.77	0.07	0.03
CAESAR	1.80	1.75	1.78	**0.05**	0.03

In Figure 4.10, we show some of the resulting model fits and their distance maps acquired with our bootstrapping algorithm. In this figure, we show the two faces per data set that achieved the smallest and largest difference in residual errors in the same order as in Table 4.3. Visual inspections of the fitted models shows an improved single component fit of the enhanced model to the scan data (S^{+}_{single}), compared with the single component fit using the initial morphable model (S_{single}). This can be seen in the residual error maps as well. That the bootstrapping algorithm successfully incorporated the 16 face scans in the morphable face model, can be seen by face instances S_{mult} and S^{+}_{single}, which are very similar. The initial morphable face model consisted of neutral expression scans only. Nevertheless, the use of multiple components allows for

correspondence estimation among some expression data as well.

Figure 4.10 Automatic correspondence estimation. From left to right, the segmented and pose normalized faces, the single component fit S_{single}, its distance map, the multiple component fit S_{mult}, its distance map, and the single component fit S^{+}_{single} with its distance map. Faces in rows two, four, and five were selected as being new to the model. Copyright © 2009, IEEE

Redundancy Estimation

To distinguish between new and redundant face data, we computed the residual errors for face instances S_{single} and S_{mult} using the RMS distance measure. In Table 4.3, we reported the RMS errors for our set of 16 faces. These differences in RMS error for $S_{\text{single}} - S_{\text{mult}}$ vary between 0.05 and 0.45. The maximum difference of 0.45 was achieved for the sad looking person on row four in Figure

4.10. With the use of a threshold *t* for the difference in RMS error, we decide whether a face is redundant or new. On the basis of the visual inspection of the faces in Figure 4.10, we decided to select *t*=0.17 for our experiments. In case the RMS difference is higher than *t*, we consider a face to be new and otherwise as redundant. With this threshold, we classify only the faces in row two, four, and five as being new.

We applied our bootstrapping algorithm to the 277 UND scans, and let our algorithm automatically select potential faces to add to the model, without actually adding them. This way we can see which face scans (persons) are new to the model. For these persons, we may assume a difficulty in identifying them, because a coefficient-based recognition system may confuse that person with a different person that has those coefficients. Out of the 277 UND scans, 35 scans were found as being new to the system, that is, having a decrease in RMS error of $S_{\text{single}} - S_{\text{mult}}$ higher than threshold *t*. Some of these produced fits are shown in Figure 4.11. Most of the selected faces have indeed new face features and should be added to the morphable face model. However, some of the faces that are covered by facial hair produce less reliable fits. To improve on these fits, one could apply a skin detection algorithm and remove the hair beforehand.

Figure 4.11 Automatic bootstrapping and redundancy estimation. From left to right, the segmented and pose normalized faces, the single component fit S_{single}, its distance map, and the multiple component fit S_{mult} with its distance map. The instances S_{mult} were automatically selected as being new

4.5 Face Matching

Our model-fitting algorithm determines a set of model coefficients that morphs the mean face to a clean model instance that resembles the 3D face scan. On the basis of this model instance, we use three different methods to perform face matching. Two methods use the newly created 3D geometry as input, namely the landmarks-based and contour-based methods. The third method uses the model coefficients as a feature vector to describe the generated face instance.

4.5.1 Comparison

Landmarks. All vertices of two different instances of the morphable model are assumed to have a one-to-one correspondence. Assuming that facial landmarks such as the tip of the nose and corners of the eyes are morphed toward the

correct position in the scan data, we can use them to match two 3D faces. So, we assigned 15 anthropometric landmarks to the mean face and obtain their new locations by fitting the model to the scan data. To match two faces **A** and **B** we use the sets of c=15 corresponding landmark locations:

$$d_{\text{corr}}(\mathbf{A}, \mathbf{B}) = \sum_{i=1}^{c} d_p(a_i, b_i), \quad (4.4)$$

where distance d_p between two correspondences a_i and b_i is the squared difference in Euclidean distance e to the nose tip landmark p_{nt}

$$d_p(a_i, b_i) = (e(a_i, p_{nt}) - e(b_i, p_{nt}))^2. \quad (4.5)$$

Contour curves. Another approach is to fit the model to scans **A** and **B** and use the new clean geometry as input for a more complex 3D face recognition method. To perform 3D face recognition, we extract from each fitted face instance three 3D facial contour curves, and match only these curves to find similar faces, see ter Haar and Veltkamp (2009).

Model coefficients. The iterative model-fitting process determines an optimal weight w_i for each of the m eigenvectors. These weights, or model coefficients, multiplied by σ describe a path along the linearly independent eigenvectors through the m dimensional face space. For two similar scans one can assume these paths are alike, which means that the set of m model coefficients can be used as a feature vector for face matching.

In the case of multiple components, each component has its own set of m model coefficients. Blanz *et al.* (2007) simply concatenated sets of model coefficients into a single coefficient vector. Here, we also concatenate the coefficient vectors of multiple components. To determine the similarity of faces with these coefficient vectors, we use four distance measures, namely the L_1 and L_2 distance before and after normalizing the vector's length. Amberg *et al.* (2008) assume that caricatures of an identity lie on a vector from the origin to any identity and use the angle between two coefficient vectors as a distance measure. Normalizing the length of the coefficient vectors and computing the L_2 distance has this same effect and results in the same ranking of retrieved faces.

4.5.2 Results

We can use the morphed face instances to perform 3D face recognition. For this experiment, we computed the 953 × 953, 244 × 244, and 700 × 700 dissimilarity matrices and sorted the ranked lists of face models on decreasing similarity.

From these ranked lists, we computed the recognition rate (RR), the mean average precision (MAP) and the verification rate at 0.1% false acceptance rate (VR@0.1%FAR). A person is recognized (or identified) when the face retrieved on top of the ranked list (excluding the query) belongs to the same subject as the query. For 77 subjects in the UND set only a single face instance is available which cannot be identified, so for this set the RR is based on the remaining 876 queries. The mean average precision (MAP) of the ranked lists are also reported, to elaborate on the retrieval of all relevant faces, that is, all faces from the same subject. Instead of focusing on 3D face retrieval application, one could use 3D face matching for imposter detection as well. For an imposter/client detection system, all face matches with a dissimilarity above a carefully selected threshold are rejected. Lowering this threshold means that more imposters are successfully rejected, but also that less clients are accepted. We use the dissimilarity threshold at which the false acceptance rate is 0.1%, which is also used in the face recognition vendor test. Because the VR@0.1%FAR depends on similarity values, it is not only important to have relevant faces on top of the ranked lists, but also that their similarity values are alike and differ from irrelevant faces.

Because the VR@0.1%FAR evaluation measure depends on the acquired similarity values, there are several ways to influence this measure. Rank aggregation with the use of Consensus Voting or Borda Count (Faltemier et al., 2008), for instance, reassigns similarity values on the basis of the ranking. This way one can abstract from the actual similarity values, which allows for the selection of a different imposter threshold and change the VR@0.1%FAR. Of course, a rank-based threshold cannot be used in case of a one-to-one face matching, that is, a scenario in which someone's identity must be confirmed or rejected. The application domain for rank-based measures is the one-to-many face matching, that is, a scenario in which we search for the most similar face in a large database.

In case of the face matching based on model coefficients, we assume that caricatures of an identity lie on a vector from the origin to any identity. If we normalize the lengths of these vectors, we neglect the caricatures and focus on the identity. This normalization step also regulates the similarity values and thus influences the VR@0.1%FAR. In Table 4.4, we report the face-matching results on the basis of the L_1 and L_2 distances between coefficient vectors, before and after length normalization. Remarkable is the significant increase of the VR@0.1%FAR for the normalized coefficient vectors, whereas the rankings are similar as shown by the MAPs. Although we show in Table 4.4 only the results

for the face model fitting using seven components, it also holds for the one and four component case. Because the L_1 distance between normalized coefficient vectors slightly outperforms the L_2 distance measure, we use this measure whenever we evaluate the performance of model coefficients.

Table 4.4 The effect of normalizing coefficient vectors to unit length

Data set	measure	RR	MAP	VR@0.1%FAR
UND	L1	1.00	0.99	0.94
	L2	1.00	0.99	0.94
	L1 norm*	1.00	1.00	0.99
	L2 norm*	1.00	1.00	0.98
GAVAB	L1	0.97	0.91	0.69
	L2	0.95	0.90	0.69
	L1 norm*	0.97	0.94	0.85
	L2 norm*	0.97	0.93	0.84
BU-3DFE	L1	0.99	0.90	0.59
	L2	0.99	0.88	0.56
	L1 norm*	0.98	0.94	0.80
	L2 norm*	0.97	0.93	0.78

*A significant increase of the VR@0.1%FAR is achieved by matching the normalized coefficient vectors.

Face retrieval and verification results based on anthropometric landmarks, contour curves, and model coefficients are shown in Table 4.5. To each set of face scans we fitted the morphable face model using one, four, and seven components. Each fitted component produces a 99 dimensional model coefficient vector with a different face instance as a result. The performance of our face matching depends on both the number of components as well as the applied feature set. The two main observations are that (1) the coefficient-based method outperforms the landmark-and contour-based methods, and (2) that the use of multiple components can improve the performance of 3D face matching. In the next paragraphs, we elaborate on these observations.

Table 4.5 Performance measures based on landmarks, contour curves, and coefficient vectors for single and multiple component fits

Data set	features	model	RR	MAP	VR@0.1%FAR
UND	landmarks	1 comp	0.85	0.86	0.71
	landmarks	4 comp	0.90	0.89	0.74
	landmarks	7 comp	0.89	0.90	0.77
	contours	1 comp	0.95	0.94	0.85
	contours	4 comp	0.97	0.92	0.98
	contours	7 comp	0.98	0.97	0.92
	coefficients	1 comp	0.98	0.98	0.95
	coefficients	4 comp	1.00	1.00	0.98
	coefficients	7 comp	1.00	1.00	0.99
GAVAB	landmarks	1 comp	0.72	0.73	0.46
	landmarks	4 comp	0.60	0.66	0.38
	landmarks	7 comp	0.64	0.67	0.43
	contours	1 comp	0.91	0.86	0.67
	contours	4 comp	0.82	0.79	0.58
	contours	7 comp	0.83	0.80	0.59
	coefficients	1 comp	0.96	0.93	0.82
	coefficients	4 comp	0.98	0.95	0.88
	coefficients	7 comp	0.97	0.94	0.85
BU-3DFE	landmarks	1 comp	0.61	0.49	0.27
	landmarks	4 comp	0.63	0.50	0.28
	landmarks	7 comp	0.66	0.54	0.31
	contours	1 comp	0.84	0.66	0.43
	contours	4 comp	0.88	0.67	0.43
	contours	7 comp	0.88	0.67	0.44
	coefficients	1 comp	0.96	0.88	0.73
	coefficients	4 comp	0.98	0.92	0.78
	coefficients	7 comp	0.98	0.94	0.80

The automatically selected anthropometric landmarks have a reasonable performance on the UND face scans, but are not reliable enough for effective 3D face matching in the two other sets. The contours perform well for retrieval and verification purposes in the UND face set. However, their performance drops significantly for the other two sets, because the contour curves cannot be effectively used in case of facial deformations. The use of the model coefficients consistently outperforms both the landmark-based and contour-based face matching. Besides the difference in performance, the three methods differ in running time as well. The landmark-based method matches two faces using only 15 coordinates, whereas the contour-based method matches two faces using 135 coordinates. The coefficient-based method matches faces using 99 weights times the number of fitted components. So, the coefficient-based method using four components has approximately the same running time as the contour-based method.

The observation that multiple (four or seven) components increases the performance of our face matching holds for all results except the landmark-and

contour-based methods in the GAVAB set. The problem with this set is that a low quality scan of a person looking up or down causes artifacts on and around the nose. In such cases a more accurate fit of the face model's nose harms, because the performance of landmark-and contour-based methods are heavily dependent on an accurate selection of the nose tip. Although the face matching improves from the single to multiple component case, there is no consensus for the four or seven component case. The use of either four or seven components causes either a marginal increase or decrease of the evaluation scores. Although face matching with the use of 1000 model coefficients is usually referred to as time efficient, one could argue for the use four components instead of seven, because the number of coefficients is smaller.

Comparison. Blanz *et al.* (2007) achieved a 96% RR for 150 queries in a set of 150 faces (from the FRGC v.1). To determine the similarity of two face instances, they computed the scalar product of the 1000 obtained model coefficients. In this chapter, we achieved 98% RR on the UND set using the three selected contour curves, and 100% RR with the use of our model coefficients.

4.6 Concluding Remarks

Where other methods need manual initialization, we presented a fully automatic 3D face morphing method that produces a fast and accurate fit for the morphable face model to 3D scan data. On the basis of a global-to-local fitting scheme the face model is coarsely fitted to the automatically segmented 3D face scan. After the coarse fitting, the face model is either finely fitted as a single component or as a set of individual components. Inconsistencies at the borders are resolved using an easy to implement post processing method. Results show that the use of multiple components produces a tighter fit of the face model to the face scan, but assigned anthropometric landmarks may lose their reliability for 3D face identification.

We also presented a method for establishing accurate correspondences among 3D face scans. With the use of an initial morphable face model and a set of predefined components, we are able to produce accurate model fits to 3D face data with noise and holes. Afterwards, the components still have defects on their border, for which we presented a new blending technique that retains the full correspondence between the face instance and the morphable model. These newly generated face instances can be added to the example set from which a

new and enhanced morphable model can be built. We tested our fully automatic bootstrapping algorithm on a set of 16 new face scans, to show that the new morphable face model has indeed an enhanced expressiveness. By adding data to an initial morphable face model, we can fit the model to face scans of a larger population and reduce the confusion among identities. Therefore, the building of a strong morphable face model is essential for the difficult task of 3D face recognition. However, adding data to a morphable model increases the computation costs for both model fitting and face comparison. Therefore, we developed an automatic redundancy check, which discriminates between new face data to add to the model and redundant face data to reject. On the basis of the initial 16 scans, we selected a residual error threshold for the automatic redundancy check. Then, we applied our bootstrapping algorithm to 277 UND scans. Our bootstrapping algorithm successfully established full correspondence between these scans and the initial morphable model, and selected 35 persons that were new to the model.

With our new bootstrapping algorithm, we are able to successfully update an initial face model, which we use to produce more accurate fits to new scan data. The algorithm is fully automatic, reuses initial face statistics, checks for redundancy, and retains the full correspondence even in case of noisy scan data with holes. The algorithm successfully enhances the neutral morphable face model with new (close to) neutral face scans. It should also apply to, for instance, a surprised face model and surprised scans, but not to a neutral face model and surprised scans. To establish correspondences among face scans with different expressions, new automatic algorithms are required.

Face matching using facial contours shows higher recognition rates on the basis the multiple component fits than for the single component fits. This means that the obtained 3D geometry after fitting multiple components has a higher accuracy. Our face modeling approach in combination with the three selected contour curves achieves 98% RR on the UND set. The obtained model coefficient that were used to produce the accurate face instances, turned out to have the highest performance. With the use of four components, we achieve 100% correct identification for 876 queries in the UND face set, 98% for 244 queries in the GAVAB face set, and 98% for 700 queries in the BU-3DFE face set. These high recognition rates based on normalized coefficient vectors proves that our model-fitting method successfully fits the morphable face model consistently to scan data with varying poses and expressions.

Exercises

1. The pose normalization based on fitting a coarse nose template fails when a scan is made from the side (such that half the nose is occluded), or when the scanning range is not right (such that the nose is not scanned). (a) Think of a way to detect such cases. (b) Develop an alternative way for pose normalization that does not suffer from such cases.

2. When a morphable model is fitted to a frontal face scan, the resulting weight values are not well commensurable to the weight values resulting from fitting a morphable model to a side face scan. Invent a strategy to cope with this situation.

3. Design an experiment to evaluate the effect of different distance measures between two vectors of model-fitting weights.

4. How would you place landmarks to build a morphable model that is suitable for fitting to faces that show surprise, or bitterness?

References

Amberg B, Knothe R, Vetter T. Expression invariant 3D face recognition with a morphable model automatic face and gesture recognition. In: Proceedings of the 8th IEEE International Conference on Automatic Face and Gesture Recognition (FGR'08); 2008 Sept 17–19, Amsterdam, The Netherlands. New York: IEEE. 2008. p. 1–7.

Amberg B, Romdhani S, Vetter T. Optimal step nonrigid ICP algorithms for surface registration. IEEE Conference on Computer Vision and Pattern Recognition, CVPR'07; 2007 Jun 17–22, Minneapolis, MN. New York: IEEE. 2007. p. 1–8.

Basso C, Verri A. Fitting 3D morphable models using implicit representations. Journal of Virtual Reality and Broadcasting 2007;4(18):1–10.

Basso C, Paysan P, Vetter T. Registration of Expressions Data using a 3D Morphable Model. In: Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition (FGR'06); 2006 Apr 2–6, Southampton, UK. New York: IEEE. 2006. p. 205–210.

Berretti S, Del Bimbo A, Pala P, Silva Mata F. Face recognition by matching 2D and 3D geodesic distances. In: Sebe N, Liu Y, Zhuang Y, editors. Proceedings of

the International Workshop on Multimedia Content Analysis and Mining (MCAM'07): Lecture Notes in Computer Science. Volume 4577; 2007 Jun 29–30, Weihai, China. Berlin Heidelberg: Springer. 2007. p. 444–453.

Besl PJ, McKay ND. A method for registration of 3D Shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence 1992;14(2):239–256.

Blanz V, Vetter T. A morphable model for the synthesis of 3D faces. In: Proceedings of the 26th Annual Conference on Computer Graphics, SIGGRAPH 99; 1999 Aug 8–13, Los Angeles, CA. New York: ACM Press. 1999. pp. 187–194.

Blanz V, Vetter T. Face recognition based on fitting a 3D morphable model. IEEE Transactions on Pattern Analysis and Machine Intelligence 2003;25(9):1063–1074.

Blanz V, Scherbaum K, Seidel HP. Fitting a morphable model to 3D scans of faces. In: Proceedings of IEEE 11th International Conference on Computer Vision; 2007 Oct 14–21, Rio de Janeiro. New York: IEEE. 2007. p. 1–8.

Bronstein AM, Bronstein MM, Kimmel R. Three-dimensional face recognition. International Journal of Computer Vision 2005;64(1):5–30.

CAESAR-survey. The Civilian American and European Surface Anthropometry Resource. Available at: http://store.sae.org/caesar, accessed Mar 1, 2013. Warrendale, PA: SAE International; 2008.

Chang KI, Bowyer KW, Flynn PJ. An evaluation of multimodal 2D+3D face biometrics. IEEE Transactions on Pattern Analysis and Machine Intelligence 2005;27(4):619–624.

Cignoni P, Rocchini C, Scopigno R. Metro: Measuring error on simplified surfaces. Computer Graphics Forum 1998;17:167–174.

Davis J, Marschner SR, Garr M, Levoy M. Filling holes in complex surfaces using volumetric diffusion. In: Proceedings of the First International Symposium on 3D Data Processing, Visualization and Transmission; 2002 Jun 19–21, Padua, Italy. New York: IEEE. 2002. p. 428–861. 2002

Faltemier T, Bowyer K, Flynn P. A region ensemble for 3-D face recognition. IEEE Transactions on Information Forensics and Security 2008;1(3):62–73.

Huang X, Paragios N, Metaxas DN. Shape registration in implicit spaces using information theory and free form deformations. IEEE Transactions on Pattern Analysis and Machine Intelligence 2006;28(8), 1303–1318.

Kakadiaris I, Passalis G, Toderici G, Murtuza N, Theoharis T. 3D face

recognition. In: Chantler M, Fisher M, Trucco M, editors. Proceedings of the British Machine Vision Conference; 2006 Sept 4–7, Edinburgh, UK. Edinburgh: BMVA Press. 2006. p. 869–878.

Kirkpatrick S, Gelatt CD, Vecchi MP. Optimization by Simulated Annealing. Science, New Series 1983; 220(4598): 671–680.

Lu X, Jain A. Deformation modeling for robust 3D face matching. IEEE Transactions on Pattern Analysis and Machine Intelligence 2008;30(8):1346–1356.

Moreno AB, Sanchez A 2004a Gavabdb: A 3D face database. In: Proceedings of the 2nd COST 275 Workshop on Biometrics on the Internet; 2004 Mar 25–26, Vigo, Spain. Luxembourg: COST European Cooperation in Science and Technology. 2004. p. 75–80.

ter Haar FB, Veltkamp RC. A 3D face matching framework. IEEE International Conference on Shape Modeling and Applications; 2008 Jun 4–6, Stony Brook, NY. New York: IEEE. 2008. p. 103–110.

ter Haar FB, Veltkamp RC 2009. A 3D face matching framework for facial curves. Graphical Models 71(2):77–91.

ter Haar FB, Daoudi M, Veltkamp RC. SHREC contest session on retrieval of 3D face scans. IEEE International Conference on Shape Modeling and Applications; 2008 Jun 4–6, Stony Brook, NY. New York: IEEE. p. 225–226.

Turk G, Levoy M. Zippered polygon meshes from range images. In: Proceedings of the 21st International ACM Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '94; 1994 Jul 24–29; New York: ACM Press. 1994. p. 311–318.

Vetter T, Jones MJ, Poggio T. A bootstrapping algorithm for learning linear models of object classes. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition; 1997 Jun 17–19, San Juan, Puerto Rico; New York: IEEE Computer Society. 1997. p. 40–46.

Whitmarsh T, Veltkamp RC, Spagnuolo M, Marini S, ter Haar FB. Landmark detection on 3d face scans by facial model registration. In: Proceedings of the First International Workshop on Shape and Semantics; 2006 Jun 17, Matsushima, Japan. New York: IEEE. 2006. p. 71–76.

Yin L, Wei X, Sun Y, Wang J, Rosato MJ. A 3D facial expression database for facial behavior research. In: Proceedings of 7th International Conference on Automatic Face and Gesture Recognition; 2006 Apr 2–6, Southampton, UK.

New York: IEEE. p. 211–216.

Zhao H, Xu G. Triangular surface mesh fairing via Gaussian curvature flow. Journal of Computational and Applied Mathematics 2006;195(1):300–311.

5

Applications

Stefano Berretti[1], Boulbaba Ben Amor[2], Hassen Drira[2], Mohamed Daoudi[2], Anuj Srivastava[3], Alberto del Bimbo[1], and Pietro Pala[1]

[1]Dipartimento di Sistemi e Informatica, University of Firenze, Firenze, Italy

[2]Institut Mines-Télécom/Télécom Lille 1, France

[3]Department of Statistics, Florida State University, Tallahassee, USA

5.1 Introduction

In the previous chapters, techniques that can effectively represent the surface of 3D face scans to perform face analysis were presented. These techniques were mainly developed to perform face identification and verification to enable provison of complementary and/or alternative solutions with respect to traditional 2D approaches. In fact, a lot of research effort has been invested on this topic in the last few years, resulting in the development of many different techniques that help obtain very high recognition accuracy on benchmark data sets. However, as 3D face recognition methods become more robust, effective, and efficient, new tasks are emerging where 3D techniques can be usefully applied.

The main objective of this last chapter of the book is to give an overview of emerging 3D face processing applications and new trends of research interest. In particular, in the next sections of the chapter, first an overview of the 3D face databases that are released for public use are presented, evidencing their main characteristics in terms of expression variations, pose changes, presence of occlusions (Section 5.2), then different applicative scenarios will be presented, as follows:

- In Section 5.3, state-of-the-art methods for 3D facial recognition are reviewed and discussed.

