ABSTRACT

Title of dissertation:	Subspace Representations for Robust Face and Facial Expression Recognition
	Sima Taheri, Doctor of Philosophy, 2013
Dissertation directed by:	Professor Rama Chellappa Department of Computer Science

Analyzing human faces and modeling their variations have always been of interest to the computer vision community. Face analysis based on 2D intensity images is a challenging problem, complicated by variations in pose, lighting, blur, and non-rigid facial deformations due to facial expressions. Among the different sources of variation, facial expressions are of interest as important channels of non-verbal communication. Facial expression analysis is also affected by changes in view-point and inter-subject variations in performing different expressions. This dissertation makes an attempt to address some of the challenges involved in developing robust algorithms for face and facial expression recognition by exploiting the idea of proper subspace representations for data.

Variations in the visual appearance of an object mostly arise due to changes in illumination and pose. So we first present a video-based sequential algorithm for estimating the face albedo as an illumination-insensitive signature for face recognition. We show that by knowing/estimating the pose of the face at each frame of a sequence, the albedo can be efficiently estimated using a Kalman filter. Then we

extend this to the case of unknown pose by simultaneously tracking the pose as well as updating the albedo through an efficient Bayesian inference method performed using a Rao-Blackwellized particle filter.

Since understanding the effects of blur, especially motion blur, is an important problem in unconstrained visual analysis, we then propose a blur-robust recognition algorithm for faces with spatially varying blur. We model a blurred face as a weighted average of geometrically transformed instances of its clean face. We then build a matrix, for each gallery face, whose column space spans the space of all the motion blurred images obtained from the clean face. This matrix representation is then used to define a proper objective function and perform blur-robust face recognition.

To develop robust and generalizable models for expression analysis one needs to break the dependence of the models on the choice of the coordinate frame of the camera. To this end, we build models for expressions on the affine shape-space (Grassmann manifold), as an approximation to the projective shape-space, by using a Riemannian interpretation of deformations that facial expressions cause on different parts of the face. This representation enables us to perform various expression analysis and recognition algorithms without the need for pose normalization as a preprocessing step.

There is a large degree of inter-subject variations in performing various expressions. This poses an important challenge on developing robust facial expression recognition algorithms. To address this challenge, we propose a dictionary-based approach for facial expression analysis by decomposing expressions in terms of action

units (AUs). First, we construct an AU-dictionary using domain experts' knowledge of AUs. To incorporate the high-level knowledge regarding expression decomposition and AUs, we then perform structure-preserving sparse coding by imposing two layers of grouping over AU-dictionary atoms as well as over the test image matrix columns. We use the computed sparse code matrix for each expressive face to perform expression decomposition and recognition.

Most of the existing methods for the recognition of faces and expressions consider either the expression-invariant face recognition problem or the identity-independent facial expression recognition problem. We propose joint face and facial expression recognition using a dictionary-based component separation algorithm (DCS). In this approach, the given expressive face is viewed as a superposition of a neutral face component with a facial expression component, which is sparse with respect to the whole image. This assumption leads to a dictionary-based component separation algorithm, which benefits from the idea of sparsity and morphological diversity. The DCS algorithm uses the data-driven dictionaries to decompose an expressive test face into its constituent components. The sparse codes we obtain as a result of this decomposition are then used for joint face and expression recognition.

SUBSPACE REPRESENTATIONS FOR ROBUST FACE AND FACIAL EXPRESSION RECOGNITION

by

Sima Taheri

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2013

Advisory Committee:
Professor Rama Chellappa, Chair/Advisor
Professor David Jacobs
Professor Amitabh Varshney
Professor Larry Davis
Professor Wojtek Czaja

© Copyright by
Sima Taheri
2013

Dedication

Dedicated to Mom, Dad, Farshad and Elissa

Acknowledgments

I owe my gratitude to all the people who have made this dissertation possible and because of whom my graduate experience has been one that I will cherish forever.

First and foremost I'd like to thank my advisor, Professor Rama Chellappa for giving me an invaluable opportunity to work on challenging and extremely interesting projects over the past five years. The extremely high standards he sets for himself and the professional heights he has scaled will always remain an inspiration for me. Inspite of his professional achievements, his polite persona and humility teach valuable lessons.

I have been fortunate to find an opportunity to work with Prof. A. N. Rajagopalan. It has been a precious experience to work with and learn from him. I would like to thank Professor David Jacobs and Professor Amitabh Varshney for serving on both my proposal and dissertation committee. Thanks are also due to Professor Larry Davis and Professor Wojtek Czaja for agreeing to serve on my dissertation committee and for sparing their invaluable time reviewing the manuscript.

My graduate life has been enriched in many ways by former and current fellow graduate students at the Computer Vision Laboratory, among whom I should particularly mention Mahesh Ramachandran, Vishal Patel, Soma Biswas, Ruonan Li, Dikpal Reddy, Kaushik Mitra, Raghuraman Gopalan, Nitesh Shroff, Nazre Batool, Qiang Qiu, Ming Du, Ming Yu Liu, Jaishanker Pillai, Jie Ni and Yi-Chen Chen. Also special gratitude is due to Dr. Aswin Sankaranarayanan and Dr. Pavan Turaga -

colleagues and collaborators - whose great mentoring during my early graduate days proved invaluable to me.

I would also like to acknowledge the help and support from several staff members who make it possible for us to find our ways through the administrative process – Janice Perrone (CfAR), Jennifer Story (CS), Fatima Bangura (CS), the UMIACS staff, Arlene Schenk, and Edna Walker, and the UMIACS computing staff.

I do not think I can do justice with words in acknowledging the role of my husband - Farshad. I also owe my deepest thanks to my family - my mother and father who have always stood by me and guided me through my career, and my brother and siter, Saeed and Sara. Words cannot express the gratitude I owe them.

I would like to acknowledge financial support from the Office of Naval Research (ONR), Physics, for all the projects discussed herein.

It is impossible to remember all, and I apologize to those I've inadvertently left out.

Lastly, thank you all and thank God!

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

Two important problems that computer vision researchers have been dealing with are object and action recognition. Effective solutions to these problems can have a wide range of applications including human-computer interaction (HCI), visual surveillance, video editing, search, retrieval and indexing. Previous works have demonstrated the challenges involved in solving these problems due to object and camera motion, variations in expression, posture, motion and clothing, illumination variations, blur effects and occlusion. These challenges motivate the need for robust algorithms for object and action recognition.

Human faces are arguably the most extensively studied object in computer vision. This is partly due to the remarkable face recognition capability of the human visual system and partly due to numerous important applications for automatic face recognition technologies. In addition, technical issues associated with face recognition are representative of object recognition problem in general. Face analysis based on 2D intensity images requires accounting for various sources of variation due to changes in illumination, view point, as well as blur on the face and non-rigid facial deformations. Therefore, it is necessary to have proper models for these changes which requires understanding the sources that contribute to the variations and the ways in which they interact with the faces. Robust face recognition vastly benefits from such model-based approaches.

The actions related to the human faces are facial expressions which are important channels of non-verbal communications. Facial expression analysis refers to algorithms that attempt to automatically analyze and recognize facial motions and facial feature changes from visual information. It has been an active research topic for behavioral scientists since the work of Darwin in 1872 [9]. The facial changes can be either identified as facial action units, proposed by Ekman *et al.* [10], or prototypic emotional expressions (e.g. Happy, Sad, Surprise, Disgust, Fear and Anger).

Facial expression analysis is also affected by changes in view-point and inter-subject variations in performing different expressions. Therefore, developing robust algorithms for expression recognition is of interest. Depending on the facial feature extraction methods as well as the types of input data (2D or 3D), the effects of these variations are different and they can be eliminated using different approaches. For example, the effect of in-plane face rotation and different scales of the faces can be eliminated by face normalization before extracting features or by proper feature representation before attempting expression recognition.

In order to develop recognition algorithms, we often make some simplifying assumptions for the image-formation process such as the Lambertian reflectance model and convexity assumptions for face, an affine model for the variations due to pose changes or the convolutional model for the blurred images etc. These assumptions lead to constraints on the set of features thus obtained which usually cause the features to lie on particular subspaces or a manifold. Therefore once the underlying assumptions and constraints are well understood, the next important step is to design recognition algorithms that are consistent with the algebra and geometry of the constraint set.

This dissertation makes an attempt to address some of the object and action recognition challenges by exploiting the idea of proper subspace representations for data. Our endeavor in this direction had been focused on the following problems.

1.1 Joint Video-based Albedo Estimation and Pose Tracking from Video

The albedo of a Lambertian object is a surface property that contributes to an object's appearance under changing illumination. As a signature independent of illumination, the albedo is useful for object recognition. Single image-based albedo estimation algorithms suffer due to shadows and non-Lambertian effects of the image. In this dissertation, we propose a sequential algorithm to estimate the albedo from a sequence of images of a known 3D object in varying poses and illumination conditions. We first show that by knowing/estimating the pose of the object at each frame of a sequence, the object's albedo can be efficiently estimated using a Kalman filter. We then extend this for the case of unknown pose by simultaneously tracking the pose as well as updating the albedo through a Rao-Blackwellized particle filter. More specifically, the albedo is marginalized from the posterior distribution and estimated analytically using the Kalman filter, while the pose parameters are estimated using importance sampling and by minimizing the projection error of the face onto its spherical harmonic subspace, which results in an illumination-insensitive pose tracking algorithm.

1.2 Spatially Varying Blur Descriptor for Robust Face Recognition

Understanding the effect of blur is an important problem in unconstrained visual analysis. Especially, with the increase in the usage of hand-held cameras, which usually generate images with lots of motion blur, the problem of blur-robust face recognition has become more and more important. While most of the existing methods use the convolutional model with a blur kernel to describe the process of forming a blurred image from a clean image, it is known that blur due to camera shake is significantly non-uniform across the image. In this dissertation, we propose a blur-robust face recognition algorithm for images with space-variant blur. By modeling a blurred face as a weighted average of geometrically transformed instances of its clean face, we build a matrix, for each gallery face, whose column space spans the space of all the motion blurred images obtained from the clean face. Considering the sparse representation of the blurred face in this space, we optimize a proper energy function with l_1 constraint on the coefficients to obtain the sparse coefficients as well as the identity of the face. We use a proper multiscale implementation to make the process efficient in terms of both computations and memory requirements.

1.3 Towards View-Invariant Expression Analysis Using Analytic Shape Manifolds

To develop robust and generalizable models for expression analysis one needs to break the dependence of the models on the choice of the coordinate frame of the camera, i.e. expression models should generalize across facial poses. To perform this systematically, one needs to understand the space of observed images subject to projective transformations. However, since the projective shape-space is cumbersome to work with, we address this problem by deriving models for expressions on the affine shape-space as an approximation to the projective shape-space by using a Riemannian interpretation of deformations that facial expressions cause in different parts of the face. We use landmark configurations to represent facial deformations and exploit the fact that the affine shape-space can be studied using the Grassmann manifold. This representation enables us to perform various expression analysis and recognition algorithms without the need for normalization as a preprocessing step.

1.4 Structure-Preserving Sparse Decomposition for Facial Expression Analysis

Although facial expressions can be decomposed in terms of action units (AUs) as suggested by the Facial Action Coding System (FACS), there have been only a few attempts that recognize expression using AUs and their composition rules. We propose a dictionary-based approach for facial expression analysis by decomposing expressions in terms of AUs. First, we construct an AU-dictionary using domain experts' knowledge of AUs. To incorporate the high-level knowledge regarding expression decomposition and AUs, we then perform structure-preserving sparse coding by imposing two layers of grouping over AU-dictionary atoms as well as over the test image matrix columns. We use the computed sparse code matrix for each expressive face to perform expression decomposition and recognition. Since domain experts' knowledge may not always be available for constructing an AU-dictionary, we also propose a structure-preserving dictionary learning algorithm which we use to learn a structured dictionary as well as divide expressive faces into several semantic regions.

1.5 Component-based Recognition of Faces and Facial Expressions

Most of the existing methods for the recognition of faces and expressions consider either the expression-invariant face recognition problem or the identity-independent facial expression recognition problem. We propose a joint face and facial expression recognition algorithm using a dictionary-based component separation algorithm (DCS). In this approach, the given expressive face is viewed as a superposition of a neutral face component with a facial expression component which is sparse with respect to the whole image. This assumption leads to a dictionary-based component separation algorithm which benefits from the idea of sparsity and morphological diversity. This entails building data-driven dictionaries for neutral and expressive components. The DCS algorithm then uses these dictionaries to decompose an expressive test face into its constituent components. The sparse codes we obtain as a result of this decomposition are then used for joint face and expression recognition.

1.6 Organization of the Dissertation

In Chapter 2, we discuss our algorithm for joint video-based albedo estimation and illumination-invariant head pose tracking. Chapter 3 describes the proposed blur-robust face recognition algorithm for the cases that we have spatially varying blur due to camera motion. Chapter 4 discusses a step towards view-invariant expression recognition using analytic shape manifolds and shows its potential for further extension. In Chapter 5, we propose a dictionary-based approach for facial expression analysis which is based on structure-preserving dictionary learning algorithm. In Chapter 6, a joint face and facial expression recognition algorithm using a dictionary-based component separation algorithm (DCS) is presented. Chapter 7 discusses the directions for future work.

Chapter 2: Joint Video-based Albedo Estimation and Pose Tracking from Video

2.1 Introduction

Variations in the visual appearance of an object mostly arise due to changes in illumination and pose [11]. Therefore, understanding the interaction of the objects' surface with irradiated light (illumination) and subsequent imaging with a camera (pose) is important for a wide range of computer vision applications. When a surface exhibits Lambertian reflectance, an illumination-insensitive property of the surface is the *albedo* which is a surface reflectance property that contributes to the object's appearance under changing illumination.

Estimating the reflectance properties of human faces has been of interest for decades [1, 2, 12–16]. But the proposed algorithms and assumptions underlying the development of algorithms are application dependent. Although faces are neither exactly Lambertian nor entirely convex, many algorithms make convex Lambertian assumption for the face. Such assumptions are reasonable for applications where the goal is to find a signature that is independent of illumination for representing and recognizing faces across illumination variations [1, 17–19] and *not* to analyze the

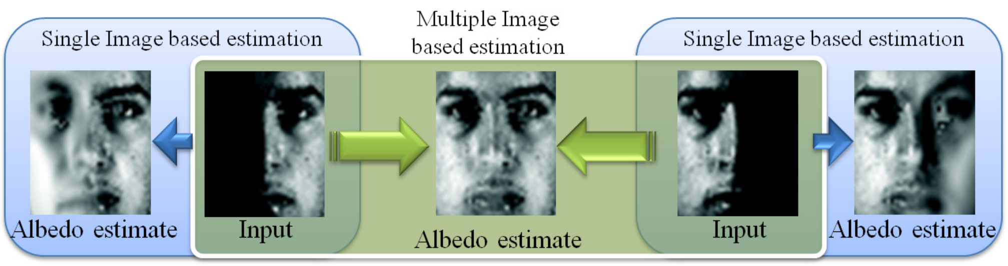

Single Image based estimation
Multiple Image based estimation
Single Image based estimation
Albedo estimate
Input
Albedo estimate
Input
Albedo estimate

Figure 2.1: Benefit of albedo estimation using multiple images. Two views of a face are shown as input images in the left and right columns and their corresponding albedos are estimated using [1]. The albedo maps shown in the right and left columns are noisy and partly based on the mean albedo. The middle column shows the improved estimated albedo map using both views of the face.

reflectance field of the face or to render a face [20,21]. In this work, we also make the convex Lambertian assumption for faces and explore a robust and computationally efficient method for recovering the albedo of a known 3D facial surface[1] from multiple images or a video. While the discussion of this chapter is mainly on estimating the albedo of human face, the proposed algorithm can be applied to generic objects under Lambertian reflectance assumption. However, since we do not explicitly model nonrigid deformations, the objects either should be rigid or only have small nonrigid deformations.

Much of the progress in facial albedo estimation has been achieved using a single image of the object under unknown lighting conditions. However, most of the existing approaches are based on restrictive assumptions on objects and illumination conditions, [1, 2], or are computationally intense due to iterative optimization procedures used for obtaining the solution, [22]. On the other hand, the ability to handle multiple images goes a long way in overcoming the shortcomings of single image-based algorithms due to its inherent advantage of having more information

[1]An average 3D face model is used.

(see Fig. 2.1 for a motivating example).

Multiple image-based algorithms mostly process the data in the batch mode [15, 23–27]. However, recursive processing of a set of images is important especially when a video is being processed. Hence our main focus is on recursive estimation of albedo from multiple frames in a video. Since the presence of multiple images comes with the possibility of additional variations (e.g. in the pose), efficient fusion of available information over the images is important as it leads to a more accurate and robust estimate of albedo. The goal of this work is recursive/sequential albedo estimation for improved pose tracking and recognition of faces modeled as Lambertian objects.

For a Lambertian object in a known pose and illumination, the observed image is linear in its albedo. This allows us to formulate the problem of multi-view albedo estimation as one of Kalman filtering. In particular, the unknown albedo is defined as the static state vector of the Kalman filter. However, since the pose of the face is usually unknown, the albedo estimation step is coupled with a pose tracking step to realize a joint albedo and pose estimation algorithm. We set this problem in a Bayesian inference framework and efficiently solve it using a Rao-Blackwellized particle filter. This allows us to perform efficient analytical inference using a Kalman filter over the albedo state-space and computationally intensive inference using particle filters over a smaller state-space encompassing just the pose parameters.

The joint tracking and albedo estimation approach allows us to incorporate illumination-insensitivity into a tracking algorithm. This is achieved by defining the particle weights using the projection error onto the spherical harmonic subspace of the current observation. We demonstrate the computational and numerical advantages as well as the limitations of our algorithm using several experiments.

It is worth pointing out that the problem studied in this chapter has close connections to the photometric stereo problem. Specifically, photometric stereo [28] refers to surface reconstruction of a static scene from multiple images taken under varying illumination. Under appropriate reflectance model (Lambertian [12], specular [29]), the image intensities at each pixel can be expressed in terms of unknown surface parameters (typically, the surface normal) and illumination. This static scene assumption obviates the need for accurate registration, and in many cases, surface estimates can be individually obtained at each pixel. As a result, photometric stereo and its variants (which includes structured lighting) are among the most precise methods for accurate shape recovery. In this chapter, we consider the scenario of a *non-static* scene under *changing* illumination. While this puts us beyond the traditional setup of photometric stereo, some of the core concepts in photometric stereo are highly relevant to our problem formulation and solution.

Contributions: In this chapter we address joint video-based albedo estimation and illumination-insensitive pose tracking. Our contributions are as follows:

- When the pose of the object is known, we propose an efficient video-based sequential albedo estimation using the Kalman filter.
- When the pose of the object is unknown, we show that pose and albedo estimation of an object from a video sequence can be performed using a computationally efficient Rao-Blackwellized particle filter.

- Finally, we propose an approach that eliminates the need for recalculating the spherical harmonic bases at each pose by exploiting the physical properties inherent to Lambertian objects.

Outline of the chapter: We discuss related work in Section 2.2. The problem of albedo estimation is addressed in Section 2.3. Subsequently, a Rao-Blackwellized particle filter is proposed in Section 2.4 for joint pose tracking and albedo estimation. Experimental results are presented in Section 2.5.

2.2 Related Work

In this section, we review some previous efforts undertaken for recovering the albedo of an object from an intensity image or a sequence of images/video of an object in different poses and illuminations. Since pose estimation is required for video-based albedo recovery, we also discuss some related work on illumination-insensitive pose tracking.

2.2.1 Albedo estimation

Estimating the facial albedo and the surface shape, as intrinsic factors pertinent to establishing facial identity, has been the focus of computer vision researchers for a long time. While significant efforts have been made to reduce the impact of extrinsic factors such as illumination and pose, the underlying problems still persist and are difficult to solve. We categorize the existing approaches into single image-based and multiple image-based approaches.

Single image-based approaches: Estimating the albedo, illumination direction and surface normals given a single intensity image is inherently ill-posed. Two approaches, namely shape-from-shading (SFS) approaches and model-based approaches, have been employed to make the problem more tractable.

Shape-from-shading approaches for object shape and albedo estimation make simplifying assumptions such as constant or piecewise constant albedo and known illumination direction [12,13]. These assumptions are not valid for many real objects and limit the practical applicability of these algorithms. Other SFS algorithms reduce the intractability of general albedo maps and surface normal estimation by using appropriate domain specific constraints, such as symmetry [30], or employing a statistical model for the shape [31–33]. In most of these approaches the main goal is shape estimation and albedo is incorporated to completely specify the image formation process.

The Retinex algorithm [34, 35] is one of the first approaches to estimating the lightness of surfaces. This algorithm uses a simplified model of intrinsic image statistics and makes the assumption that image derivatives with a large magnitude are caused by changes in the albedo of the surface, while derivatives with a small magnitude are caused by changes in illumination. Under this model, a shading image can be constructed by calculating the derivatives of the observed image, eliminating derivatives with a large magnitude, then reconstructing the image. However, this simple characterization of shading does not hold for many surfaces and this is the main disadvantage of this algorithm as noted in [36].

Model-based approaches, on the other hand, use the statistical knowledge of the 3D object model which regularizes this problem significantly [2, 37]. Blanz *et al.* [37] recovered the shape and albedo parameters of a 3D morphable model (3DMM) in an analysis-by-synthesis fashion. In order to handle more general lighting conditions, Zhang *et al.* [2] integrated the spherical harmonic illumination representation [11, 17] into the 3DMM approach, by modeling the texture component of the face using spherical harmonic bases. They proposed a feature point-based shape recovery algorithm followed by iterative estimation of albedo and illumination coefficients. However, their method can not handle the harsh lighting conditions due to the limited information that can be extracted from a single image.

To address this problem, Wang *et al.* [22, 38] proposed an optimization algorithm for albedo estimation which is robust to harsh illumination conditions and partial occlusion. By decoupling texture, geometry and illumination and modeling them separately they handle challenging conditions such as cast shadows and saturated regions. Their algorithm works by optimizing the energy function of a Markov Random field over albedo, shape and light resulting in a computationally expensive algorithm that may converge to a local optimum solution.

Biswas *et al.* [1, 39] proposed a stochastic filtering framework for albedo estimation for a frontal face as well as a face with unknown pose. They explicitly accounted for the error in the estimate of surface normals, illumination coefficients and pose to improve the albedo estimate. In their framework, the albedo estimate for pixels corrupted with a large noise is mainly based on the prior albedo and as a consequence it leads to unreliable estimation in noisy situations. Figure 2.1 shows the albedo estimated using their algorithm on two poorly illuminated faces of a subject. Each face by itself gives a poor estimate of the albedo, which is partly based on the mean albedo. However, estimating the albedo by fusing the information from both images leads to a much more accurate albedo map. This motivates the problem of multi-image or video-based albedo estimation.

Multiple image-based approaches: When an object rotates in front of a camera under distant and varying illumination, the appearance of the object changes both geometrically and photometrically. These changes provide clues to both shape and albedo of the object. Most of the approaches in this category combine multi-view stereo with photometric stereo to find correspondences across views and subsequently estimate the shape and albedo of the object [15,23–27]. But these algorithms operate in a batch processing mode and are computationally demanding [15].

Zhou *et al.* [14] proposed a factorization-based approach to fully recover the albedo and surface normal by imposing a rank, integrability and face symmetry constraints. But the important limitation of this algorithm as well as other multi-image based approaches is that they process all the images in a batch mode. However, it is necessary to develop algorithms that can work in the sequential mode and fuse the estimated parameters in previous frames with the newly available data while accounting for the various sources of error.

Non-Lambertian face modeling: As discussed earlier, the Lambertian assumption for faces is valid depending on the application. Recognizing faces across illumination variations using the albedo of the face as a signature independent of illumination has shown promising results [1,39]. On the other hand, there are other approaches that propose non-Lambertian models for analyzing the apparent bidirectional reflectance distribution function (BRDF) of the face [20,40]. While these non-Lambertian models are appropriate for computing photo-realistic facial animations as well as face relighting, they only bring slight improvements for face recognition as compared to the results obtained by making the Lambertian assumption [20].

2.2.2 Illumination-insensitive Pose Tracking

An important challenge in video-based albedo estimation is to find the pose of the face at each frame. Since albedo estimation is often coupled with a 3D shape model, a pose tracking algorithm is required to obtain the 3D configuration of the face at each frame. Several methods have been proposed for 3D face tracking, [16, 41–44], however, they are often sensitive to illumination variations.

Cascia *et al.* [43] formulated the tracking problem as an image registration problem in the cylinder's texture map image. To account for lighting variations, they modeled the residual error of registration as a linear combination of texture warping templates and orthogonal illumination templates. Marks *et al.* [44] proposed a generative model and stochastic filtering algorithm for 3D nonrigid object tracking with the aim of addressing the inefficiencies of template matching and optical flow-based algorithms. They combined the advantages of template matching and flow-based algorithms and performed the joint inference of 3D position, orientation and nonrigid deformations.

Xu and Roy-Chowdhury [16] proposed a bilinear space of motion and illumination in which they estimated the pose and illumination parameters by iterative optimization. They assumed that the illumination in the first frame is uniform and hence the intensity can be used as an estimate of the albedo. This assumption narrows down the domain of videos on which the algorithm can successfully perform tracking. Although explicit illumination modeling makes the tracking algorithm reasonably robust to illumination variations, their algorithm does not adequately model the albedo while tracking the face. The readers are referred to [45] for more detailed discussions on pose tracking.

2.3 Video-based Albedo Estimation

The key idea behind the sequential albedo estimation framework proposed in this chapter revolves around the linear relationship between the image observation and albedo. Under the Lambertian assumption for the face, the intensity reflected by a point p_i on the face, with the surface normal $\mathbf{n}_i$ and the albedo ρ_i, due to the lighting function l coming from direction $\mathbf{u}_l$ is modeled as:

$$I(p_i) = I_i = \rho_i \int l(\mathbf{u}_l) \max(\mathbf{n}_i.\mathbf{u}_l, 0) d\mathbf{u}_l \tag{2.1}$$

Lambert's cosine law is non-linear due to $\max(\mathbf{n}.\mathbf{u}_l, 0)$ which accounts for the formation of attached shadows. However, in a seminal work, Basri and Jacobs [17] showed that images of a face (specifically, any convex Lambertian object) under varying illumination are closely approximated by a 9-dimensional (linear) subspace using a spherical harmonic decomposition. Let y_{nm} be the spherical harmonic basis of *order* n and *degree* m. Note that spherical harmonic bases are functionals on a sphere, i.e, $\mathsf{y}_{nm} : \mathbb{S}^2 \mapsto \mathbb{R}$. For the rest of the chapter, we parametrize $\mathbb{S}^2$ using unit norm vectors in $\mathbb{R}^3$. Any arbitrary lighting function l over a scene can be described as a function in $\mathbb{S}^2$ if the light sources are at infinity (or in practice, sufficiently far away). In such a case, the lighting function l can be described using the spherical harmonic bases as,

$$l(\mathbf{u}_l) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} l_{nm} \mathsf{y}_{nm}(\mathbf{u}_l) \tag{2.2}$$

Basri and Jacobs also showed that the Lambertian kernel, $\max(\mathbf{n}.\mathbf{u}_l, 0)$, acts as a smoothing filter on the light source, and the image of the object produced in that lighting condition depends heavily on the lower order spherical harmonic basis elements. Therefore the generated image can be well approximated using just the first 9 basis elements as,

$$\begin{aligned} I_i &\approx \rho_i \sum_{n=0}^{2} \sum_{m=-n}^{n} l_{nm} \alpha_n \mathsf{y}_{nm}(\mathbf{n}_i) \\ &= \rho_i \sum_{n=0}^{2} \sum_{m=-n}^{n} l_{nm} Y_{nm}(\mathbf{n}_i) \end{aligned} \tag{2.3}$$

where $\{\alpha_n\}$ are the coefficients of the spherical harmonic expansion of the Lambertian kernel. For a given object of d pixels described using a set of albedos $\{\rho_i\}$ and normal vectors $\{\mathbf{n}_i\}$, we can now construct the so called *spherical harmonic basis images* (SHBI) $\{\mathbf{Y}_{nm} \in \mathbb{R}^d | n = 0, 1, 2; m = -n, \ldots, n\}$ such that $\mathbf{Y}_{nm} = \{Y_{nm}(\mathbf{n}_i)\} = (\alpha_n \mathsf{y}_{nm}(\mathbf{n}_1), \alpha_n \mathsf{y}_{nm}(\mathbf{n}_2), \ldots, \alpha_n \mathsf{y}_{nm}(\mathbf{n}_d))^T$.

Using (2.3), the intensity observed at the ith pixel of the face with a known pose and illumination is written as

$$I_i = \rho_i Y^T(\mathbf{n}_i) L + \nu_i, \tag{2.4}$$

where $Y(\mathbf{n}_i) = (Y_{00}(\mathbf{n}_i), \ldots, Y_{22}(\mathbf{n}_i))^T \in \mathbb{R}^9$ encodes the pose using spherical harmonic basis values at that pixel and $L = (l_{00}, \ldots, l_{22})^T \in \mathbb{R}^9$ encodes the lighting positions. Moreover, the observation noise, $\nu = \mathcal{N}(0, \Sigma_v)$, is defined to have a multivariate Gaussian distribution and it models the surface deviation from the Lambertian assumption (specularity, cast shadow, and saturated pixels). Then, the intensity vector for the whole face can be expressed as,

$$I = diag(\rho)\mathbf{Y}L + \nu, \tag{2.5}$$

where $\rho = (\rho_1, \ldots, \rho_d)^T \in \mathbb{R}^d$ and the $d \times 9$ matrix $\mathbf{Y} = [\mathbf{Y_{00}}, \ldots, \mathbf{Y_{22}}]$ encodes the spherical harmonic basis images. Here, the subject intrinsic matrix $B = [diag(\rho)\mathbf{Y}]$ defines an illumination-insensitive subspace of all images of the face under arbitrary illuminations. Given an observation I, we can use the projection error onto this subspace to define an observation model that is illumination-insensitive as well.