- In Section 5.3.3, the challenges related to face recognition using 3D scans with nonfrontal pose, missing parts, and occlusions are discussed, and methods that address this applicative scenario are reviewed. More insights are given for two of these methods.
- In Section 5.4, state-of-the-art methods for facial expression recognition are reviewed, and two specific solutions are then discussed in more detail. A semi-automatic approach that requires manual intervention, and a fully automatic solution that performs expression recognition without any human intervention. In this section, the new challenges posed by the analysis of 3D dynamic face sequences for the purpose of facial expression recognition are also addressed, and a recent method giving effective solution to this problem is presented.

5.2 3D Face Databases

In the last few years, several 3D face databases have been made publicly available for testing algorithms that use 3D data to perform face modeling, analysis, and recognition. These databases have progressively included face scans of subjects that exhibit non-neutral facial expressions and nonfrontal poses. Table 5.1, reports the more general data sets that are currently available for the task of 3D face recognition. For each data set information about the sensor used during acquisition, the total number of involved subjects and scans are reported, together with notes about the presence of scans with missing parts or occlusions.

Table 5.1 3D face data sets for 3D face recognition

Database	Sensor	Number of subjects	Total scans	Missing data	Occlusions
FRGC v1.0	laser	275	943	No	No
FRGC v2.0	laser	466	4007	Yes	No
GAVAB	laser	61	549	Yes	Yes
Bosphorus	structured light	105	4652	Yes	Yes

Table 5.2 summarizes the characteristics of some of the most known and used 3D face databases that include subjects with non-neutral expressions.

Table 5.2 Main characteristics of the most used 3D face databases that include non-neutral facial acquisitions

Data set	Subjects	Scans	Expressions	Pose
FRGC v2.0	466	4007	not categorized – disgust, happiness sadness, surprise	small changes
GAVAB	61	549	smile, laugh, random	up, down, left, right
BU-3DFE	100	2500	anger, disgust, fear happiness, sadness, surprise	frontal
Bosphorus	105	4666	action units, anger, disgust fear, happiness, sadness, surprise	13 yaw and pitch rotations hand, eyeglasses

Table 5.3 3D static and dynamic (3D+time) face data sets for 3D facial expressions recognition

Database	Format	Subjects	Scans	Expressions	Image	Resolution	Year
BU-3DFE	static	100	2500	7	Color	1040 × 1329	2006
Bosphorus	static	105	4666	6	Color	1600 × 1200	2008
ZJU-3DFE	static	40	360	4	Color	–	2006
ADSIP	dynamic	10	210	7	Color	601 × 549	2009
BU-4DFE	dynamic	101	606	6	Color	1024 × 681	2008
Hi4D-ADSIP	dynamic	80	3360	14	Color	1024 × 1728	2011

Some data sets have specific characteristics to perform facial expression classification (see Table 5.3). In this case, it is relevant that the number of expressions labeled in the data set and that constitute the ground truth for the purpose of expression recognition. Among these data sets, those constructed at the Binghamton University (BU-3DFE database) (Yin et al., 2006) and at the Boğaziçi University (Bosphorus database) (Savran et al., 2008), have contributed to push the research on 3D facial expression recognition and classification. Recently, some 4D databases that include dynamic sequences of 3D scans along the time have been also released for public use (Matuszewski et al., 2011, 2012; Yin et al., 2008). These 4D data are mainly intended for studying facial expressions which are intrinsically related to the time variation of the face.

More details on the individual data sets are reported in the following paragraphs, starting with 3D static data sets and then presenting the most recent 4D dynamic data sets.

The Face Recognition Grand Challenge (FRGC) database

The Face Recognition Grand Challenge version 2.0 (FRGC v2.0) database (Phillips et al., 2005) is the largest 3D database in terms of subjects enrolled and the most widely used in order to compare methods for 3D face recognition. However, it does not provide categorized facial expressions; hence it is less useful for the purpose of 3D facial expression recognition. It includes 3D face scans partitioned in three sets, namely, *Spring2003* set (943 scans of 275

individuals), *Fall2003* and *Spring2004* (4007 scans of 466 subjects in total). The *Spring2003* set is also referred to as FRGC v1.0 an is used for training, set up, and tuning of methods, whereas the *Fall2003* and *Spring2004* are referred to as FRGC v2.0 and are used for testing. Face scans are given as matrices of points of size 480 × 640, with a binary mask indicating the valid points of the face. Because of different distances of the subjects from the sensor during acquisition, the actual number of points representing a face can vary. Individuals have been acquired with frontal view from the shoulder level, with very small pose variations. About 60 percent of the faces have neutral expression, and the others show expressions of disgust, happiness, sadness, and surprise. In particular, according to a classification originally performed at Geometrix (Maurer et al., 2005) and subsequently suggested by the FRGC organizers, there are 2,469 neutral expression scans, 796 small cooperative expression scans, and 742 large noncooperative expression scans. Some scans have also occlusions because of hair. Guidelines suggest using the *Spring2003* set for training and the remaining two sets for validation.

It is also interesting to note that a 2D color image of the subject is also provided for each 3D scan. In fact, acquisitions are performed using the Minolta Vivid 910 laser scanner that permits a 2D color image capturing at the same time of the 3D acquisition (actually, the 2D image is acquired just with a very short time delay, because different sensors are used for 2D and 3D acquisitions). Figure 5.1 shows the 3D and 2D data for the gallery scan of a sample subject. The database can be summarized as follows:

- *Pros*—largest number of subjects and very high number of scans; textured images; most used for competitive evaluation; used for the FRGC contest in 2005/2006
- *Cons*—small/moderate facial expressions; expressions not categorized; some images have small misalignment between 3D and 2D data

Figure 5.1 FRGC database: 3D and 2D sample data for the subject with *id*=02463 are reported. It can be observed as the 3D and 2D acquisitions are registered and both provided with a set of 480 × 640 points/pixels

3D acquisition

2D acquisition

The GAVAB Database (GAVAB)

The Grupo de Algorítmica para la Visión Artificial y la Biometría database (GAVAB DB) (Moreno and Sánchez, 2004a) is characterized by facial scans with very large pose and expression variations and acquisition. It includes 3D face scans of 61 adult Caucasian individuals (45 males and 16 females). For each individual, nine scans are taken that differ in the acquisition viewpoint and facial expressions, resulting in a total of 549 facial scans. In particular, for each individual, there are two frontal face scans with *neutral* expression, two face scans where the subject is acquired with a *rotated* posture of the face (around $\pm 35^{\circ}$ looking up or looking down) and neutral facial expression, and three frontal scans in which the person *laughs*, *smiles*, or shows a *random* expression. Finally, there are also a right and a left side scan each nominally acquired with a rotation of $\pm 90^{\circ}$ left and right. This results in about 67% of the scans that have a neutral expression, but just 22% that have neutral expression and frontal pose.

Modified scans of this database have been used as data for the SHREC 2008 *Shape Retrieval Contest of 3D Face Scans* (Daoudi et al., 2008) and to test face recognition accuracy as well as to test recognition performance in the case parts of the face scans are missing (Drira et al., 2010; Huang et al., 2012). As an example, Figure 5.2 shows one 3D neutral scan and the left and right scans of a sample subject. In summary, the main features of the database are:

- *Pros*—accentuated facial expressions; pose variations (left/right, up/down)
- *Cons*—few subjects (61); small number of scans (549); large artifacts and noisy acquisitions; no-texture images

Figure 5.2 GAVAB database: For a sample subject, the frontal scan and the two side scans are reported. For the side scans, the given acquisition as provided in the database and its normalized counterpart are shown so as to better evidence the extent of the missing parts of the face

The Binghamton University 3D Facial Expression (BU-3DFE) Database

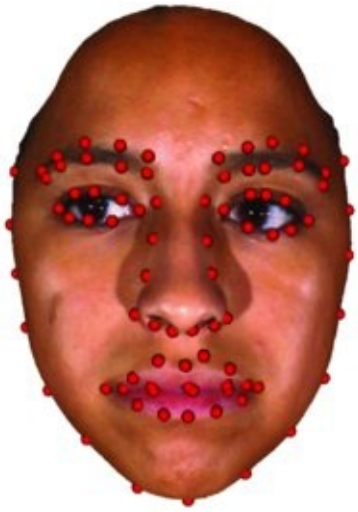
The BU-3DFE database was recently constructed at *Binghamton University* (Yin et al., 2006). It was designed to provide 3D facial scans of a population of different subjects, each showing a set of prototypical emotional states at various levels of intensities. A total of 100 subjects exist in the database, divided between female (56 subjects) and male (44 subjects). The subjects are well distributed across different ethnic groups or racial ancestries, including *White*, *Black*, *East-Asian*, *Middle-East Asian*, *Latino-Americans*, and others. During the acquisition, each subject was asked to perform the six basic facial expressions defined by Ekman, namely, *anger* (AN), *disgust* (DI), *fear* (FE), *happiness* (HA), *sadness* (SA), and *surprise* (SU), plus the *neutral* (NE). Each facial expression has four levels of intensity—*low*, *middle*, *high* and *highest*—except the neutral facial expression. Thus, there are 25 3D facial expression scans for each subject, resulting in 2500 3D facial expression scans in the database. As an example, Figure 5.3 shows the 3D faces of sample subject for the *happiness* expression (four levels of intensity) and the *neutral* expression.

Figure 5.3 BU-3DFE database: 3D textured face scans of a sample subject showing the *happiness* facial expression at the four levels of intensity (from *highest* to *low*) plus the *neutral* expression

Each of the 3D facial expression scan is also associated with a raw 3D face

mesh, a cropped 3D face mesh, a set of 83 *manually annotated* facial landmarks, and a facial pose vector. These data give a complete 3D description of a face under a specific facial expression. The cropped and textured 3D face scan, and the 83 facial landmarks are shown in Figure 5.4**a**. As summarized in Figure 5.4**b**, the landmarks are distributed in correspondence to the most distinguishing traits of the face, that is, *eyes*, *eyebrows*, *nose*, and *mouth* (plus some landmarks on the face boundary). A more detailed description of the BU-3DFE database can be found in (Yin et al., 2006).

Figure 5.4 BU-3DFE database: (a) the 83 facial landmarks evidenced on a textured 3D face scan with neutral expression; (b) the number of manually identified landmarks for different regions of the face. Copyright © 2011, Springer



(a)

face region	#points
right eye	8
left eye	8
right eyebrow	10
left eyebrow	10
nose	12
mouth	20
face boundary	15

(b)

Finally, we observe that these data are acquired with an ad hoc stereo-camera system that reconstructs the 3D shape of the face from two different left/right views. In so doing, the two cameras also acquire two 2D color images of the face with a left/right view of the face. The characteristics of the BU-3DFE data set can be summarized in the following way:

- *Pros*—categorized facial expressions; different levels of intensity for each expression; preprocessed face models; scans with annotated facial landmarks
- *Cons*—medium number of subjects (100); just frontal pose; 2D color images provided but from left/right views

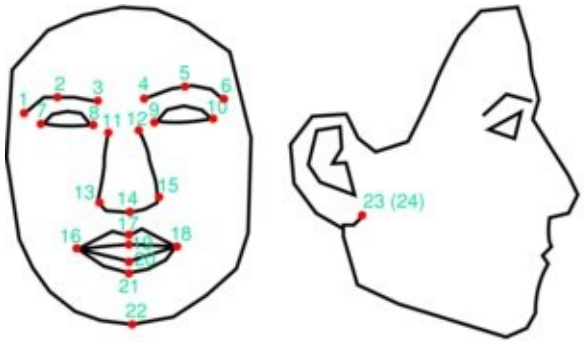
The Bosphorus Database

The Bosphorus database was collected at the Boğaziçi University and made available in 2008 (Savran et al., 2008). It consists of the 3D facial scans and images of 105 subjects acquired under different expressions and various poses and occlusion conditions. Occlusions are given by hair, eyeglasses, or predefined hand gestures covering one eye or the mouth. Many of the male subjects also have facial hair (beard and moustache. The majority of the subjects are Caucasians aged between 25 and 35 years, with a total of 60 males and 45 females. The database includes a total of 4666 face scans, with the subjects categorized into two different classes:

- 34 subjects with up to 31 scans per subject (including 10 expressions, 13 poses, 4 occlusions, and 4 neutral faces)
- 71 subjects with up to 54 different face scans. Each scan is intended to cover one pose and/or one expression type, and most of the subjects have only one neutral face, though some of them have two. There are a total of 34 expressions, 13 poses, 4 occlusions, and 1 or 2 neutral faces. In this set, 29 subjects are professional actors/actresses, the scans of whom provide more realistic and pronounced expressions.

Each scan has been manually labeled with 24 facial landmarks, such as *nose tip*, *inner eye corners*, provided that they are visible in the given scan. These feature points are summarized in Figure 5.5.

Figure 5.5 Bosphorus database: (a) the 24 facial landmarks evidenced a face sketch; (b) the number of manually identified landmarks for different regions of the face



(a)

face region	#points
right eye	2
left eye	2
right eyebrow	3
left eyebrow	3
nose	5
mouth	6
face boundary	3

(b)

In Figure 5.6, sample scans for subjects of different gender, ethnicity, and age are reported. Scans also differ for pose, occlusions, and missing parts (rendering of textured and non-textured scans are reported). With respect to the other data sets, the characteristics of the Bosphorus database can be summarized as:

- *Pros*—categorized facial expressions; facial action units performed by professional actors; occlusions and pose variations; scans with annotated facial landmarks; 2D images available
- *Cons*—medium number of subjects (105), but large number of scans (4666)

Figure 5.6 Bosphorus database: Sample scans are shown with and without texture, and from frontal and 45◦ yaw views. Some scans also evidence occlusions and missing parts

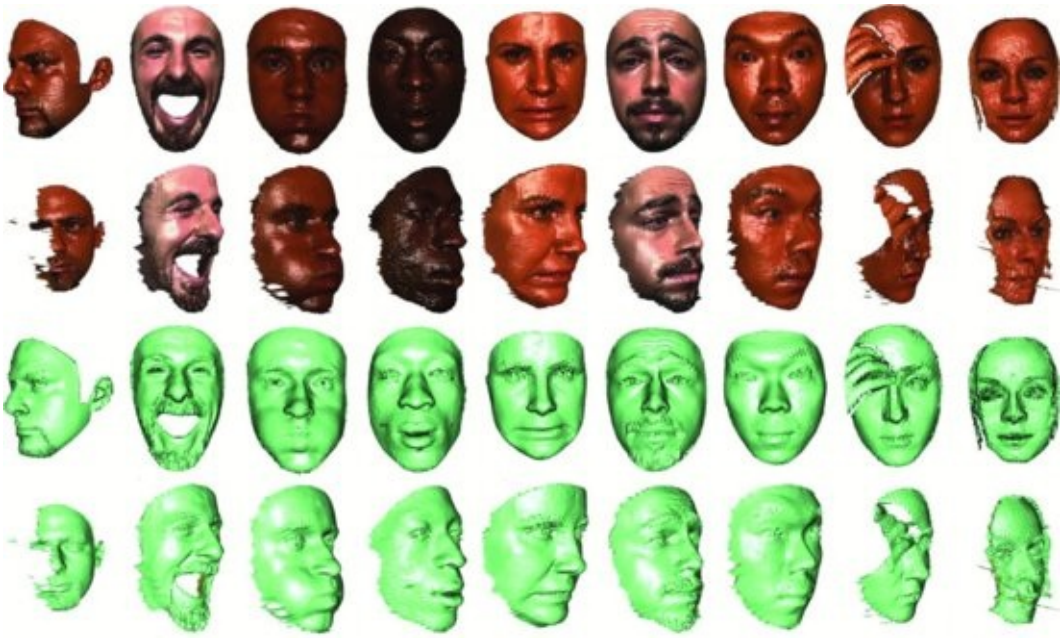


Figure 5.7 Examples of 2D and 3D frames of a 3D video from the BU-4DFE data set

The Binghamton University 4D Facial Expression (BU-4DFE) Database

To investigate the usability and performance of 3D dynamic facial sequences for facial expression recognition, a dynamic 3D facial expression database has been created at Binghamton University (Yin et al., 2008). The Dimensional Imaging's 3D dynamic capturing system (Di3D, 2006), has been used to capture a sequence of stereo images and produce the depth map according to a passive stereo-

photogrammetry approach. The range maps are then combined to produce a temporally varying sequence of high-resolution 3D images with an RMS accuracy of 0.2 mm. At the same time, 2D texture videos of the dynamic 3D models are also recorded. Each participant (subject) was requested to perform the six prototypic expressions (i.e., *anger*, *disgust*, *fear*, *happiness*, *sadness*, and *surprise*) separately. Each expression sequence contains neutral expressions in the beginning and the end, so that each expression was performed gradually from neutral appearance, low intensity, high intensity, and back to low intensity and neutral. Each 3D sequence captures one expression at a rate of 25 frames per second and each 3D sequence lasts approximately 4 seconds with about 35,000 vertices per scan (i.e., 3D *frame*). The database consists of 101 subjects (58 female and 43 male, with an age range of 18–45 years old) including 606 3D model sequences with 6 prototypic expressions and a variety of ethnic/racial ancestries (i.e., 28 Asian, 8 African-American, 3 Hispanic/Latino, and 62 Caucasian). More details on the BU-4DFE can be found in (Yin et al., 2008). An example of a 3D dynamic facial sequence of a subject with "happy" expression is shown in Figure 5.7, where 2D frames and 3D frames are reported. From left to right, the frames illustrate the extent of the facial expression from *neutral* to the *onset*, *offset*, *apex*, and *neutral* intensity of the expression.

High Resolution 4D Database from the Applied Digital Signal and Image Processing Research Centre (Hi4D-ADSIP)

This data set was acquired at the *University of Central Lancashire* and released in late 2011 (Matuszewski et al., 2011; 2012). The dynamic facial acquisition system from the Dimensional Imaging (Di3D, 2006) was used to acquire 3D facial sequences. The system consists of two pods with three cameras in each pod, and two floodlights. All six cameras have a high resolution four megapixel sensor. In each pod, two cameras are used for depth recovery, whereas the third one captures texture. The maximum recording speed is 60 frames per second. Each pod generates a range map from its own pair of corresponding stereo images using a passive stereo-photogrammetry approach and produces a related 3D triangular mesh that covers approximately half of the facial area. Two 3D meshes from both pods are subsequently stitched together to form a complete face representation with an RMS (root-mean-square) target accuracy of approximately 0.5 mm. The system is able to construct a face model covering

nearly 180° field. The participants were asked to perform a number of facial expressions one after another. Each recorded sequence starts and ends with a neutral facial appearance and is performed at a specified expression intensity. To reduce the required database storage space and to simplify time warping between expressions for construction of the common time reference frame, the participant was asked to complete each expression in less than 10s. For each sequence, six videos were recorded simultaneously with an image resolution of 2352 × 1728 pixels per frame. The recorded video sequences include four grayscale sequences for geometry reconstruction and two color sequences for texture mapping.

Currently there are 80 subjects included in the database. The majority of them, 65, are undergraduate students from the Performing Arts Department at the University of Central Lancashire. The rest are undergraduate students, postgraduate students, and members of staff from other departments of the same university without any specific training in acting. Their age range is between 18 and 60. The database consists of 48 female and 32 male subjects from a variety of ethnic origins. Each 3D model in a sequence contains approximately 20,000 vertices. For each recorded facial articulation, the following files are included in the database: (1) a sequence of OBJ files with associated MTL and texture JPG files for each individual frame; (2) a video clip; (3) animated GIF files, and (4) text file with 3D position of 84 tracked landmarks. On average, the duration of the sequence showing one of the seven basic expressions is about 3s long, whereas the durations of the mouth/eyebrow articulations and phrases reading sequences are 6s and 10s, respectively.

5.3 3D Face Recognition

Because facial biometrics is natural, contactless, and non-intrusive, it emerges as the most attractive way for identity recognition. Unfortunately, 2D-based face recognition technologies still face difficult challenges, such as pose variations, changes in lighting conditions, occlusions, and facial expressions. Instead, 3D output from laser scanners is minimally dependent on the external environmental factors and provides faithful measurements of shapes facial surfaces. It is the case the only remaining variability that is manifested within the same class (i.e., within the measurements of the same person) is the one introduced by changes in facial expressions. In fact, changes induced by facial expressions modify the shapes of facial surfaces to some extent and introduce a nuisance variability that

has to be accounted for in shape-based 3D face recognition. We argue that the variability introduced by facial expressions has become one of the most important issues in 3D face recognition. The other important issues relate to data collection and imperfections introduced in that process. It is difficult to obtain a pristine, continuous facial surface, or a mesh representing such a surface, with the current laser technology. One typically gets holes in the scanned data in locations of eyes, lips, and outside regions. For instance, scans of people with open mouths result in holes in the mouth region. Moreover, when the subject is noncooperative and the acquisition phase is unconstrained, these result in variation in pose and unforeseen extraneous occlusions. Many studies have treated pose and expression variations, but only a few have tried to focus on solving the problem of occlusions even if a face can easily be partially hidden by objects like glasses, hats, scarves, or a hand, hair, and beard. Occluded parts represent wrong information, which can degrade recognition accuracy, so locating and removing occlusions on face quickly and automatically are challenging tasks.

In the biometrics literature, recognition is a general term which includes: (1) face *verification* (or *authentication*) and (2) face *identification* (or *recognition*).

- Face verification ("Am I who I say I am?") is one-to-one match that compares a query face image against a template face image whose identity is being claimed. To evaluate the verification performance, the verification rate (the rate at which legitimate users are granted) versus false accept rate (the rate at which imposters are granted access) is plotted, called the receiver operating characteristic (ROC) curve.
- Face identification ("Who am I?") is one-to-many matching process that compares query face image against all the template images in a face database to determine the identity of the query face. The identification of the test image is done by locating the image in the database that has the highest similarity with the test image. To evaluate identification performance the cumulative matching characteristic (CMC) curve is used. This curve displays the cumulative identification rates as a function of the rank distribution. This provides an indication of how close one may be to getting the correct match if the rank-one match was incorrect.

One application of the verification task is access control where an authorized individual is seeking access to a secure facility and presents to the system his or her identity. Here, a one-to-one matching is performed: The 3D image for this individual is acquired, preprocessed, and finally compared with an enrollment

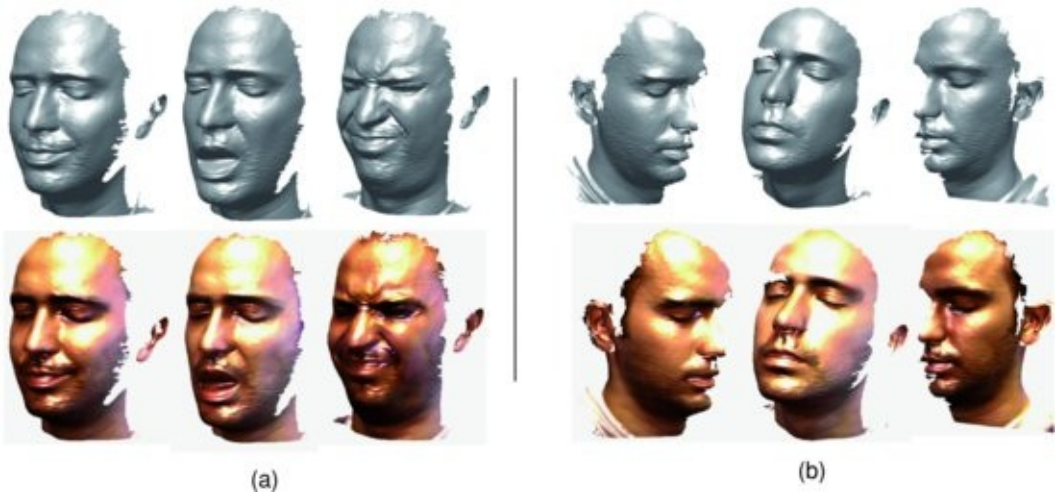
acquisition already incorporated in the system database. If the similarity is greater than a defined threshold, the subject is granted access, otherwise access is denied.

The classic application of the identification task is for identify the presence of suspect people in database of recorded identity. In this case the one-to-many matching is performed by first acquiring the 3D image of the individual, then preprocessing the scan to enhance the quality and extract appropriate shape descriptors and finally comparing against a gallery of already enrolled subjects. Depending on the specific scenario and on the size of the gallery, the computational time is typically an additional issue in this case.

5.3.1 Challenges of 3D Face Recognition

When acquired in non-controlled conditions, scan data often suffer from the problem of missing parts because of self-occlusions or laser-absorption by dark area. Actually, the 3D face needs more than one scan to be fully acquired. Especially when the pose is not frontal as illustrated in Figure 5.8**b**, the resulting scan is said to be 2.5D and not full 3D. However, this 2.5D scan is roughly approximated by 3D scan by 3D face recognition community researchers. Moreover, the bottom row in Figure 5.8**b** illustrates that dark areas (hair or eyebrows) absorb the laser and generate missing data in the 3D mesh as illustrated at the top row of the same figure.

Figure 5.8 (a) Deformations under facial expressions; (b) Missing data due to self occlusion



Additionally, variations in face data due to facial expressions cause deformations in the 3D mesh. Figure 5.8**a** illustrates expressive faces at the

bottom row (as 3D textured mesh). The top row illustrates the resulting 3D mesh with deformations.

Any 3D face recognition approach should successfully match face scans in the presence of expression-based deformations and/or missing data (as illustrated in Figs. 5.8**a,b**) to a good quality, neutral, frontal 3D model. We note that generally the enrolled face scans are collected in controlled conditions and exhibit good data quality. Past literature has tackled this fundamental issue with varying degree of success as described in the survey paper by Bowyer *et al.* (2006). In the following sections, we review some of the approaches addressing 3D face recognition, grouping them according to the general idea addressed in 3D face representation.

5.3.2 3D Face Recognition: State of the Art

Among the various solutions that have appeared in recent years, we address methods that are based on *geodesic measures* of the face, *deformable templates*, or the extraction of *local regions/features*.

Geodesic-Based Approaches

In Bronstein *et al.* (2005), the authors presented an experimental validation of the isometric model. They placed 133 markers on a face and tracked the change of both Euclidean and geodesic distances under facial expressions. The distribution of the absolute change of geodesic distances was closer to zero than the distribution of the change of Euclidean distance. Therefore, the authors assumed that the change of geodesic distance is insignificant and concluded that geodesic distance remains unchanged under facial expression. Under this assumption Bronstein *et al.* (2005) corresponded the geodesic distance between the corresponding points of the face surface to the Euclidean distance between two canonical surface points. Canonical surfaces were obtained from face surfaces by warping according to a topology preserving transformation. Finally, face models were represented with the geometric moments up to the fifth order computed for the 3D face canonical forms. However, although the effect of expressions was attenuated, a similar attenuation also occurred for discriminating features such as the eye sockets and the nose. This approach was improved in Bronstein *et al.* (2006b) where the authors handled the challenge of missing parts. They embedded the probe facial surface into that of the gallery. Faces belonging to the same subject are nearly isometric and thus result in low

embedding error, whereas different subjects are expected to have different intrinsic geometry, and thus produce higher embedding error. The open mouth corrupts the isometric model. This problem was handled later by the authors in Bronstein *et al.* (2007) by using a geodesic mask that excluded the mouth region. The authors first detected and removed the lips, then the computation of geodesic distance geodesics were calculated on the surface in the presence of a hole corresponding to the removed part. This was done while avoiding passing in mouth area.

The assumption of the isometric model has motivated several authors to use geodesic distance on facial surface. In Samir *et al.* (2009a), the geodesic distance to the tip of the nose was used as a surface distance function. Differential geometry was used to compare 3D level curves of the surface distance function. This approach was an improvement upon an earlier work by the authors Samir *et al.* (2006), where they had used the level curves of the height function to define facial curves. The use of geodesic distance in (Samir et al., 2009a) allows this approach to handle facial expressions. However, the open mouth corrupts the shape of some level curves, and this parametrization did not address this problem; hence, experiments were restricted to a small subset of FRGC v2.0 database.

A similar geodesic polar parametrization of the face surface was proposed in Mpiperis *et al.* (2007), but rather than studying the shape of curves, they studied local geometric attributes under this polar parametrization. To handle data with open mouths, they modified their geodesic polar parametrization by disconnecting the lips. Therefore, their approach required lips detection, as was the case in Bronstein *et al.* (2007).

In Berretti *et al.* (2010b), the authors used the geodesic distance on the face to extract isogeodesic facial stripes. Equal-width isogeodesic facial stripes were used as nodes of graph and edges between nodes were labeled with descriptors, referred to as 3D weighted walkthroughs (3DWWs), that captured the mutual relative spatial displacement between all the pairs of points of the corresponding stripes. Face partitioning into isogeodesic stripes and 3DWWs together provided an approximate representation of local morphology of faces that exhibits smooth variations for changes induced by facial expressions.

A common limitation of the previously described approaches is that they assume that the facial shape deforms isometrically, that is, the surface distances between points are preserved, which is not valid in the case of large expressions. Actually, the movement of mimic muscles can stretch and/or shrink the face

surface and not only bending it.

Deformable Template-Based Approaches

In recent years there has been focus on deforming surfaces, one into another, under a chosen criterion. Grenander's *deformable template* theory (Grenander, 1993) has been successfully applied to studying shapes of anatomical parts using medical images (Miller and Younes, 2001; Grenander and Miller, 1998). The set of nonrigid deformations can be subdivided into linear and nonlinear deformations. Nonlinear deformations imply local stretching, compression, and bending of surfaces to match each other and are also referred to as *elastic* deformations. Earlier attempts at elastic matching used graphs on the basis texture images of faces (Kotropoulos et al., 2000).

Kakadiaris *et al.* (2007a) used an *annotated face model* to study geometrical variability across faces. The annotated face model was deformed elastically to fit each face thus allowing the annotation of its different anatomical areas, such as the nose, eyes, and mouth. Elastic registration with models was used the points of an annotated 3D face reference model were shifted according to elastic constraints so as to match the corresponding points of 3D target models in a gallery. Similar morphing was performed for each query face. Then, face matching was performed by comparing the wavelet coefficients of the deformation images obtained from morphing. This approach was automatic. Similar approaches were based on manually annotated models (Lu and Jain, 2006, 2008; Mpiperis et al., 2008a).

Lu and Jain (2008) presented an approach that is robust to self-occlusions (due to huge pose variations) and expressions. Three-dimensional deformations learned from a small control group was transferred to the 3D models with neutral expression in the gallery. The corresponding deformation was synthesized in the 3D neutral model to generate a deformed template. The matching was performed by fitting the deformable model to a given test scan, which was then formulated as a minimization of a cost function.

ter Haar and Velkamp (2010), proposed a multiresolution approach to semi-automatically build seven morphable expression models, and one morphable identity model from scratch. The proposed algorithm automatically selects the proper pose, identity, and expression such that the final model instance accurately fits the 3D face scan.

A strong limitation of these approaches is that the fiducial landmarks needed

during expression learning have to be extracted manually for some approaches. They are usually semi-automatic and rarely full automatic.

Local Regions / Features Approaches

A different way proposed in the literature to handle expression variations is to match parts or regions of faces rather than the whole faces. Several notable local techniques were proposed in Gordon (1992) and Moreno *et al.* (2005), where the authors employed surface areas, curvatures around facial landmarks, distances, and angles between them with a nearest neighbor classifier. In Lee *et al.* (2005), on the basis of ratios of distances and angles between eight fiducial points, the authors' technique used a support vector machine classifier. Euclidean/geodesic distances between anthropometric fiducial points were employed as features in Gupta *et al.* (2007) along with linear discriminant analysis classifiers. However, a successful automated detection of fiducial points is critical here.

In Mahoor and Abdel-Mottaleb (2009) and Mousavi *et al.* (2008), the authors presented low level geometric features-based approaches and reported results on neutral faces but the performance decreased when expressions variations were introduced. Using similar features, the authors in Li *et al.* (2009) proposed designing a feature pooling and ranking scheme to collect various types of low level geometric features, such as curvatures, and rank them according to their sensitivities to facial expressions. They applied sparse representations to the collected low level features and achieved good results on a challenging data set [GAVABDB (Moreno and Sanchez, (2004b)]. This approach, however, required a training step.

Along similar lines, Wang et al., (2010) computed a signed shape difference map (SSDM) between two aligned 3D faces as an intermediate representation for the shape comparison. With regards to the SSDMs, they used three kinds of features to encode both the local similarity and the change characteristics between facial shapes. Selected the most discriminative local features optimally by boosting and trained as weak classifiers for assembling three collective strong classifiers. The individual features were of the type: Haar-like, Gabor, and local binary pattern (LBP).

McKeon and Russ (2010) used the 3D Fisherface region ensemble approach. After faces registration using the ICP algorithm, the Fisherface approach seeks to generate a scatter matrix for improved classification by maximizing the ratio of the between scatter and within scatter. Twenty-two regions were used as input

for 3D Fisherface. To select most discriminative regions, *sequential forward search* Sun *et al.* (2008) was used.

Huang *et al.* (2010), proposed to use the multiscale LBP as a new representation for 3D face jointly to shape index. They then extracted Scale Invariant Feature Transform(SIFT)-based local features. The matching also involves holistic constraint of the facial component and configuration.

In Cook *et al.* (2006), the Log–Gabor templates were used to exploit the multitude of information available in human faces to construct multiple observations of a subject, which were classified independently and combined with score fusion. Gabor features were recently used in Moorthy *et al.* (2010) on automatically detected fiducial points.

In Chang *et al.* (2006) and Mian *et al.* (2007a) the focus was on matching nose regions albeit using ICP. To avoid passing over deformable parts of the face encompassing discriminative information, the authors in Faltemier *et al.* (2008a) proposed to use a set of 38 face regions that densely cover the face and fused the scores and decisions after performing ICP on each region.

In Queirolo *et al.* (2010a), the circular and elliptical areas around the nose were used together with the forehead and the entire face region for authentication. Surface interpenetration measure (SIM) was used for the matching. Taking advantage of invariant face regions, an annealing simulated approach was used to handle expressions.

In Alyüz *et al.* (2008b), the authors proposed to use average region models (ARMs) locally to handle the missing data and the expression-inducted deformation challenges. They manually divided the facial area into several meaningful components, and registration of faces was carried out by separate dense alignments to relative ARMs. A strong limitation of this approach is the need for manual segmentation.

5.3.3 Partial Face Matching

Many of the 3D face recognition methods that have been proposed in the last few years focused on face recognition in the presence of expression variations reporting very high accuracies on benchmark databases such as the FRGC v2.0 data set (Phillips et al., 2005). However, only a few solutions explicitly addressed the problem of 3D face recognition in case only a part of the facial scan was available (*partial face match*), or parts of the face are occluded by hair, glasses, scarves, hand gestures, and such. In a traditional face recognition

experiment, both the probe and gallery scans are assumed to be acquired cooperatively so as to precisely represent the whole face. Differently, an increasing interest is targeting the development of solutions enabling recognition in uncooperative scenarios. In such cases, acquisition of the probe scan is performed in suboptimal conditions that can yield a nonfrontal face scan, missing parts, or occlusions.

In general, global approaches cannot effectively manage partial face match, whereas local approaches entail the potential to cope with the problem. To manage missing data obtained by randomly removing certain regions from frontal scans, Bronstein *et al.* (2006a) proposed a canonical representation of the face, which exploits the isometry invariance of the face surface. On a small database of 30 subjects, they reported high recognition rates, but no side scans were used for recognition. Alyüz *et al.* (2008a), proposed a part-based 3D face recognition method, which operates in the presence of both expression variations and occlusions. The approach is based on the use of ARMs for registration. Under variations, such as those caused by occlusions, the method can determine noisy regions and discard them. Savran *et al.* (2008) tested the performance of this approach tested on the Bosphorus 3D face database. However, a strong limitation of this solution was the use of manually annotated landmarks that were used for face alignment and region segmentation. Faltemier *et al.* (2008b) used a set of 38 overlapping regions that densely cover the face around the nose and selected the best-performing subset of 28 regions to perform matching using the ICP algorithm. They reported a recognition experiment accounting for missing parts in the probe faces. However, in this case, too, region segmentation across different facial scans strongly relied on the accurate identification of the nose tip. More recently, a method that addresses the partial matching problem has been proposed in Perakis *et al.* (2009). This is obtained by using an automatic face landmarks detector to identify the pose of the facial scan so as to mark regions of missing data and to roughly register the facial scan with an annotated face model (AFM) (Kakadiaris et al., 2007c). The AFM is fitted using a deformable model framework that exploits facial symmetry where data are missing. Wavelet coefficients extracted from a geometry image derived from the fitted AFM are used for the match. Experiments have been performed using the FRGC v2.0 gallery scans and side scans with 45° and 60° rotation angles as probes. In Drira *et al.* (2010), the facial surface is represented as a collection of radial curves originating from the nose tip and face comparison is obtained by the elastic matching of the curves. A quality control permits the exclusion of

corrupted radial curves from the match, thus enabling recognition even in the case of missing data. Results of partial matching are given for the 61 left and 61 right side scans of the GAVAB data set (Moreno and Sánchez, 2004a).

Local approaches based on regions are limited by the need to identify some facial landmarks to define the regions of the face to be matched. In addition, because parts of these regions can be missing or occluded, the extraction of region descriptors is difficult; hence, regions comparison is often performed using rigid (ICP) or elastic registration (*deformable models*). Methods that use keypoints of the face promise to solve some of these limitations. In particular, a few recent works have shown that local descriptors computed around salient keypoints can be usefully applied to describe 3D objects and faces. In Mian *et al.* (2008), a 3D keypoint detector and descriptor inspired to the *scale invariant feature transform* (SIFT) (Lowe, 2004) has been designed and used to perform 3D face recognition through a hybrid 2D+3D approach that also uses the SIFT detector and descriptor to index 2D texture face images. In Mayo and Zhang (2009), SIFT detectors are used to detect and represent salient points in multiple 2D depth images derived from 3D face models for the purpose of 3D face recognition. A similar idea is used in Ohbuchi and Furuya (2004) to perform 3D object retrieval by visual similarity, but in this case, points of a sampling grid are used and SIFT descriptors are computed for them. In Berretti *et al.* (2010d), SIFT feature descriptors computed in correspondence with facial landmarks of depth images are used to classify 3D facial expressions.

Partial Face Matching Using Keypoints and Facial Curves

In the following, we shortly summarize the approach for partial face matching proposed in Berretti *et al.* (2011b). The approach uses SIFT keypoints to detect relevant and stable interest points on depth images of the face and *facial curves* to model the depth of face scans along the surface path connecting pairs of SIFT keypoints. In doing so, distinguishing traits of a face scan are captured by the SIFT descriptors of detected keypoints as well as by the set of facial curves identified by each pair of keypoints. Facial curves of gallery scans are also associated with a measure of saliency so as to distinguish those that model characterizing traits of some subjects from those that are frequently observed in the face of many different subjects. In the comparison of two faces, SIFT descriptors are matched to measure the similarity between pairs of keypoints

identified on the two depth images. Spatial constrains are imposed to avoid outliers matches. Then, the distance between the two faces is derived by composing the individual distances between facial curves that originate from pairs of matching keypoints.

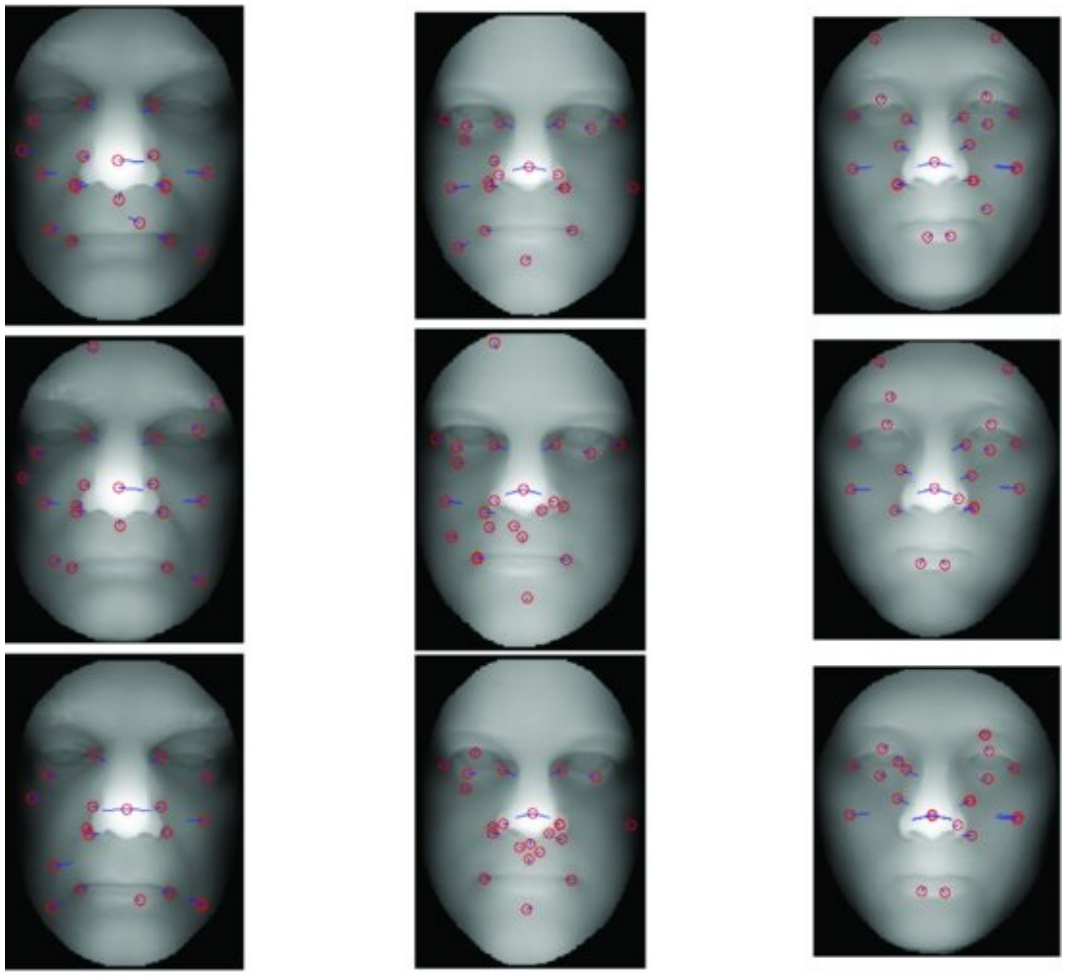
The use of keypoints of the face is advantageous with respect to using landmarks in the case of partial face matching. In fact, just few landmarks can be detected automatically with accuracy, and in the case of side scans, just a few of them are likely to be visible. On the contrary, keypoints are not constrained to specific points of the face, and many of them can be detected also on just a part of the face. According to this, the approach in Berretti et al., (2011b) does not exploit any particular assumption about the position of the keypoints on the face surface. Rather, the position of keypoints is expected to be influenced by the specific morphological traits of each subject. In particular, assuming that the process of keypoint detection incorporates a measure of the scale associated with each keypoint, the assumption that detected keypoints correspond to meaningful landmarks is relaxed and the more general assumption of within-subject *repeatability* is exploited: The position of the most stable keypoints—detected at the coarsest scales—does not change substantially within facial scans of the same subject. In particular, the approach in Berretti et al., (2011b) relies on the detection of a number of keypoints on the 3D face surface and the description of the 3D face surface in correspondence to these keypoints as well as along the paths connecting pairs of keypoints. The SIFT are used to perform keypoints detection and description. SIFT keypoints are extracted at scales of increasing σ, so as to obtain a minimum of *N* keypoints for scan. Then, the top *N* keypoints—starting from the highest σ values—are retained and used as the base for face description.

To extract depth images and detect SIFT keypoints, 3D face scans first undergo some preprocessing (Berretti et al., 2010c). First, 3D faces are cropped using a sphere of radius 100 mm centered on the nose tip (the nose tip has been detected using the approach in Mian *et al.* (2007b)). After this, face scans are transformed to depth images considering a frontal view of the scan. To this end, the pose of the scans have been normalized using a solution on the basis of an iterative *principal component analysis* (PCA) (Mian et al., 2007b). In addition, spikes were removed using median filtering in the *z*-coordinate, holes were filled using cubic interpolation, and the 3D scans were re-sampled on an uniform square grid at 1 mm resolution.

Figure 5.9 shows the depth images derived from the 3D face scans of three

different subjects. For the detected keypoints, the SIFT *descriptors* are computed. The properties of the SIFT descriptor make it capable to provide a compact and powerful local representation of the depth image and, as a consequence, of the face surface. Because the localization of SIFT keypoints only depends on the geometry of the face surface, these keypoints are not guaranteed to correspond to specific meaningful landmarks on the face. For the same reason, the detection of keypoints on two face scans of the same individual should yield to the identification of the same points on the face, unless the shape of the face is altered by major occlusions or non-neutral facial expressions. As an example, Figure 5.9 shows the keypoints identified on the depth images of three subjects.

Figure 5.9 Each column shows three depth images of the same individual. It can be noticed that a large part of the keypoints are repeatably identified at the same neighborhoods for the same individual



An important measure to validate the effectiveness of the extracted keypoints is represented by their *repeatability*. This can be evaluated using the approach in Mian *et al.* (2008). In this solution, the correspondence of the location of keypoints detected in two face scans is measured by considering ICP registration: The 3D faces belonging to the same individual are automatically registered and the errors between the nearest neighbors of their keypoints (one from each face) are recorded.

However, the information captured by combining local SIFT descriptors of the keypoints detected in a face scan is not discriminant enough for the accurate recognition of the subjects' identity. Additional information necessary to discriminate the identity of each subject is captured by considering relational information between pairs of keypoints in the form of *facial curves*, as discussed in the following sections.

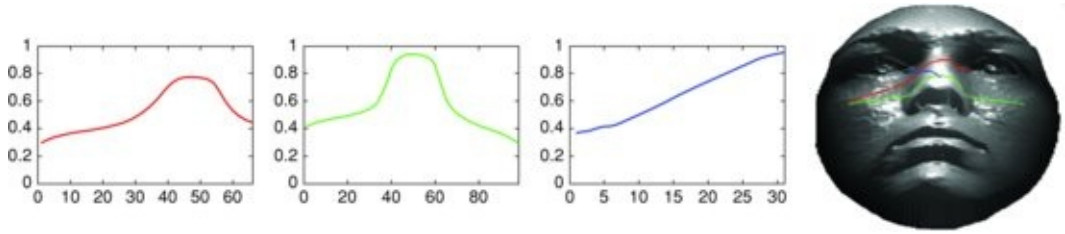
Facial Curves and Their Saliency

Each pair of SIFT keypoints detected on a depth image is used to identify a *facial curve*, that is, the 1D function of the depth values of the pixels that lay on the segment connecting the two keypoints. More formally, let $I(\mathbf{x})$ with $x \in \Re^2$ be the depth image representing a face scan, x_1 and x_2 two keypoints, then the facial curve identified by the ordered pair (x_1, x_2) is defined as

(5.1) $$\mathcal{C}^I_{x_1,x_2}(t) = I\left((1-t)x_1 + tx_2\right), \quad t \in [0,1].$$

As an example, Figure 5.10 shows three facial curves derived from the depth image of a sample individual. It can be observed that the curves capture the shape of the face across different paths and in particular the nose protrusion.