For a video of the face where the pose is known, the albedo can be optimally updated over time using the Kalman filter. However before describing the Kalman filter framework for albedo estimation, we discuss how we can avoid recalculating the spherical harmonic basis images at each frame which makes the algorithm computationally efficient.

2.3.1 Head Orientation vs. Illumination Direction

Representation of the face in (2.3) relates the 3D structure of the face, the illumination direction and the face albedo to the generated image. This representation is suited for situations where the pose is fixed and only the illumination varies. However, when the pose changes in a video, the basis images $\{\mathbf{Y_{nm}}\}$ differ from frame to frame as the normal vectors associated with a point of the face change due to rotation. Therefore, to estimate albedo, we need to recalculate the basis images at each frame which is computationally inefficient. Xu and Roy-Chowdhury [16] addressed this problem by proposing a bilinear subspace formulation for joint illumination and motion. They combined the effects of motion, illumination and 3D structure in generating a sequence of images. But their approach still requires computing the bilinear bases at each frame which is time consuming. Here, we resolve this problem by exploiting a property of Lambert's law.

For Lambertian objects, the apparent intensity of a surface patch to an observer depends on the angle between its surface normal, $\mathbf{n}_i$, and the incident illumination direction, $\mathbf{u}_l$, and is independent of each of these directions separately. Rotation of the face changes the direction of $\mathbf{n}_i$, which leads to change of angle between $\mathbf{n}_i$ and $\mathbf{u}_l$ and change of intensity at p_i as a result. But the same change in $\langle \mathbf{n}_i . \mathbf{u}_l \rangle$ is obtained if we keep $\mathbf{n}_i$ fixed and rotate the illumination source by the same magnitude and about the same axis, but in the opposite direction.

When the face is rotated by R, while the illumination is fixed, the intensity of the point p_i changes as

$$I_R(p_i) = \rho_i \sum_n \sum_m l_{nm} Y_{nm}(R(\mathbf{n}_i)) \tag{2.6}$$

where $\{Y_{nm}(R(\mathbf{n}_i))\}$ are the basis values in the new pose. Recalculating the basis images for each new pose is computationally expensive. On the other hand, the rotation of spherical harmonics using the transformation matrix $D(R)$ has been studied in [11, 46, 47]. Using this idea and (2.6), the intensity of the point p_i can be written as

$$\begin{aligned} I_R(p_i) &= \rho_i \sum_n \sum_m l_{nm} \sum_{m'} D^n_{mm'}(R) Y_{nm'}(\mathbf{n}_i) \\ &= \rho_i L^T(D(R)Y(\mathbf{n}_i)) = \rho_i Y(\mathbf{n}_i)^T(D^T(R)L) \end{aligned} \tag{2.7}$$

where D is the 9×9 spherical harmonics transformation matrix [46]. This expresses the new intensity at p_i in terms of original harmonic basis values, $Y(\mathbf{n}_i)$ and the transformed illumination coefficients. As this equation suggests, the illumination coefficients should be transformed by D^T, which is the inverse of matrix D, to compensate for the head rotation. It should be noted that transforming the 9-dimensional illumination coefficients is more robust than rotating the whole $9 \times d$ harmonic basis matrix. Moreover, this is justified by the Lambertian property, as discussed before.

We can now use the same basis images, $\{\mathbf{Y_{nm}}\}$, for representing the face throughout the image sequence and only compute the new illumination coefficients

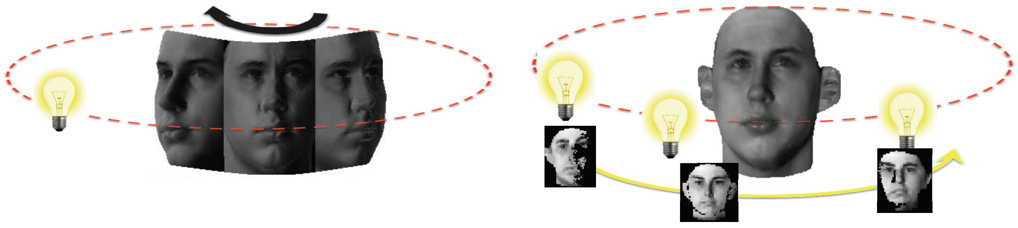


Figure 2.2: **(left)** Head rotation in front of a fixed illumination source can be replaced by **(right)** the illumination source rotation in opposite direction around the fixed face along with visibility modeling. As a consequence of the physical properties inherent to Lambertian objects, these two configurations result in the same observed intensities on the face.

for each new pose/frame. In this way we avoid recalculating the SHBI at each frame which makes the computation much faster and efficient. However, it should be noted that since rotation of the face makes some pixels disappear, a visibility test needs to be applied in order to remove the non-visible pixels from the current view. Since we have a 3D model for the face, visibility issues are solved easily. Figure 2.2 illustrates the idea proposed in this section.

2.3.2 Shape estimation

We calculate the 3D morphable shape model [48] by convex combinations of the shapes of m training examples in the Vetter dataset [49] followed by principal component analysis (PCA) as $s = \bar{s} + Sa$. The columns of S are the most significant eigenvectors s_i rescaled by their standard deviation and the coefficient a constitutes a pose-insensitive low-dimensional coding of a face. We can either use the mean shape, $\bar{s}$, throughout the process or compute a more accurate estimate of the 3D shape using the approach presented in [2].

Registering the 3D shape model to the face in the first frame can be performed

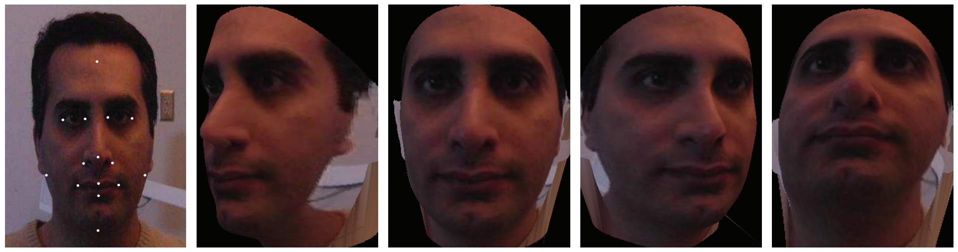


Figure 2.3: **(left to right)** Fifteen manually picked landmarks on the face, and the rendered 3D face in different views after registering the 3D model to the given face.

using the method proposed by Zhang *et. al* [2]. For a set of pre-selected feature points on the morphable model, we find the corresponding landmarks, s_{img}, on the first frame of the test video[2]. We set the initial coefficient a_0 to zero and register the average shape, $\bar{s}$, to the first frame using the algorithm proposed by [50]. This gives us the initial rotation, translation and scale parameters. We define the shape error at feature points as the difference between s_{img} and the new shape information of feature points in the model that was rendered by the recovered projection parameters. Then the vector of shape parameters, a, can be updated using the method proposed in [2]. We iterate through the shape parameter updating procedure until the amount of update falls below a threshold. Then the final shape parameters are used to get the face 3D shape model. This estimated shape model is used throughout the frames. Figure 2.3 shows a sample of the face with landmarks on it along with the registered shape with the face texture warped on it.

[2]We picked fifteen landmarks manually on the face in the first frame, but it can also be performed automatically using face and facial component detection method

2.3.3 Albedo Estimation using the Kalman Filter

As mentioned earlier, there are many single image-based algorithms for robust albedo estimation. However, to obtain an accurate estimate of the albedo from a sequence of images (sequential mode), estimates from individual images must be fused. We use the Kalman filter to fuse the information over time. Though the Kalman filter was originally designed to estimate the state of a time-varying system, it can be used on static processes as well. There are many instances where the Kalman filter has been used in this fashion [51,52]. In fact, the classic textbook by Maybeck [53] introduces the Kalman filter using a static example. It should be noted that when we apply the Kalman filter to a static process like the albedo map, the state transition model is given as $\rho_t = \rho_{t-1}$ and it is noiseless. We use the Kalman filter to sequentially update the albedo as more information becomes available over time. In such cases, as more observations are introduced, the albedo estimate converges to the true value.

The problem of video-based albedo estimation in a sequential mode can be formulated as follows. Given the estimate of albedo at frame/time $t-1$ characterized by its mean and covariance matrix, $\{\mu_{\rho,t-1}, \Sigma_{\rho,t-1}\}$, we want to update the posterior probability of the albedo $P(\rho|\mathcal{Z}^t, \Theta)$ as a new frame Z_t becomes available. Here, Z_t is the frame at time t, and $\mathcal{Z}^t = \{Z_1, ..., Z_t\}$. The parameter $\Theta = \{\theta_1, \ldots, \theta_t\}$, where θ_t denotes the surface pose at time t and, for now, is assumed to be known. Knowing this pose parameter, θ_t, an inverse warp of the 3D model of the face onto the image Z_t gives us a registered observation at time t as a d-dimensional intensity vector, $I_t = I_t(Z_t, \theta_t)$. Note that, we use a point-cloud model consisting of d points, each with a known normal $\mathbf{n}_i$ and an unknown albedo ρ_i.

Using Bayes' theorem, the posterior probability of the albedo can be sequentially updated as follows

$$P(\rho|\mathcal{Z}^t, \Theta) \propto P(Z_t|\rho, \mathcal{Z}^{t-1}, \Theta) P(\rho|\mathcal{Z}^{t-1}, \Theta) \tag{2.8}$$

where the posterior probability at time $t-1$, $P(\rho|\mathcal{Z}^{t-1}, \Theta)$, acts as the prior probability at time t to recursively update $P(\rho|\mathcal{Z}^t, \Theta)$. From (2.5), the likelihood function $P(Z_t|\rho, \mathcal{Z}^{t-1}, \Theta)$ can be written as

$$P(Z_t|\rho, \mathcal{Z}^{t-1}, \Theta) = P(I_t|\rho, \mathcal{Z}^{t-1}, \Theta) = \mathcal{N}(I_t|H_t\rho, \Sigma_{v,t})$$

where the observation matrix, $H_t = diag(h_t)$, is a $d \times d$ diagonal matrix with entries $h_{ti} = Y^T(\mathbf{n}_i)L_t$ defined for the ith pixel of the face. The illumination vector, L_t, is approximated at each frame using the albedo estimates from previous frames. Finally, the posterior estimate for the albedo at t is given by the following Kalman filter update equations:

$$\mu_{\rho,t} = \mu_{\rho,t-1} + K_t(I_t - H_t\mu_{\rho,t-1}) \tag{2.9}$$

$$\Sigma_{\rho,t} = (\mathbb{I} - K_t)\Sigma_{\rho,t-1} \tag{2.10}$$

where $\mathbb{I}$ is the identity matrix of size d and the Kalman gain K_t is defined as

$K_t = \Sigma_{\rho,t-1}(\Sigma_{\rho,t-1} + \Sigma_{v,t})^{-1}$.

Here, the prior albedo values (for faces), $\{\mu_{\rho,0}, \Sigma_{\rho,0}\}$, are estimated as the mean and covariance of the available training data in the Vetter dataset. Moreover, the initial observation noise covariance matrix, $\Sigma_{v,0}$, is learned using the training data when for each face the mean shape and mean albedo are used. We ignore the correlation among nearby pixels by defining Σ_ρ, Σ_v and therefore the Kalman gain, K, to be diagonal matrices. However, it should be noted that non-diagonal matrices can be used without significantly changing the fusion algorithm.

In order to make the albedo estimate robust against noisy pixels which are due to deviations from the Lambertian assumption, we update the observation noise covariance matrix, $\Sigma_{v,t}$, at each frame by assigning a very large value to those entries (corresponding to the pixels) whose observed intensities are above an upper threshold or below a lower threshold as well as to non-visible pixels. In this way we avoid the saturated pixels, pixels in cast shadows, pixels with specularity and occluded regions to affect the estimated value of the albedo.

As more knowledge about the albedo becomes available through new observations, the uncertainty in the static parameter ρ is updated. Equation (2.10) shows that the error covariance of the estimated ρ decreases over time (K_t's components are ≤ 1) and since the Kalman filter at each frame gives an unbiased MMSE estimate of ρ, a decrease in the error covariance indicates improvement in the estimated parameter over the time. In the ideal case where each pixel's intensity satisfies the Lambertian property in some frames over the sequence (i.e. not counting saturation or shadows), K_t and $\Sigma_{\rho,t}$ converge to zero and the final albedo estimate, μ_ρ, remains

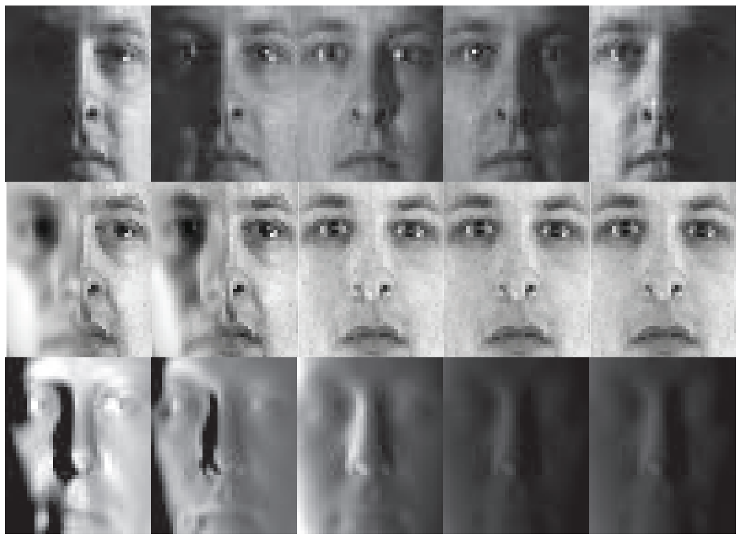


Figure 2.4: The first row shows some frames of a synthetic sequence obtained from the PIE illumination dataset. The estimated albedo maps using faces up to these frames are shown along with their corresponding Kalman gains in the second and third row, respectively. Shown below the third row are the frame numbers. Note how the Kalman gain converges to zero when all parts of the face gets well lit.

unchanged.

Figure 2.4 shows the result of applying the Kalman filter for sequential albedo estimation on a synthetic sequence. In this sequence, the head pose is fixed so as to focus on the performance of the Kalman filter for updating the state of a static parameter. It should be noted that while for a multiple image problem in which the pose is fixed (and so the correspondences are known) batch processing algorithms such as photometric stereo can be applied to get an accurate estimate of the object shape and albedo, our emphasis is on developing a sequential approach.

Figure 2.4 shows some frames of a sequence synthesized using the PIE illumination images [54]. The sequence starts with a face under harsh illumination conditions and then the light source rotates in front of the face. The figure also shows the albedo map estimated for the face up to each frame along with the corresponding Kalman gain, K_t, at each frame. The Kalman gain images show how the algorithm assigns large weights to informative pixels in the current frame and reduces the weights of badly illuminated pixels as well as pixels that violate the Lambertian assumption. As the algorithm proceeds through the frames, the albedo estimate improves and finally stabilizes as K_t goes to zero.

2.4 Pose Tracking and Albedo Estimation

When the pose of the face is unknown, analytical inference of the albedo can still be done when the albedo posterior probability is conditioned by the head pose. This observation motivates the use of the Rao-Blackwellized particle filter (RBPF). Rao-Blackwellization of a particle filter involves splitting the state variables into two sets, such that analytical inference is possible on one set conditioned on the other [55]. Rao-Blackwellization leads to more accurate estimates of state parameters with fewer particles. It has been applied to various problems such as joint rigid and non-rigid face tracking [44] and joint face tracking and head pose estimation [55].

We characterize the head pose as a function of rotation and translation of the head, where the rotation and translation are described using 3-dimensional vectors $\mathbf{r} = \{r_1, r_2, r_3\}$ and $\mathbf{t} = \{t_1, t_2, t_3\}$, respectively. Using a 3D shape model registered to the face in the first frame, the goal is to obtain a trajectory of the pose parameter evolution $\theta_t = \{\mathbf{r}, \mathbf{t}\}_t$, over the frames as well as an estimate for albedo. To this end, at each time instant t, the RBPF uses the hybrid particle set $\{\theta_t^{(i)}, w_t^{(i)}, \mu_{\rho,t}^{(i)}, \Sigma_{\rho,t}^{(i)}\}$ to approximate the posterior $P(\theta_t, \rho | \mathcal{Z}^t)$ over the joint state vector $S_t = \{\theta_t, \rho\}$. Here

$\{\theta_t^{(i)}, w_t^{(i)}\}$ approximate the posterior of pose parameters, $P(\theta_t|\mathcal{Z}^t)$, and $\{\mu_{\rho,t}^{(i)}, \Sigma_{\rho,t}^{(i)}\}$ form the analytical estimate (in terms of the mean and covariance of a Gaussian density) of the albedo associated with each pose particle, $\theta_t^{(i)}$, obtained using the Kalman filter described in Section 2.3.3. The posterior distribution $P(\theta_t, \rho|\mathcal{Z}^t)$ can be written as,

$$\begin{aligned} P(\theta_t, \rho|\mathcal{Z}^t) &\propto P(Z_t|\theta_t, \rho) \times \\ \int_{\theta_{t-1}} \int_{\mu_{\rho,t-1}} & P(\theta_t, \rho|\theta_{t-1}, \mu_{\rho,t-1}) P(\theta_{t-1}, \mu_{\rho,t-1}|\mathcal{Z}^{t-1}) \end{aligned} \tag{2.11}$$

By integrating out the albedo part of the state vector, we obtain a marginal filter for pose parameter, θ_t, as follows,

$$\begin{aligned} P(\theta_t|\mathcal{Z}^t) &\propto \int_{\rho} P(Z_t|\theta_t, \rho) \times \\ \int_{\theta_{t-1}} \int_{\mu_{\rho,t-1}} & P(\theta_t, \rho|\theta_{t-1}, \mu_{\rho,t-1}) P(\theta_{t-1}, \mu_{\rho,t-1}|\mathcal{Z}^{t-1}) \end{aligned} \tag{2.12}$$

Here, the posterior $P(\theta_{t-1}, \mu_{\rho,t-1}|\mathcal{Z}^{t-1})$ is approximated over the previous joint state by a set of particles $\{\theta_{t-1}^{(i)}, w_{t-1}^{(i)}, \mu_{\rho,t-1}^{(i)}, \Sigma_{\rho,t-1}^{(i)}\}$ as

$$\begin{aligned} P(\theta_{t-1}, \mu_{\rho,t-1}|\mathcal{Z}^{t-1}) &= P(\theta_{t-1}|\mathcal{Z}^{t-1}) P(\mu_{\rho,t-1}|\theta_{t-1}, \mathcal{Z}^{t-1}) \\ &\propto \sum_i w_{t-1}^{(i)} \delta(\theta_{t-1}^{(i)}) \alpha_{t-1}^{(i)}(\mu_{\rho,t-1}) \end{aligned} \tag{2.13}$$

where $\alpha_{t-1}^{(i)}(\mu_{\rho,t-1})$ is defined as the density on $\mu_{\rho,t-1}$ conditioned on the pose of the

ith particle and the measurements $\mathcal{Z}^{t-1}$:

$$\alpha_{t-1}^{(i)}(\mu_{\rho,t-1}) = P(\mu_{\rho,t-1}|\theta_{t-1}^{(i)}, \mathcal{Z}^{t-1}) \tag{2.14}$$

Substituting (2.13) into the expression for the marginal filter (2.12) we obtain the following Monte-Carlo approximation to the exact marginal Bayes filter,

$$\begin{aligned} P(\theta_t|\mathcal{Z}^t) \propto \sum_i w_{t-1}^{(i)} \int_\rho P(Z_t|\theta_t, \rho) \times \\ \int_{\mu_{\rho,t-1}} P(\rho|\theta_t, \theta_{t-1}^{(i)}, \mu_{\rho,t-1}) P(\theta_t|\theta_{t-1}^{(i)}, \mu_{\rho,t-1}) \alpha_{t-1}^{(i)}(\mu_{\rho,t-1}) \end{aligned} \tag{2.15}$$

This approximation has a complicated form which makes it intractable in general. In theory, it is possible to directly sample from the approximation, but this is both computationally and analytically difficult. Hence, to obtain a practical algorithm we make one additional assumption that the head motion model for pose θ_t does not depend on the albedo value $\mu_{\rho,t-1}$ at time $t-1$, $P(\theta_t|\theta_{t-1}^{(i)}, \mu_{\rho,t-1}) = P(\theta_t|\theta_{t-1}^{(i)})$. So we can now move the motion model out of the integral in (2.15), yielding

$$\begin{aligned} P(\theta_t|\mathcal{Z}^t) \propto \sum_i w_{t-1}^{(i)} P(\theta_t|\theta_{t-1}^{(i)}) \int_\rho P(Z_t|\theta_t, \rho) \times \\ \int_{\mu_{\rho,t-1}} P(\rho|\theta_t, \theta_{t-1}^{(i)}, \mu_{\rho,t-1}) \alpha_{t-1}^{(i)}(\mu_{\rho,t-1}) \end{aligned} \tag{2.16}$$

Now we can perform importance sampling in the usual way, using the predictive density $\sum_i w_{t-1}^{(i)} P(\theta_t|\theta_{t-1}^{(i)})$ as the proposal density. We define the observation model of the particle filter as the projection error of the observed intensity vector $I_t^{(i)} = I_t(Z_t, \theta_t^{(i)})$ onto the subspace of spherical harmonic images. This makes the tracking component of the particle filter illumination-insensitive. Toward this end, the importance weights, $\{w_t^{(i)}\}$, are estimated as follows. For each pose particle $\theta_t^{(i)}$, first the spherical harmonic basis images (including albedo) are calculated as $B_t^{(i)} = diag(\mu_{\rho,t}^{(i)})\mathbf{Y}$, where $\mu_{\rho,t}^{(i)}$ is the analytical estimate of albedo obtained using (2.9). Here, $\mathbf{Y}$ is the $d \times 9$ matrix which, based on the discussion in Section 2.3.1, is fixed regardless of the head pose parameter $\theta_t^{(i)}$. By leveraging the Gaussian assumption in (2.5), the importance weights are defined as

$$w_t^{(i)} \propto \exp(-\frac{1}{2}\|I_t^{(i)} - B_t^{(i)}L_t^{(i)}\|^2_{\Sigma_v}), \text{where } L_t^{(i)} = (B_t^{(i)})^\dagger I_t^{(i)} \qquad (2.17)$$

Here, $B_t^{(i)}L_t^{(i)}$ is the projection of the observation vector $I_t^{(i)}$ onto the basis $B_t^{(i)}$, and $(B_t^{(i)})^\dagger$ is the pseudo-inverse. Figure 2.5 presents a summary of the proposed algorithm.

2.5 Experimental Results

In this section, we present experimental results on joint face tracking and albedo estimation. We report the results on some synthetic sequences generated using the 3D Vetter dataset [49] and images from the PIE-illumination dataset [54]. The PIE-illumination dataset has several images of a subject taken under different illumination conditions. These images can be used to generate a synthetic video of the fixed head under desired illumination variations. We also show results on real

Initialization:

- **model registration:** Fifteen landmark points are manually selected on the face in the first frame using which the initial pose parameters, θ_0, are estimated.
- **parameter setting:** We use the mean albedo and its variance over training data as the initial albedo, $\mu_{\rho,0}$, and the initial Kalman filter error covariance matrix, $\Sigma_{\rho,0}$, respectively. The 3DMM is calculated using the training data in the Vetter dataset (we mostly use the mean shape as the 3D model in our experiments). The initial illumination coefficients are obtained using the mean albedo and current observation.

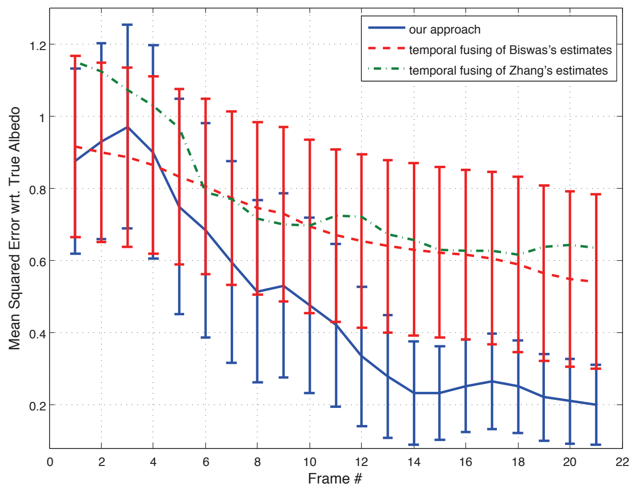
RBPF iterations: Starting from the posterior approximation $P(\theta_{t-1}, \rho|\mathcal{Z}^{t-1})$ estimated by a set of N weighted hybrid particles $\{\theta_{t-1}^{(i)}, w_{t-1}^{(i)}, \mu_{\rho,t-1}^{(i)}, \Sigma_{\rho,t-1}^{(i)}\}$, repeat for $j \leq N$:

1. Sample from the dynamic model $P(\theta_t|\theta_{t-1}^{(i)})$ for a chosen $\theta_{t-1}^{(i)}$ to obtain a predicted pose parameter, $\hat{\theta}_t^{(j)}$.
2. Get the observation vector $I_t^{(j)}$ through an inverse warp of the 3D model on the current frame using $\hat{\theta}_t^{(j)}$ as the head pose and then find the intensity at the model vertices.
3. Update $\mu_{\rho,t-1}^{(i)}$ and $\Sigma_{\rho,t-1}^{(i)}$ to $\mu_{\rho,t}^{(j)}$ and $\Sigma_{\rho,t}^{(j)}$ according to the Kalman filter equations (2.9,2.10).
4. Calculate the importance weight $w_t^{(j)}$ using (2.17).

Figure 2.5: Summary of the algorithm for joint albedo estimation and pose tracking.

sequences from the BU dataset [43]. This dataset has various sequences per subject in which significant illumination changes and 2D (in-plane) and 3D (out-of-plane) head rotations exist. The resolution of frames is 320 × 240 (non-interleaved) and the videos are collected at 30 fps.

To have examples with more extreme lighting conditions, we also collected some sequences of rotating heads in front of a fixed lighting source to evaluate the performance and limitations of our algorithm in situations where the Lambertian



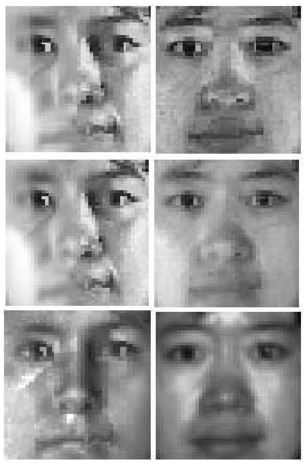


Figure 2.6: The plot in the left column shows the MSE of estimated albedo for synthetic PIE sequences averaged over 68 subjects. We compared our results at each frame with those obtained from the temporal fusion of Biswas' [1] and Zhang's [2] estimates up to each frame. The right column shows the estimated albedos for a chosen subject obtained from (from top to bottom) our approach, [1], and [2]. The faces in the left column are the estimates obtained in the first frame and the right column shows the final albedo estimates. Both visualizations indicate the superior performance of the proposed approach.

assumption for the face is largely violated. It should be noted that the ground truth albedo maps are available for the PIE and Vetter datasets. But for the BU dataset, there are sequences taken under uniform illumination which can be used to obtain the ground truth albedos with scale ambiguity for each subject. We should also mention that we did not perform shape correction in these experiments and just used the mean shape for the faces. We also evaluate the effect of using the mean shape instead of the true shape in the estimated albedo error.

2.5.1 Albedo Estimation

In this section, the goal is to evaluate the performance of our video-based albedo estimation algorithm using various synthetic and real sequences.

Synthetic sequences: We compare our results with the results of Biswas *et al.* [1] and Zhang *et al.* [2] using 68 synthetic PIE sequences with fixed frontal faces and rotating illumination source around them. To ensure a fair comparison, we apply their algorithms at each frame separately and then fuse the estimated albedos by temporal averaging over the frames (computing the mean of the estimated albedo maps up to each frame). In other words, at each frame we obtain the albedo map by averaging over the estimated albedos form all the previous frames up to (including) current frame. Figure 2.6 illustrates the error curves for the three methods. As can be seen, the proposed algorithm achieves the best final albedo estimates compared to other approaches. This result shows the impact of using the Kalman filter to fuse the information over the frames. [2] gives estimates with large error in the initial frames due to the harsh illumination condition. On the other hand, the algorithm proposed in [1] obtains better initial albedo because of incorporating the error statistics in the calculations, but the estimate is not improved through fusion.