Figure 5.10 Three facial curves between pairs of keypoints of a sample face. The horizontal axis reports the number of pixels between the two keypoints, whereas the vertical axis represents the gray-level of the depth image along the curve normalized in [0,1]



Comparison of the identity of two face representations is accomplished by matching their corresponding facial curves. Given the facial curve $\mathcal{C}_1(t)$,

$t \in [0, t_1]$ identified by the keypoint pair (x_1, x_2) of the face scan I_1, and the facial curve $\mathcal{C}_2(t)$, $t \in [0, t_2]$ identified by the keypoint pair (x_3, x_4) of the face scan I_2, the distance between the two facial curves is measured as

$$\mathcal{D}(\mathcal{C}_1(t), \mathcal{C}_2(t)) = \int_0^{\min\{t_1, t_2\}} |\mathcal{C}_1(t) - \mathcal{C}_2(t)| \, dt. \quad (5.2)$$

In general, it is expected that not all the facial curves have the same relevance in discriminating between different subjects. This suggests to evaluate the *saliency* of facial curves by assuming the higher the saliency of a curve $\mathcal{C}$, the lower the uncertainty about the identity of the person that is represented in the depth image if the curve $\mathcal{C}$ is observed. To this end, an information theoretic model is defined to associate with a generic curve a value of saliency. Formally, let's assume that the 3D face scans of *N* different subjects are available and let *X* be a discrete random variable taking values in the set $\{x_1, \ldots, x_n\}$, representing the identity of the subject. Furthermore, let *Y* be a continuous random variable representing a sample curve. Given an observation *Y=y*, the uncertainty of *X* once *Y* is observed can be measured through the *Shannon* entropy of the posterior distribution, that is defined as

$$H(X|Y = y) = -\sum_{x_i} P(X = x_i|y) \log P(X = x_i|y). \quad (5.3)$$

Values of $H(X|Y=y)$ are high for facial curves *Y* that are observed in the faces of many subjects and low for facial curves *Y* that are observed in the faces of just a few subjects. The lower the value of $H(X|Y=y)$, the more the observation of the facial curve *Y* on a face scan tells about the identity of the subject.

Operatively, since in a real application context only the gallery scan is available for each subject, estimation of $P(X=x_i|Y=y)$ is prevented. Therefore, the saliency $\mathcal{S}(y)$ of the facial curve *Y* can be approximated with a measure of the frequency of observing a curve similar to *Y* (up to a threshold τ) in the gallery scans $\mathcal{S}(y) = e^{-\frac{N}{N_g}}$, being *N* the number of occurrences of *Y* in the scans of the gallery and N_g the number of gallery scans.

Face Matching Using Keypoints and Facial Curves

Face comparison is performed by jointly matching the keypoints and the facial curves of two faces. First, SIFT descriptors of the keypoints detected in the probe and the gallery are compared so that for each keypoint in the probe, a candidate corresponding keypoint in the gallery is identified. In particular, a keypoint k_p in the probe is assigned to a keypoint k_g in the gallery if they match

each other among all keypoints, that is, if k_p is closer to k_g than to any other keypoint in the gallery and k_g is closer to k_p than to any other keypoint in the probe. For this purpose, proximity of keypoints is measured through the Euclidean distance between 128-dimensional SIFT descriptors associated with the keypoints. This analysis of the proximity of keypoint descriptors results in the identification of a candidate set of keypoint correspondences. Identification of the actual set of keypoint correspondences must pass a final constraint targeting the consistent spatial arrangement of corresponding keypoints on the probe and the gallery. The RANSAC algorithm is used to identify outliers in the candidate set of keypoint correspondences. This involves generating transformation hypotheses using a minimal number of correspondences and then evaluating each hypothesis on the basis of the number of inliers among all features under that hypothesis. In this way, corresponding keypoints whose spatial arrangement is an outlier are removed from the match.

Correspondences between inliers pairs of keypoints of two face scans are used to measure the distance between the two facial scans. Given a probe and a gallery, the correspondences identified by the spatial consistency can be formalized in terms of a function $\xi : \aleph \mapsto \aleph$ that associates with a facial curve $\mathcal{C}_i^{(p)}$ in the probe, its corresponding facial curve $\mathcal{C}_{\xi(i)}^{(g)}$ in the gallery. For each matched facial curve in the probe $\mathcal{C}_i^{(p)}$, the distance to the corresponding facial curve $\mathcal{C}_{\xi(i)}^{(g)}$ in the gallery is evaluated by weighting the result of Equation 5.2 by the *saliency* of the gallery scan

$$d_i = \mathcal{D}\left(\mathcal{C}_i^{(p)}, \mathcal{C}_{\xi(i)}^{(g)}\right) \mathcal{S}\left(\mathcal{C}_{\xi(i)}^{(g)}\right). \quad (5.4)$$

Eventually, the distance between the probe and the gallery is measured by averaging values of d_i in Equation 5.4 over all pairs of matching facial curves.

It should be noted that the matching scheme does not exploit any specific assumption about the correspondence of keypoints and facial curves to meaningful anatomical parts of the face. As a consequence, the matching scheme can be used without any change to support matching between a partial scan and a full scan of the face, thus enabling the recognition of faces with missing parts and/or occlusions.

5.3.4 Comparison of State-of-the-Art Methods

In the following sections, we report the 3D face recognition results for several

approaches performing face identification/verification on the 3D face scans. Results are first reported for the FRGC v2.0 data set for neutral and expressive scans, then the results on the GAVAB data set are shown with the performance of methods capable of managing both expression variations and large pose changes of the face, this latter one resulting in acquisitions with missing parts.

Comparative Evaluation on FRGC v2.0 Data Set

For the first evaluation, the results scored by state-of-the-art 3D face recognition methods on the FRGC v2.0 data set were presented (see Section 5.2). Facial scans are categorized, according to the classification provided in the FRGC protocol, as showing neutral expression, small expression, and large expression. The gallery consists of the first scans of each subject in the database (466 scans in total), with the remaining scans forming the probe set. Using the categories mentioned earlier, three recognition experiments are considered: (1) neutral versus neutral, (2) neutral versus non-neutral, and (3) neutral versus all. In these experiments, the first label indicates the gallery scans (neutral), whereas the second one refers to the probe scans used in the experiment (i.e., neutral, non-neutral, and all). Overall recognition at rank-1 of state-of-the-art methods are presented in Table 5.4, thus reporting about the effectiveness in identification scenario.

Table 5.4 Comparison of rank-1 recognition rates on the FRGC v2.0 data set for the state-of-the-art methods

Method	rank-1 RR
Spreeuwers (2011)	99.5%
Wang *et al.* (2010)	98.3%
ter Haar and Velkamp (2010)	97%
Berretti *et al.* (2010b)	94.1%
Queirolo *et al.* (2010a)	98.4%
Faltemier *et al.* (2008a)	97.2%
Kakadiaris *et al.* (2007b)	97%
Drira *et al.* ()	97%

The highest published results on this data set are those reported in the works of Queirolo *et al.* (2010a), Spreeuwers (2011), Wang *et al.* (2010), with a rank-1 recognition rate greater than 98% for the neutral versus neutral experiment. The overall best recognition score on FRGC v2.0 is reported by Spreeuwers (2011), which uses an intrinsic coordinate system on the basis of the vertical symmetry

plane through the nose.

To evaluate the performance of the state-of-the-art approaches in a verification scenario, receiver operating characteristic (ROC) curves for the ROC III mask of the FRGC v2.0 and the "All vs. All" experiment are reported in Table 5.5 using the verification results at FAR of 0.1 percent as performance indicator.

Table 5.5 Comparison of verification rates (i.e., true acceptance rate, *TAR*) at *FAR*=0.1% on the FRGC v2.0 data set for the state-of-the-art methods

	TAR @ FAR = 0.1%	
Method	ROC III	All vs. All
Kakadiaris *et al.* (2007b)	97%	–
Faltemier *et al.* (2008a)	94.8%	93.2%
Berretti *et al.* (2010b)	–	81.2%
Queirolo *et al.* (2010a)	96.6%	96.5%
Spreeuwers (2011)	94.6%	94.6%
Wang *et al.* (2010)	98.4%	98.13%
Drira et al., ()	97.14%	93.96%

Comparative Evaluation on GAVAB Data Set

The GAVAB DB (see Section 5.2) has many noisy 3D face scans with large expressions and also scans with missing parts because of acquisitions where subjects exhibit large pose variations with respect to the frontal one. The GAVAB experimental protocol assumes that one of the two frontal scans with the neutral expression for each person is taken as a gallery model, and the remaining are used as probes.

Table 5.6 compares published results of methods using the same evaluation protocol on the GAVAB. The approaches in Drira *et al.* (2010) and Huang *et al.* (2012) attain the highest recognition rate for faces with non-neutral expressions and side scans.

Table 5.6 Recognition results comparison between different methods on the GAVAB data set

	neutral	expressive	looking-down	looking-up	right side	left side
Li et al. (2009)	96.67%	93.33%	–	–	–	–
Moreno et al. (2005)	90.16%	77.9%	–	–	–	–
Mahoor et al. (2009)	–	72%	85.3%	88.6%	–	–
Huang et al. (2012)	100%	93.99%	96.72%	96.72%	93.44%	78.69%
Drira et al. (2010)	100%	94.54%	100%	98.36%	70.49%	86.89%

5.4 Facial Expression Analysis

3.4 Facial Expression Analysis

As machines become more and more involved in everyday human lives and take part in both their living and work spaces, they need to become more intelligent in terms of understanding the moods and emotions of humans. Embedding these machines with systems capable of recognizing human emotions and mental state is precisely what the human–computer interaction research community is focusing on in the affective computing and human–machine interaction communities. The following are some interesting areas where automatic facial expression recognition systems find applications:

- Human–machine interface: Facial expression is a way of communication as many other ways (e.g., speech signal). Emotional detection is natural for humans, but it is a very difficult task for machines; therefore, the purpose of an emotion recognition system is to use emotion-related knowledge in such a way that human–machine communication can be improved and make machines and robots more human-like.
- Medical care and cure field: Facial expressions are the direct means to identify when specific mental processes (e.g., pain, depression) occur.
- Psychological field: Expression detection is tremendously useful for the analysis of the human psychology.
- Security field: Decoding the language of micro-expressions is crucial for establishing or detracting from credibility, and to determine any deception from suspects during interrogations. This is because micro-expression is a momentary involuntary facial expression that people unconsciously display when hiding an emotion.
- Education field: Pupils' facial expressions inform the teacher of the need to adjust the instructional message.

The first studies on this subject date back to the late 1970s with the pioneering work of Ekman (1972). In these studies, it is evidenced that a number of *basic* facial expressions exist that can be categorized into six classes, namely, *anger*, *disgust*, *fear*, *happiness*, *sadness*, and *surprise*, plus the *neutral* expression. This categorization of facial expressions has been also proved to be consistent across different ethnicities and cultures; hence, these expressions are in some sense “universally” recognized.

In their studies, Ekman and Friesen (1977) also defined the *Facial Action Coding System* to code the facial expressions through the movement of face points as described by the *action units*. This work inspired many researchers to analyze facial expressions in 2D by tracking facial features (e.g., facial

landmarks) and measuring the amount of facial movements these landmarks undergo from expression to expression in still images and videos. Almost all of the methods developed in 2D use distributions or facial distances of these facial landmarks as features to be used as inputs to classification systems. Then, the outcome of the classifiers is one of the facial expression classes. These approaches mainly differ in the facial features selected and the classifier used to distinguish among the different facial expressions.

Recently, there has been a progressive shift from 2D to 3D in face analysis approaches, mainly motivated by the robustness of the 3D facial shape to illumination changes, pose, and scale variations. Although many studies have appeared to perform 3D face recognition (Berretti et al., 2010c; Gupta et al., 2010; Kakadiaris et al., 2007c; Mian et al., 2008; Queirolo et al., 2010b; Samir et al., 2009b), very few have taken advantage of the 3D facial geometric information to perform facial expression recognition. A few years ago, the first solutions to automatically perform facial expression recognition based on 3D face scans were proposed using very small databases and categorizing only a few facial expressions (Ramanathan et al., 2006). The availability of new facial expression databases, like those constructed at the BU-3DFE database (Yin et al., 2006), and at the Boğaziçi University (Bosphorus database) (Savran et al., 2008) have now propelled research on this topic. In particular, the BU-3DFE database has become the de facto standard for comparing facial expression recognition algorithms. This is because unlike other 3D face data sets, the BU-3DFE database provides a precise categorization of facial scans according to the Ekman's six basic facial expressions plus the neutral one, also providing different levels of expression intensities (see also the description in Section 5.2).

In the following paragraphs, the problem of facial expression recognition is introduced by first reviewing the more recent and influencing state-of-the-art solutions then presenting in detail some specific solutions. For the characterizing features of the main 3D face databases for expression analysis, we refer to Section 5.2.

5.4.1 3D Facial Expression Recognition: State of the Art

Most of the works on 3D facial expression recognition can be categorized as those based on the *generic facial model* or *feature classification*.

In the first category, a general 3D face model (*template* model) is trained with prior knowledge, such as feature points, shape and texture variations, or local

geometry labels. A dense correspondence between points of 3D faces is usually required to build the template model. For example, in Ramanathan *et al.* (2006) a correspondence is established between faces with expression and their neutral pair by minimizing an energy function. A *morphable expression model* (MEM) is constructed by applying the PCA to different expressions, so that new expressions can be projected into points in a low dimensional space constructed by the eigen-expressions obtained by MEM. Expression classification is performed by comparing the Euclidean distances among projected points in the eigen-expression space, and a recognition rate of over 97% is reported on a small and private data set (just 25 subjects with 4 expressions per subject are included in the data set). An Approach inspired by the advances in the artificial intelligence techniques such as *ant colony* optimization (ACO) and *particle swarm* optimization (PSO) is proposed in Mpiperis *et al.* (2008c). In this work, first anatomical correspondence between faces is established using a generic 3D deformable model and the 83 manually detected facial landmarks of the BU-3DFE database. Then, surface points are used as a basis for classification, according to a set of classification rules that are discovered by an ACO/PSO-based rule-discovery algorithm. The performance of the algorithm evaluated on the BU-3DFE database scored a total recognition rate of 92.3%. In Mpiperis *et al.* (2008b), face recognition and facial expression recognition are performed jointly by decoupling identity and expression components with a bilinear model. An elastically deformable model algorithm that establishes correspondence among a set of faces is proposed. Construction of the model relies on manually identified landmarks that are used to establish points correspondence in the training stage. Fitting these models to unknown faces enables face recognition invariant to facial expressions and facial expression recognition with unknown identity. A quantitative evaluation of the technique is conducted on the BU-3DFE database with an overall 90.5% facial expression recognition. In Gong *et al.* (2009), the shape of an expressional 3D face is approximated as the sum of a basic facial shape component, representing the basic face structure and neutral-style shape, and an expressional shape component that contains shape changes caused by facial expressions. The two components are separated by first learning a reference face for each input non-neutral 3D face then, on the basis of the reference face and the original expressional face, a facial expression descriptor is constructed, which accounts for the depth changes of rectangular regions around eyes and mouth. Average recognition rates of 71.63% and 76.22% have been reported on the BU-3DFE database, respectively, not using and using a reference

neutral scan for each subject.

Approaches in the second category, extract features from 3D scans and use these features to classify the facial scans into different expressions. In Wang *et al.* (2006), a feature-based facial expression descriptor is proposed and the BU-3DFE database is used for the first time. The face is subdivided into seven regions using manually annotated landmarks and classified primitive surface features are classified into basic categories, such as *ridge*, *ravine*, *peak*, and *saddle*, using surface curvatures and their principal directions. They reported the highest average recognition rate of 83.6% using the primitive facial surface features and an LDA classifier. The facial expressions of *happiness* and *surprise* were reported to be the best well-identified with accuracies of 95% and 90.8%, respectively. Comparison with the results obtained using the *Gabor-wavelet* and the *Topographic Context* 2D appearance feature-based methods on the same database showed that the 3D solution outperforms the 2D methods. Soyel and Demirel (2007) also performed 3D facial expression recognition on the BU-3DFE database. Among the 83 facial landmarks labeling the 3D faces of the BU-3DFE database, only six distance measures maximizing the differences of facial expressions were selected. These six distance values were used to form a distance vector for the representation of facial expressions as defined by the MPEG-4 *Facial Definition Parameter Set* (Pandzic and Forchheimer, 2005). The results obtained from a neural-network classifier using the 3D distance vectors reached up to 98.3% in the recognition of the *surprise* facial expression, whereas the average recognition performance is 91.3%. Tang and Huang (2008) first extracted a set of candidate features composed of normalized Euclidean distances between the 83 facial landmarks of the BU-3DFE database. Then they used a feature-selection method on the basis of the maximizing the average relative entropy of marginalized class-conditional feature distributions to retain only the most informative distances. Using a regularized multiclass *AdaBoost* classification algorithm, they obtained a 95.1% average recognition rate for the six basic facial expressions on a subset of the BU-3DFE database. The neutral facial expression was not classified rather, as a preprocessing step, its features served as fiducial measures that were subtracted from the features of the six basic facial expressions of the corresponding subject. The approach proposed in Venkatesh *et al.* (2009) on the other hand, used a modified PCA to classify facial expressions using only the shape information at a finite set of fiducial points that were extracted from the 3D neutral and expressive faces of the BU-3DFE database. The approach used 2D texture images of the face to mark interest

regions around the eyebrows, eyes, nose, and mouth, and extracted facial contours in those regions with the help of an active contour algorithm. Then, these contours were uniformly sampled, and the sampled points were mapped onto the 3D data set to generate a shape and color descriptor of the interest-regions. An average recognition rate of 81.67% was reported. Maalej *et al.* (2010) proposed an approach based on the shape analysis of local facial patches. The patches were extracted around the 83 manually annotated facial landmarks of the BU-3DFE database, and the shape of each patch described by a set of curves representing the surface points at the same Euclidean distance from the landmark. A Riemannian framework was then applied to compare the shape of curves undergoing to different facial expressions. The length of the geodesic path that separates corresponding curves provided quantitative information about their shape similarity. The best expression recognition results on the BU-3DFE database have been obtained using these measures as entries of a MultiBoost classifier. An improved version of this approach is reported in Maalej *et al.* (2011) with new experiments and results given. The work presented in Berretti *et al.* (2010e) exploits the local characteristics of the face around a set of facial landmarks to classify facial expressions. In particular, the facial landmarks are a subset of the 83 facial landmarks of the BU-3DFE plus a set of facial keypoints automatically identified starting from the given landmarks. SIFT descriptors are computed around the facial keypoints, combined together and used as feature vector to represent the face. Before to perform classification of the extracted descriptors, a feature selection approach is used to identify a subset of features with *minimal-redundancy* and *maximal-relevance* among the large set of features extracted with SIFT. The set of selected features is finally used to feed a set of classifiers on the basis of *support vector machines* (SVM).

From the previous discussion, it emerges that the large part of existing works on 3D facial expression recognition relies on the presence of landmarks accurately identified on the face. Methods based on the *generic facial model* use landmarks to establish correspondences between faces in the construction of a deformable template face. Usually, these approaches are also computationally demanding because of the deformation process. Solutions based on *feature classification* in many cases compute distances between landmarks and evaluate how these distances change between expressional and neutral scans. That several landmarks are not automatically detectable and the precision required for their positioning demand for manual annotation in both training and testing stages. Furthermore, several solutions require a neutral scan for each subject in order to

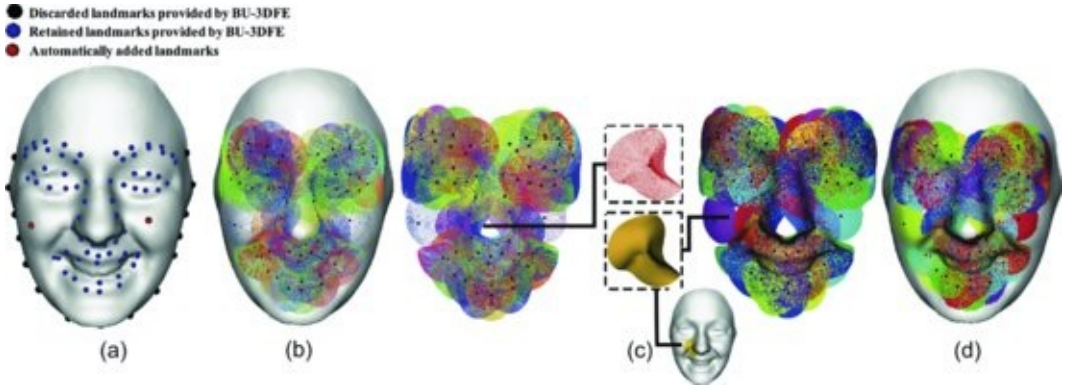
evaluate the differences generated in the 3D scans by facial expressions with respect to neutral reference scans. In practice, these factors limit the applicability of many approaches.

In the following sections, we provide more details on two facial expression recognition methods that are semi-automatic and fully automatic.

5.4.2 Semi-automatic 3D Facial Expression Recognition

In the following paragraphs, we discuss an approach that uses local descriptors called *local patches* of the face represented by a set of local curves to perform person independent 3D facial expression recognition. This approach was originally proposed in Maalej *et al.* (2010, 2011). In this work, sets of level curves $\{c_\lambda^l\}_{1 \le \lambda \le \lambda_0}$ are associated to *N* reference points (landmarks) $\{r_l\}_{1 \le l \le N}$ (Figure 5.11(a)) (Figure 5.11(b)).

Figure 5.11 (a) 3D annotated facial shape model (70 landmarks); (b) 3D closed curves extracted around the landmarks; (c) 3D curve-based patches composed of 20 level curves with a size fixed by a radius $\lambda_0 = 20$ mm; (d) extracted patches on the face



These curves are extracted over the patches centered at these points. Here λ stands for the value of the distance function between the reference point r_l and the point belonging to the curve c_λ^l, and λ_0 stands for the maximum value taken by λ. Accompanying each facial model are 83 manually picked landmarks; these landmarks are practically similar to the MPEG-4 feature points and are selected on the basis of the facial anatomy structure. Given these points, the feature region on the face can be easily determined and extracted. We were interested in a subset of 68 landmarks laying within the face area, discarding those marked on the face border. Contrary to the MPEG-4 feature points specification that annotates the cheeks center and bone, in BU-3DFE there were

no landmarks associated with the cheek regions. Thus, two extra landmarks at both cheeks, obtained by extracting the middle point along the geodesic path between the mouth corner and the outside eye corner was added.

We propose to represent each facial scan by a set of patches around the landmarks. Let r_i be the reference point and P_i a given patch centered on this point and localized on the facial surface denoted by S. Each patch will be represented by an indexed collection of level curves. To extract these curves, we use the Euclidean distance function $\|r_i - p\|$ to characterize the length between r_i and any point p on S. Indeed, unlike the geodesic distance, the Euclidean distance is sensitive to deformations. Besides it enables deriving curve extraction in a fast and simple way. Using this function, the curves as level sets is defined as the following:

(5.5) $$\|r_i - .\| : c_\lambda^i = \{p \in S \mid \|r_i - p\| = \lambda\} \subset S, \quad \lambda \in [0, \lambda_0].$$

Each c_λ^i is a closed curve, consisting of a collection of points situated at an equal distance λ from r_i. Figure 5.11 summarizes the scheme of patches extraction.

Framework for 3D Shape Analysis

Once the patches are extracted, we aim to study their shape and design and a similarity measure between corresponding ones on different scans under different expressions. This is motivated by the common belief that people smile, or convey any other expression, the same way, or more appropriately certain regions taking part in a specific expression undergo practically the same dynamical deformation process. We expect that certain corresponding patches associated with the same given expression will be deformed in a similar way, whereas those associated with two different expressions will deform differently. The following sections describe the shape analysis of closed curves in $\mathbb{R}^3$, initially introduced by Joshi *et al.* (2007), and its extension to analyze shape of local patches on facial surfaces.