While estimating the 3D shape of the subjects will increase the accuracy of the results, it is time consuming. On the other hand, in many cases using the mean shape of the face can produce results with an acceptable level of accuracy. To show the effect of using the mean shape instead of the true shape for each face, we use 3D faces in the Vetter dataset to synthesize sequences with the head fixed at a position and rotating illumination, consisting of 20 frames. The albedos are estimated using both the mean shape and the true subject specific shapes of the faces. Figure 2.7 illustrates the effect of using the mean shape instead of the true shapes on the mean squared error of the estimated albedos with respect to

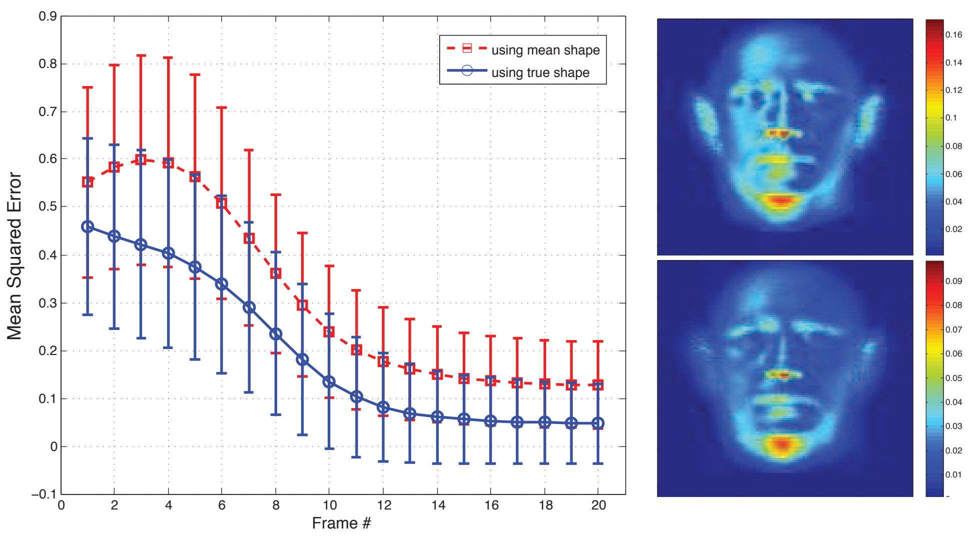


Figure 2.7: The effect of using the mean shape instead of the true 3D shape on the estimated albedo error with respect to the ground truth. The estimates are obtained for the synthesized Vetter sequences (with 20 frames, fixed face and rotating illumination source around the face). The mean estimated albedo errors are also shown using color coded images **top:** when the mean shape is used for albedo estimation, and **bottom:** when the true subject specific shapes are used for albedo estimation. The figure is best viewed in color.

the ground truth for 100 synthetic sequences. The figure also shows the spatial distribution of the mean albedo errors for the two situations. As it can be seen, the albedo errors are comparable for the two cases even for the regions around eyes and nose. Moreover, in both cases the error in the final albedo is considerably smaller than the initial error. Therefore, although using the mean shape leads to the larger final error, the fact that there is no need to know or estimate the true 3D shape of the face compensates for that.

Real sequences: Albedo estimation in real video sequences with changing pose is more challenging due to tracking errors as well as deviations from the Lambertian assumption.[3] Figure 2.8 illustrates the decrease in the error of the estimated

[3]This source of error exists in the PIE sequences as well

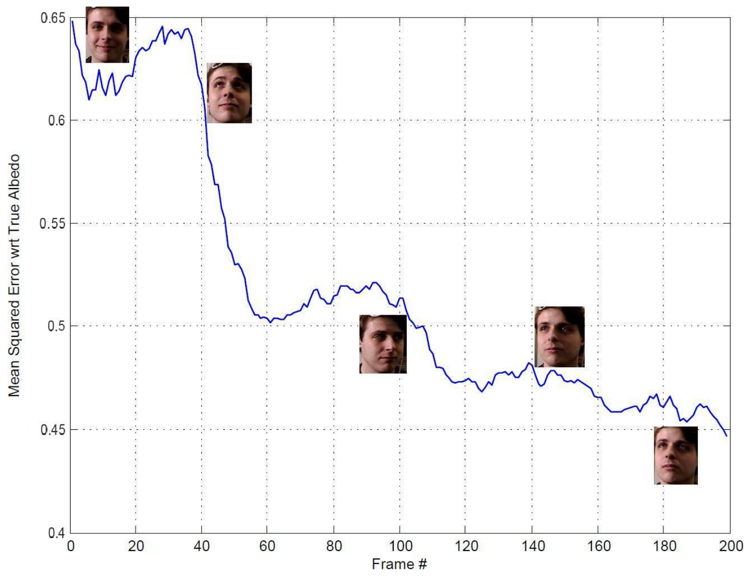


Figure 2.8: The MSE of the albedo estimate versus frames for a sequence in the BU dataset along with the appearance of the face at some frames. Rotation of the face brings more information for albedo estimation and therefore reduces the estimate error.

albedo map over the frames for a sequence from the BU dataset with varying head pose and illumination. The figure also shows the appearance of the face at places where the error decreases drastically. As it can be seen, such sudden reductions in the albedo error happens when some previously shadowed parts of the face become illuminated, which means they bring new information for albedo estimation. It should also be noted that slight increases in errors are mainly due to tracking errors and errors created as a consequence in the albedo estimation process.

Figure 2.9 shows the albedo maps obtained for some of the subjects in the BU dataset. The rows show some frames of the selected sequences (including the very first and last frames), along with the estimated albedo maps using the proposed algorithm up to each frame. The selected sequences in this figure usually start with the face in a harsh illumination condition and then the motion of the face in

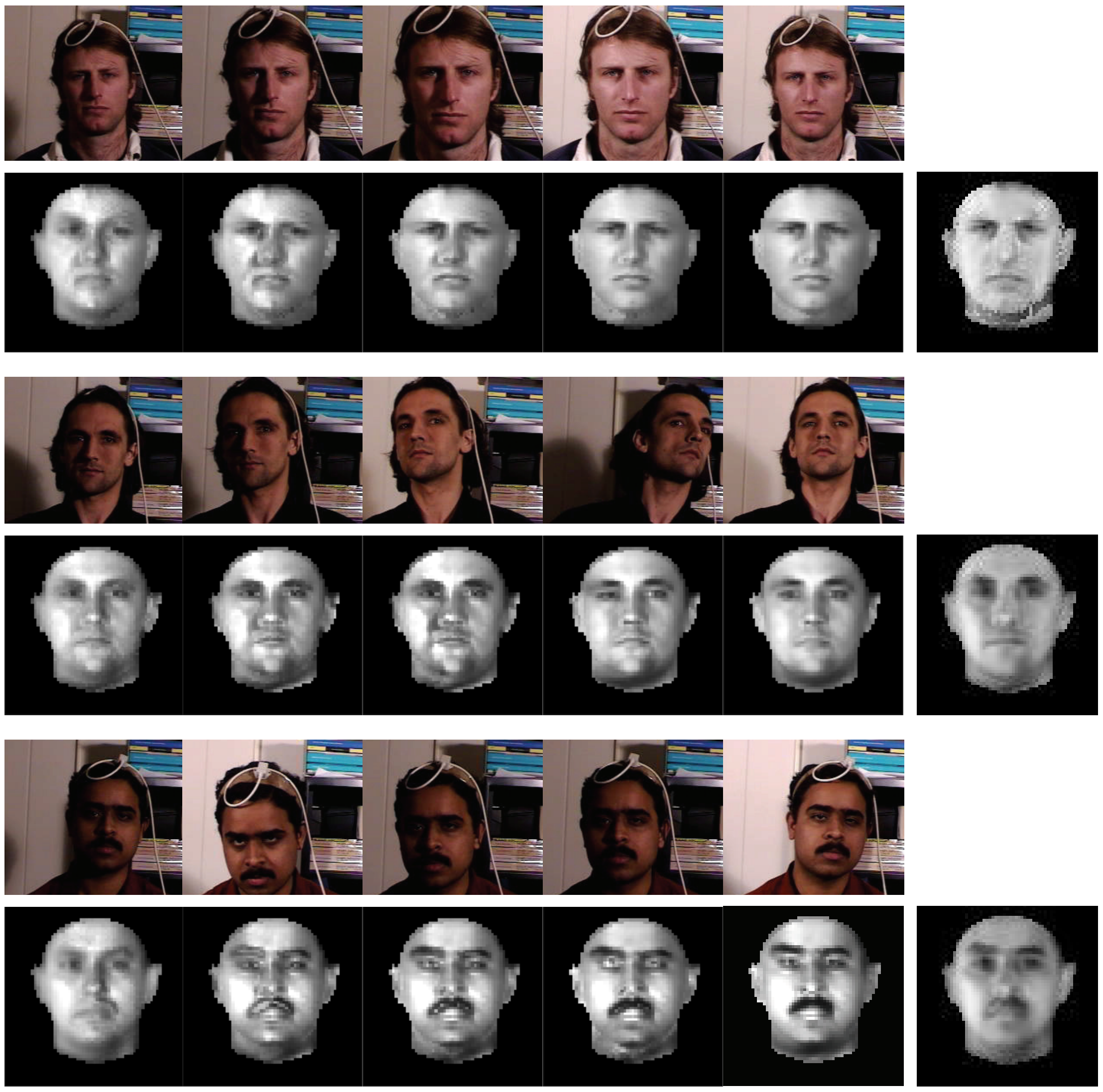


Figure 2.9: Albedo maps estimated for three sequences in the BU dataset. The rows show some of the frames in each sequence followed by the estimated albedo using the proposed algorithm up to that frame. The last column shows the averaging of the estimated albedos obtained using [1] at each frame.

subsequent frames brings more information regarding the reflectance properties of those shadow pixels and hence improves the albedo estimate obtained from the first frames. The last column result of the albedo rows shows the temporal averaging of the estimated albedo maps [1] over the frames. These results are blurry due to incorporating information from all the frames with equal weights.

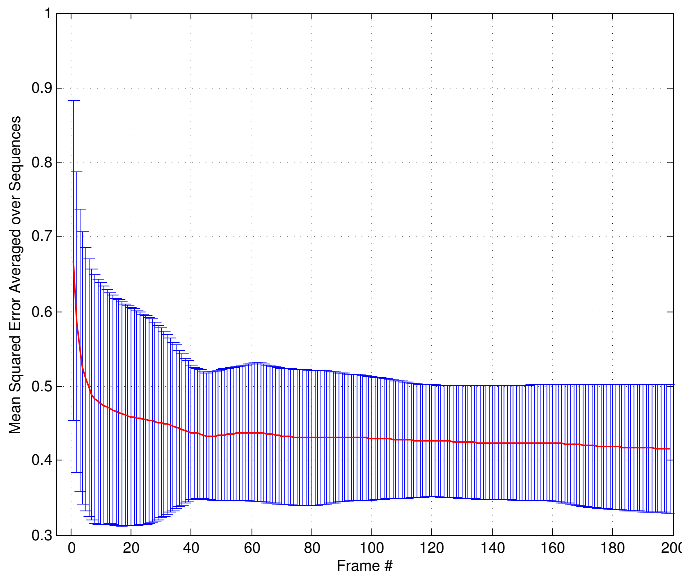


Figure 2.10: The MSE in the final estimate of albedo with respect to the ground truth averaged over 80 sequences in BU dataset along with the standard deviation of the error at each frame.

The amount of reduction in the albedo error over the frames for various sequences depends on the illumination conditions throughout the frames as well as tracking accuracy. For some sequences the shape of the face is far from the average shape and also for some cases the illumination condition is not improving enough over the frames, so the final albedo estimate still has considerable error with respect to the ground truth, although it is less than the initial error. Figure 2.10 shows the average behavior of the albedo error with respect to the ground truth over 80 BU sequences along with the standard deviation of error at each frame.

Finally, to evaluate our algorithm for some extreme situations, we applied the algorithm on some sequences, collected indoors, where the face rotates in front of a fixed illumination source. Figure 2.11 shows some frames along with their estimated albedos for the two sequences one with a good lighting condition and the other one

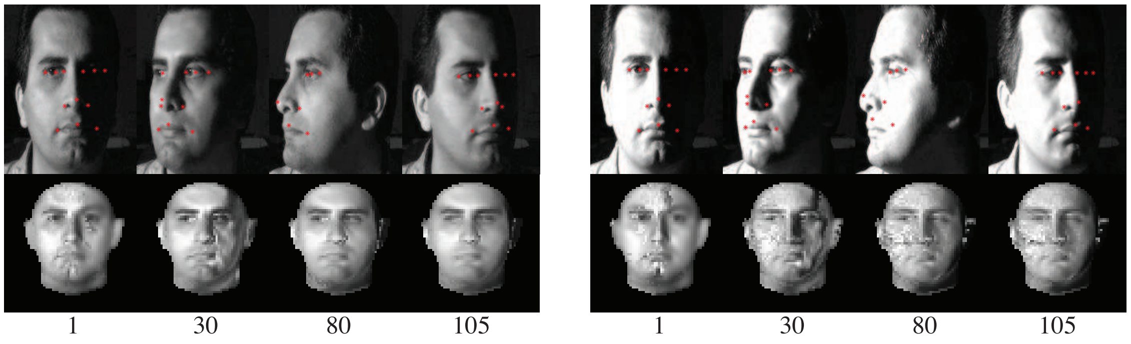
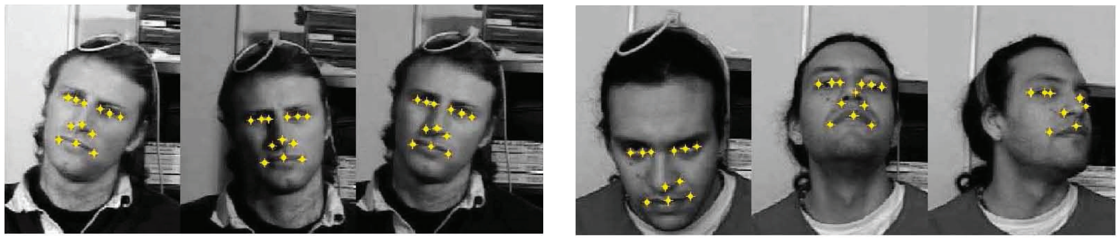


Figure 2.11: Tracking results and albedo estimates for two sequences in **left:** good illumination and **right:** harsh illumination conditions. Shown below is the frame number of each image (out of 150 frames).

in a harsh illumination condition with saturated pixels and cast shadows. As can be seen, the albedo estimated using the second sequence is noisy which is mainly due to the saturated pixels. This example illustrates a limitation of our algorithm which occurs when a part of the object is not Lambertian. Since our algorithm excludes such pixels from the updating framework, their albedo estimates will not improve over the frames. But if specularity (or any other example of violation from Lambertian assumption) occurs in a limited number of frames and the surface parts that have such errors in some frames show their Lambertian properties in some other frames, our algorithm will be able to ignore the occurrences of such errors and get a good albedo estimate out of good frames in the sequence.

2.5.2 Illumination-Insensitive Tracking

Illumination-insensitive head pose tracking is an important part of the proposed algorithm. While pose estimation is necessary for our sequential albedo estimation algorithm, updating the albedo map at each frame also helps to have ac-



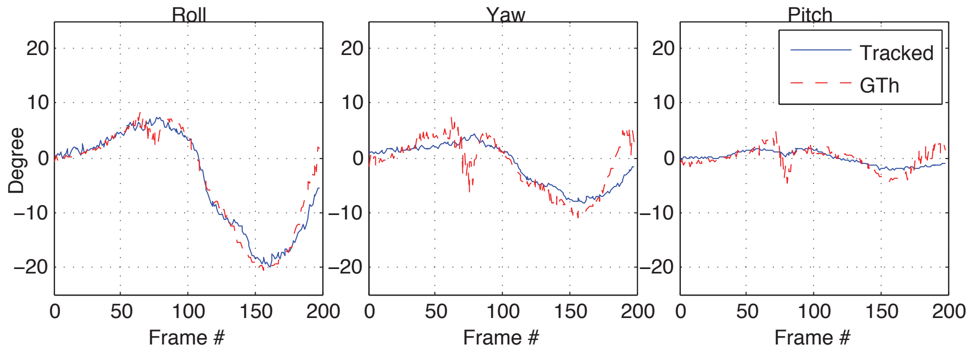


Figure 2.12: Tracking results for two sequences under illumination changes and in/out-of-plane rotation. The **first row** shows faces with tracked landmarks on them, the **second row** presents a comparison between the estimated rotation angles (roll, yaw, pitch) with the ground truth for the sequence in the top-left column.

curate pose tracking. We evaluate the performance of the tracking algorithm using sequences with both uniform and varying illumination in the BU dataset for which the ground truth pose information is available. The dataset with uniform illumination has 45 sequences for 5 subjects (9 sequences per subject) and the dataset with varying illumination has 27 sequence for 3 subjects, each sequence has 198 frames in which the face goes through several in-plane and out-of-plane rotations as well as translations.

Figure 2.12 shows some frames of two sequences with the tracked landmarks on the face. These examples show the ability of the tracker to maintain tracks in spite of illumination changes and large out-of-plane rotations. Figure 2.12 also presents a comparison between the rotation angles estimated for the left sequence, in the top

	Uniform Illumination	**Varying Illumination**
Rotation	0.822 ± 0.44	1.01 ± 0.15
Translation	0.83 ± 0.42	0.95 ± 0.095

Table 2.1: Averaged tracking error in terms of both position (translation) and orientation (rotation) for two subsets of BU dataset one with uniform illumination and other one with varying illumination.

row, and its ground truth. It can be seen that the tracker accurately estimates the pose of the face in almost all frames.

To have a quantitative evaluation of the precision of our tracking algorithm and evaluate its robustness against illumination variation, we use the metric introduced in [43] which is based on the Mahalanobis distance between the estimated and measured positions and orientations. Two normalized errors, position error and orientation error, are defined at each frame of the sequence. The precision of the tracker is then defined for each sequence as the root mean square error computed over the sequence up to the point where the track was lost. For this purpose we defined the track as lost when the position error at that frame exceeded a fixed threshold.

Table 2.1 shows the evaluation results for both subsets of the BU dataset with uniform and varying illuminations. The percentage of tracked frames for both cases is 89.2 ± 4.003 and the averaged tracking time per frame is 2.05 ± 0.34 seconds (using Matlab software and on a 4GHz processor) where most of this time is spent for retrieving the observed intensities at the vertices of a 3D face model[4]. The timing of the joint pose and albedo estimation algorithm is the same as for tracking, since

[4]We can improve the tracking rate using a parallelized particle filter [56] as well as using a more powerful processor and by programming on C++.

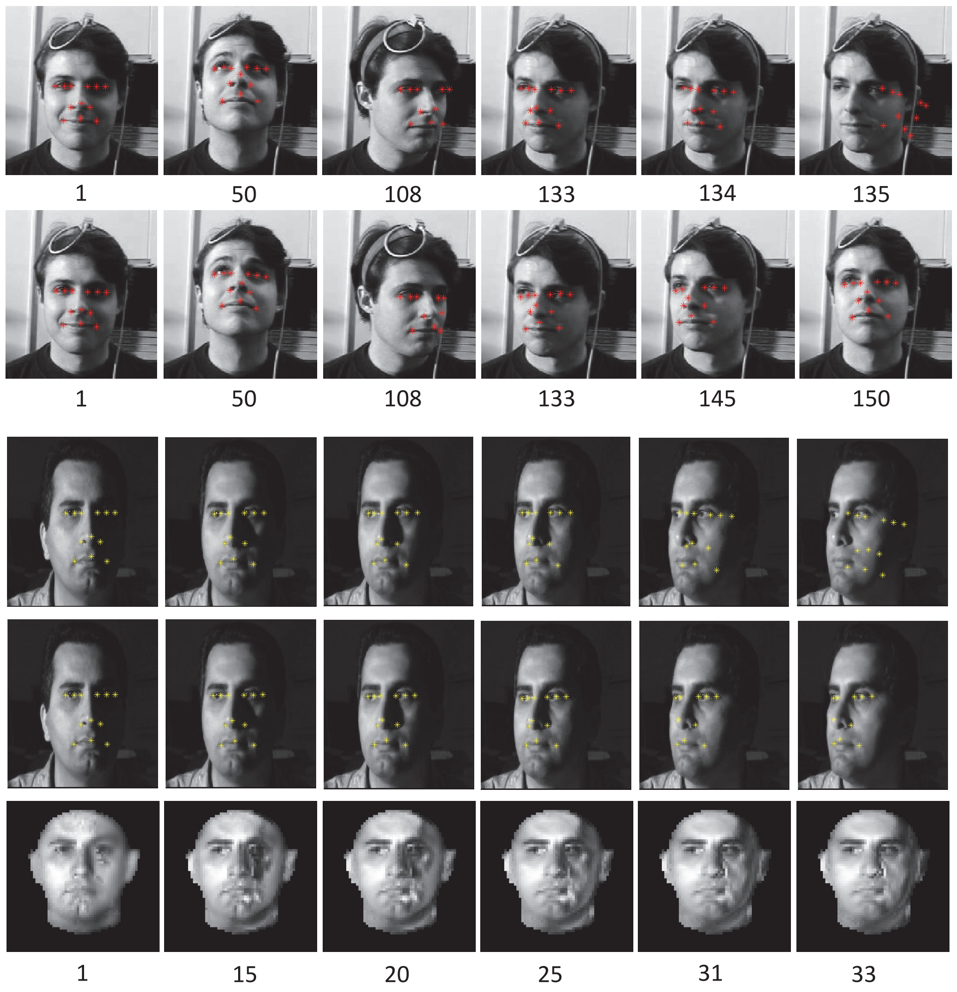


Figure 2.13: Comparing the pose tracking results with and without albedo updating step. For each sequence, the first row shows the tracking results of the particle filter without the albedo updating step and the second row has the results of the proposed algorithm. For the second sequence, we also show the estimated albedo map using the proposed algorithm at each frame. The frame numbers are shown below images.

the albedo estimation is based on just a linear operation. As Table 2.1 shows, the proposed algorithm performs well for both orientation (rotation) and position (translation) estimation. Moreover, comparable results on both datasets indicate that our algorithm is to a reasonable degree insensitive to illumination changes. However as expected, tracking on a dataset with varying illumination is more challenging.

To show the importance of the albedo update step for pose tracking, we perform the pose tracking experiment using the particle filter framework, as discussed in Section 2.4 but without the albedo update using the Kalman filter step. So the albedo is estimated using the first frame of the sequence (we assume that the face has an almost frontal pose in the first frame of the sequence and it is partially shadowed) and the estimated spherical harmonic basis images B_t are therefore fixed throughout the pose tracking step. We compare the results from this algorithm with those of our algorithm (using RBPF) on two sequences. As Fig. 2.13 shows, the first algorithm looses the track whenever the face goes through an illumination change. This is because the estimated albedo and therefore the estimated spherical harmonic basis are not accurate and changing the appearance of the face due to illumination changes causes the face to not lie on the same spherical harmonic subspace. This emphasizes the importance of updating the albedo throughout the sequence so that the available information from multiple images is used to obtain a more accurate estimate of the albedo and spherical harmonic basis as a result.

2.5.3 Video-based Face Recognition

The resulting albedo maps provide signatures of faces that can be used as inputs to many existing 2D techniques for face recognition. Here our objective is to show the improvement in face recognition obtained due to video-based albedo estimation. To this end, we use the true albedo maps of the subjects as the gallery set and the probe set includes a number of videos per subject where the videos usually starts with the face partly in the shadow. We perform the face recognition experi-

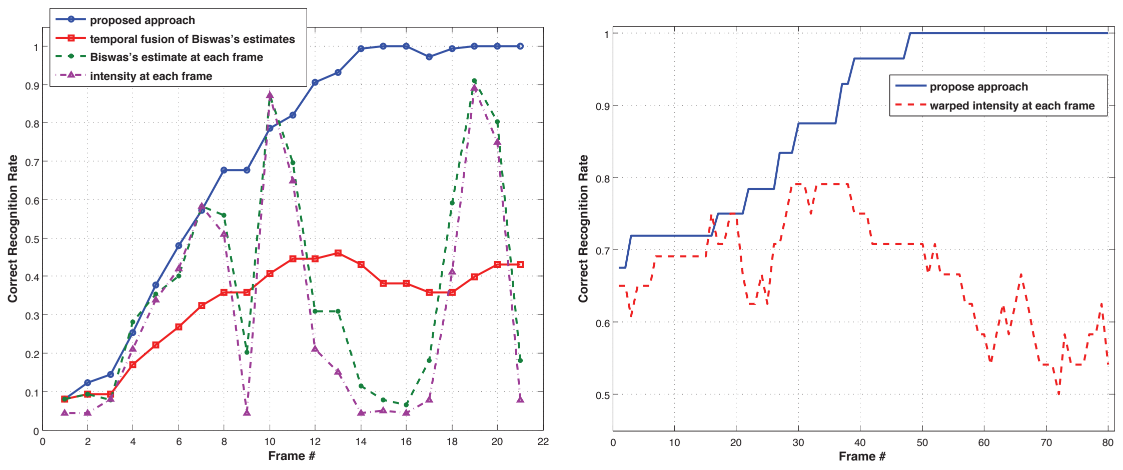


Figure 2.14: Video-based face recognition rate versus number of frames used for albedo estimation. **Left:** comparing the recognition rate on the PIE synthetic sequences using the albedo maps resulting from [1] estimated at each single frame, temporal fusion of [1] estimates up to each frame, face intensity at each frame and the proposed algorithm; **Right:** recognition rate on the BU dataset using the albedo maps produced by the proposed algorithm compared to the recognition rate using the warped intensity at each frame.

ment on the synthetic sequences from the PIE dataset as well as the BU sequences. In both cases we report the recognition rates averaged over all the sequences versus the number of frames used for albedo estimation. We expect the recognition rate to increase as more frames are used for albedo estimation, since the albedo maps improve over the frames.

Figure 2.14 illustrates the face recognition rate averaged over all the sequences versus frame number. For the PIE sequences (left column of the figure) we have 68 subjects and a sequence of 21 frames per subject in the test set. Each sequence starts with the face being partly in shadow and then the illumination rotates around the face. We apply the proposed algorithm to each sequence to obtain the estimated albedo maps up to each frame using which we perform the face recognition (blue solid curve). To show the importance of albedo estimation and proper fusion of information over frames for face recognition, we compare our results with the results from three other algorithms. First we obtain the albedo maps at each frame using Biswas' single image-based algorithm [1] and use them to perform face recognition at each frame (green dash curve). Then we temporally fuse these estimated albedo maps up to each frame (temporal averaging) and again perform recognition at each frame (red solid curve), and finally perform recognition using the face intensity at each frame (magenta dash-dotted curve).

As the plots show, recognition performance using the intensity at each frame as well as the albedo estimated from a single frame is completely dependent on the quality of the face at that frame and while it is around 90% for some frames, it decreases to below 10% for some other frames. The temporal fusion/averaging of the [1] estimates over the frames stabilizes the results but still the recognition rate is low due to the blurry albedo maps obtained through this process. But the proposed algorithm results in a considerable increase in the recognition rates over the sequences.

Similar results using the sequences in the BU dataset are presented in the right column of the figure. We have six subjects and an average of nine sequences per subject, each with 80 frames, using which we obtain the albedo maps up to each frame and perform face recognition. We compare the results from our algorithm with the case where we use the warped intensity at each frame for recognition. However since the pose of the head is changing throughout the sequence, to get the warped intensity at each frame we need to estimate the head pose. We can perform this step separately by tracking fiducial points using the Kanade-Lucas-Tomasi (KLT) tracker [57], but since the illumination is changing considerably throughout the sequences, the KLT fails to track the feature points. Therefore, we use the estimated poses from our algorithm (RBPF) and then obtain the warped intensity map at each frame for recognition. The results again show the superiority of having an accurate albedo map for face recognition compared to the intensity map at each frame.

2.5.4 Kalman Smoother

In the proposed algorithm, the estimate of ρ_t is made based on the noisy measurement set $\mathcal{Z}^t = \{Z_1, ..., Z_t\}$. But if a delay in the production of ρ_t be permitted, then more measurements become available during the delay interval and these new measurements can be used in producing the estimate of ρ_t. Thus a delay of N time units during which $Z_{t+1}, ..., Z_{t+N}$ appear allows estimation of ρ_t by

$$\mu_{\rho,t|t+N} = E[\rho_t | Z_1, ..., Z_{t+N}]$$

Such an estimate is called smooth estimate. Historically, three particular types of smoothing problems have been studied, each characterized by the particular subset of all possible smoothed estimates sought: Fixed-point smoothing, fixed-lag smoothing and fixed-interval smoothing. For our problem, the fixed-lag smoother [58] is the best since it allows "online" production of smoothed estimates.

Since more measurements are used in producing $\mu_{\rho,t|t+N}$ than in producing $\mu_{\rho,t|t}$, one expects the estimate to be more accurate, and generally, one expects smoothers to perform better than filters, although inherent in a smoother is a delay

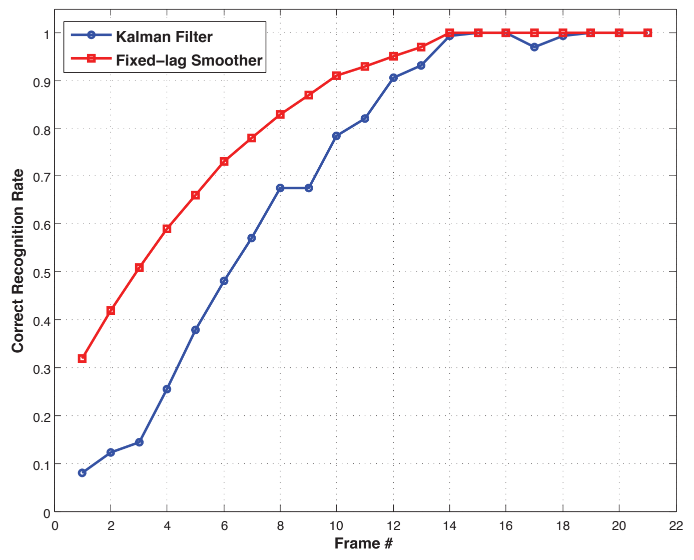


Figure 2.15: Comparing the face recognition rates obtained using our approach (Kalman filter) with those of applying the fixed-lag smoother algorithm (with lag of 4 frames) on the synthetic PIE sequences.

and, as it turns out, an increase in estimator complexity. Further, the greater the delay, the greater the increase in complexity [58]. Thus depending on the delay and complexity that the system can tolerate, some improvements in the estimates can be obtained using smoothed estimates.