We start by considering a closed curve β in $\mathbb{R}^3$. Although there are several ways to analyze shapes of closed curves, an elastic analysis of the parametrized curves is particularly appropriate in 3D curves analysis. This is because (1) such analysis uses a square-root velocity function representation which allows us to compare local facial shapes in presence of elastic deformations, (2) this method uses a square-root representation under which the elastic metric reduces to the standard $\mathbb{L}^2$ metric and thus simplifies the analysis, and (3) under this metric the Riemannian distance between curves is invariant to the reparametrization. To

analyze the shape of β, we shall represent it mathematically using a square-root representation of β as follows ; for an interval *I*=[0, 1], let $\beta : I \longrightarrow \mathbb{R}^3$ be a curve and define $q : I \longrightarrow \mathbb{R}^3$ to be its square-root velocity function (SRVF), given by

$$q(t) \doteq \frac{\dot{\beta}(t)}{\sqrt{\|\dot{\beta}(t)\|}}. \quad (5.6)$$

Here *t* is a parameter $\in I$ and $\|.\|$ is the Euclidean norm in $\mathbb{R}^3$. We note that *q*(*t*) is a special function that captures the shape of β and is particularly convenient for shape analysis, as we describe next. The classical elastic metric for comparing shapes of curves becomes the $\mathbb{L}^2$-metric under the SRVF representation (Srivastava et al., 2011). This point is very important as it simplifies the calculus of elastic metric to the well-known calculus of functional analysis under the $\mathbb{L}^2$-metric. Also, the squared $\mathbb{L}^2$-norm of *q*, given by: $\|q\|^2 = \int_{\mathbb{S}^1} < q(t), q(t) > \mathrm{d}t = \int_{\mathbb{S}^1} \|\dot{\beta}(t)\| \mathrm{d}t,$ which is the length of β. To restrict our shape analysis to closed curves, we define the set: $\mathcal{C} = \{q : \mathbb{S}^1 \longrightarrow \mathbb{R}^3 | \int_{\mathbb{S}^1} q(t)\|q(t)\| \mathrm{d}t = 0\} \subset \mathbb{L}^2(\mathbb{S}^1, \mathbb{R}^3)$. Notice that the elements of $\mathcal{C}$ are allowed to have different lengths. Because of a nonlinear (closure) constraint on its elements, $\mathcal{C}$ is a nonlinear manifold. We can make it a Riemannian manifold by using the metric: For any $u, v \in T_q(\mathcal{C})$, we define

$$\langle u, v \rangle = \int_{\mathbb{S}^1} \langle u(t), v(t) \rangle \mathrm{d}t. \quad (5.7)$$

So far we have described a set of closed curves and have endowed it with a Riemannian structure. Next we consider the issue of representing the *shapes* of these curves. It is easy to see that several elements of $\mathcal{C}$ can represent curves with the same shape. For example, if we rotate a curve in $\mathbb{R}^3$, we get a different SRVF but its shape remains unchanged. Another similar situation arises when a curve is reparametrized; a reparameterization changes the SRVF of curve but not its shape. To handle this variability, we define orbits of the rotation group *SO*(3), and the reparameterization group Γ as the equivalence classes in $\mathcal{C}$. Here, Γ is the set of all orientation-preserving diffeomorphisms of $\mathbb{S}^1$ (to itself) and the elements of Γ are viewed as reparameterization functions. For example, for a curve $\beta : \mathbb{S}^1 \to \mathbb{R}^3$ and a function $\gamma : \mathbb{S}^1 \to \mathbb{S}^1$, $\gamma \in \Gamma$, the curve $\beta \circ \gamma$ is a reparameterization of β. The corresponding SRVF changes according to $q(t) \mapsto \sqrt{\dot{\gamma}(t)} q(\gamma(t))$. We set the elements of the orbit

$$[q] = \{\sqrt{\dot{\gamma}(t)} O q(\gamma(t)) | O \in SO(3), \ \gamma \in \Gamma\} \quad (5.8)$$

to be equivalent from the perspective of shape analysis. The set of such equivalence classes, denoted by $\mathcal{S} \doteq \mathcal{C}/(SO(3) \times \Gamma)$ is called the *shape space* of closed curves in $\mathbb{R}^3$. $\mathcal{S}$ inherits a Riemannian metric from the larger space $\mathcal{C}$ because of the quotient structure.

The main ingredient in comparing and analysing shapes of curves is the construction of a geodesic between any two elements of $\mathcal{S}$, under the Riemannian metric given in Equation 5.7. Given any two curves β_1 and β_2 represented by their SRVFs q_1 and q_2, we want to compute a geodesic path between the orbits $[q_1]$ and $[q_2]$ in the shape space $\mathcal{S}$. This task is accomplished using a *path-straightening approach*, which was introduced in Klassen and Srivastava (2006). The basic idea here is to connect the two points $[q_1]$ and $[q_2]$ by an arbitrary initial path α and to iteratively update this path using the negative gradient of an energy function $E[\alpha] = \frac{1}{2}\int_s \langle \dot{\alpha}(s), \dot{\alpha}(s) \rangle ds$. The interesting part is that the gradient of E has been derived analytically and can be used directly for updating α. As shown in Klassen and Srivastava (2006), the critical points of E are actually geodesic paths in $\mathcal{S}$. Thus, this gradient-based update leads to a critical point of E, which, in turn, is a geodesic path between the given points. In the remainder of the chapter, we will use the notation $d_{\mathcal{S}}(\beta_1, \beta_2)$ to denote the length of the geodesic in the *shape space* $\mathcal{S}$ between orbits q_1 and q_2 to reduce the notation.

3D Patches Shape Analysis

Now, we extend ideas developed in the previous section from analyzing shapes of curves to the shapes of patches. As mentioned earlier, we are going to represent a number of l patches of a facial surface S with an indexed collection of the level curves of the $\|r_l - .\|$ function (Euclidean distance from the reference point r_l). That is, $P_l \leftrightarrow \{c_\lambda^l, \lambda \in [0, \lambda_0]\}$, where c_λ^l is the level set associated with $\|r_l - .\| = \lambda$. Through this relation, each patch has been represented as an element of the set $\mathcal{S}^{[0,\lambda_0]}$. In our framework, the shapes of any two patches are compared by comparing their corresponding level curves. Given any two patches P_1 and P_2, and their level curves $\{c_\lambda^1, \lambda \in [0, \lambda_0]\}$ and $\{c_\lambda^2, \lambda \in [0, \lambda_0]\}$, respectively, our idea is to compare the patches curves c_λ^1 and c_λ^2, and to accumulate these differences over all λ. More formally, we define a distance $d_{\mathcal{S}^{[0,\lambda_0]}}$ given by

(5.9) $$d_{\mathcal{S}^{[0,\lambda_0]}}(P_1, P_2) = \int_0^L d_{\mathcal{S}}(c_\lambda^1, c_\lambda^2) d\lambda.$$

In addition to the distance $d_{\mathcal{S}^{[0,\lambda_0]}}(P_1, P_2)$, which is useful in biometry and other classification experiments, we also have a geodesic path in $\mathcal{S}^{[0,\lambda_0]}$ between the two points represented by P_1 and P_2. This geodesic corresponds to the optimal elastic deformations of facial curves and, thus, facial surfaces from one to another. Figure 5.12 shows some examples of geodesic paths that are computed between corresponding patches associated with shape models sharing the same expression, and termed *intraclass geodesics*. In the first column we illustrate the source, which represents scan models of the same subject, but under different expressions. The third column represents the targets as scan models of different subjects. As for the middle column, it shows the geodesic paths. In each row we have both the shape and the mean curvature mapping representations of the patches along the geodesic path from the source to the target. The mean curvature representation is added to identify concave/convex areas on the source and target patches and equally spaced steps of geodesics. This figure shows that certain patches, belonging to the same class of expression, are deformed in a similar way. In contrast, Figure 5.13 shows geodesic paths between patches of different facial expressions. These geodesics are termed *interclass geodesics*. Unlike the intraclass geodesics shown in Figure 5.12, these patches deform in a different way.

Figure 5.12 Examples of intraclass (same expression) geodesic paths with shape and mean curvature mapping between corresponding patches

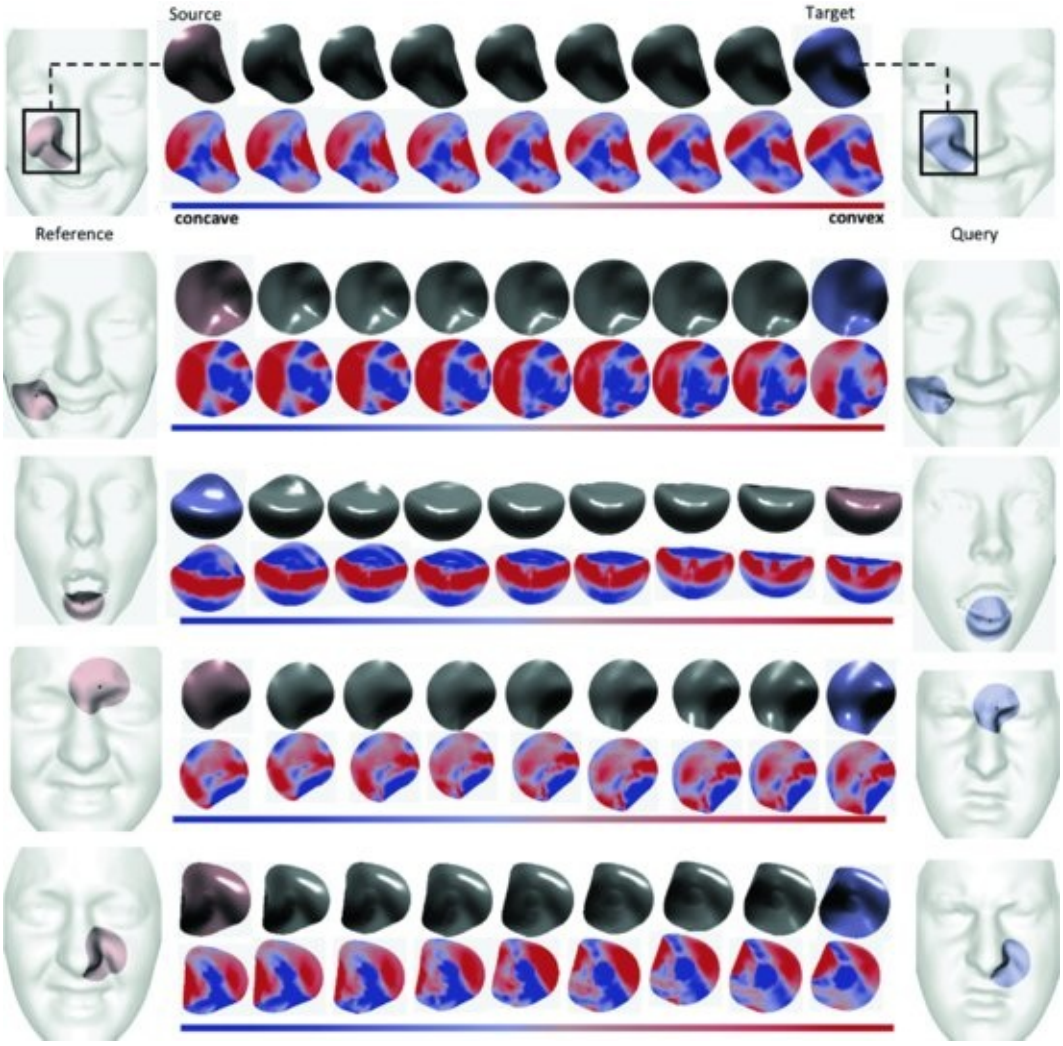
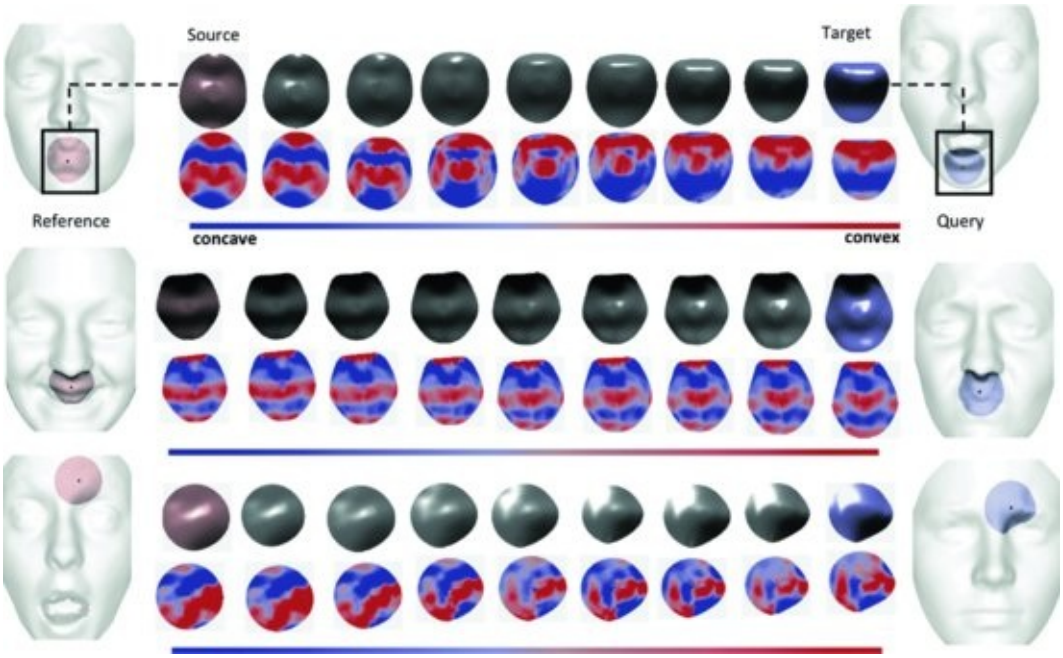


Figure 5.13 Examples of interclass (different expressions) geodesic paths between source and target patches



Feature Vector Generation for Classification

To classify expressions, we build a feature vector for each facial scan. Given a candidate facial scan of a person *j*, facial patches are extracted around facial landmarks. For a facial patch P^i_j, a set of level curves $\{c_\lambda\}^i_j$ are extracted centered on the *i*th landmark. Similarly, a patch P^i_{ref} is extracted in correspondence to landmarks of a reference scans *ref*. The length of the geodesic path between each level curve and its corresponding curve on the reference scan are computed using a Riemannian framework for shape analysis of 3D curves. The shortest path between two patches at landmark *i*, one in a candidate scan and the other in the reference scan, is defined as the sum of the distances between all pairs of corresponding curves in the two patches as indicated in Equation 5.9. The feature vector is then formed by the distances computed on all the patches and its dimension is equal to the number of used landmarks *N*=70 (i.e., 68 landmarks are used out of the 83 provided by BU-3DFED and the two additional cheek points). The *i*th element of this vector represents the length of the geodesic path that separates the relative patch to the corresponding one on the reference face scan. All feature vectors computed on the overall data set will be labeled and used as input data to machine learning algorithms such as MultiBoosting and SVM, where MultiBoosting is an extension of the successful Adadoost technique for forming decision committees.

Recognition Experiments

Recognition Experiments

To investigate facial expression aforementioned, the above approach is applied to a data set that is appropriate for this task. In this section, we describe the experiments, obtained results, and comparisons with related work.

For the goal of performing identity-independent facial expression recognition, the experiments were conducted on the BU-3DFE static database. A data set captured from 60 subjects were used, half (30) of them were female and the other half (30) male, corresponding to the high and highest intensity levels 3D expressive models (03–04). These data are assumed to be scaled to the true physical dimensions of the captured human faces. Following a similar setup as in Gong *et al.* (2009), we randomly divided the 60 subjects into two sets, the training set containing 54 subjects (648 samples), and the test set containing 6 subjects (72 samples).

To drive the classification experiments, we arbitrarily choose a set of six reference subjects with its six basic facial expressions. We point out that the selected reference scans do not appear either in the training or in the testing set. These references, with their relative expressive scans corresponding to the highest intensity level, are taken to play the role of representative models for each of the six classes of expressions. For each reference subject, we derive a facial expression recognition experience.

Several facial expression recognition experiments were conducted with changing at each time the reference. Using the *Waikato Environment for Knowledge Analysis (Weka)* (Hall et al., 2009), we applied the MultiBoost algorithm with three weak classifiers, namely, Linear Discriminant Analysis (LDA), Naive Bayes (NB), and Nearest Neighbor (NN), to the extracted features, and we achieved average recognition rates of 98.81%, 98.76%, and 98.07%. We applied the SVM linear classifier as well, and we achieved an average recognition rate of 97.75%. We summarize the resulting recognition rates in Table 5.7.

Table 5.7 Classification results using local shape analysis and several classifiers

	MultiBoost LDA	MultiBoost NB	MultiBoost NN	SVM-Linear
Recognition rate	**98.81%**	98.76%	98.07%	97.75%

We note that these rates are obtained by averaging the results of the 10 independent and arbitrarily run experiments (10-fold cross validation) and their respective recognition rate obtained using the MultiBoost–LDA classifier. We note that different selections of the reference scans do not affect significantly the

recognition results and that there is no large variation in recognition rates values. The reported results represent the average over the six performed experiments. The MultiBoost–LDA classifier achieves the highest recognition rate and shows a better performance in terms of accuracy than do the other classifiers. This is mainly as a result of the capability of the LDA-based classifier to transform the features into a more discriminative space and, consequently, result in a better linear separation between facial expression classes.

The average confusion matrix relative to the best performing classification using MultiBoost–LDA is given in Table 5.8.

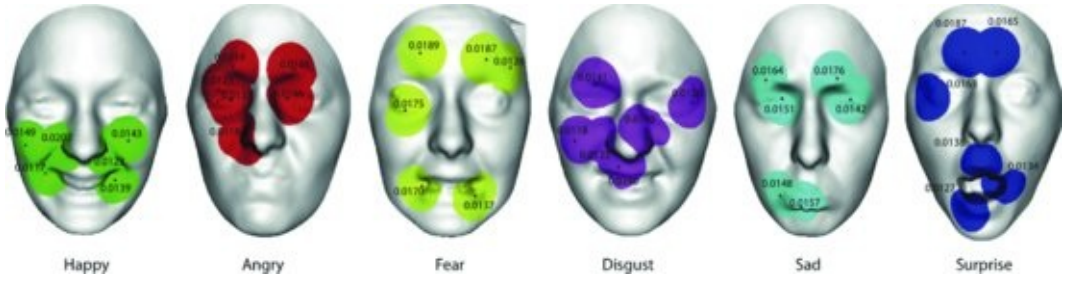
Table 5.8 Average confusion matrix given by MultiBoost-LDA classifier

	AN	DI	FE	HA	SA	SU
AN	**97.92**	1.11	0.14	0.14	0.69	0.0
DI	0.56	**99.16**	0.14	0.0	0.14	0.0
FE	0.14	0.14	**99.72**	0.0	0.0	0.0
HA	0.56	0.14	0.0	**98.60**	0.56	0.14
SA	0.28	0.14	0.0	0.0	**99.30**	0.28
SU	0.14	0.56	0.0	0.0	1.11	**98.19**

To better understand and explain the results mentioned earlier, the Multiboost algorithm is applied on feature vectors built from distances between patches for each class of expression. In this case, we consider these features as weak classifiers. Then, we look at the early iterations of the MultiBoost algorithm and the selected patches in each iteration.

Figure 5.14 illustrates for each class of expression the most relevant patches. Notice that, for example, for the *happy* expression the selected patches are localized in the lower part of the face, around the mouth and the chin. As for the *surprise* expression, we can see that most relevant patches are localized around the eyebrows and the mouth region. It can be seen that patches selected for each expression lie on facial muscles that contribute to this expression.

Figure 5.14 Selected patches at the early few iterations of MultiBoost classifier for the six facial expressions (anger, disgust, fear, happiness, sadness, and surprise) with their associated weights



Comparison with Related Work

Table 5.9 shows a comparison of semi-automatic approaches proposed in the state-of-the-art methods using the same experimental setting (54-versus-6-subject partitions) of the BU-3DFE database. In general, results are obtained by averaging many trials where the subjects partitioning between train and test are randomly changed. Although not all the methods use the same number of trials, and just the work in Maalej *et al.* (2011) evidences the differences between different trials, we can assume that the reported results can be compared giving an idea of the effectiveness of the different solutions.

Table 5.9 Comparison of state of the art methods on the BU-3DFE data set

	Maalej et al. (2011)	Gong et al. (2009)	Berretti et al. (2011a)	Zhao et al. (2011)
RR	**98.0 ±1.6%**	76.2%	77.5%	82.0%

5.4.3 Fully Automatic 3D Facial Expression Recognition

Some recent works have shown that local descriptors computed at salient keypoints of 3D objects can be usefully applied to capture relevant 3D shape features. For example, in (Mian et al., 2008) a 3D keypoint detector and descriptor has been defined for the purpose of 3D face recognition. In (Mayo and Zhang, 2009), SIFT keypoints detected on multiple 2D depth images have been used to perform 3D face recognition. SIFT descriptors computed on a sampling grid of points in 2D depth images have been used in (Ohbuchi and Furuya, 2004) for 3D object retrieval by visual similarity. Finally, SIFT descriptors have been also used in (Zheng et al., 2009) to perform 2D expression recognition from nonfrontal face images.

Grounding on these studies, in the following we discuss an approach that uses local descriptors of the face to perform person independent 3D facial expression recognition. This approach has been originally proposed in (Berretti et al., 2010e), (Berretti et al., 2010), and subsequently developed to a completely automatic solution that exploits the local characteristics of the face around a set of facial keypoints automatically detected (Berretti et al., 2011a). In this solution, some facial landmarks are first identified, and then SIFT descriptors computed at these landmarks are combined together as feature vector representing the face. A feature selection approach is then applied to these vectors inorder to extract the subset of most relevant features, and the selected features are finally classified using SVMs. As it emerges from the experimental

evaluation, this approach is capable to achieve state of the art results on the BU-3DFE database just relying on few keypoints that are automatically detected and without using neutral scans as reference.

In the rest of this section, we will first briefly provide a solution for the automatic identification of facial keypoints. Then we will address the adaptation of SIFT descriptors to the proposed case and the feature selection approach used to reduce the set of SIFT features and the SVM based classification of the selected features. We will also provide a summary of the results obtained with this approach.

Automatic Identification of Facial Keypoints

The BU-3DFE database is the standard benchmark to compare 3D facial expression recognition algorithms (see Section 5.2). However, the fact that this database provides a set of manually identified landmarks, and the inherent difficulty in automatically detecting the majority of these landmarks has oriented the research towards semi-automatic solutions for 3D facial expression recognition as illustrated in Section 5.4.2. In semi-automatic solutions, the position of facial landmarks is assumed to be known to achieve high facial expression recognition rates (see Section 5.4.1), but this hinders the applicability of these solutions to the general case in which manual annotation of the landmarks in 3D is not available or even possible. To overcome this limitation, in Berretti *et al.* (2011a) a completely automatic solution to identify fiducial points of the face is proposed, which is shortly reviewed in the following paragraphs.

As first preprocessing step, the 3D face scans were transformed to *depth images* where the gray-value of each image pixel represents the depth of the corresponding point on the 3D surface. As an example, Figure 5.15 shows the depth images derived from the 3D face scans of a same subject under three different facial expressions.