To investigate the trade-off between the delay and the improvement we obtain in the estimate of the albedo map we perform the sequential albedo estimation and face recognition experiments on the synthetic PIE sequences. We consider a lag of $N = 4$ frames and applied a fixed-lag smoother to the PIE sequences at each frame to estimate $\mu_{\rho,t|t+4}$ and then use these smoothed estimates to perform the face recognition experiment as explained in the section 2.5.3. Figure 2.15 shows the improvements that the smoothed estimates result compared to the causal estimates obtained using the Kalman filter. Increasing the value of N slightly increases the improvement.

It should be noted that the head poses are fixed in the synthetic PIE sequences, since if the head pose is varied then we need the pose information for the future frames as well which is not available at the current frame. Moreover, knowing that the delay at each frame in the proposed approach is due to the pose estimation step (the albedo estimation is performed in real-time), we need our sequential algorithm and the Kalman filter to jointly estimate the pose and albedo map at each frame.

2.5.5 Comparison with a Batch Processing Algorithm

We also compared our sequential albedo estimation algorithm with a batch processing method for albedo estimation. As we mentioned in the introduction, our algorithm has connections to the photometric stereo problem. Hence, we applied a photometric stereo algorithm (as a batch processing method)[5] to the synthetic PIE sequences to estimate the albedo (as well as the normal vectors to the surface) for each subject. Then we computed the average MSE of the estimated albedo and the ground truth albedo over 68 subjects. Note that while the photometric stereo algorithm optimizes the albedo globally (along with accounting for surface estimates etc.), our algorithm incrementally updates the albedo under a fixed shape model.

Table 2.2 shows the results for both the algorithms. As the table shows, using the batch processing method gives a slightly better error rate compared to our sequential algorithm. This result was expected since a batch processing method uses all the available information (including the future frames in our case) at once. However as we emphasized in the chapter, our algorithm is applicable to a video in

[5]http://pages.cs.wisc.edu/ csverma/CS766_09/Stereo/stereo.html

Table 2.2: MSE error of the estimated albedos using a photometric stereo algorithm as well as our proposed sequential algorithm with respect to the ground truth averaged over 68 synthetic PIE sequences.

	Photometric Stereo	**Our approach**
Albedo MSE	0.166 ± 0.042	0.21 ± 0.08

which frames come at a time and so the algorithm processes the information once it becomes available.

When pose variations are presented in the sequences, both the sequential and batch processing methods need correspondences across the images. We applied a tracking algorithm to obtain these correspondences for the low resolution images we used for our experiments. These correspondences are not very accurate and it would be interesting to investigate the effects of these inaccurate correspondences on a batch processing method as a future work.

2.6 Summary and Limitations

We proposed a joint tracking and sequential albedo estimation framework using a Rao-Blackwellized particle filter. The tracking algorithm finds the best pose at each frame which minimizes the projection error of the observed appearance onto the spherical harmonic basis images. At the same time, a Kalman filter updates the albedo map estimated in the previous frame using current observations and by incorporating useful information regarding the albedo into the prior albedo estimate. Simultaneous pose and albedo estimation at each frame improves the final albedo map. Comparisons with true poses and true albedo maps were shown to validate the effectiveness of the algorithm. Moreover, the robustness of the algorithm against errors due to deviations from the Lambertian assumption has been evaluated. The albedo estimated using our approach can be used for video-based face recognition. The proposed algorithm has limitations and we briefly discuss some of them here.

Assumptions: The main assumption in this work is the Lambertian assumption for the human face. As discussed in the introduction, while this assumption is reasonable for the application in this work, having a more accurate model for the face image formation will lead to more precise results and enables improved inferences. We also make simplifying assumptions regarding independence of pixels in the face image. This assumption can be removed by adding a prior albedo model and also having full covariance matrices in the Kalman filter.

Robustness: In this work we achieved some robustness against non-Lambertian effects (e.g. cast shadow, saturation and specularity) by updating the Kalman filter observation noise matrix at each frame in an ad-hoc manner. It might be interesting to actually use a proper robust distribution instead of Gaussian as the noise model.

Speed: Sequential estimation of albedo has many advantages in the context of real-time implementations. For one, we need a small buffer to store frames before they are processed. Further, even when the processing algorithm is not real-time, we can continue processing frames using strategies for carefully dropping frames while maintaining a finite buffer. This is especially relevant in our setting. While albedo estimation can be performed in real-time, the pose estimation step which is based on the particle filter implementation in Matlab is slow which makes the whole algorithm non-real time. However it should be noted that our choice of particle filters

— in addition to providing powerful inference capabilities — also allows parallel implementation. There has been a significant body of work on parallel and pipelined implementations of particle filters [56, 59]. A basic premise of this body of work is that particle filters are extremely parallelizable and linear speedup in the number of computing nodes is very much possible. A GPU implementation with modern GPUs that have 100-1000s of computing nodes, for example, has the capability of achieving real-time performance.

The delay in estimating the albedo at each frame can be used to interpret the problem as a smoothing problem, as we discussed in the chapter. But the interpretation as a fixed-lag smoothing problem is currently applicable only for the constant pose case. Extension to variable pose case would be interesting.

Deformation: We do not account for non-rigid deformation in our models. However since at each frame we only update the albedo for those pixels whose current intensity is in a reasonable range with respect to their intensity in the previous frame (we assume small changes between two frames), our algorithm can handle non-rigid deformation up to some degree. This approach can also be generalized for other types of objects which are rigid or can only have small non-rigid deformations.

Chapter 3: A Spatially Varying Blur Descriptor for Robust Face Recognition

3.1 Introduction

Understanding the effects of blur, which normally arise due to out-of-focus lens, atmospheric turbulence, and relative motion between the camera and objects in the scene, is an important problem in image analysis applications such as face recognition. Motion blur, among them, has long been an important challenge for developing robust face recognition algorithms. With increase in the usage of hand-held cameras, the study of motion blur is receiving considerable attention. During image capture, due to the effect of averaging of light intensities at the camera sensors, the relative motion between the scene and the camera results in motion blur. Although the main objective has usually been to remedy the effect of blur, works also exist that focus on inferring valuable information about the object or camera motion by using blur as a cue [60–64]. In this chapter, we deal with the problem of recognizing human faces blurred due to incidental motion of hand-held cameras.

The blur at each image point is characterized by a blur kernel also known as the point spread function (PSF). For the case of pure in-plane camera translations,

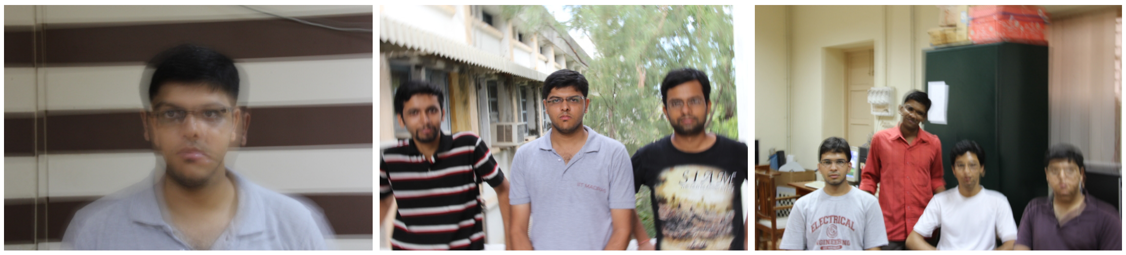


Figure 3.1: Examples of blur images under the real camera shake. The blur in the images is very non-uniform. The images are from an internally collected dataset.

the shape of the PSF reflects the camera motion. In fact, the shape of the PSF is preserved at all image points. In such cases the image formation equation modeling the blurring process can be written as, $g(x, y) = (f * k)(x, y) + \eta(x, y)$. Here (x, y) denotes the pixel location at which a 2D convolution, $*$, is performed between a clean image f and an unknown blur point-spread function k, to result in a blurred image. The ubiquitous noise present in the system, which can be due to quantization, or other sensor-induced errors, is represented by η.

The convolution model although convenient, is restrictive. Face images when captured with a hand-held camera typically exhibit non-uniform blurring under the influence of camera motion. Figure 3.1 shows some images taken with a real camera. The camera transformations can range from (a) in-plane translation and rotation to, (b) out-of-plane translation, (c) out-of-plane rotation, and even general 6D motion. As can be seen, the blur on the images and faces is very non-uniform. Such images cannot be modeled using the convolutional model for blur and we need to model these images using space-varying blur formulation.

Face recognition systems that work with clean faces have considerably lower performance when they are presented with blurred faces. Therefore, it is important to either remove the effect of blur from faces before passing them to recognition systems or use a blur-robust face recognition algorithm. Especially with the increase in the usage of hand-held cameras which usually generate images with lots of motion blur and also the emergence of new applications such as photo tagging in social media, the problem of blur-robust face recognition has become more and more important. Since the main goal of face recognition systems is to recognize the face and not to generate the clean image of the face, we focus on proposing a direct approach for blur-robust face recognition. In this work, we treat the camera motion to consist of translations and rotations along all three directions.

When a camera undergoes a general 6D motion the shape of the blur kernel will vary across different image points. If we assume a planar structure for face, the blurred image can be modeled as a weighted average of geometrically transformed instances (homographies) of the reference image. Each homography is assigned a weight that denotes the fraction of the exposure duration for that transformation. The weights of the transformations are referred to as the transformation spread function (TSF). The TSF is a generalization of the PSF notion; while the PSF denotes the motion of a point light source at a pixel during image capture, the TSF denotes the transformations undergone by the image plane due to camera motion [65]. The notion of TSF or set of homographies for describing motion blur was mooted in the recent past by different groups [65–68].

We use the TSF blur model to design a blur-robust face recognition algorithm. This model is general and mimics the realistic blurring situations very accurately. It is important to note that clean training images are available when attempting face recognition. By applying all the possible transformations that exist in a six-dimensional space (three dimensions for translations and three dimensions for rotation) on the clean image, we obtain a matrix whose column space spans all possible blur images obtained from that clean image. To compute such a matrix, we should first discretize the space of all transformations. To recognize a blurred image, we minimize the distances between the given image and the transformation matrices computed for all the gallery/clean images. For this purpose, we use a proper optimization function with l_1 constraint on the TSF weights. This optimization also gives us an estimate of the transformations applied to the clean image which resulted in that blurred image. We also propose a multiscale implementation of the algorithm to make the process efficient in terms of both the computations and the memory demands. Various experiments on both synthetic and real faces are performed to evaluate the effectiveness of the proposed algorithm.

Contributions: In this work we propose a direct approach for motion blur-robust face recognition which does not need image deblurring. We also propose

- a subspace representation for all the blurred versions of a clean image obtained under arbitrary camera shake.
- an objective function with proper sparsity constraints which after minimization gives the identity of the blurred face as well as a good estimate of the transformation that must be applied to the clean image to generate the blurred image.
- a multiscale implementation of the algorithm which makes the process efficient

in terms of both computations and memory demands.

Outline of the chapter: Related works are discussed in Section 3.2. Then we review the convolution model for uniform blur formation in Section 3.3 and discuss the shortcomings of this model. We discuss some details of the motion blur model and propose the optimization algorithm for blur kernel estimation in Section 3.4. The face recognition across motion blur is provided in Section 3.5. We also discuss our multiscale implementation in this section. Section 3.6 has several experimental results using both synthetic and real examples.

3.2 Related Work

Face recognition has received significant attention in several disciplines such as computer vision, image processing, pattern recognition and neural networks. The ubiquity of computer applications that use face recognition to help users with tagging their photos, such as Google Picasa, Adobe Photoshop Elements, Apple iPhoto, Facebook and Microsoft Windows Live Photo Gallery highlights the success of face recognition. Various recent algorithms have been proposed for face recognition [69–76], where some of them have been designed for robust face recognition against challenges such as occlusion, illumination, degraded training images, low-resolution and blurred faces.

Most of the existing face recognition systems usually have considerably lower performance when presented with degraded faces due to effects such as blur. Hence several blur-robust face recognition algorithms have been proposed [76–78]. In blur-robust face recognition the main objective is to remedy the effect of blur and have blur-insensitive recognition. Existing approaches for blur-robust face recognition can be classified as: (i) inverse methods based on deblurring, and (ii) direct methods based on invariants. While the goal of deblurring is to estimate the clean image from the observed blurred image, direct methods, which are mainly based on invariants, search for those properties of the original image that are preserved across blur and use them for robust face recognition [79].

Existing approaches for blur-insensitive face recognition usually have a spatially uniform blur assumption [77, 78, 80]. But this assumption made by most algorithms is often violated. Among them, deblurring-based methods either use the knowledge of PSF at image pixels as well as the clean image statistics to perform non-blind deconvolution or attempt to solve an under-constrained problem of estimating the PSF and the clean image without assuming any knowledge about the blur kernel using blind-deconvolution [76, 77, 80]. Zhang *et al.* [76] proposed a joint blind image restoration and recognition method based on the sparse representation prior to handle the challenging problem of face recognition from low-quality images, where the degradation model is realistic and totally unknown. The proposed algorithm achieves simultaneous restoration and recognition by iteratively solving the blind image restoration in pursuit of the sparsest representation for recognition.

There are also direct methods, mostly developed for the specific class of centrally symmetric blur PSF's which account for blur due to out-of-focus lens and atmospheric effects [81, 82] as well as methods based on moment-based invariants in both spatial and Fourier domain [83–85]. A recent direct method proposed a blur-robust descriptor for face recognition using a fusion of image-formation models under the effect of spatially uniform blur and differential geometric tools [78]. They showed that the subspace resulting from convolutions of an image with a complete set of orthonormal basis functions that could represent the blur kernel, is invariant to blur under some assumptions. They then studied the utility of this invariant for the problem of direct recognition of faces, using techniques that account for their underlying non-Euclidean geometry. In another work, Vageeswaran *et al.* [86] showed that set of all images obtained from a face image by blurring it and by changing the illumination conditions forms a bi-convex set. Based on this set-theoretic characterization, they proposed a blur and illumination-robust algorithm for face recognition whose main step involves solving convex optimization problems.

As mentioned earlier, real camera shake causes non-uniform blur which cannot be modeled using the convolutional blur formulation. The assumption of uniform blur is violated for many images/faces that are taken under arbitrary camera shake. Several deblurring algorithms have been proposed for shaken images. Some of these approaches have focused on piecewise-uniform blurs arising from multiple moving objects in the scene, spatially varying combinations of localized uniform blurs or rotations of planar objects in the scene [87–90]. But, these approaches usually make the assumption that the blur is locally uniform, and do not consider global models for continuously varying blur that results from arbitrary camera shakes.

Miskin *et al.* [91] and Fergus *et al.* [92] attempted to estimate the blur kernel using a variational inference approach. The estimated kernel is then used to deblur the observation directly using the Richardson-Lucy (RL) algorithm to give the clean image. Several algorithms have been proposed for image restoration under the motion blur effect [68,93–96]. Whyte *et al.* [68,97] proposed a new parametrized geometric model of the blurring process in terms of the rotational velocity of the camera during exposure. They modified the algorithm proposed by [91,92] to perform blind deblurring for camera shake removal using a single blurred image. They also proposed an algorithm, based on the work in [98], which uses both a blurred image and a sharp but noisy image of the same scene to remove the non-uniform blur. But they limit the freedom of camera motions to camera rotations only. Cho *et al.* [96] presented a novel blind motion deblurring method for dealing with non-uniform blurs caused by camera shake. Their method is based on a novel representation of motion blurs, which models the blur effects using a set of homographies. They fully describe the motions of camera shakes in 3D world.

3.3 Convolution Model for Space-Invariant Blur

As we mentioned in the introduction, for images with spatially-invariant blur, the image formation can be expressed as the convolution between the original sharp image and the point spread function (PSF) of the blur kernel. While this model is sufficient for describing the blur due to out-of-focus lens, atmospheric turbulence and pure in-plane camera translations (all for flat scenes), it cannot describe many other blurring effects which are mainly due to general camera shake (including out-of-plane motion and in-plane rotation).

Direct face recognition algorithms usually benefit from the resulting simple

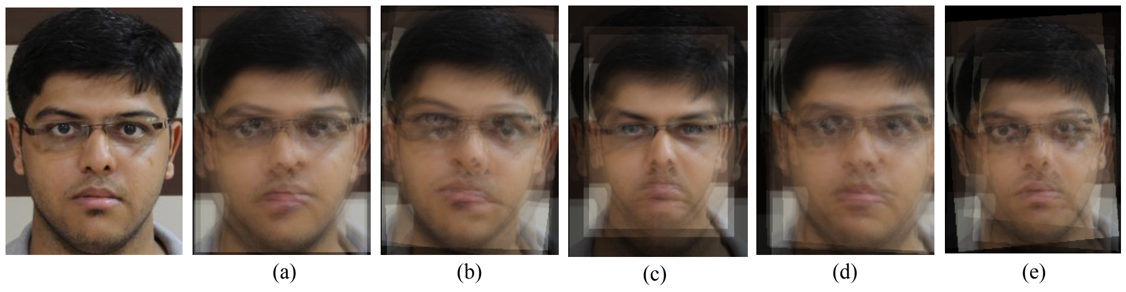


Figure 3.2: Some synthetic images obtained from the clean image (left most column) after applying random (a) in-plane translations, (b) in-plane translations and rotation, (c) out-of-plane translation, (d) out-of-plane rotation, and (e) 6D motion, i.e. 3D translations and 3D rotations. The image is from an internally collected dataset.

convolutional model to yield a blur-insensitive descriptor for recognition. In a recent work, Gopalan *et al.* [78] represented the blur kernel as a linear combination of a complete set of orthonormal basis functions, $k = \sum_{i=1}^{K} \alpha_i \phi_i$. Hence under no noise assumption, the blur image lies in the subspace generated by convolving the clean image with each of these basis functions,

$$g = f * \sum_{i=1}^{K} \alpha_i \phi_i = \sum_{i=1}^{K} \alpha_i (f * \phi_i) = D\alpha \tag{3.1}$$

where $D = [f * \phi_1, f * \phi_2 ... f * \phi_K]$ and $\alpha \in I\!R^K$. Using this representation, the blur-insensitive face recognition can be performed by minimizing the distance between the blurred test face and each of the gallery face subspaces.

$$\begin{aligned} n^* &= \arg\min_n \|g - \sum_{i=1}^{K} \alpha_i (f_n * \phi_i)\|_2^2 \qquad (3.2) \\ &= \arg\min_n \|g - D_n \alpha\|_2^2 \end{aligned}$$

An important limitation of space-invariant blur modeling for face recognition is that this model can only handle face misalignments due to in-plane translations. If the faces have out-of-plane geometric degradation (due to out-of-plane camera rotation) then the model cannot handle such misalignments between the faces and the recognition algorithm will fail[1]. Especially the scale differences between images which is because of camera motion orthogonal to the image plane causes a serious problem for the convolutional model of the blur.

In order to show the weakness of the convolutional model in handling most of the motion-blurred images, we compare the reconstruction errors of the reconstructed faces using the convolutional model and the motion blur model (will be introduced in Section 3.4) for different synthetic blur faces generated using various camera motions, as shown in Fig. 3.2, ((a) in-plane translations, (b) in-plane translations and rotation, (c) out-of-plane translation, (d) out-of-plane rotation, and (e) 6D motion, i.e. 3D translations and 3D rotations.). Figure 3.3 shows the reconstructed faces as well as the RMS errors. As the figure shows, except for the in-plane translations (case (a)) where as expected the RMS is the same for the both models, in all other cases the correct motion blur model gives much smaller RMS than the convolutional model and it is clear from the figure that the convolutional model is restrictive and cannot handle out-of-plane transformations (cases (c), (d), (e)) as well as in-plane rotation (cases (b), (e)).

[1]Unless the displacement is quite small or the camera focal length is large enough so to approximate the out-of-plane camera rotation with the in-plane translation

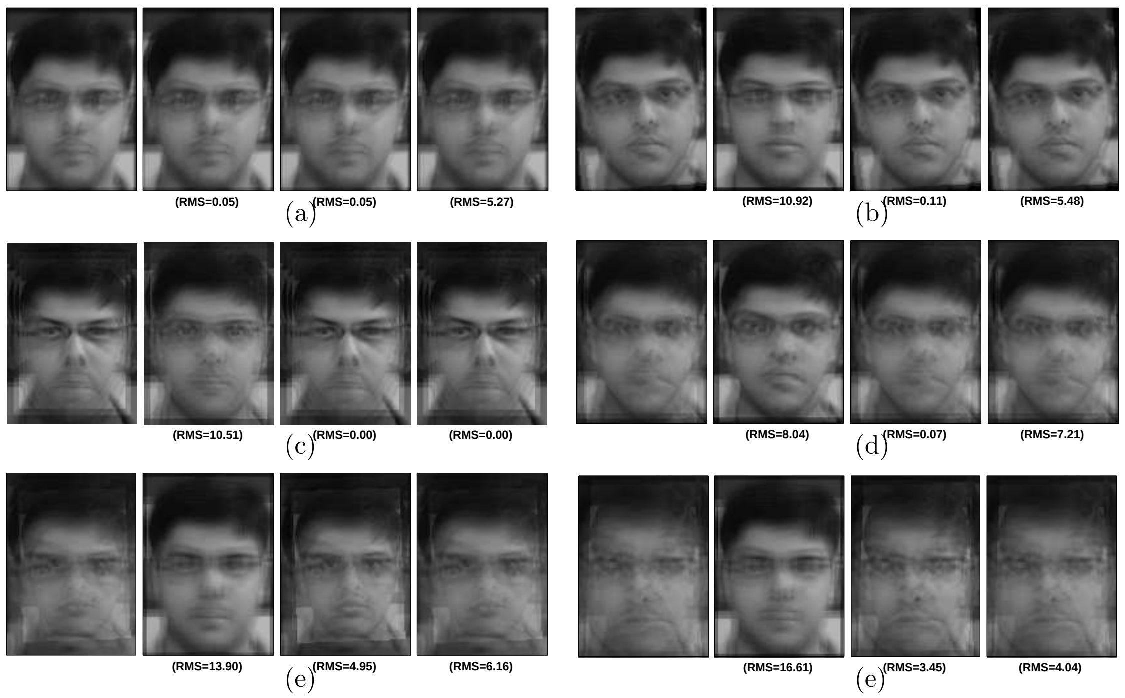


Figure 3.3: Comparing the reconstructed faces and reconstruction errors (RMS) using the convolutional model for the blur and the motion blur model with the brute force and multiscale implementations. The results are shown for 5 different camera motions, (a) in-plane translations, (b) in-plane translations and rotation, (c) out-of-plane translation, (d) out-of-plane rotation, and (e) 6D motion. We show two examples for case (e). For each motion case images from left to right are: original blur, convolutional model, motion blur model and multiscale implementation.

3.4 Motion Blur Model For Faces

When the camera motion is not restricted to in-plane translations, the apparent motion of scene points in the image will vary at different locations resulting in space-variant blurring. In such a scenario, the convolution model with a single blur kernel does not hold. However, as we mentioned earlier [2], because we model face by a flat surface, the blurred face can be accurately modeled as a weighted average of the warped instances of the original image [65, 67, 68, 93]. In this section, we

[2]The focus of this chapter is on the images of faces, but the proposed algorithm is equally applicable to other planar scenarios.

present the motion blur model and illustrate how this model can explain geometric degradations of faces. Next, we propose an optimization algorithm to recover the motion blur kernels.

Note that in the face recognition problem, the original training images are available. Let the image of a training face captured by a still camera be denoted by $f : \mathbb{R}^2 \rightarrow \mathbb{R}$. Let $\mathbf{X} = [X \ Y \ Z]^T$ denote the spatial coordinates of a point on the face with the camera center as the origin. The projection of $\mathbf{X}$ in the image plane (x, y) is given by $x = \frac{\nu X}{Z}$ and $y = \frac{\nu Y}{Z}$ where ν denotes the *focal length* of the camera. Using homogeneous coordinates, the image point $\mathbf{x} = [x \ y \ 1]^T$ can be written as $K_\nu \mathbf{X}$. In this discussion, K_ν is assumed to be of the form

$$K_\nu = \begin{bmatrix} \nu & 0 & 0 \\ 0 & \nu & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

Due to camera motion during image capture, at each instant of time τ, the coordinates of the 3D point $\mathbf{X}$ changes to $\mathbf{X}_\tau = R_\tau \mathbf{X} + T_\tau$ with respect to the camera, where $T_\tau = [T_{X_\tau} \ T_{Y_\tau} \ T_{Z_\tau}]^T$ is the translation vector and R_τ is the rotation matrix. This rotation matrix is parameterized in terms of θ_X, θ_Y and θ_Z (the angles of rotation about the three axes) using the matrix exponential

$$R_\tau = e^{\Theta_\tau} \text{ where } \Theta_\tau = \begin{bmatrix} 0 & -\theta_{Z_\tau} & \theta_{Y_\tau} \\ \theta_{Z_\tau} & 0 & -\theta_{X_\tau} \\ -\theta_{Y_\tau} & \theta_{X_\tau} & 0 \end{bmatrix} \tag{3.4}$$

We consider that all of the face points in the training image are at a distance d_o from the camera. Since the depth is constant, the point $\mathbf{x}_\tau$, at which $\mathbf{X}_\tau$ gets projected in the camera, can be obtained through a homography H_τ as $\mathbf{x}_\tau = H_\tau \mathbf{x}$ where

$$H_\tau = K_\nu \left(R_\tau + \frac{1}{d_o} T_\tau \left[0\ 0\ 1 \right] \right) K_\nu^{-1} \tag{3.5}$$

Let g_τ denote the blurred image captured at time instant τ. For the sake of simplicity, we use the same notation $(\mathbf{x})$ for the homogeneous coordinates as well as for the coordinates in the image plane. Then we can write $g_\tau(\mathbf{x}) = f(H_\tau^{-1}\mathbf{x})$ where H_τ^{-1} denotes the inverse of H_τ (since $g_\tau(H_\tau \mathbf{x}) = f(\mathbf{x})$). Now the blurred face g can be interpreted as the average of the light intensities observed in the image plane during exposure. Therefore, the blurred face intensity at an image point $\mathbf{x}$ is given by

$$g(\mathbf{x}) = \frac{1}{t_e} \int\limits_0^{t_e} f\left(H_\tau^{-1}\mathbf{x}\right) d\tau \tag{3.6}$$

where t_e is the total exposure duration.

The blurred face can be more appropriately modeled in terms of the clean face by averaging it over the set of possible transformations resulting from the camera motion. We define the *transformation spread function* (TSF) $h_T : \mathbf{T} \rightarrow \mathbb{R}_+$ as a mapping from the set of all possible transformations $\mathbf{T}$ to non-negative real numbers. For each transformation $\lambda \in \mathbf{T}$, the value of the TSF, $h_T(\lambda)$, denotes the fraction of the total exposure duration for which the camera stayed in the position that caused the transformation H_λ^{-1} on the image coordinates. The blurred image can be written as an average of the warped images weighted by the TSF, h_T. i.e.,

$$g(\mathbf{x}) = \int\limits_{\lambda \in \mathbf{T}} h_T(\lambda) f\left(H_\lambda^{-1}(\mathbf{x})\right) d\lambda \tag{3.7}$$

In this model, the order of the set of transformations undergone by the reference image during exposure is lost due to the averaging effect; however, this is not relevant information. According to this model, the apparent motion of pixels may be significantly non-uniform across the image.

When the camera motion is not restricted, the paths traced by scene points in the image plane can vary across the image resulting in space-variant blur; however, the blurring operation can be described by a single TSF using (3.7). So the TSF depicts camera motion during exposure. For instance, if the camera undergoes only in-plane rotations, the TSF will have non-zero weights only for the rotational transformations. Analogous to a blur kernel, the TSF satisfies the relation

$$\int\limits_{\lambda \in \mathbf{T}} h_T(\lambda) = 1 \tag{3.8}$$

It should be noted that convolution model for the blur is a special case of motion blur which is valid when the camera motion is confined to 2D translations. In such a case the PSF and TSF will be identical.

3.4.1 Recovery of Homographies

Having a clean image and its blurred version under unknown camera shake, it is of interest to estimate the homographies applied to the clean image and the corresponding TSF values. Having an estimate of these values can help for blur-insensitive face recognition through minimizing the distance between the reconstructed blur image, using the estimate of the homographies and TSF, with the original blur image.

In practice, we represent the transformation space $\mathbf{T}$ with coordinate axes that correspond to the camera motion and quantize each axis to get a discrete set of transformations. The TSF, h_T, defined on the discrete transformation space $\mathbf{T}$ can be considered as a vector in $\mathbb{R}^{N_T}$ where N_T is the total number of transformations present in $\mathbf{T}$. N_T is controlled by the number of translation steps along each axis as well as the number of rotation steps about each axis, so $N_T = N_{tx} \times N_{ty} \times N_{tz} \times N_{rx} \times N_{ry} \times N_{rz}$. Discretizing (3.7), each observed pixel g_i is modeled as:

$$g_i = \sum_{k=1}^{N_T} \left(\sum_j C_{ijk} f_j \right) h_k \tag{3.9}$$

where i, j and k index into the blurred image, the clean image and the blur kernel, respectively. Here the sum $\sum_j C_{ijk} f_j$ interpolates the point $f(\mathbf{H}_k x_i)$ in the clean image. Based on this equation, the blurred image can be linearly expressed in terms of the clean image as

$$g = \mathbf{B}h \tag{3.10}$$

where $\mathbf{B}_{ik} = \sum_j C_{ijk} f_j$. If $f \in I\!R^N$, with $N = R \times C$ pixels for an image with R

rows and C columns, then $\mathbf{B} \in \mathbb{R}^{N \times N_T}$ is a matrix whose columns each contain a projectively transformed copy of the clean image [68]. Therefore, if we apply all possible transformations in the space T (after discretization as discussed before) to the clean image f and build the matrix $\mathbf{B}$ using all the transformed copy of f as the columns, then the column space of $\mathbf{B}$ spans the space of all motion blurred images obtained from the clean image f.

As we mentioned earlier, space-invariant blur which results from the convolutional blur model is a special case of general motion blur model. Hence (3.10) is a general version of (3.1) where each element h_k corresponds to a camera orientation (translation followed by rotation), and consequently to a homography $\mathbf{H}_k$. However, since a sparse number of camera motions usually happen while capturing an image, h will be very sparse and moreover, since components of h represent the fraction of time that each homography has been applied to the image, those components must be nonnegative and sum up to one.

3.5 Face Recognition Across Motion Blur

Let us consider an M class problem with $\{f_m\}_{m=1}^{M}$ denoting the gallery faces which are clean faces, one face per subject. Let g denote the blurred probe image which belongs to one of the M classes. The problem we are looking at is, given f_m's and g, find the identity $m^* \in \{1, 2, ..., M\}$ of g. Based on the discussion above, the first step is to generate the matrix $\mathbf{B}_m$ for each gallery face. Then since g belongs to one of the M classes, it falls into the span of one of these matrices. Therefore, the identity of the probe image can be found by minimizing the projection error of g onto $\{\mathbf{B}_m\}$'s considering the proper constraints as

$$\begin{aligned} m^* &= \arg\min_m \|g - \mathbf{B}_m h_m\|_2^2 + \lambda \|h_m\|_1 \qquad (3.11) \\ s.t. \quad & \|h_m\|_2 = 1 \quad \& \quad h_m \in \mathbb{R}^{p+} \end{aligned}$$

This equation is similar to (3.2), but it has proper constraints to guarantee the properties we expect for h.

In equation (3.11) we assume observation noise, ν, to be Gaussian, zero mean and unit variance distribution so that $g = \mathbf{B}h + \nu$. Minimizing the l_1 norm of h guaranties the sparsity of h while the constraints force the elements of h to be non-negative and sum up to one. This optimization problem can be solved using a proper variant of the Lasso algorithm [99] which considers additional constraints. It should be noted that the transformation space T should be defined and discretized properly so that the algorithm be able to find correct homographies. We refer to this implementation of our algorithm as *brute force*, since it searches over all possible combinations of homographies and tries to minimize the reconstruction error considering all the constraints.

Figure 3.3 shows some examples of reconstructed faces after estimating the TSF and homographies. As the figure shows, the algorithm is able to perfectly reconstruct the synthetic blurred faces. In all the reconstructed examples using the blur motion model shown in this figure, the RMS value is smaller than one, except for the case (e), 6D motion. As the figure shows, the RMS is larger for this case and the reason is that the algorithm needs to search through a very large set of transformations and this makes the optimization to be slow and non-efficient which leads to possibly a local minimum as a result. Latter in Section 3.5.1 we will discuss how using multiscale implementation we can make the process more efficient.

3.5.1 Multiscale Implementation

Appropriate discretization of the space of transformations will affect our ability to accurately estimate the blur kernel and homographies. Since we are fundamentally limited by the resolution of the images, having a very fine discretization leads to redundant computations. So in practice, we need a grid spacing which corresponds to a minimum displacement of one pixel in the image. It should be noted that since the kernel is defined over 6 dimensions, $t_x, t_y, t_z, r_x, r_y, r_z$, doubling the sampling resolution for them increases the number of kernel elements, N_T, by a factor of 2^6. We set the size of the blur kernel along each dimension in terms of the size of the blur we need to model, typically a few pixels for translation along each dimension, e.g. $[-6 : 1 : 6]$, and a few degrees along each dimension of rotation, e.g. $[-5^o : 0.5^o : 5^o]$.

Considering the discretization defined above, the number of transformations in the space $\mathbf{T}$ becomes in the order of 20×10^6. Optimization over a vector with this number of elements is very time consuming and inefficient, especially since only few of those elements have nonzero values. Moreover the resulting matrix $\mathbf{B}$ should have this number of columns which is too many to fit in the computer's memory.

Therefore, we need to apply a multiscale framework to solve this problem. We follow the multiscaling approach presented in [68]; however, here we are dealing with multiscaling in 6D instead of 3D. The idea is to start from a coarse representation of image and kernel, and successively refine the estimated kernel at higher resolutions.

We first build Gaussian pyramids of n_l levels for both the clean and blurred images. Then starting from the coarsest scale $s = 0$, we estimate the homographies by optimizing (3.11) over a smaller space of transformations. In fact downsampling a blurred image by scale of s reduces the amount of pixel displacements due to the camera translation along X and Y axes by the same scale, and if the focal length of the camera is large enough, it has the same effect on pixel displacements due to camera rotation about X and Y. Hence, downsampling the images will also reduce the space of allowed transformations[3].

So at each scale s, we find the optimal blur kernel $\hat{h}_s$ for that scale. We then up-sample $\hat{h}_s$ to the next scale $(s + 1)$ using bilinear interpolation, find the nonzero elements of this up-sampled and interpolated kernel. Also, using a proper threshold we remove very small nonzero values (resulted from interpolation). This process gives us several 6D nonzero regions inside the transformation space. When finding the optimal kernel $\hat{h}_{s+1}$, we only search for the valid homographies which lie within these nonzero regions. This corresponds to discarding many columns of $\mathbf{B}$, reducing both the computation and memory demands of the algorithm. We repeat this process at each scale, until we find the optimal kernel at the finest scale.

Figure 3.3 compares the RMS value we obtain using this multiscale imple-

[3]translation and rotation along the Z axis remain unchanged after downsampling the image

mentation with the RMS from the brute force search. As the figure shows, using multiscale implementation we obtain a much faster algorithm which may yield less accurate homography estimation and face reconstruction. Also from the figure we can claim that the larger the search intervals for the transformation is (larger basis matrix) the more gain we obtain by applying our multiscale implementation (e.g. case (e)).

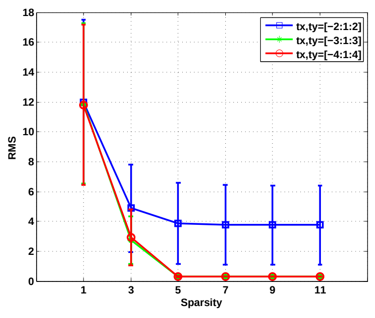
Using the multiscale implementation, we obtain a considerable speed up in the process. For example, while performing the homography estimation and face reconstruction using the brute force algorithm on 200 synthetic blurred images in case (b) (each image is of size 64×64) takes an average of 52.3 ± 11.3 cputime (on a 4Gb linux machine, and using the MATLAB software), the multiscale implementation performs the task in 0.71 ± 0.15 cputime. This means a speed up of around 70 times. For the case (e) where the multiscale implementation has its most benefits, the speed up we gain is around 80 times. This speed up, although obtained at the cost of lower accuracy, makes the multiscale implementation perfect for face recognition applications, where the speed is the main concern and the reconstruction accuracy is of less importance.

3.6 Experimental Results

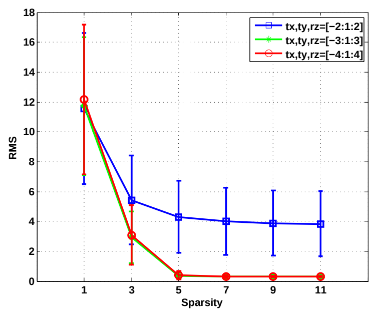
In this section, we show the effectiveness of the proposed algorithm for blur-insensitive face recognition using some synthetic and real blurred faces. We also study the robustness of the algorithm to noise due to facial variations other than blur such as lighting, expression, alignment and occlusion, which are common in an unconstrained face recognition setting.

Experiments on FERET dataset First we generate some synthetically blurred faces using the FERET dataset [100]. This dataset includes 'fa'-gallery and 'fb'-probe subsets which have faces of 1001 subjects, with one image per subject. Faces of the same person across 'fa' and 'fb' have small variations in expression and alignment. The original image size of 256×256 is first resized to 64×64. Then we generate five different synthetically blurred sets of 'fb' by applying random camera transformations. These fives sets are synthesized using following camera motions: (a) in-plane translations, (b) in-plane translations and rotations, (c) out-of-plane translations, (d) out-of-plane rotations and (e) 6D motions.

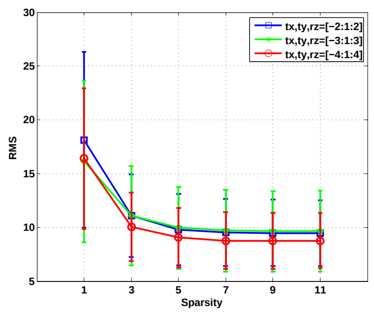
To show the effects of different parameters in the performance of the algorithm, we plot the RMS values versus the values of sparsity of h for various transformation intervals. We randomly select 200 images among 1001 available synthetically blurred faces from FERET dataset. To generate the synthetic blurred faces, we fix the sparsity to 5, and the transformation intervals are as, $tx = [-3 : 1 : 3], ty = [-3 : 1 : 3], tz = [0.8 : 0.1 : 1.2], rx = [-2 : 1 : 2], ry = [-2 : 1 : 2], rz = [-3 : 1 : 3]$. As can be seen in Fig. 3.4, increasing the sparsity for the reconstruction decreases the RMS up to a certain point and it remains unchanged afterward. Moreover, increasing the search intervals for various transformations also initially improves the RMS and then the RMS remains unchanged for transformation intervals which are larger than the original ones. These plots show that if we choose large enough values for the transformation intervals and for the sparsity value, we will obtain



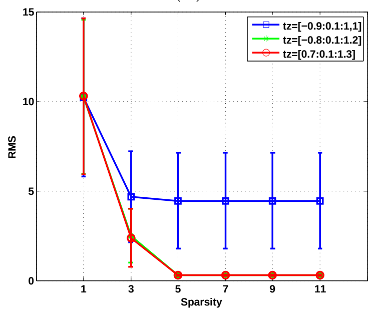
(a)



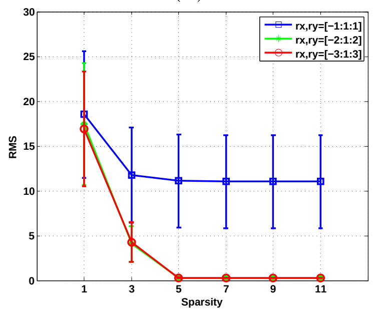
(b)



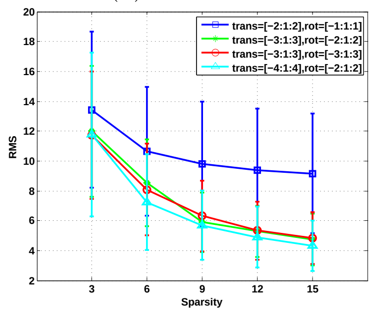
(b)-multiscale



(c)



(d)



(e)-multiscale

Figure 3.4: RMS versus sparsity of the blurred coefficients for synthetic blurred faces from FERET dataset. For the cases (a) to (d) we used brute force search to estimate the homographies and reconstruct the face. We applied our multiscale implementation to the case (b) and (e).

good reconstructed faces and RMS values. But this comes at the cost of having a slower algorithm with higher memory demands.

We also investigate the effect of various parameters in the performance of the multiscale implementation. The last column in Fig. 3.4 has the plots for multiscale implementation for cases (b) and (e). As can be seen, the RMS values is higher for multiscale implementations, but same as brute force case the increase in the intervals and sparsity improves the results. Even in case (e) we see that the RMS error keeps improving as we increase the sparsity and intervals; however, again here we have a trade off between the speed and accuracy.

Face Recognition: We perform face recognition while (i) we consider 'fb' as

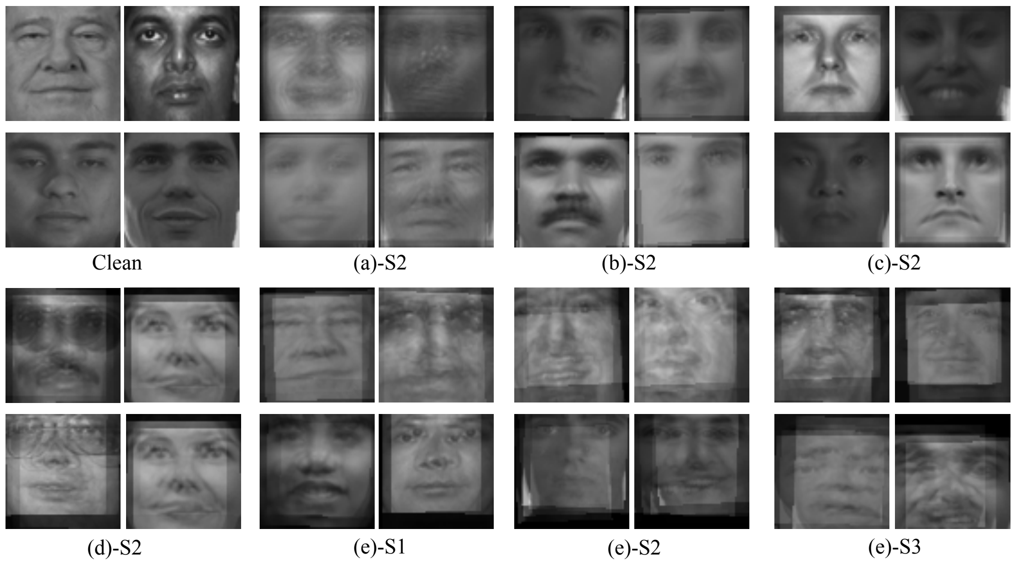


Figure 3.5: Some examples of the blurred faces in different set-ups from the FERET dataset.

the gallery set, which means we are ignoring the expression changes and misalignments, and (ii) we use 'fa' as the gallery set, so we have noise due to expression changes and misalignments between faces.

For generating synthetic blurred faces, we use three settings for the transformation intervals. Table 3.1 shows the range of transformations we use at each setting. The sparsity value in all the cases is set to 10. For recognizing faces, we use a multiscale implementation and the number of scales in multiscale implementation is set to 3. In all cases, the camera focal length is set to 200. The transformation

Table 3.1: Transformation settings to generate synthetic blurred faces using FERET dataset.

Set-ups	Transformation intervals			
	tx, ty	tz	rx, ry	rz
S1	[-2:1:2]	[0.8:0.05:1.2]	[-1:0.5:1]	[-2:0.5:2]
S2	[-3:1:3]	[0.8:0.05:1.2]	[-2:0.5:2]	[-3:0.5:3]
S3	[-5:1:5]	[0.8:0.05:1.2]	[-3:0.5:3]	[-3:0.5:3]

Table 3.2: Face recognition rates (%) on synthetic sequences from FERET dataset. 'fb' set is the gallery set.

Sparsity	Set-ups						
	(a)-S2	(b)-S2	(c)-S2	(d)-S2	(e)-S1	(e)-S2	(e)-S3
5	100	97	100	100	100	93	75
10	100	98	100	100	100	95	82
15	100	100	100	100	100	100	92

intervals are fixed to S2 and we change the sparsity parameter to be 5, 10 and 15. Figure 3.5 illustrates some examples of the blurred faces in different set-ups.

Table 3.2 shows the recognition rates for different set-ups of the experiments on the FERET dataset, when 'fb' set is our gallery faces (no expression changes or misalignment). To evaluate the robustness of the face recognition algorithm against other sources of noise such as expression changes and misalignment, we perform the experiments on the synthetic blurred faces using 'fa' set as the gallery images. Table 3.3 shows the results. It should be noted that if we use the convolutional blur model to perform face recognition, we obtain 47% recognition rate on the set (e)-S2, when the sparsity is set to 20.

Experiments on real motion-blurred dataset: We perform face recognition experiments on an internally collected dataset. The images have been captured while the camera has gone through various 6D motion and so the captured images have non-uniform blur. Each image contains one or more faces (Fig. 3.1) so we

Table 3.3: Face recognition rates (%) on synthetic sequences from FERET dataset. 'fa' set is the gallery set.

Sparsity	Set-ups				
	(a)	(b)	(c)	(d)	(e)
10	72	70	74	72.5	68
15	76	75	76	74	72

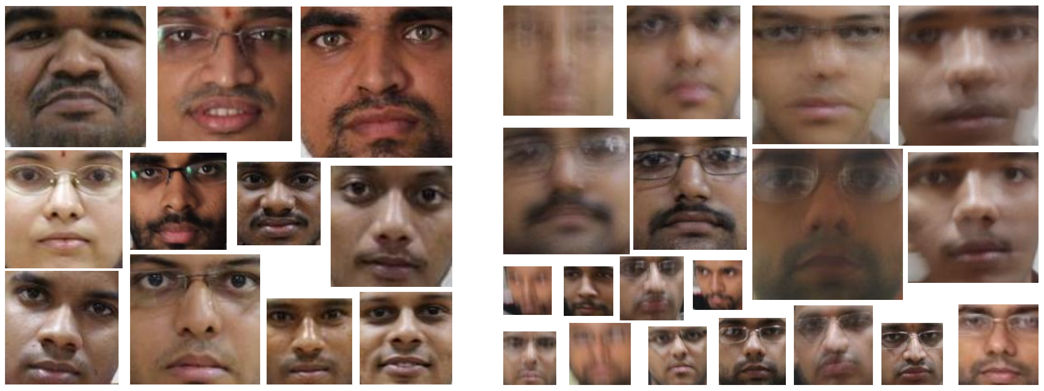


Figure 3.6: Some examples of the real blurred images from gallery and query set.

crop the faces from images. Since the proposed algorithm can handle out-of-plane translation (scaling), we do not need to resize the images and the gallery and probe images can all be of different sizes. We append the images with zero so to have two equal size images for comparison and recognition. The query images have been usually taken indoor, under the same lighting condition. Expression and head pose variations are other sources of noise available in this dataset. Some examples of the blurred images from both indoor and outdoor sets are shown in Fig. 3.6.

We perform face recognition on these images. The dataset includes 6 gallery faces and 250 test images. We use multiscale implementation with 3 scales and set the transformation intervals as $tx = [-3 : 1 : 3], ty = [-3 : 1 : 3], tz = [1 : 0.1 : 3], rx = [-2 : 1 : 2], ry = [-2 : 1 : 2], rz = [-3 : 1 : 3]$. The focal length is chosen to be 200 and the sparsity value is set to 15. As a result we obtain 80% recognition rate on this dataset.

3.7 Summary

We proposed a blur-robust face recognition algorithm for space-variant blur images. Using the transformation spread function, which is a generalization of PSF, we modeled the blurred face as a weighted average of geometrically transformed instances of the clean face. This enabled us to construct a matrix, for each clean face, whose column space spans the space of all the motion blurred images obtained from the clean face. This matrix was then used in a proper objective function along with l_1 minimization to obtain the TSF coefficients and the identity of the blurred test face. We also proposed a multiscale implementation of our algorithm to increase the efficiency and speed of it. The experimental results on both synthetic and real faces prove the effectiveness of the proposed algorithm on handling non-uniform blur on faces. Since there is no publicly available dataset for this problem, we could not perform comparisons. Our future work is to collect proper dataset with outdoor images and other variations such as illumination changes and to perform comparison with other available approaches for blur-robust face recognition.

Chapter 4: Towards View-Invariant Expression Analysis Using Analytic Shape Manifolds

4.1 Introduction

The goal of facial expression analysis is to create systems that can automatically analyze and recognize facial feature changes and facial motion from visual information. This has been an active research topic for several years and has attracted the interest of many computer vision researchers and behavioral scientists, with applications in behavioral science, security, animation, and human-computer interaction [101].

Facial expressions occur along with the head motions and pose variations, especially when there are spontaneous human-to-human interactions. Therefore, it is necessary for facial expression analysis algorithms to be able to jointly analyze the head pose and facial expressions, or in other words be invariant to pose changes. But this is a challenging task especially due to large variations in the appearance of facial expressions in different views and also the nonlinear coupling of these different sources of variations in the observed images.

While most of the proposed methods for facial expression analysis can only handle frontal-view faces, [102–104], there has been recent progress in designing pose-invariant facial expression recognition algorithms. Previous work treating pose invariance in facial expression analysis can be generally divided into two groups of approaches as those based on a 3D face model and those based on a 2D face model. It should be noted that since facial geometry conveys important information about a human's emotional state, one of the common approaches for analyzing facial expressions is by using face shape models. Therefore, our focus in this work is on geometry-based approaches.

There are several approaches that use a 3D face model and jointly estimate the rigid and nonrigid facial deformations [105–109]. In these approaches, the estimated rigid motion of the face is a byproduct of the system and can be used for tracking facial landmarks, while the non-rigid motion is further processed for expression analysis. The main disadvantages of these 3D shape model-based approaches is that they are computationally expensive, require time-consuming initialization process, and the 3D model fitting techniques may not converge. Moreover in a HCI application, 2D images and 2D shapes are far more easily available than 3D shapes. Thus, the focus of our work is on using 2D facial geometries.

Since 2D face images are projections of 3D faces, the rigid head motions and non-rigid facial expressions are non-linearly coupled in the captured 2D images. This fact has made the pose-invariant facial expression analysis based on 2D shape models hard to solve [110]. Most of the existing approaches that use a 2D shape model and facial landmarks, decouple the rigid and nonrigid motions via normalization by aligning all the available configurations to a reference frame [111–113]. But these normalization-based approaches depend on the choice of the reference frame which is usually made arbitrarily [114]. There is also a very recent normalization-based algorithm proposed by Rudovic *et al.* [110] in which using some trained regression functions, the 2D landmark locations in non-frontal poses are mapped to the corresponding locations in the frontal pose. This method shows promising results for pose-invariant expression recognition, however, requires a pose estimation phase before performing pose normalization and errors in pose estimation may contribute to recognition errors.

The main drawback of all these normalization-based approaches is that they ignore the intrinsic geometry of shape-space and instead they consider the aligned shapes as points in the Euclidean space. In this chapter, we emphasize the importance of understanding shapes as equivalence classes across view-changes instead of as a vector derived from features such as active shape models. This would enable expression models to generalize across views. Since the 2D face images are the projections of 3D faces, the projective shape-space, which carries information about the configuration of the facial landmarks that are invariant to the camera view point, is of most importance in expression analysis.

Equivalence classes are difficult to work with from a statistical perspective and we need a canonical representative from them that can be used for statistical analysis. For the similarity shape-space (Kendall's shape-space) [115], concepts such as pre-shape and Procrustes analysis are well-studied. The affine shape-space for m landmark points in $\mathbb{R}^k$ is identified with the set of all k-dimensional subspaces of $\mathbb{R}^m$, [114,116], which is a Grassmann manifold. This manifold also has well-studied mathematical structure that can be used for statistical analysis [3, 117–119]. But similar advances in projective shape-space have been slow due to overemphasis on the importance of similarity shapes in image analysis [120]. Thus, suitable metrics are hard to define for comparing projective shapes.

On the other hand, projective transformations can in many cases be approximated by affine transformations. Therefore, here we perform expression analysis using the affine shape-space since its structure is well-understood. But the eventual goal is to achieve invariance to large view-changes via projective shape-spaces and this work is a small step in that direction so that the advantages of using shape spaces for pose-invariant expression analysis can be realized.

Contributions: Our main contribution is to show the advantages of using a proper shape-space for pose-invariant facial expression analysis. Modeling the facial landmark configurations as equivalence classes on the affine shape-space, as an approximation to the projective shape-space, not only decouples the rigid and nonrigid facial motions, but also offers a well-defined underlying structure for the data. Most of the available algorithms for expression analysis can easily be extended to this shape-space.

Outline of the chapter: In Section 4.2, we discuss the landmark-based representation of facial geometry. We then discuss the affine shape-space and show that the facial landmark configurations can be identified as points on the Grassmann manifold. Some mathematical discussions on the Grassmann manifold are also provided. Then in Section 4.3, we describe the extension of some of the facial expression analysis algorithms to this shape-space and present experimental results

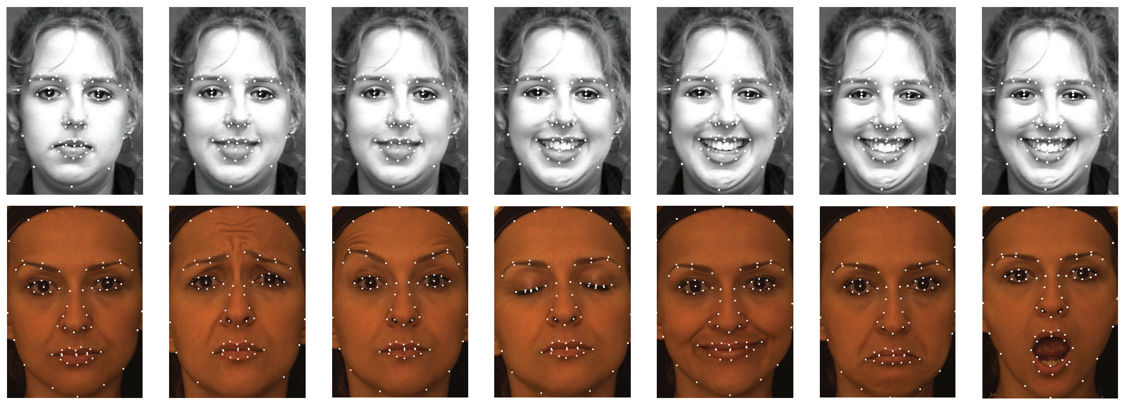


Figure 4.1: A sequence from the Cohn-Kanade database (first row), and a subject in the Bosphorus database performing various AUs (second row). The landmark locations are shown on the faces.

on facial expression recognition. Finally Section 4.4 illustrates the effectiveness of our representation for expressions synthesis and expression mapping among subjects.

4.2 Facial Expressions on the Manifold

Non-rigid facial deformations can be encoded using facial action coding system (FACS) [10], where each action unit (AU) determines the shape of its corresponding facial components. Figure 4.1 (second row) shows the face of a subject with different AUs. As can be seen, while AU-1 implies a raised inner brow, AU-27 corresponds to a wide open mouth. As the figure shows, the geometry of facial components is a good cue for representing and recognizing most of the AUs/expressions. In our work we use the landmarks on the face to represent the facial geometry at each frame of an expression sequence.

4.2.1 Facial Geometry

Landmark-based face shape representation is one of the most widely used approaches for geometric modeling of faces. Here we model the facial geometry using a $m \times 2$ matrix $L = [(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)]^T$ in $\mathbb{R}^2$. Figure 4.1 shows the locations of 2D landmarks on faces in different databases. To model the non-rigid deformations corresponding to each expression, the first step is to decouple the rigid and non-rigid deformations of landmarks. Since the shape observed in an image of a face is a perspective projection of 3D locations of the landmarks, the projective shape-space is an appropriate choice to realize invariance with respect to the camera angle. Modeling the facial geometry as equivalence classes in the projective shape-space introduces a new way of statistical analysis of 2D faces which is independent of the face poses. But advances in statistical analysis of projective shapes are still preliminary. On the other hand, projective shapes in constrained situations can be approximated with affine shapes. Therefore, we limit our discussions to affine shape-spaces.

All the affine transformations of a shape can be derived from the base shape simply by multiplying the centered shape matrix, $\tilde{L}_{\text{base}}$, by a 2×2 full-rank matrix on the right (translations are removed by centering). Multiplication by a full-rank matrix on the right preserves the column space of the matrix, thus the 2D subspace of $\mathbb{R}^m$ spanned by the columns of the centered shape, i.e. $\text{span}(\tilde{L}_{\text{base}})$, is invariant to affine transformations of the shape. Subspaces such as these can be identified as points on a Grassmann manifold [117].

Given a sequence of a face performing an expression, we would like to model the facial deformations that generate such a sequence. Based on the above discussions, a sequence of faces is represented as a sequence of points on a Grassmann manifold. So, we can model the facial deformations via geodesics on the manifold, where a geodesic is a path of shortest length on the manifold between two given points. The geodesic emanating from a point on the manifold can be characterized by a velocity vector on the tangent plane at that point. Therefore, we parametrize the facial deformations corresponding to each expression/AU as a velocity (or sequence of velocities) with which a point on the manifold (neutral face) should move in order to reach the final point (apex) in unit-time. In the following, we briefly describe how to compute these parameters on the Grassmann manifold. The readers are referred to [3, 119] for a more in-depth discussion of the mathematical details.

4.2.2 Geometry of the Grassmann Manifold

As we mentioned in the previous chapter, a Grassmann manifold, $\mathcal{G}_{m,k}$, is the space of k-dimensional subspaces in $\mathbb{R}^m$. By fixing m, k throughout this chapter we avoid adding suffixes to index the set $\mathcal{G}$. Each element of $\mathcal{G}$ can be identified by a unique $m \times m$ projection matrix, P, onto the k-dimensional subspace of $\mathbb{R}^m$. It should be noted that there are two approaches for representing points on the Grassmann manifold, either as tall-thin $m \times k$ matrices, or as square idempotent projection matrices. The former while more efficient, requires sophisticated quotient-space interpretations. The projection matrix representation, on the other hand, has

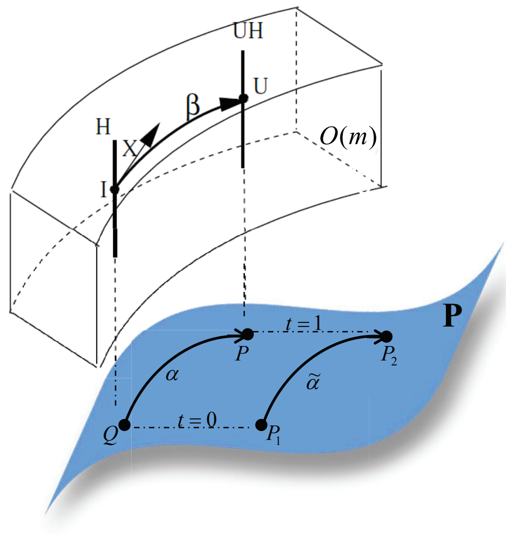


Figure 4.2: Process of computing a geodesic on the Grassmann manifold by lifting it to the particular geodesic in $O(m)$, [3].

relatively simpler analytical and geometric properties, but it is computationally intensive. However, since we are using sparse landmarks, m is typically in the order of $50 - 100$, thus the extra computational burden is not very significant. Therefore, we work with the projection matrix representation for the Grassmann points. Let $\mathbb{P}$ be the set of all $m \times m$ symmetric, idempotent matrices of rank k. Then, $\mathbb{P}$ is the set of all projection matrices and hence is diffeomorphic to $\mathcal{G}$. The identity element of $\mathbb{P}$ is defined as $Q = diag(I_k, 0_{m-k,m-k})$, where $0_{a,b}$ is an $a \times b$ matrix of zeros and I_k is the $k \times k$ identity matrix.