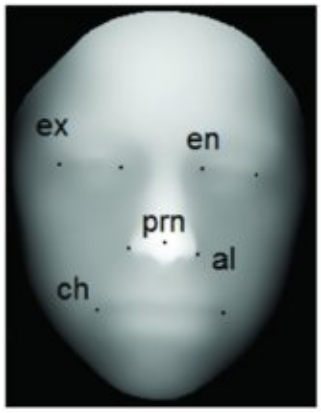
Figure 5.15 BU-3DFE database: depth images derived from the 3D face scans of subject. The highest level of intensity is reported for expressions *anger*, *disgust*, and *fear*.

On the depth images, the point with maximum gray value has been used as initial estimate of the tip of the nose. This point was used to crop a rectangular region of the face (following anthropometric statistical measures (Farkas, 1994), the cropped region extends 50 mm on the left and 50 mm on the right of the nose tip, and 70 mm above and 50 mm below the nose tip). The cropped region of the face is used for all the subsequent steps of the processing.

Then the approach starts from the consideration that just a few fiducial points of the face can be automatically identified with sufficient robustness across different individuals and ethnicities. This is supported by recent studies as that in Gupta *et al.* (2010), where methods are given to automatically identify 10 facial fiducial points on the 3D face scans of the *Texas 3D Face Recognition Database*. Following this idea, a general method to automatically identify nine keypoints of the face is given. The points are the tip of the nose (*pronasale*, *prn*), the two points that define the nose width (*alare*, *al*), the points at the *inner* and *outer* eyes (*endocanthion*, *en* and *exocanthion*, *ex*, respectively), and the outer mouth points (*cheilion*, *ch*), as evidenced on the 3D face scan and depth image of Figures 5.16**a,b**.

Figure 5.16 BU-3DFE database: The 9 facial keypoints that are automatically detected. Points are evidenced on a textured 3D face scan and on the depth image of the scan in (a).

(a)



(b)

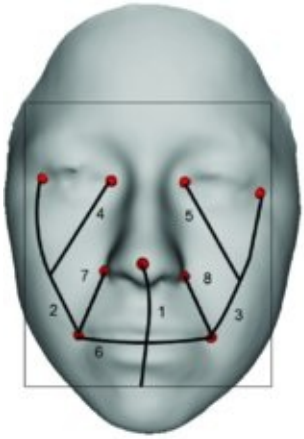
Different algorithms are used in order to detect the keypoints.

- For the nose tip and the two alare points, the proposed solution develops on the methods given in Gupta *et al.* (2010). In particular, the nose tip is determined in the convex part of the central region of the face as the point with a local maximum of the elliptic Gaussian curvature, which is closest to the initial estimate of the nose tip. For the alare points, first the edges of the facial depth images are identified using a *Laplacian of Gaussian* (LoG) edge detector and points along the nasal boundary with high curvature ("critical" points) and negative curvature values are detected. Then, the two alare points are selected as the outer left and outer right critical points.
- For the remaining points (inner and outer eyes and outer mouth) a solution based on the SIFT detector applied to local search windows is applied. In particular, the extraction of these points proceeds in cascade using the location of the nose tip and the alare points to identify a set of the search windows on the face. In the *en-en*, *ex-ex* and *ch-ch* search windows the SIFT *detector* algorithm is ran. In fact, SIFT has been defined on 2D grayscale images and includes a *keypoints detector* and a *feature extractor* (Lowe, 2004). By definition, keypoints detected by SIFT are mainly located at corner points of an image, so that they can be useful to capture significant anthropometric face points. The SIFT point detected at the *highest scale* is retained as the keypoint of the search window. Experiments on the accuracy of the keypoints detection using this approach can be found in Berretti *et al.* (2011a).

Face Representation with Selected SIFT Features at Facial Keypoints

The nine keypoints automatically detected are used as reference to derive a set of sampling points of the face. This is obtained by considering 8 lines that connect pairs of keypoints, as shown in Figure 5.17**a**. In particular, these lines connect the *nose tip* to the lower point of the face (*line 1*), the *outer mouth* with the *outer eyes* (*lines 2, 3*), the *inner eyes* with the mid points of lines 2 and 3 (*lines 4, 5*), the *outer mouth* points each other (*line 6*), and the *alare* points with the *outer mouth* (*lines 7, 8*). Lines are sampled uniformly with a different number of points as reported in the table of Figure 5.17**b**. According to this, the face is sampled with a total number of 79 keypoints.

Figure 5.17 (a) The eight lines along which the sample points are located (the cropped region of the face is also reported); (b) the number of points and their indices grouped according to the surface line they belong to. Copyright © 2011, Springer



(a)

line	*#points*	*indices*
1	12	1–12
2	16	13–28
3	16	29–44
4	9	45–53
5	9	54–62
6	9	63–71
7	4	72–75
8	4	76–79

(b)

To capture salient features that characterize different facial expressions in 3D, local descriptors are computed around the 79 sample points of the face. The SIFT *feature extraction* algorithm has been used for this purpose to derive SIFT *descriptors*. By computing the 128-dimensional SIFT descriptor at each of the 79 keypoints, a feature vector with 10,112 components is obtained to represent each depth image.

To reduce the dimensionality and improve the significativeness of the features, only the features with maximal relevance and minimal redundancy have been selected using a *feature selection* analysis. In particular, feature selection is performed using the *minimal redundancy maximal relevance* (mRMR) model (Peng et al., 2005). For a given classification task, the aim of mRMR is to select a subset of features by taking into account the ability of features to identify the

classification label, as well as the redundancy among the features. These concepts are defined in terms of the *mutual information* between features.

In our approach, the mRMR algorithm is applied to the set of 10112-dimensional feature vectors representing the faces. Each vector $v_f = (f_1, \ldots, f_{10112})$ is constructed by concatenating the 128-dimensional SIFT descriptors computed at the face keypoints, orderly from 1 to 79. A data discretization into three levels is applied to the vectors as preprocessing step. This is obtained by computing the mean value μ_k and the standard deviation σ_k for every feature f_k. Then, discretized values $\hat{f}_k$ are obtained. The overall set of discretized feature vectors is used to feed the mRMR algorithm so as to determine the features which are most relevant in discriminating between different facial expressions of 3D face scans of different subjects.

The facial expression recognition problem is a multiclassification task that is faced as a combination of separated instances of *one-versus-all* classification subproblems. For each subproblem, face scans showing one expression are assumed as targets (positive examples), whereas all the other scans with any different expression are considered as negative examples. Repeatedly, the target expression is changed among the six basic expressions provided by the BU-3DFE database, hence, the sets of positive and negative examples change. Because of this, mRMR feature selection is performed independently for each classification subproblem. In general, this results into different features providing the minimal redundancy and maximal relevance for the purpose of discriminating across different facial expressions. Then, just the most relevant features identified for every expression are retained from the original feature vectors in order to train the classifiers. This results into vectors $v_{\hat{f}}^{\mathrm{expr}} = (\hat{f}_{p_1}, \ldots, \hat{f}_{p_{N\,expr}})$, where $p_1, \ldots, p_{N\,expr}$ are the indices of the features components selected in the original vector, and the subscript the label of a particular expression.

The selected features are then used to perform facial expression recognition using a *maxima rule* between six *one-versus-all* SVM classifiers, each with a radial basis function kernel of standard deviation equal to one (the publicly available SVMLight implementation of SVM has been used: http://svmlight.joachims.org/).

5.5 4D Facial Expression Recognition

The use of 4D data for face analysis applications is still at its nascent stages,

with no research on face recognition from sequences of 3D face scans and very little research focusing on facial expression recognition. The first approach addressing the problem of facial expression recognition from dynamic sequences of 3D scans was proposed by Sun *et al.* Sun and Yin (2008). Their approach basically relies on the use of a generic deformable 3D model whose changes are tracked both in space and time in order to extract a spatiotemporal descriptor of the face. In the temporal analysis, a vertex flow tracking technique is applied to adapt the 3D deformable model to each frame of the 3D face sequences, and the vertex flow estimation is derived by establishing point to point correspondences between 3D meshes on the basis of a conformal mapping approach. The vertex correspondence across 3D model sequences provides a set of motion trajectories (vertex flow) of 3D face scans. Consequently, the vertex flow can be depicted on the adapted generic model (tracking model) through estimation of the displacement vector from the tracked points of the current frame to the corresponding points of the first frame (assumed to have a neutral expression). A facial motion vector is then obtained to describe the dynamics of facial expression across a 3D frame sequence. In the spatial analysis, an automatic surface labelling approach is applied on the tracked locations of the range models in order to classify the 3D primitive surface features into eight basic categories. As a result, each depth scan in the sequence can be represented by a spatiotemporal feature vector that describes both shape and motion information and provides a robust facial surface representation. Once spatiotemporal features are extracted, a two-dimensional Hidden Markov Model (HMM) is used for classification. In particular, a spatial HMM and a temporal HMM were used to model the spatial and temporal relationships between the extracted features. Exhaustive analysis was performed on the BU-4DFE database, with a reported average recognition rate equal to 83.7% for identity-independent facial expression recognition. The main limit of this solution resides in the use of the 83 manually annotated landmarks of the BU-4DFE that are not released for public use.

The approach proposed by Sandbach *et al.* (2011) exploits the dynamics of 3D facial movements to analyse expressions. This is obtained by first capturing motion between frames using free-form deformations and extracting motion features using a quad-tree decomposition of several motion fields. GentleBoost classifiers are used to simultaneously select the best features to use and perform the training using two classifiers for each expression: one for the onset temporal segment, and the other for the offset segment. Then, HMMs are used for

temporal modeling of the full expression sequence, which is represented as the composition of four temporal segments, namely, neutral-onset-apex-offset, which model a sequence with an initial neutral segment followed by the activation of the expression, maximum intensity of the expression, deactivation of the expression and closing of the sequence again with a neutral expression. The average correct classification results for three prototypic expressions (i.e., happiness, anger, surprise) of the BU-4DFE database is equal to 81.93%.

In Le *et al.* (2011) a level curve–based approach is proposed to capture the shape of 3D facial models. The level curves are parametrization using the arclength function. The Chamfer distance is applied to measure the distances between the corresponding normalized segments, partitioned from these level curves of two 3D facial shapes. These measures are then used as spatiotemporal features to train HMM, and since the training data were not sufficient for learning HMM, the authors proposed to apply the universal background modeling to overcome the overfitting problem. Using the BU-4DFE database to evaluate their approach, they reached an overall recognition accuracy of 92.22% for three prototypic expressions (i.e., happiness, sadness, surprise).

The work of Fang *et al.* (2011) proposes a fully automatic 4D facial expression recognition approach with a particular emphasis on 4D data registration and dense correspondence between 3D meshes along the temporal line. The variant of the local binary patterns (LBP) descriptor proposed in Zhao and Pietikäinen (2007), which computes LBP on three orthogonal planes is used as face descriptor along the sequence. Results are provided on the BU-4DFE database for all expressions and for the subsets of expressions used in Sandbach *et al.* (2011) and Le *et al.* (2011), showing improved results with respect to competitor solutions.

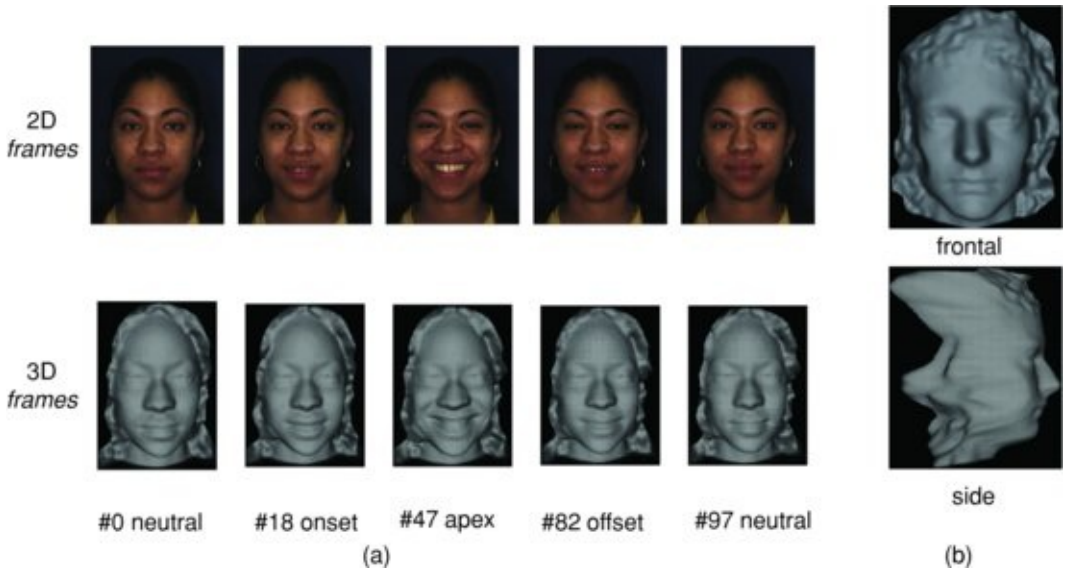
5.5.1 The BU-4DFE Database

The 3D and 4D databases used for facial expression recognition have been summarized in Section 5.2 and further details on can be found in Yin *et al.* (2008). To make the discussion simpler, we recall, in the following sections, the main features of the dynamic 3D facial expression database that recently created at Binghamton University (Yin et al., 2008).

The 3D scans have been constructed by capturing a sequence of stereo images of subjects exhibiting facial expressions and producing a depth map for each frame. The range maps are then combined to produce a temporally varying

sequence of 3D scan. Subjects were requested to perform the six prototypic expressions, separately, in such a way that each expression sequence contained neutral expressions in the beginning and at the end. In particular, each expression was performed gradually from neutral appearance, low intensity, high intensity, and back to low intensity and neutral. Each 3D sequence captures one expression at a rate of 25 frames per second and each 3D sequence lasts approximately 4 seconds with about 35,000 vertices per scan (i.e., 3D *frame*). The database consists of 101 subjects including 606 3D model sequences with six prototypic expressions. Examples of 2D and 3D frames sampled from the *happy* 4D sequences of subject F021 are given in Figure 5.18.

Figure 5.18 BU-4DFE: (a) Frames from the dynamic sequence of subject F021 (happy expression). For each row, five frames are reported out of 98 total frames of the sequence; (b) a raw 3D frame before preprocessing is shown from frontal and side view



An example of a 3D dynamic facial sequence of a subject with the *happy* expression is shown in Figure 5.18**a**, where 2D frames (not used in our solution) and 3D frames before preprocessing are reported. For each row, five frames are given (out of the 98 total frames of the specific sequence) each representing a sample of the start and end of the sequence (first and last frame), of the intensity of the facial expression in the *onset* and *offset* intervals of the sequence, and of the interval of the sequence with the larger intensity of the expression (*apex*).

From a preliminary analysis, we note that the resolution of the individual scans of 3D sequences is not very high. In fact, the average number of vertices per scan is reasonable (about 35,000), but the number of vertices used to represent

the face region is considerably lower as a result of the large outliers acquired in the hair and shoulder regions (see Figure 5.18**b**). The lack of facial geometric details makes the 3D sequences quite challenging to be used for facial expression recognition and face recognition.

5.5.2 3D Shape Motion Analysis

The raw data obtained from even the most accurate 3D scanners is far from being perfect and clean because it contains spikes, holes, and significant noise. A preprocessing step must be applied to remove these anomalies before any further operations can be performed. Thus the preprocessing is an important stage of the recognition systems, especially when knowing that all the features will be extracted from the output of this step. An automatic preprocessing pipeline is developed and applied according to the following steps:

1. Filling holes: Several BU-4DFE scans are affected with holes that often lie in the mouth region and that take part in an acquisition session that meets an open mouth expression. In this case the mouth area is not visible and cannot be acquired by the stereo sensors, which causes missing data. A linear interpolation technique is used to fill the missing regions of a given raw 3D face image.

2. Nose detection: The nose tip is a keypoint that is needed for both preprocessing and facial surface representation stages. Knowing that in most 3D images, the nose is the closest part of the face to the 3D acquisition system, in this step the nose tip is detected using horizontal and vertical slicing of the facial scan in order to search for the maximum value of the *z*-coordinate along these curved profiles. Once this is done for the first frame, for the remaining frames of the sequence this detection technique is refined and the search area in a current frame is reduced to a small sphere centered on the nose tip detected in the previous frame.

3. Cropping: Face boundaries, hair, and shoulders are irrelevant parts for the study, and they are usually affected with outliers and spikes. The nose tip detected in the previous step is used to crop out the required facial area from the raw face image. Using a sphere, centered on the nose tip and of a radius determined empirically, the 3D range model is cut and the mesh structure kept inside the sphere is retained.

4. Pose correction: In this step a global registration technique (i.e., ICP) is applied to align meshes of the current frame and the first frame of the

sequence. After this rigid alignment, the pose of the 3D face is adjusted and made similar enough to the pose in the first frame.

Geometric Facial Deformation

One basic idea to capture facial deformations across 3D video sequences is to track meshes vertices densely along successive 3D frames. Because the meshes resolutions vary across 3D video frames, establishing a dense matching on consecutive frames is necessary. For this purpose, Sun and Yin (2008) proposed to adapt a generic model (a tracking model) to each 3D frame using a set of 83 predefined facial landmarks to control the adaptation based on radial basis functions. The main limitation of this approach was that 83 landmarks were manually annotated in the first frame of each sequence. Moreover, the adaptation decreased the accuracy of expression recognition when emotions were manifested by subtle changes of the face. In another solution presented by Sandbach *et al.* (2011, 2012), the authors used an existing nonrigid registration algorithm (FFD) (Rueckert et al., 1999) on the basis of the B-splines interpolation between a lattice of control points. Here, dense matching was a preprocessing step used to estimate a motion vector field between 3D frames *t* and *t*-1. The problem of quantifying subtle deformations along the sequence still remains a challenging task, and the results presented in Sandbach *et al.* (2011) were limited to just three facial expressions: happiness, anger, and surprise.

In the following paragraphs we discuss an approach that uses a collection of radial curves to represent a 3D face. A Riemannian shape analysis is applied to effectively quantify deformations induced by facial expressions in subsequent 3D frames. This is obtained by a dense scalar field defined in Chapter 3 (Section 3.8.5), which denotes the shooting directions of the geodesics paths constructed between pairs of corresponding radial curves of two expressive faces. This approach has been originally proposed in Drira *et al.* (2012). The first step to capture the deformation between two given 3D faces F_1 and F_2 is to extract the radial curves originating from the nose tip. Let β_α^1 and β_α^2 denote the radial curves that make an angle α with a reference radial curve on faces F_1 and F_2, respectively. The tangent vector field $\dot{\psi}_\alpha^*$ that represents the energy *E* needed to deform β_α^1 to β_α^2 is then calculated for each index α. We consider the magnitude of this vector field at each point *k* of the curve to construct the DSFs of the facial surface. In this way, the DSF quantifies the local deformation between points of radial curves β_1 and β_2, respectively, of the faces F_1 and F_2. In

the practice, we represent each face with 100 radial curves and 50 points on each curve so that the DSFs between two 3D faces is expressed by a 5000-dimensional vector.

Mean Shape Deformation with Random Forest Classifier

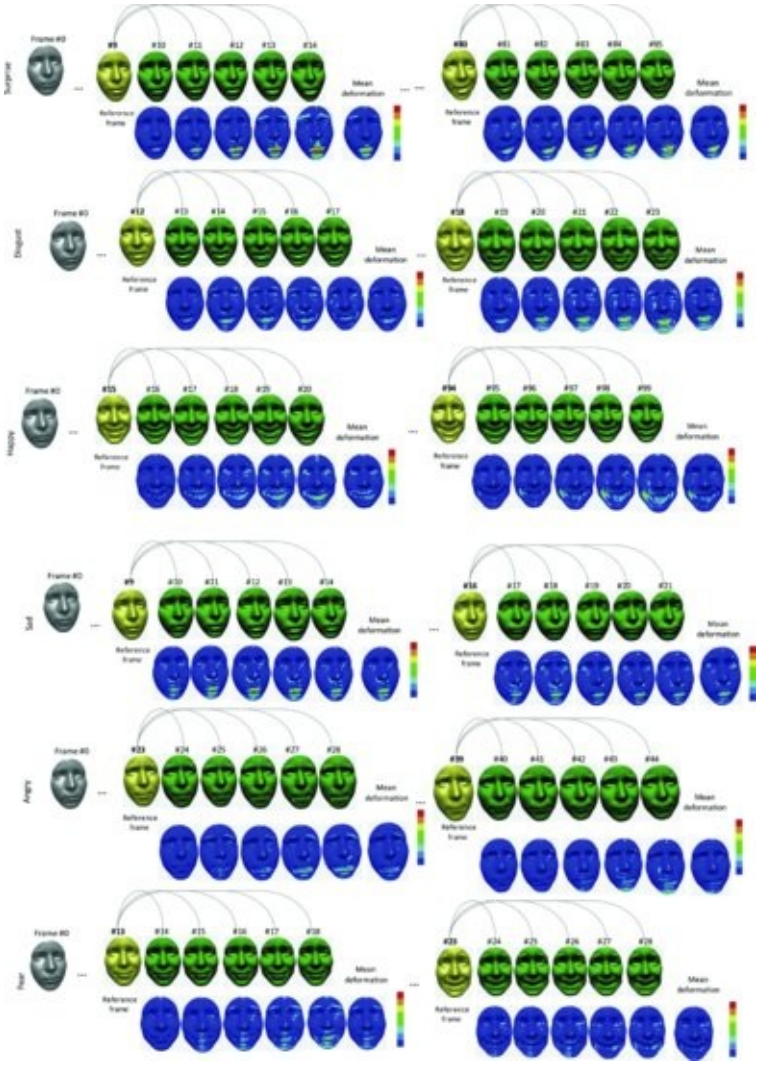
In this section, we propose a pattern-based approach for expression classification in 3D video. The idea is to capture a mean deformation of the face in the sliding window on the 3D expression sequence, and consider it as a pattern for classification. To get this pattern, the first frame of each subsequence is considered as the reference one, and the dense deformation is computed from this frame to each of the remaining frames of the subsequence. Let F_{ref} denote the reference frame of a subsequence and F_i the ith successive frame in the subsequence; the successive frame, for example, is denoted by F_1. The DSF is calculated between F_{ref} and F_i, for different values of i ($i = 1 \ldots n-1$), and the mean deformation is then given by

$$\overline{\mathrm{DSF}} = \frac{1}{n-1} \sum_{i=1}^{n-1} \mathrm{DSF}(F_{\mathrm{ref}}, F_i). \quad (5.10)$$

Figure 5.19 illustrates one subsequence for each expression with n=6 frames. Each expression is illustrated in two rows, the upper row gives the reference frame of the subsequence and the $n-1$ successive frames of the subsequences. Later, the corresponding dense scalar fields computed for each frame are shown. The mean deformation map is reported at the right and represents the feature vector for each subsequence. The feature vector for this subsequence is built on the basis of the mean deformation of the $n-1$ calculated deformations. Thus, each subsequence is represented by a feature vector of size equal to the number of points on the face. To provide a visual representation of the scalar field, an automatic labeling scheme is applied: The warm colors (red, yellow) are associated with high DSF(F_{ref}, F_t) values and correspond to facial regions affected by high deformations, whereas the cold colors are associated with regions that remain stable from one frame to another. Thus, this dense deformation filed summarizes the temporal changes of the facial surface when a particular facial expression is conveyed.

Figure 5.19 Calculus of dynamic shape deformation on subsequences taken from the BU-4DFE database. Each expression is illustrated by two rows: The upper one gives the reference frame of the subsequence and the N-1 successive frames

of the subsequences. The corresponding deformation fields computed for each frame with respect to the reference one are illustrated in the lower row. The mean deformation map is given at the right of each lower row



According to this representation, the deformation of each subsequence is captured by the mean DSF $\bar{V}_{\alpha}^{k}$ defined in Equation 5.10. Because the dimensionality of the feature vector is high, we use LDA-based transformation to map the present feature space to an optimal one that is relatively insensitive to different subjects while preserving the discriminative expression information. LDA defines the within-class matrix S_w and the between-class matrix S_b. It transforms a N-dimensional feature to an optimized d-dimensional feature where $d<n$. For our experiments, the discriminative classes are the 6 expressions, thus the reduced dimension d is 5.

For the classification, we used the multiclass version of the random forest

algorithm. The random forest algorithm was proposed by Breiman (2001) and defined as a meta-learner comprising many individual trees. It was designed to operate quickly over large data sets and more importantly to be diverse by using random samples to build each tree in the forest. A tree achieves highly nonlinear mappings by splitting the original problem into smaller ones, solvable with simple predictors. Each node in the tree consists of a test, whose result directs a data sample towards the left or the right child. During training, the tests are chosen to group the training data in clusters where simple models achieve good predictions. Such models are stored at the leaves, computed from the annotated data, which reached each leaf at train time. Once trained, random forest classifies a new expression from an input feature vector by putting it down each of the trees in the forest. Each tree gives a classification decision by voting for that class. Then, the forest chooses the classification having the most votes (over all the trees in the forest).

Experimental Results

Deformations following facial expressions across 3D video sequences are characterized by subtle variations induced mainly by the motion of facial points. These subtle changes are important to perform effective expression recognition, but they are also difficult to be analyzed because of the face movements. To handle this problem, as described in the previous section, we propose a curve-based parametrization of the face that consists in representing the facial surface by a set of radial curves. According to this representation, the problem of comparing two facial surfaces, a reference facial surface and a target one, is reduced to the computation of the DSFs between them.

To make possible starting the recognition process from any frame of a given video, we considered subsequences of N frames. Thus, we chose the first N frames as the first subsequence. Then, we chose N-consecutive frames starting from the second frame as the second subsequence. This process was repeated by shifting the starting index of the sequence every one frame till the end of the sequence.

Following the experimental protocol proposed in Sun and Yin (2008), a large set of subsequences were extracted from the original expression sequences using a sliding window. The subsequences have been defined in Sun and Yin (2008) with a length of six frames with a sliding step of one frame from one subsequence to the following one. For example, with this approach, a sequence

of 100 frames originates a set of $6 \times 95 = 570$ subsequences, each subsequence differing from one frame from the previous one. This accounts for the fact that, in general, the subjects can come into the system not necessarily starting with a neutral expression, but with a generic expression. Classification of these short sequences is regarded as an indication of the capability of the expression recognition framework to identify individual expressions.

According to this, we first computed for each subsequence the Mean Deformation, which was then presented to multiclass Random Forest, as outlined in Section 5.5.2. The performance of the Random Forest classifier varied with the number of trees. Thus, we performed the experiments with different numbers of trees; the results of this experimentation are shown in Figure 5.20. As illustrated by this figure, the average recognition rate rose with the increasing number of trees until 40, when the recognition rate reached 93.21%, and then became stable. Thus, in the following paragraphs, we considered 40 trees and the detailed results (confusion matrix) with this number of trees are shown in Table 5.10. We recall that the rates were obtained by averaging the results of the 10-independent and arbitrarily run experiments (10-fold cross validation).

Figure 5.20 4D expression recognition results using Random Forest classifier when varying the number of trees

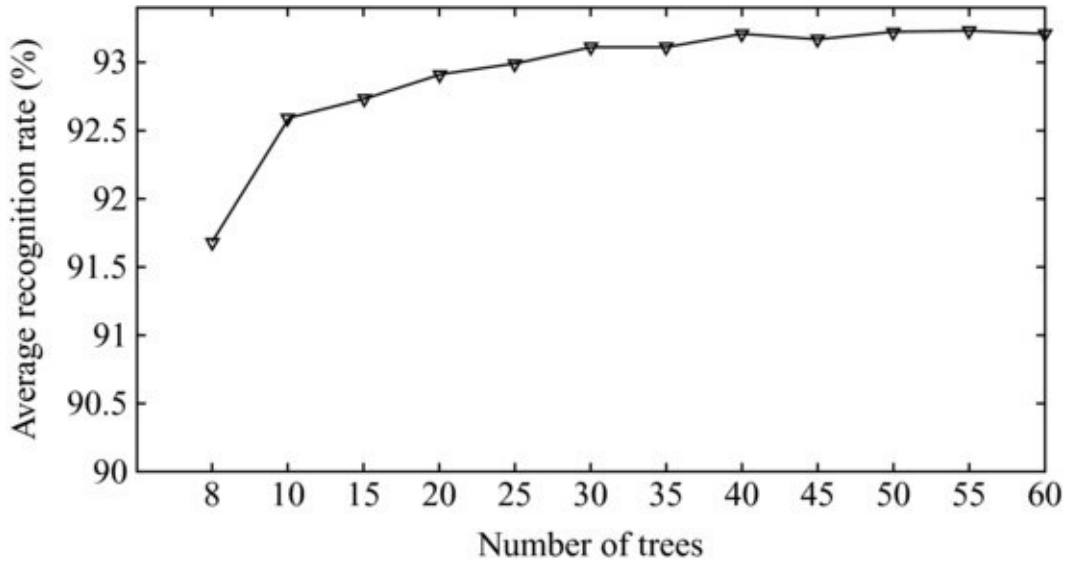


Table 5.10 Confusion matrix for Mean Deformation and Random Forest classifier (for 6-frames window)

%	An	Di	Fe	Ha	Sa	Su
An	**93.11**	2.42	1.71	0.46	1.61	0.66
Di	2.3	**92.46**	2.44	0.92	1.27	0.58
Fe	1.89	1.75	**91.24**	1.5	1.76	1.83
Ha	0.57	0.84	1.71	**95.47**	0.77	0.62
Sa	1.7	1.52	2.01	1.09	**92.46**	1.19
Su	0.71	0.85	1.84	0.72	1.33	**94.53**

Average recognition rate = ***93.21%***

It can be noted that the lower recognition was obtained for the *fear (Fe)* expression (91.24%), which is mainly confused with the *anger (An)* and *disgust (Di)* expressions. Interestingly, these three expressions capture negative emotive states of the subjects so that similar facial muscles can be activated. The best classified expressions are *happiness (Ha)* and *surprise (Su)* with recognition accuracy of 95.47% and 94.53%, respectively.

5.5.3 Discussion and Comparative Evaluation

To the best of our knowledge, the only four works reporting results on expression recognition from dynamic sequences of 3D scans are Sun and Yin (2008), Sandbach *et al.* (2011), Le *et al.* (2011), and Fang *et al.* (2011). These works have been evaluated on the BU-4DFE data set, but the testing protocols used in the experiments are sometimes different, so that a direct comparison of the results reported in these papers is difficult. In the following, we present in Table 5.11 the last results published on 3D facial expression recognition from dynamic 3D scans.

Table 5.11 Comparison of 3D Dynamic Facial expression Recognition Approachs

Authors	Features	Classification	RR[3] (%)
Sun and Yin (2008)	12 Motion Units	HMM	70.31
Sun and Yin (2008)	Tracking model	HMM	80.04
Sun and Yin (2008)	Curvature+Tracking model	HMM	82.19
Sun and Yin (2008)	Curvature+Tracking model	2D-HMM	90.44
Sandbach et al. (2011)	FFD+Quad-tree	GentleBoost+HMM	73.61, 81.93
Le et al. (2011)	Level curve-based	HMM	92.22
Fang et al. (2011)	LBP-TOP	SVM-RBF	74.63
Drira et al. (2012)	Geometric Mean Deformation	LDA-Random Forest	93.21

Exercises

1.What are the main limitations of 2D facial expression recognition systems?

2. We consider the following ROC Figure 5.21 curves of two 3D face authentication approaches evaluated using the same experimental protocol:

1. How do you interpret these curves?

2. You have been asked as an expert in Biometrics to choose between these two approaches to design two different access control systems. Complete the table below. Explain your choices?

	Approach 1	Approach 2
Access Control system for Canteen for children		
Access Control system for safe of a bank		

Figure 5.21 ROC Curves

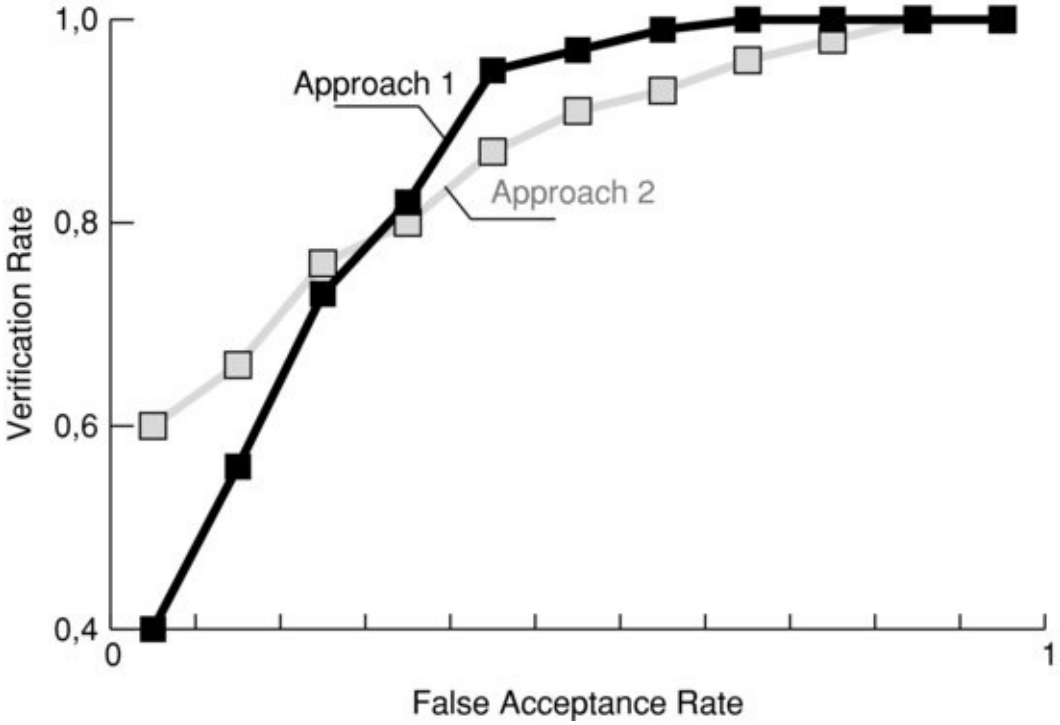


Figure 5.22 (a) Linearly separable data samples represented in a plane and separated by a straight line; (b) nonlinearly separable data samples represented in a plane and separated by a curved line

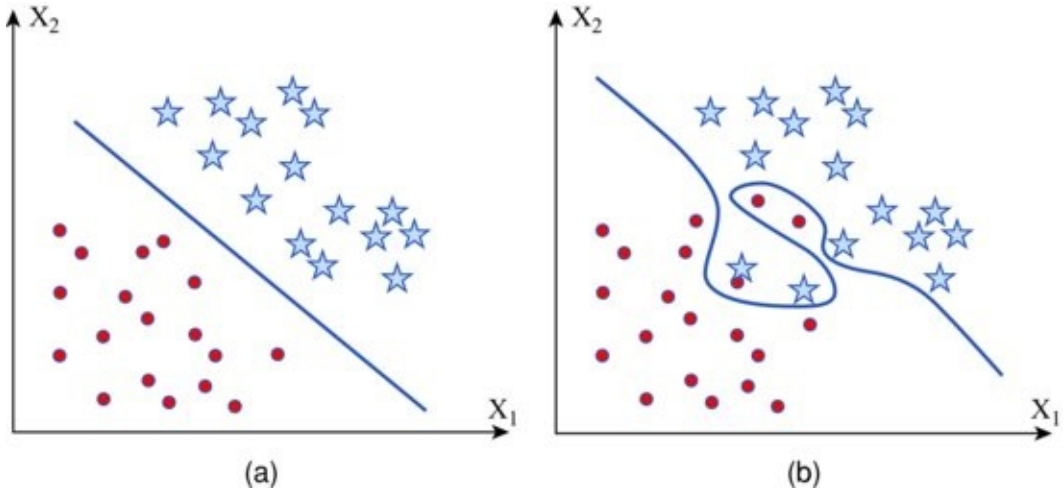
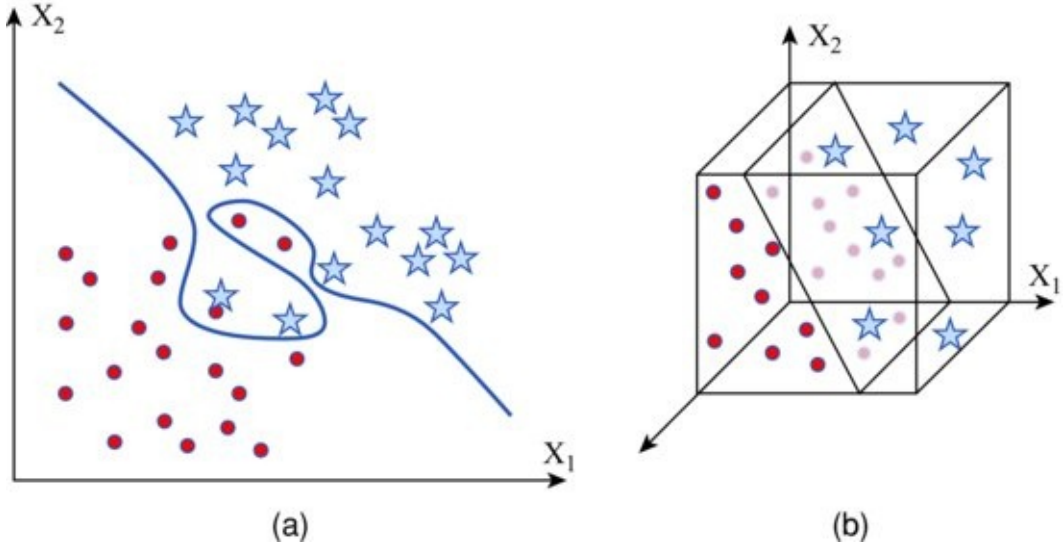


Figure 5.23 (a) Nonlinearly separable data samples represented in a plane and separated by a curved line; (b) plan separation after a transformation of of the same data samples into a 3D space



3. Give one application of facial expression recognition in medical application.

Glossary

In the following we provide a brief introduction to the machine learning algorithms used in the manuscript. For more details, please refer to some textbooks on machine learning such as Bishop (2006).

AdaBoost

AdaBoost is a very successful machine-learning method that permits to build an

accurate prediction rule, its principle is based on finding many rough rules of thumb instead of finding a one highly accurate rule. More simpler, the idea is to build a strong classifier by combining weaker ones. AdaBoost is proven to be an effective and powerful classifier in the category of ensemble techniques. The algorithm takes as input a training examples $(x_1, y_1), \ldots, (x_N, y_N)$ where each x_i ($i = 1, \ldots, N$) is an example that belongs to some domain or instance space *X*, and each label y_i is a boolean value that belongs to the domain $Y = \{-1, +1\}$, indicating whether x_n is positive or negative example. Along a finite number of iterations $t = 1, 2, \ldots, T$ the algorithm calls, at each iteration *t*, the weak classifier (or learner). After *t* times it generates a set of hypothesis $\{h_t\}_{t=1}^{T}$ such that $h_t \longrightarrow \{-1, 1\}$. The final classifier *H*(*X*) is the strongest one, and is given by the combination of these hypothesis, ponderated by their respective weight factors $\{\alpha_t\}_{t=1}^{T}$. The hypothesis h_t and its corresponding weight α_t are determined at each iteration *t*, the selection of the best hypothesis h_t, at each time *t*, is done among a set of hypothesis $\{h_j\}_{j=1}^{J}$, where *j* stands for the number of features considered for the classification task. h_t is equal to h_j that gives the smallest error of classification ϵ_j. The error ϵ_j corresponds to samples that are misclassified, and that will see their associated weight increased in the next iteration *t*+1. These procedures are presented in Algorithm 9.

Algorithm 9 AdaBoost algorithm

- Input: set of examples $(x_1, y_1), \ldots, (x_N, y_N)$ where $x_i \in X$ and $y_i = \{-1, +1\}$.
- Let m be the number of negatives examples and l be the number of positive examples. Initialize weights $w_{1,n} = \frac{1}{2m}, \frac{1}{2l}$ depending on the value of y_n.
- For $t = 1, \ldots, T$ x:
  1. Normalize the weights $w_{t,n}$ so that $\sum_{n=1}^{N} w_{t,n} = 1$.
  2. For each feature f_j, train a weak classifier h_j.
  3. The error ϵ_j of a classifier h_j is determined with respect to the weights $w_{t,1}, \ldots, w_{t,N}$:

$$\epsilon_j = \sum_{n}^{N} w_{t,n} |h_j(x_n) - y_n|$$

  4. Choose the classifier h_j with the lowest error ϵ_j and set $(h_t, \epsilon_t) = (h_j, \epsilon_j)$.
  5. Update the weights $w_{t+1,n} = w_{t,n} \beta_t^{1-e_n}$, where $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$ and $e_n = 0$, if example x_n is classified correctly by h_t and 1 otherwise
- The final strong classifier is given by:

$$H(x) = \begin{cases} 1 & \text{if } \sum_{t=1}^{T} log\frac{1}{\beta_t} h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} log(\frac{1}{\beta_t}); \\ 0 & \text{otherwise.} \end{cases}$$

MultiBoost

Given a set of training data $(x_1, c_1), \ldots, (x_n, c_n)$ where x_i is the input, and each output $c_i \in 1, \ldots, K$ is a class label. We use k to denote the number of possible class labels. Using training data, MultiBoost, which is an extension of the original AdaBoost method, permits to find a classification rule so that when given a new input X, we can assign it a class label c from $1, \ldots, K$. Let $T(x)$ denote a weak multiclass classifier that assigns a class label to X. Then the MultiBoost algorithm, called also AdaBoostM1, proceeds as reposted in Algorithm 10.

Algorithm 10 AdaBoostM1 algorithm

- Initialize the observation weights $w_i = \frac{1}{n}, i = 1, 2, \ldots, n.$
- For $m = 1$ to M:
  - Fit a classifier $T_m(x)$ to the training data using weights w_i
  - Compute $\text{err}_m = \frac{\sum_{i=1} nw_i \prod(c_i \neq T_m(x_i))}{\sum_{i=1} nw_i}$.
  - Compute $\alpha_m = \log(\frac{1-\text{err}_m}{\text{err}_m})$
  - Set $w_i \leftarrow w_i \cdot \exp(\alpha_m \cdot \prod(c_i \neq T_m(x_i))), i = 1, \ldots, n.$
  - Re-normalize w_i.
- Output$C(x) = \arg\max_k \sum_{m=1} M\alpha_m \prod(T_m(x) = K).$

MultiBoost with LDA Classifier

MultiBoost with LDA classifier incorporates linear discriminant analysis (LDA) algorithm to implement linear combinations between selected features and generate new combined features. The combined features are used along with the original features in boosting algorithm for improving classification performance. Given a binary classification problem with linear classifiers that are specified by discriminant functions, LDA assumes the covariance matrices of both classes to be equal, $\sum$. We denote the means by μ_1 and μ_2, and arbitrary feature vector by *X* define

$$D(x) = [b; w]^T . [1; x]$$
$$w = \sum^{-1} .(\mu_2 - \mu_1)$$
$$b = -w^T .\mu$$
$$\mu = \tfrac{1}{2}.(\mu_1 + \mu_2).$$

D(*x*) is the difference in the distance of the feature vector *X* to the separating hyperplane described by its normal vector *w* and the bias *b*. If *D*(*x*) is greater than 0, the observation *X* is classified as class 2 and otherwise as class 1.

MultiBoost with NB Classifier

The Naive Bayes classifier estimates the posterior probability that an instance belongs to a class, given the observed attribute values for the instance. It builds a simple conditional independence classifier. Formally, the probability of a class label value *Y* for an unlabeled instance *X* containing *N* attributes $\langle A_1, A_2, \ldots, A_n \rangle$ is given by

$$
\begin{aligned}
P(y|x) &= \\
&= P(x|y).\frac{P(y)}{P(x)} \\
&\propto P(A_1, A_2, \ldots, A_n).P(y) \\
&= \prod_{j+1} nP(A_j|y).P(y).
\end{aligned}
$$

The preceding probability is computed for each class and the prediction is made for the class with the largest posterior probability. The probabilities in the aforementioned formulas must be estimated from the training set.

MultiBoost with NN Classifier

The Nearest Neighbor pattern classifier has shown to be a powerful tool for multiclass classification. The basic idea of the NN classifier is that whenever we have a new instance to classify, we find its *k* nearest neighbors from the training data. Given a query instance x_i to be classified

- let $x_1, x_2, \ldots, x_k$ denote the *k* instances from training examples that are nearest to x_q;
- return the class that represents the maximum of the *k* instances.

Support Vector Machine (SVM)

SVM is based on the use of functions that can optimally separate data. When considering the case of two classes for which data are linearly separable, there exists an infinite number of hyperplanes for separating the observations. SVM's goal is to find the optimal hyperplane that separates data with maximizing the distance between the two classes and that goes middle of the two points classes of examples. The nearest points, which are used only for the determination of the hyperplane, are called support vectors. Among the models of SVM, there is linear-SVM and nonlinear-SVM. The first are the simplest SVM because they can linearly separate data, whereas the second ones are used for data that are not linearly separable. In the last case the data are transformed to be represented in a large space where they are linearly separable.

The evaluation of classification techniques is a recurrent problem which often depends on the difficult task of measuring generalization performance, that is, the performance on new, previously unseen data. For most real-world problems we can only estimate the generalization performance. To evaluate a certain learning algorithm, we usually apply a cross-validation scheme.

Cross-validation

The traditional approach of cross-validation consists in dividing the data to classify into training and testing partitions a number of times. In our studies, we applied *k*-fold cross-validation procedure (with *k*=10), where the data was first partitioned into *k* equally (or nearly equally) sized segments or folds. Subsequently *k* iterations of training and validation were performed such that within each iteration a different fold of the data was held out for validation, while the remaining *k*−1 folds were used for learning. Following this procedure we measured the classification accuracy of the considered classifier.

Hidden Markov Model

HMM consists of <*S*, s_0, *O*, *A*, *B*>:

- *S*: the set of states;
- s_0: the initial state where everything begins;
- *O*: the sequence of observations, each of which is drawn from a vocabulary, $\langle o_1, o_2, \ldots, o_T \rangle$;
- *A*: the transitional probability matrix;
- *b*: the emission probabilities, where $b_i(o_t)$ is the probability of observation o_t generated from state *i*.

Forward–Backward Probability

- Forward probability: At time *t*, the probability that
  – we are in state *i*
  – the observation thus far has been $o_1 \ldots o_t$
  – $\alpha_t(i) = P(s_t = i, o_1, \ldots, o_t|\lambda)$.
- Backward probability: At time *t* and we are in state *i*, the probability that
  – the observation that follows will be $o_{t+1} \ldots o_T$
  – $\beta_t(i) = P(o_{t+a} \ldots o_T|s_t = i, \lambda)$.

Baum–Welch Algorithm

The Baum–Welch algorithm is used to estimate the parameters of a Hidden Markov model. It can compute maximum likelihood estimates and posterior mode estimates for the parameters (transition and emission probabilities) of an HMM, when given only emissions as training data.