A Grassmann manifold $\mathcal{G}$ (or $\mathbb{P}$) is a quotient space of the orthogonal Lie group, $O(m)$. Therefore, the geodesic on this manifold can be made explicit by lifting it to a particular geodesic in $O(m)$ [3]. This process is illustrated by Fig. 4.2. Then the tangent, X, to the lifted geodesic curve in $O(m)$ defines the velocity associated with a curve on $\mathbb{P}$. The tangent space of $O(m)$ at identity is $o(m)$, the

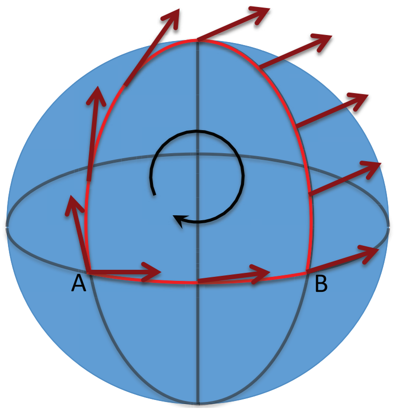


Figure 4.3: Parallel transport of a vector around a closed loop on the manifold. The direction and orientation of the vector changes to match the local structure of the destination point.

space of $m \times m$ skew-symmetric matrices, X. In this space, the Riemannian metric is defined as the inner product $\langle X_1, X_2 \rangle = \text{trace}(X_1 X_2^T)$. This property is inherited by $I\!\!P$ as well.

The geodesics in $I\!\!P$ passing through the point Q (at time t=0) are of the type $\alpha : (-\epsilon, \epsilon) \mapsto I\!\!P$, $\alpha(t) = \exp(tX) Q \exp(-tX)$, where X is a skew-symmetric matrix belonging to the set M, where

$$M = \left\{ \begin{bmatrix} 0 & A \\ -A^T & 0 \end{bmatrix} : A \in I\!\!R^{k,n-k} \right\} \subset o(m) \tag{4.1}$$

Therefore, the geodesic between Q and any P is completely specified by an $X \in M$ such that $\exp(X) Q \exp(-X) = P$. We can then construct a geodesic between any two $P_1, P_2 \in I\!\!P$ by rotating them to Q and some $P \in I\!\!P$.

One important concept is the parallel transport which is a smooth operation between tangent spaces that allows us to transfer tangent vectors between points while locally preserving direction and orientation [119]. In Euclidean space, the parallel transport is simply performed by moving the base of the arrow. However, moving a tangent vector by this technique on a manifold will not generally be a tangent vector. Figure 4.3 illustrates the parallel transport on the manifold. As the figure shows the result of parallel transport depends on the path along which we move the tangent vector. Readers are referred to [119] for more details on parallel transport on the Grassmann manifold. Some Grassmann related algorithms which will be of use in expression analysis are as follows.

```
1. Let $U \in \Phi^{-1}(P_1)$ so that $P_1 = UQU^T$
2. Define $P = U^T P_2 U$
3. Find $X$ that takes $Q$ to $P$.(using Algorithm 2)
4. Find the geodesic between $Q$ and $P$:    $\alpha(t) = \exp(tX)Q\exp(-tX)$
5. Shift $\alpha(t)$ to $P_1$ and $P_2$ as:
$\tilde{\alpha}(t) = (U\exp(tX)U^T)P_1(U\exp(-tX)U^T)$
*** Here the sub-matrix $A$, where $X = \text{cdiag}(A, -A^T)$, is the velocity that
takes $P_1$ to $P_2$ in unit time.
```

Algorithm 1: Find the geodesic between two points P_1, $P_2 \in I\!\!P$

```
1. Define $B = Q - P$
2. Find the Eigen decomposition of $B = W\Sigma W^T$.
*** The eigenvalues of $B$ are either 0's or occur in pairs of the form $(\lambda_j, -\lambda_j)$
where $0 < \lambda_j \leq 1$. Then $Qw_j$ and $Qw_{j'}$ are chosen to be positive multiples of
each other, where $w_j$, $w_{j'}$ are the columns of $W$ corresponding to the
eigenvalues $\lambda_j$ and $-\lambda_j$. This is achieved by multiplying $w_j$ by an appropriate
unit number.
3. Set $X = W\Omega W^T \in M$, where $\Omega$ is derived from $\Sigma$ by replacing all the
$2 \times 2$ blocks, $\widehat{\text{diag}}(\lambda_j, \lambda_{j'})$, by $cdiag(-\sin^{-1}(\lambda_j), \sin^{-1}(\lambda_j))$ and keep the rest
unchanged.
```

Algorithm 2: Given $P \in I\!\!P$, find an $X \in M$ such that $\alpha(1) = \exp(X)Q\exp(-X) = P$

```
1. Let $U \in \Phi^{-1}(P_1)$ so that $P_1 = UQU^T$
2. Form a skew-symmetric matrix $X = cdiag(A, -A^T)$
3. Define $V(t) = U\exp(tX)U^T$
4. Then $P_2 = V(1)P_1V(1)^T$
```

Algorithm 3: $P_2 \in I\!\!P$ that is reached in unit time by following a geodesic starting at P_1 with velocity A

```
1. Let $U_1 \in \Phi^{-1}(P_1)$ so that $P_1 = U_1QU_1^T$
1. Let $U_3 \in \Phi^{-1}(P_3)$ so that $P_3 = U_3QU_3^T$
2. Define $V = U_1\exp(X)U_1^T$
3. Compute $P_4 = VP_3V^T$
4. Having $P_3$ and $P_4$, the parallel transport of $X$ to $P_3$ is calculated using
   Algorithm 1
```

Algorithm 4: Given P_1 and the direction vector $X \in M$, find the parallel transport of X to point P_3

4.3 Facial Expression Analysis

The goal is to perform expression analysis using the equivalence classes of face shapes on the Grassmann manifold. We show how we can extend most of the available expression analysis algorithms to the Grassmann manifold in order to perform expression analysis in an affine-invariant manner. In particular, we discuss the linear modeling of facial landmark deformations using ASM as well as modeling the nonlinear deformations using a nonlinear dimensionality reduction technique. We also discuss the AUs and basic expression recognition by learning statistical models on the Grassmann manifold.

We use three databases to evaluate the strength of this approach. The first one is the Bosphorus database [121] that is composed of a selected subset of AUs as well as the six basic emotions for 105 subjects. For each subject, the neutral face and the face in the apex of various AUs and emotions are presented. In addition, 22 landmarks per face are provided by the database. However, we manually marked 75 landmarks for some of the subjects (Fig. 4.1). The second database is the Cohn-Kanade's DFAT-504 database (CK) [122], which consists of more than 100 subjects, each performing a set of emotions. The sequences begin from neutral or nearly neutral faces and end at the apex state of the expression. The sequences were annotated by certified FACS coders. We also manually labelled the sequences into the six basic expressions. Moreover, 59 landmarks per faces are available [123]. The third database is a sequence of a talking face [1] with 5000 frames which shows the face of a subject while talking. The facial landmarks are also provided for this database. Figure 4.1 shows some examples of the images in these databases.

4.3.1 Facial Deformations Modeling

By representing the facial landmark configurations as equivalence classes on the affine shape-space, we transform the data to the space of nonrigid deformations. In other words, starting from a projection matrix on the Grassmann manifold corresponding to a neutral face, it is ensured that moving in each direction on the manifold is corresponding to a nonrigid deformation of the initial configuration. These nonrigid facial deformations can be statistically modeled, depending on the linear or nonlinear assumptions for the deformations, by calculating the principal directions of variations using principal geodesic analysis (PGA) or by estimating the expression manifold through nonlinear dimensionality reduction.

[1]http://personalpages.manchester.ac.uk/staff/timothy.f.cootes/data

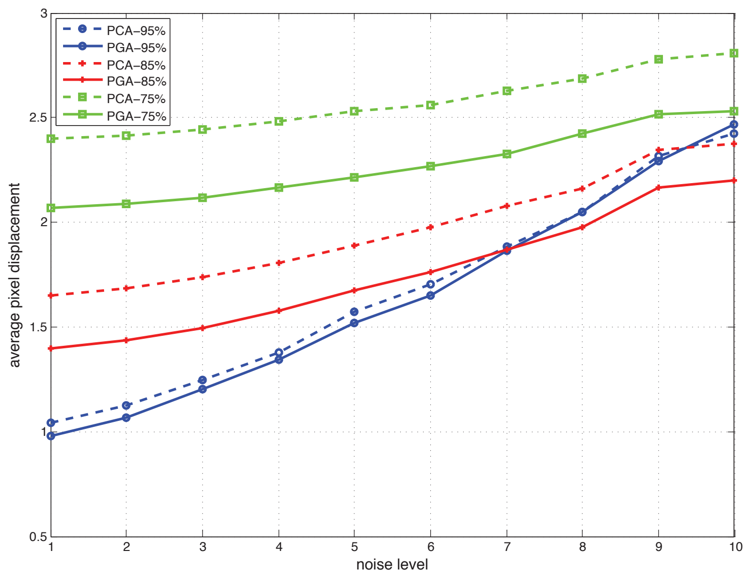


Figure 4.4: PCA versus PGA bases for facial landmarks representation and reconstruction.

4.3.1.1 Linear Deformations Modeling

As we know, assuming a linear structure for the facial deformations, active shape models (ASM) learn the principal directions of facial geometric variations using PCA. The same idea can be extended to the Grassmann manifold using PGA, [124], which is a generalization of PCA to the manifold setting. Representing the facial landmarks with different expressions as points on the Grassmann manifold, the principal directions of variations can be learned. For this purpose, the first step is to find the intrinsic mean of the points on the manifold using the Karcher mean algorithm for the Grassmann manifold [3, 125]. Then the principal geodesic directions are calculated using the warped data to the tangent plane at the mean point.

Figure 4.4 compares the PCA and PGA approaches for recovering the face shape, in the talking face database, under the noise. Using manually marked 2D landmarks on the faces in this database as ground truth, we perturb the position of landmarks independently with different levels of Gaussian noise. Then we use these two approaches to reconstruct the shapes from the noisy observations. We apply the similarity alignment method on the faces to bring them to the same co-ordinate frame before performing PCA. As the figure shows the PGA bases have higher resilience against noise. This can be due to the fact that modeling the variations on the shape-space ensures that the variability being computed is from shape changes only and not due to rigid transformations.

4.3.1.2 Nonlinear Manifold Learning

Since the linear assumption for facial deformations is not always valid, there are several approaches that consider the geometric variations of facial components on a low-dimensional nonlinear manifold and learn such a manifold using nonlinear dimensionality reduction algorithms [107, 112, 113]. The nonlinear dimensionality reduction algorithms preserve the local structure of the data while reducing dimensionality. Therefore, considering the true structure of the data is important for these algorithms.

As emphasized earlier, the face shapes after quotienting the affine group lie on the Grassmann manifold. The local structure of the data on this space can be employed to learn the low-dimensional nonlinear expression manifold. A simple dimensionality reduction algorithm is locally linear embedding (LLE), [126], which

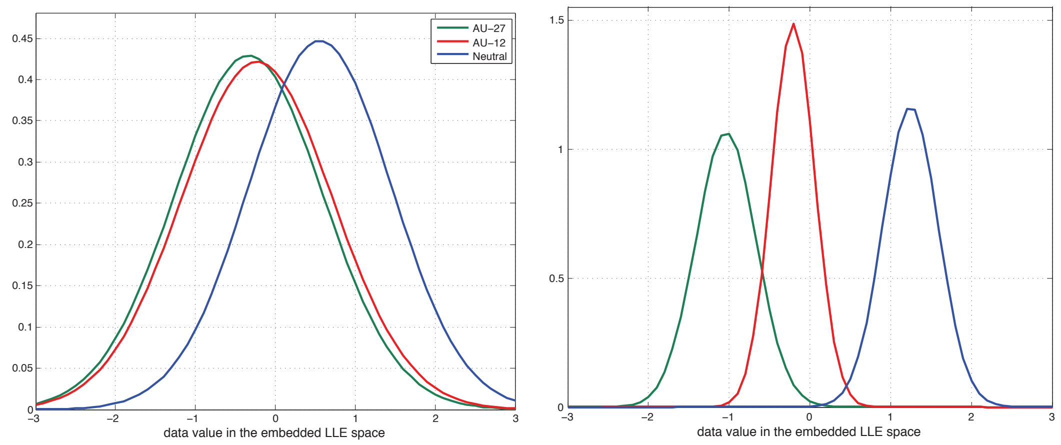


Figure 4.5: Dimensionality reduction using LLE from data in the Euclidean space (left) and on the Grassmann manifold (right). Plots show the distributions of the 1D projected data separated by the classes. Better separation is seen among classes on the right.

is a neighborhood-preserving embedding of high-dimensional inputs. This algorithm can be extended to the data on the Grassmann manifold to nonlinearly project the subspaces to the lower dimensional space.

We compare the results of low-dimensional manifold learning using the data on the Grassmann manifold versus normalized data in the Euclidean space. For this purpose the dimensionality of the training data, composed of the facial landmarks for 80 subjects in the Bosphorus database having three different expressions, Neutral, AU-12, and AU-27, is reduced to one-dimension using the LLE method. Figure 4.5 illustrates the distribution of the data in the LLE embedded space for both Euclidean and manifold cases. As the figure clearly shows, for the data on the Grassmann manifold the one-dimensional representations are well-separated for different classes compared to that for the data on the Euclidean space.

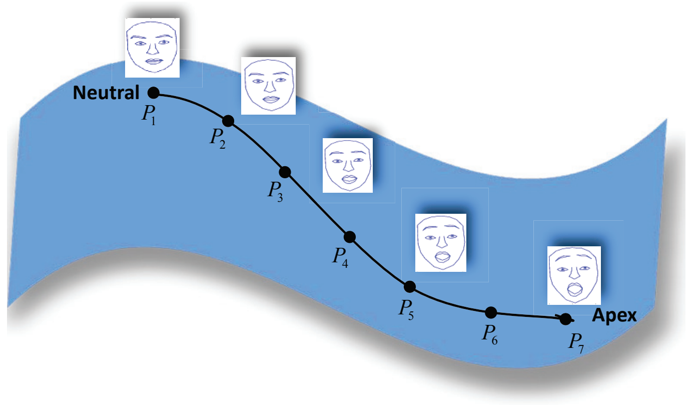


Figure 4.6: A sequence of facial expression is a curve on the Grassmann manifold.

4.3.2 AUs Template Learning

Facial expressions are combinations of several AUs occurring simultaneously or sequentially in different parts of the face. Recognizing these AUs is a proper way for expression recognition. To this end, we learn a template for each AU on the Grassmann manifold. A sequence of faces performing an AU is represented as a sequence of facial landmarks $\mathcal{L} = \{L_i\}_1^n$ where L_1 corresponds to the neutral face and L_n to the apex point (in our cases). This sequence is equivalent to a sequence of subspaces/projection matrices $\mathcal{P} = \{P_i\}_1^n$ which can be considered as samples of a curve on the Grassmann manifold (Fig. 4.6).

We represent each sequence as a piecewise-geodesic curve and model each piece using its velocity vector. Therefore, we have a sequence of N velocity vectors $\mathcal{A} = \{A_i\}$, where $A_i = \text{velocity}(P_i \to P_{i+1})$ and N is the number of segments. For the case of CK database, we choose N to be equal to the number of sequence frames minus one. But for the Bosphorus database, since only the initial and final images of each sequence are available, each AU is represented using the velocity vector corresponding to the geodesic between P_1 and P_n ($n = 2$). Although this geodesic is an approximation to the real sequence, our experiments show that it is a good approximation since AUs are simple and represented by short sequences and the geodesic between the initial and end point is almost the same as the curve connecting the intermediate frames on the manifold.

In order to learn a template model for each AU on the manifold, an important step is to parallel transport each curve to a common tangent plane so that we can learn the statistics of the set of vectors corresponding to an AU. As mentioned earlier, parallel transport on the manifold is different from that of Euclidean space. Figure 4.7 compares the results of parallel transport on the Euclidean space and the manifold. The tangent vector from the sequence in the first row is learned and then we parallel transport it to a new face in the second and third rows. Before applying the tangent vector, we affine transform the new face so that we can better show the effect of parallel transport. As the figure shows, parallel transport on the manifold generates the face with the correct deformations while the corresponding result on the Euclidean space is distorted. This again emphasizes the importance of exploiting the shape-space geometry for our problem.

The natural choice for the common tangent plane is the one at the average of neutral faces on the manifold. For this purpose, we calculate the Karcher mean for the neutral faces. Then the velocity vectors corresponding to each AU, for the Bosphorus database, is parallel transported to the mean point and the Gaussian distribution is learned as the model for each AU. It should be noted that in the

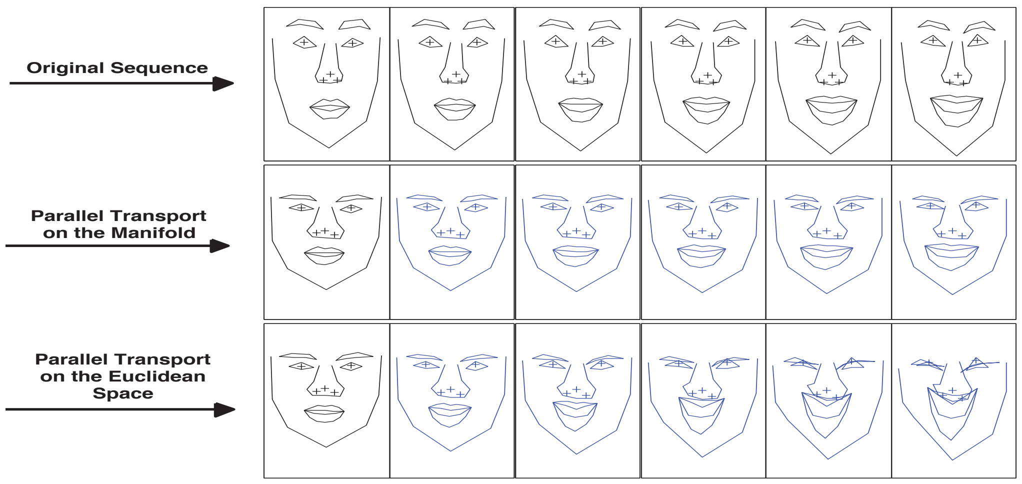

Original Sequence
Parallel Transport on the Manifold
Parallel Transport on the Euclidean Space

Figure 4.7: Comparison between the results of parallel transport on the manifold versus that of Euclidean space. The first sequence (its tangent vector from the leftmost to the rightmost shape) is parallel transported to the face on the second and third row, and the new sequence is synthesized on the manifold (second row) and Euclidean space (third row).

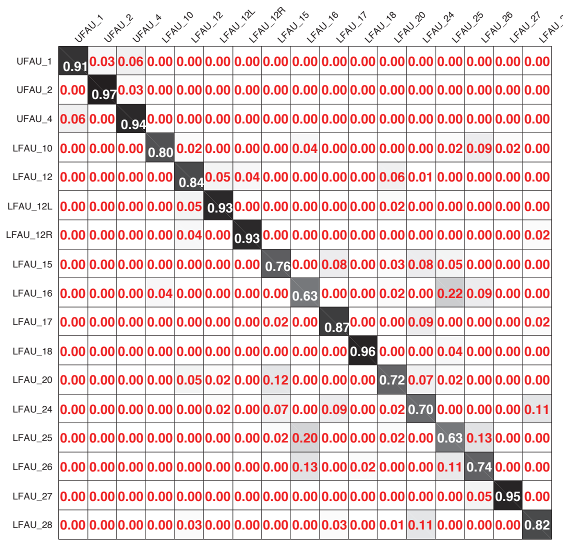
Bosphorus database each AU is represented using a tangent vector. Therefore, the final model is the mean and standard deviation over the tangent vectors at the mean neutral face.

Another method for learning an AU template model is using dynamic time warping (DTW) on various sequences corresponding to each AU [127]. Especially for the CK database, since each AU is represented using a sequence of projection matrices and since expressions occur at different rates, it is necessary to time warp the sequences in order to learn a rate-invariant model for them. Adapting the DTW algorithm to the sequences that reside on a Riemannian manifold is a straightforward task, since DTW can operate with any measure of similarity between the different temporal features. Here, we use the geodesic distance between the projection matrices of different sequences as a distance function and warp all the sequences (corresponding to an AU) to a randomly selected sequence. Then the final model for each AU is obtained by computing the Karcher mean of all the warped sequences. This is a simple and fast approach that works fairly well.

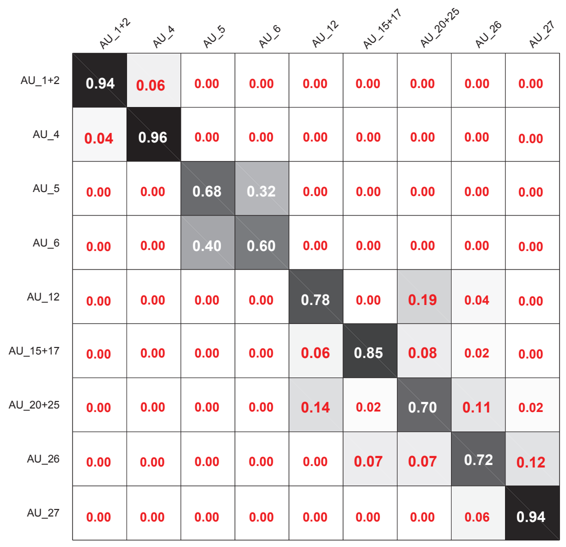
4.3.3 Action Units Recognition

Using the learned AU models for the Bosphorus and CK databases, we perform AU recognition. We report the results on seventeen single AUs in the Bosphorus database and nine single or combined AUs in the CK database. The training samples are chosen as images/sequences containing only the target AU occurring in the corresponding local facial components (brow, eye, nose, and mouth). In the Bosphorus database the lack of sufficient landmarks on the faces limits our recognition capabilities. For example we cannot recognize the AU-43 since no landmarks are provided for the eyes. Also for the CK database, since the sequences are mainly corresponding to the expressions and not AUs, we only chose those AUs for which enough training sequences are available. We divide both databases into eight sections, each of which contains images from different subjects. Each time, we use seven sections for training and the remaining sections for testing so that the training and testing sets are mutually exclusive. The average recognition performance is computed on all the sections.

For the Bosphorus database, we perform maximum likelihood (ML) recognition where we find the probability of each test velocity vector comes from the learned Gaussian distribution. But for the CK database, we first warp each test



	UFAU_1	UFAU_2	UFAU_4	LFAU_10	LFAU_12	LFAU_12L	LFAU_12R	LFAU_15	LFAU_16	LFAU_17	LFAU_18	LFAU_20	LFAU_24	LFAU_25	LFAU_26	LFAU_27	LFAU_28
UFAU_1	0.91	0.03	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
UFAU_2	0.00	0.97	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
UFAU_4	0.06	0.00	0.94	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
LFAU_10	0.00	0.00	0.00	0.80	0.02	0.00	0.00	0.00	0.04	0.00	0.00	0.00	0.00	0.02	0.09	0.02	0.00
LFAU_12	0.00	0.00	0.00	0.00	0.84	0.05	0.04	0.00	0.00	0.00	0.00	0.06	0.01	0.00	0.00	0.00	0.00
LFAU_12L	0.00	0.00	0.00	0.00	0.05	0.93	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00
LFAU_12R	0.00	0.00	0.00	0.00	0.04	0.00	0.93	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02
LFAU_15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.76	0.00	0.08	0.00	0.03	0.08	0.05	0.00	0.00	0.00
LFAU_16	0.00	0.00	0.00	0.04	0.00	0.00	0.00	0.00	0.63	0.00	0.00	0.02	0.00	0.22	0.09	0.00	0.00
LFAU_17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.87	0.00	0.00	0.09	0.00	0.00	0.00	0.02
LFAU_18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.96	0.00	0.00	0.04	0.00	0.00	0.00
LFAU_20	0.00	0.00	0.00	0.00	0.05	0.02	0.00	0.12	0.00	0.00	0.00	0.72	0.07	0.02	0.00	0.00	0.00
LFAU_24	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.07	0.00	0.09	0.00	0.02	0.70	0.00	0.00	0.00	0.11
LFAU_25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.20	0.00	0.00	0.02	0.00	0.63	0.13	0.00	0.00
LFAU_26	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.13	0.00	0.02	0.00	0.00	0.11	0.74	0.00	0.00
LFAU_27	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.95	0.00
LFAU_28	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.03	0.00	0.01	0.11	0.00	0.00	0.00	0.82



	AU_1+2	AU_4	AU_5	AU_6	AU_12	AU_15+17	AU_20+25	AU_26	AU_27
AU_1+2	0.94	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00
AU_4	0.04	0.96	0.00	0.00	0.00	0.00	0.00	0.00	0.00
AU_5	0.00	0.00	0.68	0.32	0.00	0.00	0.00	0.00	0.00
AU_6	0.00	0.00	0.40	0.60	0.00	0.00	0.00	0.00	0.00
AU_12	0.00	0.00	0.00	0.00	0.78	0.00	0.19	0.04	0.00
AU_15+17	0.00	0.00	0.00	0.00	0.06	0.85	0.08	0.02	0.00
AU_20+25	0.00	0.00	0.00	0.00	0.14	0.02	0.70	0.11	0.02
AU_26	0.00	0.00	0.00	0.00	0.00	0.07	0.07	0.72	0.12
AU_27	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.06	0.94

Figure 4.8: Confusion matrices for AU recognition on the Bosphorus (left) and CK (right) databases. LFAU and UFAU refer to the lower and upper face AUs.

sequence to the learned template using DTW and then use the distance between the two sequences for recognition. Figure 4.8 shows the confusion matrices for both databases. As the results indicate, for the AUs that are mainly identified by their facial deformations the recognition rate is high, e.g. AU-2, AU-4, and AU-27. However, for AUs whose distinction is more due to the appearance deformations than the geometries, the algorithm may confuse them with the AUs with similar geometries, e.g. AU-16 and AU-25. In these cases, AUs occurring in other parts of the face can be used as cues to remove the ambiguity and improve the recognition.

We also performed a recognition experiment using the Bosphorus database on the Euclidean space, where the normal parallel transport is performed before learning the distributions. While the average recognition rate for AUs on the Grassmann manifold is 83%, this value is 79% on the Euclidean space. Although the recognition rate is improved on the Grassmannian, but it is not considerable. A possible reason can be the fact that in the Bosphorus database the faces are almost always frontal and there are not significant affine variations between the faces.

4.3.4 Basic Expression Recognition

To have a comparison with baseline results, we perform the basic expression recognition using the CK database. For this purpose, we manually label the sequences into one of six basic expressions, namely, {Happy, Sad, Fear, Surprised, Disgust, Angry} and then from each sequence we select the last five frames. We apply the linear discriminant analysis (LDA) and multi-class SVM to the training data and perform leave-one-subject-out cross-validation over 89 subjects. Table 4.1 compares the results of applying LDA and SVM to the normalized data in the Euclidean space as well as subspace representations on the Grassmann manifold. For the Grassmann data, we use the projection Grassmann kernel, [129], to perform SVM as well as kernel LDA. The table also shows the latest results reported by Cohn and Kanade *et al.* [128]. However, it should be noted that these results are reported on the extended Cohn-Kanade dataset (CK+) which has more subjects and accurate expression coding. The results show some improvements in using the

Table 4.1: Basic expression recognition on the CK dataset using algorithms on both Euclidean (E) and Grassmann (G) spaces. The last row shows the results on CK+ dataset.

	Ha	Sa	Fe	Su	Di	An	Averaged
E-LDA	91.3	75.0	70.2	96.1	76.3	60.0	78.15
G-LDA	88.9	78.2	74.4	97.3	80.5	68.0	81.2
G-KLDA	95.1	85.7	83.0	98.6	86.8	65.7	85.8
E-SVM	91.3	80.3	74.4	97.2	78.9	62.8	80.8
G-SVM	95.0	85.7	74.5	97.2	78.9	65.7	82.8
SVM [128]	98.4	4.0	21.7	100.0	68.4	35.0	54.6

geometry of the Grassmannian over performing the analysis in the Euclidean space. Although we see only modest improvements, but as discussed before, while performing normalization in the Euclidean space suffers from being arbitrary and thus is highly sensitive to noise, the analysis on the Grassmann is more stable and resilient to noise.

4.4 Facial Expression Synthesis

In this section, we illustrate the capability of this representation for expressions synthesis and mapping among subjects. We also evaluate the performance of our approach in decomposing an expression into its constituent AUs. Our results show that using proper geometric modeling, we not only can recognize most of the AUs, based on templates learned for them, but also can recognize expressions by decomposing them into their AUs building blocks, perform novel expression synthesis, as well as transfer expressions among subjects.

4.4.1 Facial Expression Synthesis

As mentioned earlier, a facial expression is composed of several AUs where they may occur in combinations or show a serial dependence. Transition from AUs or combination of actions to another may involve no intervening neutral state (Fig. 4.9). In our geometric-based representation framework we can easily combine the velocity vectors corresponding to various AUs in order to generate a curve of new expression on the manifold. Since the velocity vectors are all defined on the same tangent plane

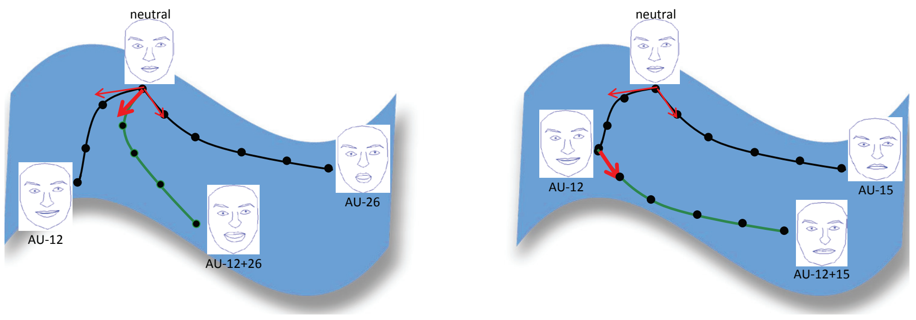


Figure 4.9: AUs combinations lead to new expression sequences. The combination can occur simultaneously (left) or sequentially (right).

$(o(m))$ addition of velocity vectors is well-defined. We can also compute weighted sums of velocity vectors to create new expressions. In addition, we can apply the AUs one after another in order to generate the sequentially occurring AUs.

Figure 4.9 illustrates the process of synthesizing new expression for both simultaneous and sequential cases. The combinations of AUs may be either additive or non-additive. The figure shows an example of an additive combination, AU-12+26, where the effect of each individual AU is still observed in the final combination. However, non-additive combinations represent further complexity. An example is AU-12+15, shown in Fig. 4.9, where the AU-15 modifies the effect of AU-12 on the lip corners. In these cases the resulting appearance change is highly dependent on timing. For non-additive AUs the combinations mostly occur sequentially [101].

We performed this experiment on the Bosphorus database. In this database, the combination of velocities for different AUs gives a new geodesic, sampling along which generates a sequence of new expressions, as can be seen in Fig. 4.10. The figure shows examples of generating *Surprise* and *Disgust* expressions using their corresponding AUs. For visualization, the generated sequence of subspaces on the

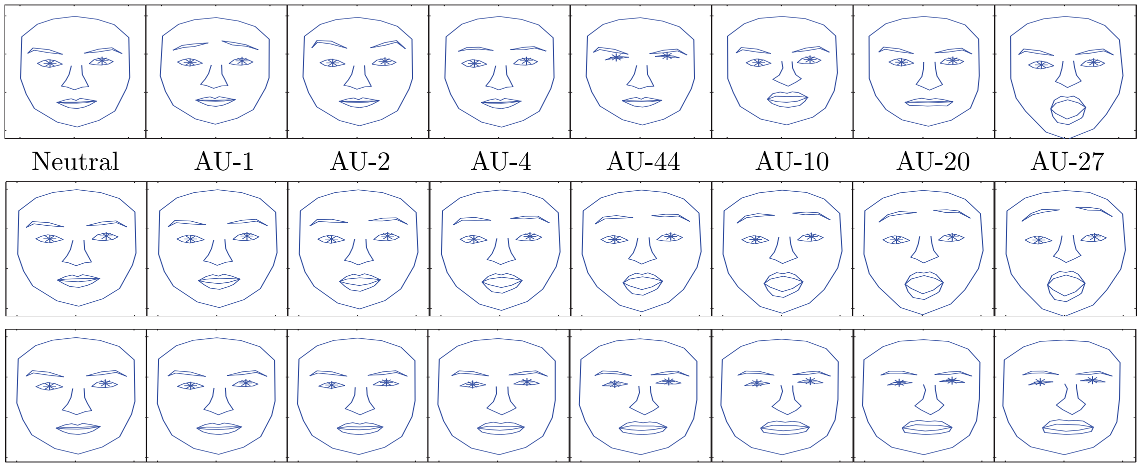


Figure 4.10: AUs combinations. First row: some of the AUs from Bosphorus database, middle row: combination of AUs 1+2+27 which gives *Surprise* expression, and last row: combination of AUs 4+44+10+20 which gives *Disgust* expression

manifold is transformed to the landmark space. For this purpose, we first find an orthonormal bases Y for each subspace P. Then using the locations of global landmarks on the face and their corresponding values in the matrix Y, the 2×3 affine matrix B is obtained. Applying this matrix to Y gives the landmarks on the image coordinate-system.

4.4.2 Expression Mapping

Expression mapping (also called performance-driven animation) has been a popular technique for generating realistic facial expressions. Given an image of the person's neutral face as well as the same person's face with an expression, the goal is to generate the same expression on a new person's neutral face. In this section we are not dealing with the appearance modeling of the expressions, but our goal is to show how we can transport an expression model between subjects. The appearance

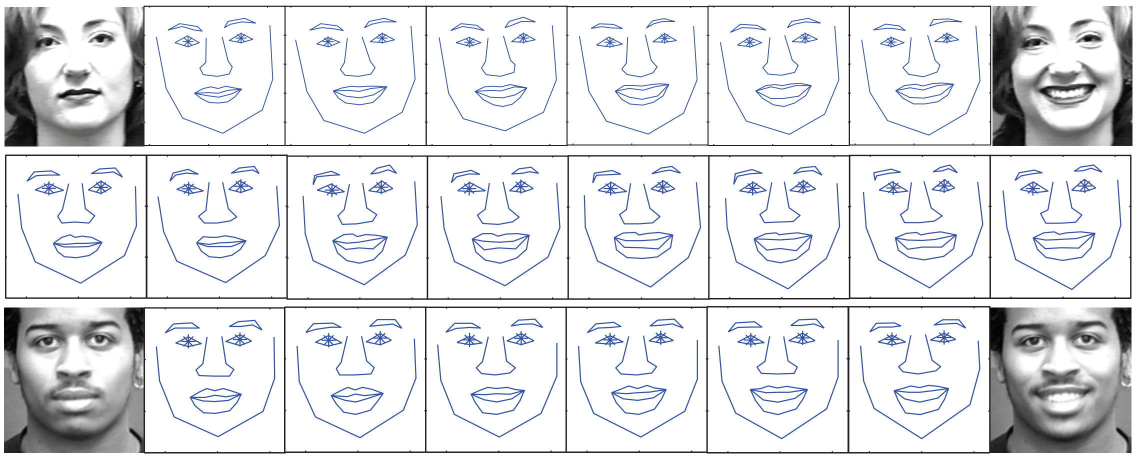


Figure 4.11: Expression mapping. The happy expression is mapped from a subject in the first row to another subject in the middle row. The last row shows the real happy expression for the second subject. The synthesized sequence has the style of the first subject.

modeling can always be done on top of geometric modeling to create photorealistic faces [130].

Mapping an expression from a point on the manifold to another point can be performed by parallel transporting the geodesics between subspaces using Algorithm 4. Figure 4.11 shows an example of mapping the *Happy* expression between two subjects in the Cohn-Kanade database. It also shows the real expression sequence available for the second subject. It should be noted that mapping an expression, from a subject to another subject, maps the style of the first subject as well. Therefore, the real sequence may not necessarily be the same as the synthesized one.

4.4.3 Expression Decomposition

Parsing an expression-stream into its constituent AUs is an essential requirement of a robust facial expression analysis system. Most of the existing algorithms

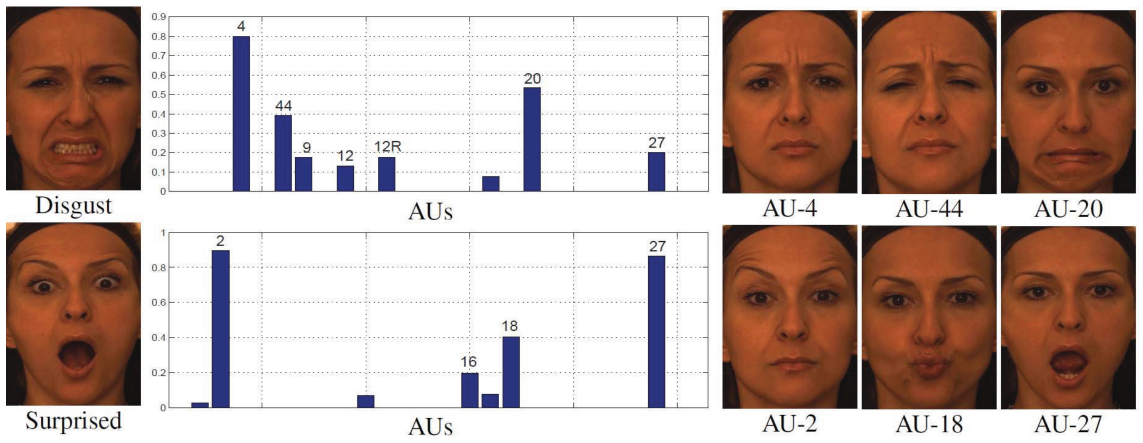


Figure 4.12: Expression decomposition into AUs. First row is *Disgust* expression and second row is *Surprise* expression. Only the labels for the AUs with the intensities greater than 0.2 are shown.

require training using various combination of AUs, additive and non-additive. But in the proposed algorithm we can decompose an expression only using the learned template models for the individual AUs. For this purpose, we build a dictionary matrix, $D = [A_{au-i}]$, where the columns are the velocity vectors corresponding to various AUs. This dictionary matrix can be built either for a subject using the mean velocity vectors learned for that subject, or for the whole database using AU template models learned using all the training data. Now having the velocity vector for a test expression, A_e, the coefficient vector $\mathbf{x}$ is obtained by solving for the linear system $A_e = D\mathbf{x}$ using linear programming and with the constraint that $\{x_j\}$'s must be all positive. The reason for this constraint is that each expression is the combination of several AUs with positive weights.

Figure 4.12 shows two examples of expression decomposition for a subject. As the figure shows, the *Disgust* and *Surprised* expressions are decomposed into some AUs. The figure also shows the AUs with considerable weights/intensities (greater than 0.2) for each expression. Knowing the AUs that an expression is composed of, we can easily recognize the expression. This experiment can be extended for stream of expressions to detect transitions among various AUs as well as different intensity level of an AU. (e.g. different level of mouth opening).

4.5 Summary

This work is a step toward breaking the dependence of facial expression analysis systems to the choice of the coordinate frame of the camera. We discussed that using the equivalence class of shapes in a proper shape-space, one can remove the need for a pre-processing step to align the data to a common coordinate frame. While we claim that the projective shape-space is the proper space to model the facial variations, we have limited our discussions to the affine shape-space since it is mathematically well understood compared to the projective space. We showed that the affine shape-space for our facial landmark configurations has Grassmannian properties and therefore nonrigid facial deformations due to various expressions can be represented as points on the Grassmann manifold. By modeling the facial expressions on this manifold we ensure that the variability being computed is from shape changes only and not the coordinate frame. We extended some of the available statistical algorithms for facial expressions, e.g. ASM, nonlinear manifold learning, and expression template learning, to the Grassmann manifold and showed the benefits of this representation. It should be noted that while similarity alignment in the Euclidean space can remove the effect of camera coordinate frame to a good extent, working with equivalence class of shapes in the shapes-spaces is a systematic way of dealing with alignment issue and the main benefits become more obvious when we move to the projective shape-space.

Chapter 5: Structure-Preserving Sparse Decomposition for Facial Expression Analysis

5.1 Introduction

Some emotions motivate human actions and others enrich the meaning of human communications [131]. Therefore, understanding the users' emotions is a fundamental requirement of human-computer interaction systems (HCI). Facial expressions are important means of detecting several emotions. Following the work of Ekman *et al.* [10], many studies have focused on the analysis and recognition of facial expressions. The goal of facial expression analysis is to create systems that can automatically analyze facial feature changes and map them to facial expressions. This has been an active research topic for several years and has attracted the interest of many computer vision researchers and behavioral scientists, with applications in behavioral sciences, security, animation and human-computer interaction.

Facial expressions are combinations of a set of AUs introduced in the Facial Action Coding System [10]. Action units are the smallest visibly discriminable muscle actions that combine to perform expressions and FACS is a human-observer-based system designed to code these subtle changes in facial features [4]. Such

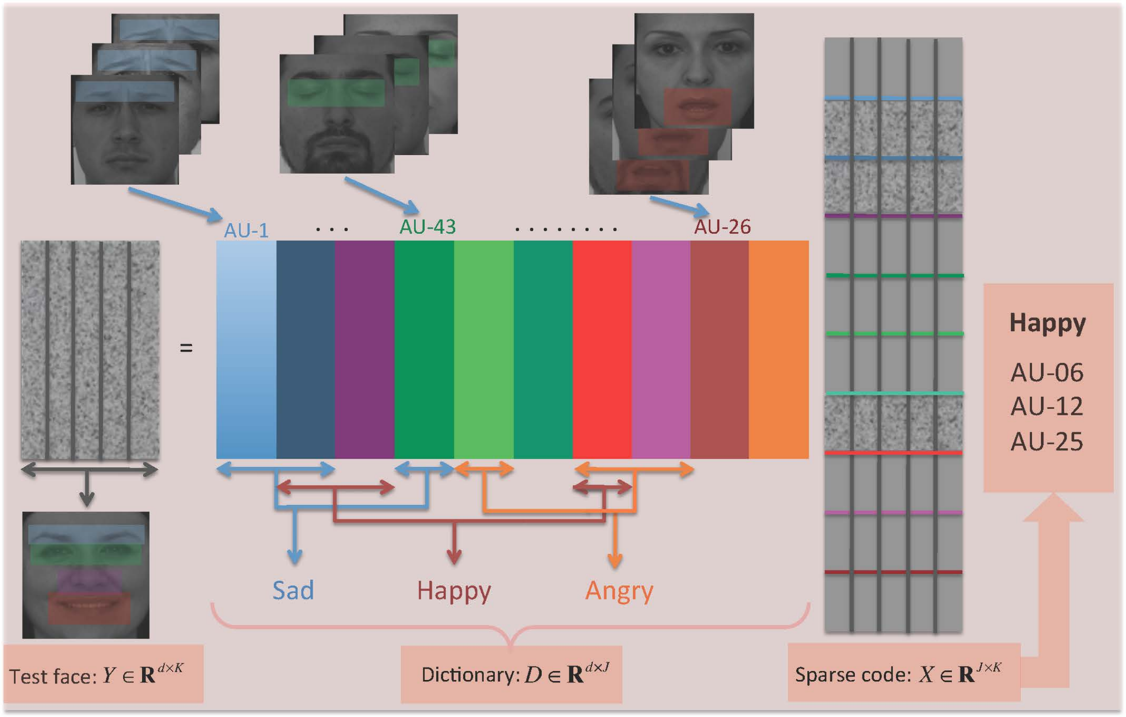

AU-1
. . .
AU-43
. . . .
AU-26
=
Happy
AU-06
AU-12
AU-25
Sad
Happy
Angry
Test face: $Y \in \mathbf{R}^{d \times K}$
Dictionary: $D \in \mathbf{R}^{d \times J}$
Sparse code: $X \in \mathbf{R}^{J \times K}$

Figure 5.1: Summary of the proposed algorithm. The AU-dictionary D is constructed from various blocks of AU atoms. The test face is represented using a matrix of features (image matrix Y) and is decomposed using the AU-dictionary and coded as a structure-preserving sparse code matrix X ($Y = DX$). This representation enables expressive face classification as well as decomposition into its constituent AUs.

changes happen locally in the face and result in both local appearance changes and shape deformations. Previous research studies on expression analysis indicate the importance of proper modeling of such local deformations for automatic expression analysis. AUs are suitable as mid-level representations in automatic facial expression analysis systems as they reduce the dimensionality of the problem [132, 133]. However, there have been only a few attempts that exploit the domain experts' knowledge on AU composition rules and expression decompositions for designing systems to analyze and recognize expressions.

In this chapter, we propose a dictionary-based approach for facial expression analysis including expression decomposition, classification and synthesis. Using the domain experts' knowledge on various AUs and how local facial regions are affected by these AUs, we first learn an AU-dictionary, D. This dictionary, as shown in Fig. 5.1, consists of AU-blocks, i.e., dictionary atoms corresponding to each AU, and so it has a particular structure which helps capture the high-level knowledge regarding AUs and their composition rules extracted from FACS. To encode this knowledge as sparse codes while designing the dictionaries, we propose a two-layer approach for grouping the dictionary atoms. The lower layer is the AU-layer which groups dictionary atoms corresponding to each AU. The top layer is called the expression-layer which uses the high-level knowledge to group different AUs that are composed to form a particular expression (e.g. Sad, Happy, Angry). This two-layer approach suggests a multi-layer structure-preserving sparse coding problem. The sparse code matrix, X, approximates an expressive face, Y, using this AU-dictionary.

As shown in Fig. 5.1, the test face is represented using a matrix of features, Y, which is referred to as the image matrix. Therefore, we are dealing with a multi-layer as well as multi-variable (columns of Y) sparse coding problem. The image matrix also has a structure in which local descriptors (columns) corresponding to each AU region on the face are grouped together and in the top layer all the columns are grouped together to represent a particular expression. In order to preserve this structure we define two grouping layers for this image matrix as well. Then by employing a multi-layer, multi-variable group sparse coding algorithm, we minimize a proper objective function which imposes these groupings (for both dictionary and image matrix) into the sparse code matrix X. This is effective for expression classification as well as decomposing an unknown expression. We can also synthesize new expressions through valid composition of AU-blocks of the dictionary.

Learning an AU-dictionary requires a dataset of subjects performing various AUs which is not always available. On the other hand, since the definition of AU is an ambiguous semantic description in FACS [134], it is hard to define proper structures for AUs. Hence, we further propose a structure-preserving dictionary learning algorithm to jointly learn several semantic structures on expressive faces and their corresponding dictionary atoms (structure-blocks). For this purpose, we introduce an appropriate objective function and propose a greedy algorithm to optimize that. We then use the learned dictionary (concatenation of structure-blocks) for expression classification in a similar way as discussed for the AU-dictionary. We evaluate the proposed algorithm on two publicly available datasets and demonstrate the effectiveness of algorithms for expression decomposition and classification. We also illustrate some preliminary examples of expression synthesis using our generative algorithm.

Contributions: Our main contributions in this work are:

- We learn an AU-dictionary by defining proper semantic regions on the face.
- We incorporate the high-level knowledge from FACS regarding AUs and their composition rules as a two-layer grouping over dictionary atoms. We also impose a similar two-layer grouping over the test image matrix.
- We extend the single multi-layer group sparse coding algorithm proposed in

[135] to the multi-layer, multi-variable group sparse coding case.

- We also propose a structure-preserving dictionary learning algorithm to replace the AU-dictionary.

Outline of the chapter: We review related work in Section 5.2. Then Section 5.3 discusses the structured AU dictionary, our approach to generate it and the groupings we impose over the dictionary atoms. Then in Section 5.4 we present the multi-layer multi-variable group sparse coding, the objective function and the algorithm to optimize it. A structure-preserving dictionary learning algorithm is then proposed in Section 5.5. Experimental results are presented in Section 5.6.

5.2 Related Work

Many previous approaches for expression analysis have proposed discriminative classifiers for AUs and/or universal emotions [102, 106, 110, 128, 134, 136–138]. Among these, Littlewort *et al.* [136] presented the Computer Expression Recognition Toolbox (CERT) which is a software tool for fully automatic real-time facial expression recognition. CERT can automatically code the intensity of 19 different facial actions from FACS and 6 different universal facial expressions. Although some of these approaches show very promising recognition rates on emotions/AUs, they do not benefit from the connection among AUs and emotions provided in FACS as well as they are pure discriminative classifiers. For surveys on recent developments in universal emotions and AU recognition, we refer the readers to [131, 132].

A few algorithms that employed the knowledge presented in FACS regarding

AUs and emotions for expression analysis are reviewed here. Tong *et al.* [106] systematically combined prior knowledge about facial expressions and AUs with image measurements through a dynamic Bayesian network (DBN) to achieve accurate, robust, and consistent facial expression analysis. They modeled the relationship between AUs and the local facial components as well as the relationships among AUs themselves using a complex graphical model and used that to infer facial expressions.

Yang *et al.* [134] interpreted facial expressions by learning some compositional features based on local appearance features around AU areas. They avoided AU detection, and tried to interpret facial expression by learning these compositional appearance features. They showed the consistency of the built compositional features with respect to the interpretation of FACS.

There are some generative approaches for expression analysis in which new expressions are recognized as compositions of simpler/basic expressions. Active shape models (ASM) and active appearance models (AAM) [139] are used to approximate deformable face models using linear subspace analysis. However, linear subspace methods are inadequate to represent the underlying structure of real data and so nonlinear manifold learning approaches are proposed. Liao *et al.* [107] decomposed each expression using a basis of eight one-dimensional manifolds each learned offline from sequences of labeled universal expressions. They applied tensor voting to learn these nonlinear deformation manifolds and showed results for both expression recognition and synthesis.

By observing that images of all possible facial deformations of an individual make a smooth manifold embedded in a high dimensional image space, Chang *et* *al.* [140] proposed a probabilistic video-based facial expression recognition method on manifolds. They represented a complete expression sequence as a path on the expression manifold and used a probabilistic model to perform expression recognition as well as synthesize image sequences. Taheri *et al.* [141] modeled AUs as geodesic pathways on the Grassmann manifold. This representation enables expression models to be generalized across view changes. Moreover, it enables the decomposition of an expression into constituent AUs, synthesis of new expressions and expression mapping between different subjects.

These studies have limitations in terms of the expressions they recognize or features they use. Most of them are pure discriminative classifiers or only use the facial shape information and so can only synthesize the shape sequences. But they all suggest that it is important to propose a generative expression analysis system that models AUs and employ them for expression analysis by incorporating domain experts' knowledge provided by FACS regarding expression decomposition and AU composition rules.

Dictionary learning and sparse coding have been effective for robustly modeling data with some level of noise and intra-class variations. Algorithms for data-driven learning of domain-specific overcomplete dictionaries are widely employed for reconstruction and recognition applications [142–145]. Local variations in the appearance of the faces due to various expressions can also be modeled using a set of dictionary atoms. But facial expressions are structured actions (e.g. deformations corresponding to each AU occur at a particular local region of the face) and maintaining this structure is important for expression analysis. Yu *et al.* [146] propose a computa-

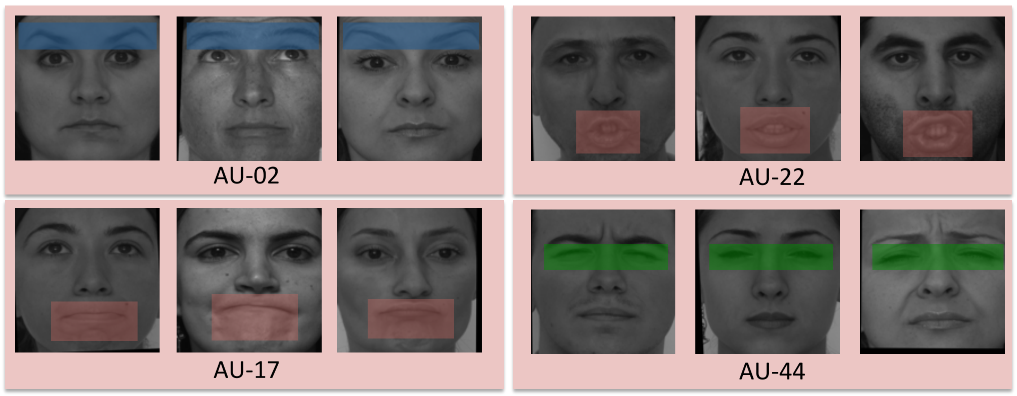
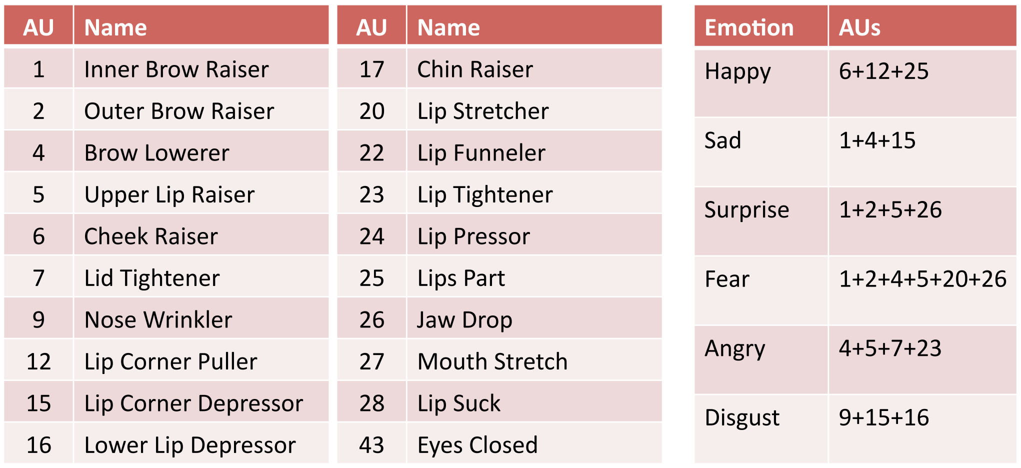


Figure 5.2: Some samples of various AUs and different subjects performing them from the Bosphorus dataset. The regions that each AU affects are illustrated on the faces.

tionally efficient MAP-EM algorithm for structure-preserving dictionary learning. This approach regularizes the sparse estimation by assuming dependency on the selection of active atoms. In this work, we also need to learn a structure-preserving dictionary for modeling facial expressions and AUs.

5.3 Action Unit Dictionary

Action units are the basic components of each expression and they are usually different for various expressions. Therefore, breaking an expression into a set of AUs is an important step toward facial expression analysis. Facial action coding proposed in FACS serves only as an initial step. However, the next step for modeling these AUs and exploiting them for expression analysis is particularly difficult due to the ambiguous semantic nature of AUs. In this section, we propose a dictionary learning framework for facial AUs.



AU	Name
1	Inner Brow Raiser
2	Outer Brow Raiser
4	Brow Lowerer
5	Upper Lip Raiser
6	Cheek Raiser
7	Lid Tightener
9	Nose Wrinkler
12	Lip Corner Puller
15	Lip Corner Depressor
16	Lower Lip Depressor

AU	Name
17	Chin Raiser
20	Lip Stretcher
22	Lip Funneler
23	Lip Tightener
24	Lip Pressor
25	Lips Part
26	Jaw Drop
27	Mouth Stretch
28	Lip Suck
43	Eyes Closed

Emotion	AUs
Happy	6+12+25
Sad	1+4+15
Surprise	1+2+5+26
Fear	1+2+4+5+20+26
Angry	4+5+7+23
Disgust	9+15+16

Figure 5.3: FACS action units and their compositions. As the right table shows, combinations of different AUs generate universal facial emotions. [4]

5.3.1 Modeling Action Units

Each AU determines the deformation of its corresponding facial components, as shown in Fig. 5.2 and Fig. 5.3. Figure 5.2 shows faces of a few subjects performing different AUs. As can be seen, each AU acts in a local area of the face while keeping other parts unchanged. This motivates using local features extracted from expressive faces to model each AU. Moreover, Fig. 5.2 shows that there is a large degree of inter-subject variations in performing various AUs. These individual differences in expressiveness relates to the degree of facial plasticity, morphology, frequency of intense expression, and overall rate of expression [4]. These intra-subject variations should also be considered while modeling various AUs. As discussed earlier, dictionary learning is effective for modeling such data with large degree of inter-class variations.

While dictionary learning using all local descriptors extracted from faces can be effective for expression analysis, it does not preserve the structure of facial deformations. For example, a deformation in a particular local area of the face (e.g. mouth) can be reconstructed using the same set of dictionary atoms used for representing a deformation in different parts of the face (e.g. brow). The coherence of such a dictionary limits its descriptive and discriminative power and the large degree of freedom in choosing dictionary atoms may become a source of instability in decomposition [146].

Therefore, it is important to preserve the structure of deformations while modeling AUs. To this end, we learn a dictionary per AU using local features extracted from AU semantic regions on faces performing that AU. These semantic regions are defined as regions on the face in which local deformations corresponding to various AUs occur. Such regions can be subjectively defined by looking at various faces performing particular AUs. Figure 5.2 illustrates some examples of these semantic regions we use for different AUs. After defining these regions, we apply the well-known K-SVD algorithm [142] to learn data-driven AU dictionary blocks, i.e., atoms corresponding to each AU. There are also some extensions proposed for the K-SVD algorithm, [143, 144], to learn dictionaries that are compact, discriminative as well as reconstructive. We can apply these extensions to learn dictionaries for the AUs acting on the same semantic region of the face in order to learn dictionaries that are discriminative for those AUs.

Finally, we have several blocks of dictionaries (AU-blocks) which can be combined to generate an AU-dictionary, as illustrated in Fig. 5.1. This dictionary has a structure indicated by its AU-blocks. This structure is later imposed as a constraint for structure-preserving sparse coding (Section 5.4). Details of feature representation and extraction are discussed in Section 5.3.3.

5.3.2 Composing Action Units

There are some subsets of AUs that usually co-occur on the face to generate meaningful facial emotions. The number of these subsets is much smaller than all possible combinations of AUs. These subsets of AUs are especially well-known for universal expressions. For example, it is known that a happy face is usually a combination of AUs-$\{6+12+25\}$. Figure 5.3 shows some combinations of AUs that generate universal facial emotions. After learning the AU-dictionary, we should define these high-level groupings for AU-blocks in the dictionary. Such a grouping is particularly useful for expression classification into one of universal facial emotion classes as well as expression decomposition.