Algorithm 11 Baum–Welch algorithm

1. Initialize the parameters to some values;
2. Calculate "forward-backward" probabilities based on the current parameters;
3. Use the forward-backward probabilities to estimate the expected frequencies;
   - Expected number of transitions from state i (to state j);
   - Expected number of being in state j (and observing o_t);
   - Expected number of starting in state j;
4. Use the expected frequencies to estimate the parameters;
5. Repeat 2 to 4 until the parameters converge.

References

Alyüz N, Gökberk B, Akarun L. 3D face recognition system for expression and occlusion invariance. In: *Proceedings of the IEEE 2nd International Conference on Biometrics: Theory, Applications, and Systems*; 2008 Sept 29–Oct 1, Washington, DC. 2008a. New York: IEEE. p. 1–7.

Alyüz N, Gökberk B, Dibeklioglu H, Savran A, Salah AA, Akarun L, Sankur B. 3d face recognition benchmarks on the bosphorus database with focus on facial expressions. In: *Biometrics and Identity Management: 1st European Workshop*, BIOID; 2008 May 7–9, Roskilde, Denmark. 2008b. Heidelberg: Springer-Verlag. 2008. p. 57–66.

Berretti S, Ben Amor B, Daoudi M, del Bimbo A. Person independent 3D facial expression recognition by a selected ensemble of SIFT descriptors. In: *Proceedings of the 3rd Eurographics/ACM SIGGRAPH Symposium on 3D Object Retrieval, 2010a May 2*, Norrköping, Sweden. New York: ACM Press/Addison-Wesley Publishing Company. 2010. p. 47–54.

Berretti S, Bimbo AD, Pala P. 3d face recognition using isogeodesic stripes. IEEE Transactions on Pattern Analysis and Machine Intelligence 2010b;32(12):2162–2177.

Berretti S, del Bimbo A, Pala P. 3D face recognition using isogeodesic stripes. IEEE Transactions on Pattern Analysis and Machine Intelligence 2010c;32(12):2162–2177.

Berretti S, del Bimbo A, Pala P. Recognition of 3D faces with missing parts based on profile networks. In: *Proceedings of the 1st ACM Workshop on 3D Object Retrieval; 2010 Oct 25–29, Firenze, Italy*. New York: ACM Press. 2010d. p. 81–86.

Berretti S, Ben Amor B, Daoudi M, del Bimbo A. 3d facial expression recognition using sift descriptors of automatically detected keypoints. The Visual Computer 2011a;27(11):1021–1036.

Berretti S, del Bimbo A, Pala P. Facial curves between keypoints for recognition of 3d faces with missing parts. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops: Workshop on Multi Modal Biometrics*; 2011b Jun 20, Colorado Springs, CO. New York: IEEE. 2010. p. 49–54.

Berretti S, del Bimbo A, Pala P, Ben Amor B, Daoudi M. Selected SIFT features for 3d facial expression recognition. In: *Proceedings of 20th International Conference on Pattern Recognition*; 2010, Aug 23–26, Istanbul, Turkey. New York: IEEE. 2010e. p. 4125–4128.

Bishop C. *Pattern Recognition and Machine Learning Information Science and Statistics*. New York: Springer; 2006.

Bowyer KW, Chang K, Flynn P. A survey of approaches and challenges in 3d and multimodal 3d + 2d face recognition. Computer Vision and Image Understanding 2006;101(1):1–15.

Breiman L. Random forests. Machine Learning 2001;45(1):5–32.

Bronstein AM, Bronstein MM, Kimmel R. Three-dimensional face recognition. International Journal of Computer Vision 2005;64(1):5–30.

Bronstein AM, Bronstein MM, Kimmel R. Robust expression-invariant face recognition from partially missing data. In: Leonardis A, Bischof H, Pinz A, editors. *Proceedings of the 9th European Conference on Computer Vision*; 2006 May 7–13, Gratz, Austria. Hedielberg: Springer. 2006a. p. 396–408.

Bronstein EM, Bronstein MM, Kimmel R. Robust expression-invariant face recognition from partially missing data. In: Leonardis A, Bischof H, Pinz A, editors. *Proceedings of the 9th European Conference on Computer Vision*. Lecture Notes on Computer Science; 2006 May 7–13, Gratz, Austria. Heidelberg: Springer. 2006b. p. 396–408.

Bronstein AM, Bronstein MM, Kimmel R. Expression-invariant representations of faces. IEEE Transactions on Image Processing 2007;16(1):188–197.

Chang K, Bowyer W, Flynn P. Multiple nose region matching for 3d face recognition under varying facial expression. IEEE Transactions on Pattern Analysis and Machine Intelligence 2006;28(10):1695–1700.

Cook J, McCool C, Chandran V, Sridharan S. Combined 2d/3d face recognition

using Log–Gabor templates. In: *Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*; 2006, Nov 22–24, Sydney, Australia. New York: IEEE. 2006. p. 83.

Daoudi M, ter Haar F, Veltkamp R. 3D face models. SHREC contest session on retrieval of 3D face scans. *IEEE International Conference on Shape Modeling and Applications*; 2008 Jun 4–6, Stony Brook, NY. New York: IEEE. p. 215–216.

Di3D 2006 http://www.di3d.com.

Drira H, Amor BB, Daoudi M, Srivastava A, Berretti S. 3d dynamic expression recognition based on a novel deformation vector field and random forest. In: Proceedings of the 21th International Conference on Pattern Recognition; 2012 nov 11–15, Tsukuba, Japan. 2012. p. 1104–1107.

Drira H, Amor BB, Srivastava A, Daoudi M, Slama R. 3d face recognition under expressions, occlusions and pose variations. IEEE Trans. Pattern Anal. Mach. Intell. Minor revision. To appear in IEEE Transactions on Pattern Analysis and Machine Intelligence.

Drira H, Ben Amor B, Daoudi M, Srivastava A. Pose and expression-invariant 3D face recognition using elastic radial curves. In: *Proceedings of the British Machine Vision Conference*; 2010 Aug 30–2 Sept, Aberystwyth, UK. Edinburgh: BMVA Press. 2010. p. 1–11.

Ekman P. Universals and cultural differences in facial expressions of emotion. In: J. Cole, editor. *Nebraska Symposium on Motivation, 1971*. Volume 19. Lincoln: University of Nebraska Press; 1972. p. 207–282.

Ekman P, Friesen WV. *Manual for the the Facial Action Coding System*. Palo Alto, CA: Consulting Psychologist Press; 1977.

Faltemier TC, Bowyer KW, Flynn PJ. A region ensemble for 3-d face recognition. IEEE Transactions on Information Forensics and Security 2008a;3(1):62–73.

Faltemier TC, Bowyer KW, Flynn PJ. A region ensemble for 3D face recognition. IEEE Transactions on Information Forensics and Security 2008b;3(1):62–73.

Fang T, Zhao X, Shah S, Kakadiaris I. 4d facial expression recognition. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*; 2011 Nov 6–13, Barcelona, Spain. New York: IEEE. 2011. p. 1594–1601.

Farkas LG. *Anthropometry of the Head and Face*. New York: Raven Press; 1994.

Gong B, Wang Y, Liu J, Tang X. Automatic facial expression recognition on a single 3D face by exploring shape deformation In: *Proceedings of the ACM International Conference on Multimedia*; 2009 Oct 19–22, Beijing, China. New York: ACM Press. 2009. p. 569–572.

Gordon G. Face recognition based on depth and curvature features. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*; 1992 Jun 15–18, Champaign, IL. New York: IEEE. 1992. p. 108–110.

Grenander U. *General Pattern Theory*. Oxford: Oxford University Press; 1993.

Grenander U, Miller MI. Computational anatomy: An emerging discipline. Quarterly of Applied Mathematics 1998;56(4):617–694.

Gupta S, Aggarwal JK, Markey MK, Bovik AC. 3d face recognition founded on the structural diversity of human faces. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*; 2007 Jun 18–23, Minneapolis, MN. New York: IEEE. 2007. p. 1–7.

Gupta S, Markey MK, Bovik AC. Anthropometric 3D face recognition. International Journal of Computer Vision 2010;90(3):331–349.

Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH. The WEKA data mining software: An update. ACM SIGKDD Explorations Newsletter 2009;11(1): 10–18.

Huang D, Ardabilian M, Wang Y, Chen L. 3D face recognition using eLBP-based facial representation and local feature hybrid matching. IEEE Transactions on Information Forensics and Security 2012;7(5):1551–1564.

Huang D, Zhang G, Ardabilian M, Wang Y, Chen L. 3d face recognition using distinctiveness enhanced facial representations and local feature hybrid matching *Fourth IEEE International Conference on Biometrics: Theory Applications and Systems (BTAS)*; 2010 Sep 27–29, Washington, DC. New York: IEEE. 2010. p. 1–7.

Joshi SH, Klassen E, Srivastava A, Jermyn IH. An efficient representation for computing geodesics between n-dimensional elastic shapes. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*; 2007 Jun 18–23, Minneapolis, MN. New York: IEEE. 2007.

Kakadiaris IA, Passalis G, Toderici G, Murtuza MN, Lu Y, Karampatziakis N,

Theoharis T. Three-dimensional face recognition in the presence of facial expressions: An annotated deformable model approach. IEEE Transactions on Pattern Analysis and Machine Intelligence 2007a;29(4):640–649.

Kakadiaris IA, Passalis G, Toderici G, Murtuza MN, Lu Y, Karampatziakis N, Theoharis T. Three-dimensional face recognition in the presence of facial expressions: An annotated deformable model approach. IEEE Transactions on Pattern Analysis and Machine Intelligence 2007b;29(4):640–649.

Kakadiaris IA, Passalis G, Toderici G, Murtuza MN, Lu Y, Karampatziakis N, Theoharis T. Three-dimensional face recognition in the presence of facial expressions: An annotated deformable model approach. IEEE Transactions on Pattern Analysis and Machine Intelligence 2007c;29(4):640–649.

Klassen E, Srivastava A. Geodesics between 3D closed curves using path-straightening. In: Leonardis A, Bischof H, Pinz A, editors. *Proceedings of the 9th European Conference on Computer Vision*, Volume 1; 2006 May 7–13, Gratz, Austria. Hedielberg: Springer. 2006. p. 95–106.

Kotropoulos C, Tefas A, Pitas I. Frontal face authentication using morphological elastic graph matching. IEEE Transactions on Image Processing 2000;9(4):555–560.

Le V, Tang H, Huang TS. Expression recognition from 3d dynamic faces using robust spatiotemporal shape features. In: *Proceedings of the IEEE Conference on Automatic Face and Gesture Recognition*; 2011 Mar 21–25, Santa Barbara, CA. New York: IEEE. 2011. p. 414–421.

Lee Y, Song H, Yang U, Shin H, Sohn K. Local feature based 3d face recognition. In: Kanade T, Jain AK, Ratha NK, editors. *Proceedings of the 5th International Conference on Audio and Video-Based Biometric Person Authentication*; 2005 Jul 20–22, Hilton Rye Town, NY. 2005. Heidelber: Springer. 2005. p. 909–918.

Li X, Jia T, Zhang H. Expression-insensitive 3d face recognition using sparse representation. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*; 2009 Jun 20–26, Miami, FL. New York: IEEE. 2009. p. 2575–2582.

Lowe D. Distinctive image features from scale-invariant key points. International Journal of Computer Vision 2004;60(2):91–110.

Lu X, Jain AK. Deformation modeling for robust 3d face matching. IEEE Transactions on Pattern Analysis and Machine Intelligence 2008;30(8):1346–

1357.

Lu X, Jain AK. Deformation modeling for robust 3d face matching. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*; 2006 Jun 17–22, New York, NY. New York: IEEE. p. 1377–1383.

Maalej A, Ben Amor B, Daoudi M, Srivastava A, Berretti S. Local 3D shape analysis for facial expression recognition. In: Proceedings of the 20th International Conference on Pattern Recognition, Istanbul, Turkey. p. 4129–4132, 2010.

Maalej A, Ben Amor B, Daoudi M, Srivastava A, Berretti S. Shape analysis of local facial patches for 3d facial expression recognition. Pattern Recognition 2011;44(8):1581–1589.

Mahoor MH, Abdel-Mottaleb M. Face recognition based on 3d ridge images obtained from range data. Pattern Recognition 2009;42(3):445–451.

Matuszewski B, Quan W, Shark LK. High-resolution comprehensive 3-d dynamic database for facial articulation analysis. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*; 2011 Nov 6–13, Barcelona, Spain. New York: IEEE. 2011. p. 2128–2135.

Matuszewski BJ, QuanW, Shark LK, McLoughlin AS, Lightbody CE, Emsley HC and Watkins CL. Hi4d-adsip 3-d dynamic facial articulation database. Image and Vision Computing 2012; 30(10):713–727.

Maurer T, Guigonis D, Maslov I, Pesenti B, Tsaregorodtsev A, West D, Medioni G. Performance of geometrix activeid 3D face recognition engine on the FRGC data. In: *Proceedings of the IEEE Workshop on Face Recognition Grand Challenge Experiments*; 2005 Jun 20–25, San Diego, CA. New York: IEEE. p. 154.

Mayo M, Zhang E. 3D face recognition using multiview key point matching. In: *Proceedings of the 6th IEEE International Conference on Advanced Video and Signal Based Surveillance*; 2009 Sept 2–4, Genoa, Italy. New York: IEEE. p. 290–295.

McKeon R, Russ T. Employing region ensembles in a statistical learning framework for robust 3d facial recognition. In: *Proceedings of the 4th IEEE International Conference on Biometrics: Theory Applications and Systems*; 2010 Sep 23–26, Washington, DC. New York: IEEE. 2010. p. 1–7.

Mian A, Bennamoun M, Owens R. An efficient multimodal 2d-3d hybrid

approach to automatic face recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 2007a;29(11):1927–1943.

Mian AS, Bennamoun M, Owens R. An efficient multimodal 2D-3D hybrid approach to automatic face recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 2007b;29(11):1927–1943.

Mian AS, Bennamoun M, Owens R. Keypoint detection and local feature matching for textured 3D face recognition. International Journal of Computer Vision 2008;79(1):1–12.

Miller MI, Younes L. Group actions, homeomorphisms, and matching: A general framework. International Journal of Computer Vision 2001;41(1/2):61–84.

Moorthy A, Mittal A, Jahanbin S, Grauman K, Bovik A. 3d facial similarity: automatic assessment versus perceptual judgments. *Fourth IEEE International Conference on Biometrics: Theory Applications and Systems*; 2010 Sep 27–29, Washington, DC. New York: IEEE. 2010. p. 1–7. 2010

Moreno AB, Sánchez A 2004a Gavabdb: A 3D face database. In: *Proceedings of the 2nd COST 275 Workshop on Biometrics on the Internet*; 2004 Mar 25–26, Vigo, Spain. Luxembourg: COST European Cooperation in Science and Technology. 2004a. p. 75–80.

Moreno AB and Sanchez A 2004b Gavabdb: A 3d face database. In: Proceedings of the 2nd COST 275 Workshop on Biometrics on the Internet; 2004 Mar 25–26, Vigo, Spain. Luxembourg: COST European Cooperation in Science and Technology. 2004. p. 77–85.

Moreno AB, Sanchez A, Velez JF, Daz FJ. Face recognition using 3D local geometrical features: PCA vs. SVM. *4th International Symposium on Image and Signal Processing and Analysis*; 2005. Sept 15–17. New York: IEEE. p. 185–190.

Mousavi MH, Faez K, Asghari A. Three dimensional face recognition using SVM classifier ICIS '08. In: *Proceedings of the Seventh IEEE/ACIS International Conference on Computer and Information Science*; 2008, May 14–16, Washington, DC. New York: IEEE/ACIS. 2008. p. 208–213.

Mpiperis I, Malassiotis S, Strintzis MG. 3-d face recognition with the geodesic polar representation. IEEE Transactions on Information Forensics and Security 2007;2(3–2):537–547.

Mpiperis I, Malassiotis S, Strintzis M. Bilinear models for 3-d face and facial

expression recognition. IEEE Transactions on Information Forensics and Security 2008a;3(3), 498–511.

Mpiperis I, Malassiotis S, Strintzis MG. Bilinear models for 3-D face and facial expression recognition. IEEE Transactions on Information Forensics and Security 2008b;3(3), 498–511.

Mpiperis I, Malassiotis S, Petridis V, Strintzis MG. 3D facial expression recognition using swarm intelligence. In: *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*; 2008 Mar 31–Apr 4, Las Vegas, Nevada. New York: IEEE. 2008c. p. 2133–2136.

Ohbuchi R, Furuya T. Scale-weighted dense bag of visual features for 3D model retrieval from a partial view 3D model. In: *Proceedings of the IEEE 12th International Conference on Coumputer Vision Workshops: Workshop on Search in 3D and Video*; 2004 Sept 27–Oct 4, Kyoto, Japan. New York: IEEE. 2004. p. 63–70.

Pandzic I, Forchheimer R. *MPEG-4 Facial Animation: The Standard, Implementation and Applications*. Chichester: Wiley; 2005.

Peng H, Long F, Ding C. Feature selection based on mutual information: Criteria of max-dependency, max-relevance and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence 2005;27(8):1226–1238.

Perakis P, Passalis G, Theoharis T, Toderici G, Kakadiaris IA. Partial matching of interpose 3D facial data for face recognition. In: *Proceedings of the 3rd IEEE International Conference on Biometrics: Theory, Applications and Systems, BTAS'09*; 2009 Sept 28–30, Washington, DC. New York: IEEE. 2009. p. 1–8.

Phillips PJ, Flynn PJ, Scruggs T, Bowyer KW, Chang J, Hoffman K, Marques J, Min J, Worek W. Overview of the face recognition grand challenge. In: *Proceedings of the IEEE Workshop on Face Recognition Grand Challenge Experiments*; 2005 Jun 20–25, San Diego, CA. New York: IEEE. 2005. p. 947–954.

Queirolo CC, Silva L, Bellon OR, Segundo MP. 3d face recognition using simulated annealing and the surface interpenetration measure. IEEE Transactions on Pattern Analysis and Machine Intelligence 2010a;32:206–219.

Queirolo CC, Silva L, Bellon OR, Segundo MP. 3d face recognition using simulated annealing and the surface interpenetration measure. IEEE Transactions on Pattern Analysis and Machine Intelligence 2010b;32:206–219.

Ramanathan S, Kassim A, Venkatesh YV, Wah WS. Human facial expression

recognition using a 3D morphable model. In: *Proceedings of the IEEE International Conference on Image Processing*; 2006 Oct 8–11, Atlanta, GA. New York: IEEE. 2006 p. 661–664.

Rueckert D, Sonoda L, Hayes C, Hill D, Leach M, Hawkes D. Nonrigid registration using free-form deformations:application to breast MR images. IEEE Transactions on Medical Imaging 1999;18(8):712–721.

Samir C, Srivastava A, Daoudi M. Three-dimensional face recognition using shapes of facial curves. IEEE Transactions on Pattern Analysis and Machine Intelligence 2006;28:1858–1863.

Samir C, Srivastava A, Daoudi M, Klassen E. An intrinsic framework for analysis of facial surfaces. International Journal of Computer Vision 2009a;82(1):80–95.

Samir C, Srivastava A, Daoudi M and Klassen E. An intrinsic framework for analysis of facial surfaces. International Journal of Computer Vision 2009b;82(1):80–95.

Sandbach G, Zafeiriou S, Pantic M, Rueckert D. A dynamic approach to the recognition of 3d facial expressions and their temporal models. In: *Proceedings of the IEEE Conference on Automatic Face and Gesture Recognition*; 2011 Mar 21–25, Santa Barbara, CA. New York: IEEE. 2011. p. 406–413.

Sandbach G, Zafeiriou S, Pantic M, Rueckert D. Recognition of 3D facial expression dynamics. Image and Vision Computing. Image and Vision Computing, 2012;30(10):683–697.

Savran A, Alyüz N, Dibeklioğlu H, Céliktutan O, Gökberk B, Sankur B, Akarun L. Bosphorus database for 3D face analysis. In: Proceedings of the First COST 2101 Workshop on Biometrics and Identity Management; 2008 May 7–9, Roskilde University, Denmark. Luxembourg: COST European Cooperation in Science and Technology. 2008. p. 1–11.

Soyel H, Demirel H. Facial expression recognition using 3D facial feature distances. In: Mohamed K, Aurélio C, editors. *Proceedings of the International Conference on Image Analysis and Recognition*; 2007 Aug 22–24, Montreal, Canada. Heildelberg: Springer. 2007. p. 831–838.

Spreeuwers L. Fast and accurate 3d face recognition using registration to an intrinsic coordinate system and fusion of multiple region classifiers. International Journal of Computer Vision 2011;93(3):389–414.

Srivastava A, Klassen E, Joshi SH, Jermyn IH. Shape analysis of elastic curves

in euclidean spaces. IEEE Transactions on Pattern Analysis and Machine Intelligence 2011;33(7):1415–1428.

Sun Y, Yin L. Facial expression recognition based on 3D dynamic range model sequences. In: David F, Philip T, Andrew S, editors. *Proceedings of the European Conference on Computer Vision*; 2008 Oct 12–18, Marseille, France. Heildelberg: Springer. 2008. p. 58–71.

Sun Y, Todorovic S, Goodison S. A feature selection algorithm capable of handling extremely large data dimensionality. In: *Proceedings of the 8th Society of Industrial and Applied Mathematics (SIAM) International Conference on Data Mining*; 2008 Apr 24–26, Atlanta, GA. Cambridge: SIAM/Cambridge University Press. 2008. p. 530–540.

Tang H, Huang TS. 3D facial expression recognition based on automatically selected features. In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*; 2008 Jun 24–26, Anchorage, AK. New York: IEEE. 2008. p. 1–8.

ter Haar F, Velkamp RC. Expression modeling for expression-invariant face recognition. Computers and Graphics 2010;34(3):231–241.

Venkatesh YV, Kassim AA, Murthy OVR. A novel approach to classification of facial expressions from 3D-mesh datasets using modified PCA. Pattern Recognition Letters 2009;30(12):1128–1137.

Wang J, Yin L, Wei X, Sun Y. 3D facial expression recognition based on primitive surface feature distribution. In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, Volume 2; 2006 Jun 17–22, New York, New York: IEEE. 2006. p. 1399–1406.

Wang Y, Liu J, Tang X. Robust 3d face recognition by local shape difference boosting. IEEE Transactions on Pattern Analysis and Machine Intelligence 2010;32:1858–1870.

Yin L, Chen X, Sun Y, Worm T, Reale M. A high-resolution 3d dynamic facial expression database. In: *Proceedings of the 8th IEEE International Conference on Automatic Face and Gesture Recognition (FGR08)*; 2008 Sept 17–19, Amsterdam, The Netherlands. New York: IEEE. 2008. p. 1–6.

Yin L, Wei X, Sun Y, Wang J, Rosato M. A 3D facial expression database for facial behavior research. In: *Proceedings of the 7th IEEE International Conference on Automatic Face and Gesture Recognition (FGR06)*; 2006 Apr 2–6, Southampton, UK. New York: IEEE. 2006. p. 211–216.

Zhao G, Pietikäinen M. Dynamic texture recognition using local binary patterns with an application to facial expressions. IEEE Transactions on Pattern Analysis and Machine Intelligence 2007;29(6):915–928.

Zhao X, Dellandra E, Chen L. Building a statistical AU space for facial expression recognition in 3D. In: *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. 2011; Dec 6–8, Noosa, Queensland, Australia. New York: IEEE. 2011. p. 406–409.

Zheng W, Tang H, Lin Z, Huang TS. A novel approach to expression recognition from nonfrontal face images. In *Proceedings of the IEEE International Conference on Computer Vision*; 2009 Sept 29–Oct 2, Kyoto, Japan. New York: IEEE. 2009. p. 1901–1908.