It should be noted that while most of the expressions can be reconstructed as linear combinations of additive AUs, there are some non-additive AU combinations as well. An example of an additive AU combination is smiling with mouth open, which can be coded as AU-$\{12+25\}$, AU-$\{12+26\}$, or AU-$\{12+27\}$ depending on the degree of lip parting [4]. However, non-additive combinations usually affect the same area of the face where the outcome of their simultaneous occurrence is different from the effect of each of the constituent AU. An example is AU-$\{12+15\}$, which often occurs during embarrassment. Although AU-12 raises the cheeks and lip corners, its action on lip corners is modified by the downward action of AU-

15 [4]. Although such non-additive combinations usually occur sequentially, the simultaneous occurrence of them imposes further constraints. Since the number of such non-additive AUs is limited, this issue can be addressed by adding these non-additive combinations as new AUs to the dictionary.

5.3.3 Feature Extraction

Feature representation plays an important role in facial expression recognition. The features used for this purpose generally fall into two categories, geometric features [107, 140, 141] and appearance features [106, 134, 147]. In this work we use proper appearance features to model local appearance deformations.

Selection of appropriate features is critical in facial expression analysis. Popular local appearance descriptors include Gabor filter, Haar-like features, SIFT and Local Binary Patterns (LBP). While all these features are powerful for describing local appearances, the SIFT features are effective in describing the edges and finer appearance features. Since deformations corresponding to facial expressions are mainly in the form of lines and wrinkles, we chose SIFT features for our experiments.

To extract the local features, we divide each face image into overlapping patches so that they cover all the AUs' semantic regions (Fig. 5.4). We extract SIFT features at three scales from the center of all the patches, denoted as $\{P_1, P_2, ...P_K\}$, where K is the number of image patches. We choose the local patches to be of size $(n/8 \times n/8)$ where the size of the face images are $(n \times n)$ (after aligning and

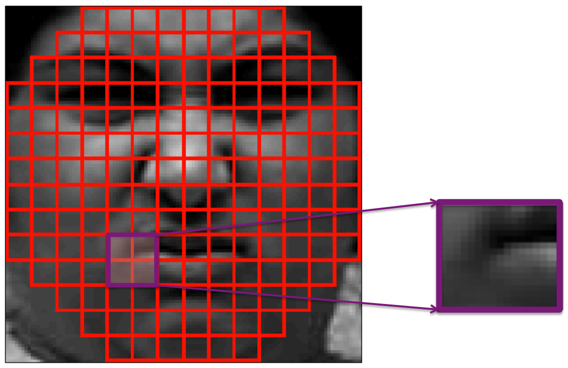


Figure 5.4: Local overlapping patches are placed on the face image and local descriptors are extracted from each patch.

resizing). The amount of overlaps between patches may vary and in the experiments presented in this chapter it is set to $n/16$.

5.4 Structure-Preserving Sparse Coding

In the previous section we discussed how to learn an AU-dictionary and defined some compositional rules for grouping AUs to form various emotions. The particular structure in the dictionary and the AUs grouping can be imposed as constraints for structure-preserving sparse coding. In this section, we discuss the formulation of our multi-layer, multi-variable group sparse coding and present methods for predicting the class label of an expressive face and decomposing an expression into its constituents using sparse codes.

5.4.1 Problem Formulation

Having the structured AU-dictionary, an expressive face can be decomposed into combinations of some AUs. Our goal is to impose proper grouping-based constraints on this decomposition so that a sparse subset of AU-blocks in the dictionary is used for expression reconstruction and the subset is among the valid compositions of AUs that we discussed in Section 5.3.2. Such grouping-based constraints encode the high-level knowledge regarding expression formations on the face.

Dictionary grouping: There are two layers of grouping constraints to be imposed on the dictionary atoms. In the lower layer, to emphasize that the dictionary atoms used for reconstructing the test face should come from a sparse set of AUs, we impose the sparsity constraint on the AU-blocks in the dictionary and force the number of blocks with non-zero coefficients to be as few as possible. On the top layer, we impose AUs co-occurrence information through valid AUs composition (or expression-layer grouping) by forcing the groups having non-zero coefficients for their AU-blocks to be as few as possible. We refer to these two layers of grouping constraints as the *dictionary-AU-layer* and the *dictionary-expression-layer*.

Image matrix grouping: As mentioned before, an expressive face is represented as a set of local descriptors extracted from overlapping patches on the face. So we have a multi-variable (multi-column) representation for each test face. However, the sparse representation for each of the local descriptors is not independent from others. Hence, we define two layers of grouping constraints on the image matrix. On the top layer, we force the same sparsity pattern for all the local descriptors by grouping them together in one group. This means that we want all these local descriptors to have the same expression label. In the lower layer, we want the descriptors extracted from the same AU semantic regions on the face to have similar sparsity patterns. This implies that these descriptors are reconstructed using the atoms in the same dictionary-AU-layer. We refer to these two layers of grouping on the image matrix as the *test-AU-layer* and the *test-expression-layer*. An illustration of these different layers of grouping constraints and the sparse code matrix is depicted in Fig. 5.1.

The AU-dictionary consists of N blocks for N action units, $D = \{D_1, D_2, ..., D_N\}$, where the i^{th} block has J_i dictionary atoms. Therefore, the AU-dictionary, D, is a matrix of size $d \times J$ where d is the dimensionality of local descriptors extracted from a patch on the face and $J = \sum_i J_i$. The dictionary-AU-layer is formed by grouping atoms in each D_i and the dictionary-expression-layer is formed by grouping some valid subsets of $\{D_i\}$'s as discussed in Section 5.3.2. We express these two grouping layers using a set of indices, $G_{dic} = \{g_1, g_2, ..., g_{|G_{dic}|}\}$, where $g_i \subset \{1, ..., J\}$ includes indices of those dictionary atoms grouped together in either AU- or expression-layer. $|G_{dic}|$ is the total number of such groupings which is a summation of the number of groups in two layers.

Image matrix Y is a $d \times K$ matrix where K indicates the number of local descriptors extracted from each expressive face (number of patches). We also define a set of indices, $G_{tst} = \{g_1, g_2, ..., g_{|G_{tst}|}\}$ where $g_i \subset \{1, ..., K\}$ includes indices of grouped dictionary atoms in either the AU- or the expression-layer. $|G_{tst}|$ is the total number of such groupings which is the summation of number of groups in two layers.

The matrix of sparse codes, $X \in \mathbb{R}^{J \times K}$ should be computed such that the grouping structures indicated by G_{dic} and G_{tst} are satisfied. Therefore, we formulate the objective function of multi-layer, multi-variable structure-preserving sparse coding as follows:

$$\begin{aligned} \min_{X \in \mathbb{R}^{J \times K}} f(X) = \min_{X} \quad & \frac{1}{2}\|Y - DX\|_F^2 \\ & + \gamma_{dic} \sum_{k=1}^{K} \sum_{g \in G_{dic}} \omega_g^{dic} \|X_g^{(k)}\|_2 \\ & + \gamma_{tst} \sum_{j=1}^{J} \sum_{g \in G_{tst}} \omega_g^{tst} \|X_g^{(j)}\|_2 + \lambda \|X\|_1 \end{aligned} \tag{5.1}$$

Here γ_{dic}, γ_{tst}, ω_g^{dic}, ω_g^{tst} and λ are weights on different layers and different groups. $X^{(k)}$ and $X^{(j)}$ are the k^{th} column and the j^{th} row of matrix X, respectively, $X_g^{(k)}$ is a part of the column $X^{(k)}$ indicated by the indices in $g \in G_{dic}$ and $X_g^{(j)}$ is a part of the row $X^{(j)}$ indicated by the indices in $g \in G_{tst}$. Also $\|X\|_1$ is the l_1-norm of the matrix X which is defined as the l_1-norm of the vector formed by concatenating all the columns of the matrix. This l_1-norm penalty encourages the solution to be generally sparse, and λ is the regularization parameter that controls the sparsity level. For the $X_g^{(k)}$ and $X_g^{(j)}$ we use the l_2-norm to encode the sparse codes within each group as a unit. We adopt the Proximal Gradient method proposed recently in [135] to optimize this objective function and extend it to the multi-layer, multi-variable sparse coding case, as discussed in the next section.

5.4.2 Objective Optimization

In this section we discuss an extension of the Proximal Gradient method [135] to optimize the objective function (5.1). This objective function consists of three terms as follows,

$$\min_{X \in \mathbb{R}^{J*K}} f(X) = \min_{X} g(X) + \Omega(X) + \lambda \|X\|_1 \tag{5.2}$$

where $g(X) = \frac{1}{2}\|Y - DX\|_F^2$ is the squared-error loss and $\Omega(X)$ is called the structured-sparsity-inducing penalty [135]. The main challenge in optimizing this objective function arises from the overlapping group structure in the non-smooth penalty term $\Omega(X)$. The overlaps among $\{X_g^{(k)}\}_{g \in G_{dic}}$ and $\{X_g^{(j)}\}_{g \in G_{tst}}$ make the block coordinate descent methods [148, 149] which are commonly used for the problem with non-overlapping groups (group Lasso) not applicable. The most widely adopted method for addressing this problem is to formulate it as a second-order cone programming (SOCP) and solve it by the interior method (IPM) [150]. But this approach is computationally prohibitive even for problems of moderate size. Very recently, Chen *et al.* [135, 151] proposed the Proximal Gradient method for estimating regression parameters with the overlapping group structure encoded in the structured-sparsity-inducing norm. They showed that using the dual norm, the non-separable structured-sparsity-inducing penalty $\Omega(X)$ can be approximated using a smooth function such that its gradient can easily be calculated. The approximation problem can then be solved by the first-order proximal gradient method: fast iterative shrinkage-thresholding algorithm (FISTA) [152]. In this work, we adopt this method and extend it to the multi-layer, multi-variate group sparse coding in order to optimize (1).

The non-smooth penalty term $\Omega(X)$ can be formulated as

$$\begin{aligned} \Omega(X) &= \gamma_{dic} \sum_{k=1}^{K} \sum_{g \in G_{dic}} \omega_g^{dic} \|X_g^{(k)}\|_2 \\ &+ \gamma_{tst} \sum_{j=1}^{J} \sum_{g \in G_{tst}} \omega_g^{tst} \|X_g^{(j)}\|_2 \\ &= \max_{A_1, A_2} \quad \langle C_1 X, A_1 \rangle \quad + \quad \langle C_2 X^T, A_2 \rangle \end{aligned} \tag{5.3}$$

where A_1 and A_2 are auxiliary matrices associated with $X_g^{(k)}$ and $X_g^{(j)}$ respectively. The matrix A_1 is of size $\sum_{g \in G_{dic}} |g| \times K$ and its k^{th} column is defined as $\alpha^k = [\alpha_{g_1}^{k\ T}, ..., \alpha_{g_{|G_{dic}|}}^{k\ \ T}]^T$ with the domain $\mathcal{Q} \equiv \{\alpha^k | \|\alpha_g^k\|_2 \leq 1, \forall g \in G_{dic}\}$, where $\mathcal{Q}$ is the Cartesian product of unit balls in Euclidean space and thus a closed and convex set. A_2 is also a matrix of size $\sum_{g \in G_{tst}} |g| \times J$ and its j^{th} column is defined as $\alpha^j = [\alpha_{g_1}^{j\ T}, ..., \alpha_{g_{|G_{tst}|}}^{j\ \ T}]^T$ with the similar domain as defined before. There are also two highly sparse matrices, C_1 and C_2, which help separating the overlapping groups in X. In the matrix $C_1 \in \mathbb{R}^{\sum_{g \in G_{dic}} |g| \times J}$, the rows are indexed by all pairs of $(i, g) \in \{(i, g) | i \in g, i \in \{1, ..., J\}\}$, the columns are indexed by $j \in \{1, ..., J\}$, and each element is given as:

$$C_{(i,g),j}^{(1)} = \begin{cases} \gamma_{dic} \omega_g^{dic} & \text{if } i = j, \\ 0 & \text{otherwise} \end{cases}$$

and similarly the elements of $C_2 \in \mathbb{R}^{\sum_{g \in G_{tst}} |g| \times K}$ are also defined (replacing $\gamma_{dic}\omega_g^{dic}$ with $\gamma_{tst}\omega_g^{tst}$).

The smooth approximation of $\Omega(X)$ is formulated as follows,

$$f_\mu(X) = \max_{A_1, A_2} \big(\langle C_1 X, A_1 \rangle + \langle C_2 X^T, A_2 \rangle - \mu (d(A_1) + d(A_2)) \big) \tag{5.4}$$

where μ is the positive smoothness parameter which controls the degree of approximation and $d(A_{(.)}) = \frac{1}{2}\|A_{(.)}\|_F^2$. Chen *et al.* [135] proved that for any $\mu > 0$, $f_\mu(X)$ in a convex and continuously-differentiable function in X, and the gradient of $f_\mu(X)$ takes the following form:

$$\bigtriangledown f_\mu(X) = C_1^T A_1^* + A_2^{*T} C_2$$

where A_1^* and A_2^* are optimal solutions to (5.4). [135] also provides the closed-form equations for these optimal solutions. The equations presented in [135] can be easily extended to our problem. Given the smooth approximation of the non-smooth structured-sparsity-inducing penalties, the fast iterative shrinkage-thresholding algorithm (FISTA) can be applied to minimize the objective function in (5.2). Readers are referred to [135, 151] for more details on this optimization technique and the smoothing proximal gradient algorithm for structured sparse coding.

5.4.3 Expression Decomposition and Classification

Using the structure-preserved sparse code matrix X, we can decompose an expressive face into its constituent AUs. Then the magnitude of each AU (the l_2-norm of the corresponding block in the sparse code matrix X) can be an indication of the intensity of that AU in the face. This decomposition is not limited to faces with universal emotions and can be performed for any expressive face. However since FACS specifically has the information regarding AUs composition rules for the universal emotions, we can enforce these particular groupings into the dictionary-expression-layer and employ that for universal expression recognition.

Using the information presented in Fig. 5.3, right table, we form the grouping indices for the dictionary-expression-layer. Now having a test face with universal emotion, we can predict the class label for this face based on the reconstruction error in the dictionary-expression-layer such that we assign the test image to the class with the minimum residual error computed as:

$$c^* = \arg\min_{c} \|Y - D\delta_c(X)\|_F \tag{5.5}$$

where $\delta_c(X)$ is obtained by setting all the coefficients in X except those in the c^{th} dictionary-expression-layer to be zero.

5.5 Structure-Preserving Dictionary Learning

Constructing the AU-dictionary needs expert-level knowledge regarding AUs and the regions they affect on the face as well as a dataset with subjects performing various AUs for data-driven dictionary learning. Such information and dataset may not always be available. Therefore, it is necessary to have an automatic approach for structure-preserving dictionary learning for facial expression analysis.

To this end, we propose a structure-preserving dictionary learning algorithm for facial expressions. The goal is to jointly estimate some semantic structures on different expressive faces and their corresponding dictionary atoms. Our approach is motivated by Yu *et al.* [146] which proposed a computationally efficient MAP-EM algorithm for structure-preserving dictionary learning. With the goal of preserving the image directional regularity, they defined an initial set of dictionary bases using directional PCAs. Then using the EM framework, patches from the input image are clustered based on their residual errors over the initial dictionary bases and the dictionary bases are updated. This process converges after some iterations and finally the directional structures on the image as well as their dictionary bases are obtained.

We model a face as a collection of local subspaces so that each subspace element is well reconstructed using that subspace basis and its approximation via other subspaces results in a large residual error. So the goal is to find these subspace structures over various expressive faces. In other words, we want to find some clusters $\{S_i\}$ over the facial patches and their corresponding dictionary atoms $\{D_i\}$

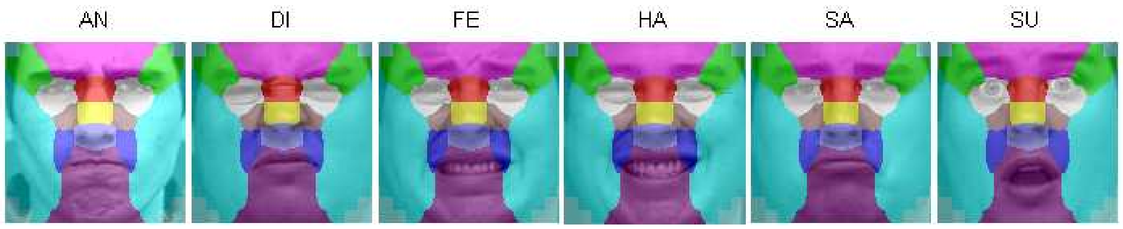


Figure 5.5: Learned structures for the universal expressions from the CK+ dataset (best viewed in color). AN: Angry, DI: Disgust, FE: Fear, HA: Happy, SA: Sad, SU: Surprise.

so that the final clusters (or their corresponding subspace representations) are as separate as possible. This can be achieved by maximizing the summation over cross-residual errors which we define as the error in representing each cluster's descriptors using other cluster's dictionary atoms. The objective function can be formulated as follows,

$$\max_{\{S_i\},\{D_i\}} \sum_i \sum_{j \neq i} \|Y(S_i) - D_j X_{ij}\|_F^2 \tag{5.6}$$

where $Y(S_i)$ is a matrix with the columns of all the descriptors extracted from the patches at cluster S_i, and X_{ij} is the matrix of sparse coefficients resulting from decomposition of $Y(S_i)$ on dictionary D_j where $j \neq i$. This problem can be solved by a greedy algorithm based on a bottom-up pair-wise merging procedure. The algorithm starts with some initial clustering of input patches and then in order to maximize (5.6), at each step we greedily merge two clusters with minimum cross-residual costs.

We initialize the clusters at each of the patch locations, p, on expressive faces with a same expression label, e, and learn the initial dictionary atoms, $D_{e,p}$, using patch descriptors at each initial cluster, $S_{e,p}$. This means that if we have K local patches at each face image and E different expression labels, we start with a $K \times E$ initial set of dictionary blocks. The reason to incorporate the expression labels for structure initialization is that the patches at the same locations but on different expressive faces do not necessarily encode the same semantic knowledge. For example, the patches corresponding to the lip corners are in different locations for Happy and Surprise faces.

As mentioned earlier, we adopt a greedy bottom-up procedure to merge pairs of clusters. At each iteration of the algorithm (before the stopping criterion is met), the cross-residual error between each pair of clusters, C_{rc}, is calculated and then the two clusters with the minimum merging cost, $C_{\tilde{r}\tilde{c}}$, are merged to form a new cluster,

$$\begin{aligned} (\tilde{r}, \tilde{c}) &= \arg\min_{r,c} C_{rc} \\ &= \arg\min_{r,c} \left(\|Y(S_r) - D_c X_{rc}\|_F^2 \right. \\ &+ \left. \|Y(S_c) - D_r X_{cr}\|_F^2 \right) \end{aligned} \tag{5.7}$$

The dictionary for the new cluster is updated using the K-SVD algorithm [142]. The procedure stops when the minimum merging cost of two clusters goes above a threshold. A summary of this algorithm is presented in Algorithm 5. Here $\{y^e_{:,p}\}$ is the feature pool extracted from the p^{th} patch location in all the images with expression label e. Figure 5.5 illustrates the result of applying this algorithm on expressive faces with universal emotions from the CK+ dataset. As it can be seen, the learned structures are almost similar for different expression classes. Now in order to construct the structure-preserving dictionary, we learn the dictionary blocks for the structures on each of the expression classes. Then the final dictionary is formed by putting these dictionary blocks together. In this case the two-layer grouping is formed in the lower layer by grouping the dictionary atoms corresponding to each structure and in the top layer by grouping the structures corresponding to each expression class. We employ this dictionary for the universal expression recognition.

```
Data: features from all images
Result: final structures and dictionaries: ${\{S_s\}, \{D_s\}}$
initialization:    $S_{e,p} \leftarrow \{y^e_{.,p}\}$,    $D_{e,p} \leftarrow$ K-SVD$\big(Y(S_{e,p})\big)$;
while stopping criterion has not been met do
    Cost-Matrix = $\{C_{rc} = \|Y(S_r) - D_c X_{rc}\|_F^2 + \|Y(S_c) - D_r X_{cr}\|_F^2\}$;
    $(\tilde{r}, \tilde{c}) = \arg\min_{r,c}$ Cost-Matrix;
    Update Step:
        $S_{new} \leftarrow$ merge$\{S_{\tilde{r}} \& S_{\tilde{c}}\}$;
        $D_{new} \leftarrow$ K-SVD$\{Y(S_{new})\}$;
        Update Cost-Matrix;
    if $C_{\tilde{r}\tilde{c}} >$ cost-threshold then
        stopping criterion is met;
    end
end
```

Algorithm 5: Summary of the algorithm for structure-preserving dictionary learning.

5.6 Experimental Results

We conducted experiments using two publicly available datasets. The first is the Bosphorus dataset [121] that is composed of a selected subset of AUs as well as the six universal emotion categories: Anger, Disgust, Fear, Happy, Sadness and Surprise, for 105 subjects. For each subject, the neutral face and the face in the apex of various AUs and emotions are presented. Some AUs or emotions are not available for some subjects. The second dataset is the Extended Cohn-

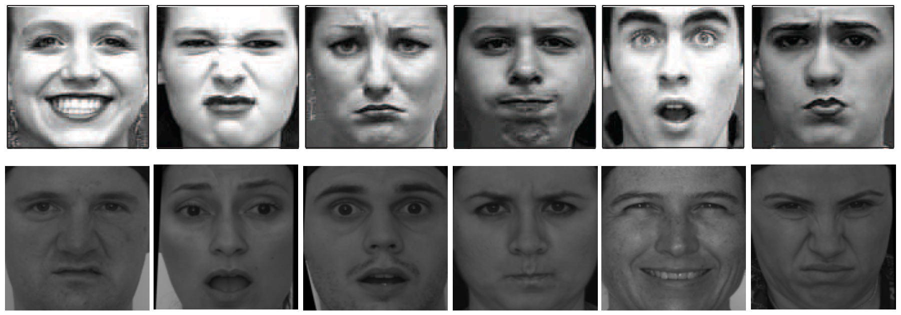


Figure 5.6: Some example of expressive faces in (first row) CK+ and (second row) Bosphorus datasets after the preprocessing step is completed.

Kanade dataset (CK+) [128] which consists of 593 sequences from 123 subjects. The image sequences incorporate the onset (neutral face) to peak formation of facial expressions. Only 327 out of 593 sequences have emotion labels from each of the six universal emotion categories. Again some emotion sequences are not available for some subjects.

In the preprocessing step for the CK+ dataset, we first detect and crop faces at the apex of sequences. Then for both datasets, face images are resized to 128×128 and using the coordinates of eye corners and nose tip (provided by the datasets) the faces are properly aligned. Figure 5.6 shows some examples of faces from both datasets after the preprocessing step is completed.

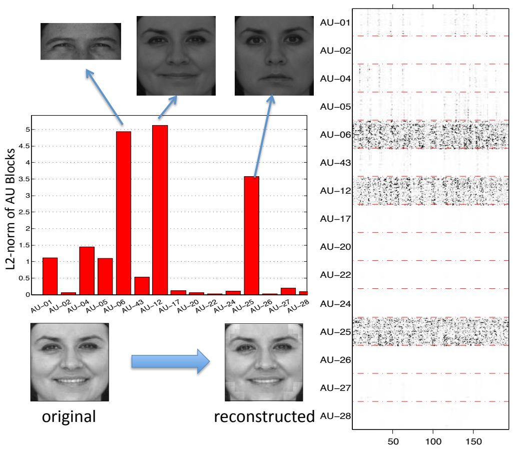
5.6.1 Parameters Setting

Multi-layer, multi-variate group sparse coding has several parameters and it is important to assign appropriate values to them for better performance of the algorithm. We follow the weighting strategy proposed by Chen *et al.* [135] and also adopted by Gao *et al.* [153]. In this strategy the weight for each group is proportional to the square root of the length of the group, so $\omega_g^{dic} = \sqrt{|g|}$ $(g \in G_{dic})$ and $\omega_g^{tst} = \sqrt{|g|}$ $(g \in G_{tst})$. Then we set $\gamma_{dic} = \gamma_{tst} = \lambda = \theta$. In this way there is only one parameter, θ, in the whole objective function. In our experiments we set $\theta = 10^{-3}$.

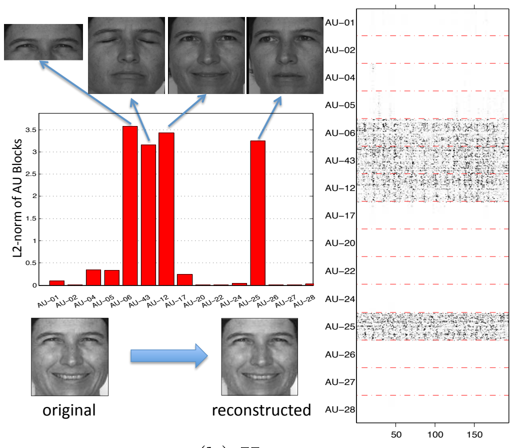
The number of atoms in a dictionary block (AU/structure-block) and the sparsity are other parameters needed by the K-SVD [142] algorithm which is used for data-driven dictionary learning. In our experiments, we learn a dictionary of size $J_i = 20$ per structure/AU and set the sparsity to be half of the dictionary size, i.e. 10. Our experiments show that these choices ensure a good trade-off between the accuracy of the representation and the speed of the algorithm. A larger dictionary (up to a point) may slightly improve the results but at the cost of slower convergence of the sparse coding algorithm.

5.6.2 Expression Decomposition

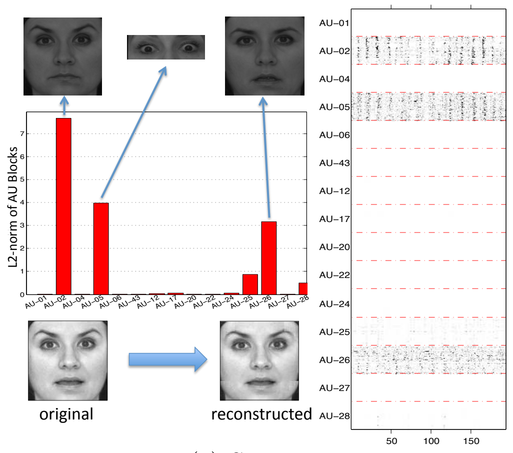
As discussed in Section 5.4.3, the AU-dictionary can be used to decompose an expressive face into its constituent AUs. In fact the main advantage of modeling AUs for expression analysis is that we are not limited to the six universal expressions, and many other expressions that often occur on the face but do not belong to these universal expressions can also be analyzed by predicting the AUs they are composed of. For this experiment, we learn the AU-dictionary using a selected subset of 15 AUs in the Bosphorus dataset. This subset consists of six upper face AUs (1, 2, 4, 5, 6, and 43) and nine lower face AUs (12, 17, 20, 22, 24, 25, 26, 27 and 28). To learn the



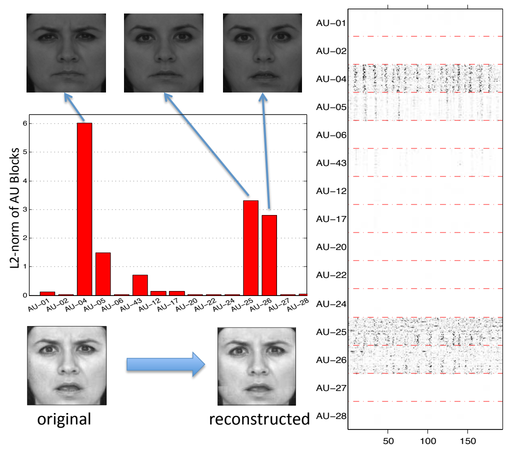
(a) Happy



(b) Happy



(c) Surprise



(d) Fear

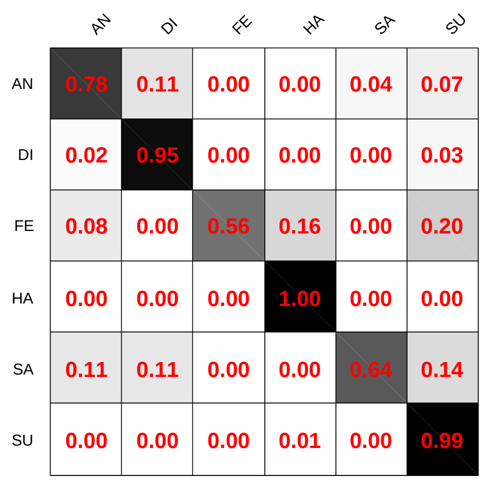
Figure 5.7: Expression decomposition into constituent AUs. Each example includes the matrix of sparse codes, X, on the right column, the AU bar plot which shows the l_2-norm of each AU-block in the sparse code matrix, the sample faces with those AUs that have significant magnitude on top of the bar plot, and finally the original and reconstructed expressive faces below the bar graph. The decompositions reveal the correct constituent AUs for each expressive face.

AU-dictionary, we first delineate the patches that correspond to each AU semantic region. For simplicity we only define seven regions on the face corresponding to brows, eyes, nose, mouth and forehead. It should be noted that AU-5 and AU-6 are not included in the Bosphorus dataset; however, these two AUs occur in many expressions, so we learn their corresponding dictionary blocks using the happy and surprise samples in the dataset. However, we avoid using the same samples for dictionary training and decomposition testing.

Observations: Figure 5.7 shows some examples of expression decomposition we performed using expressive samples from the Bosphorus dataset. The figure shows decomposition results for two happy faces of different subjects. Both decompositions predict AUs-{06+12+25} which is in accordance with the information in Fig. 5.3. However in part (b), AU-43 has also been reported as one of the constituent components of the happy face. This can be due to the extreme smile on the face which makes the eyes look almost closed. Parts (c) and (d) illustrate the decomposition for two expressive faces with Surprise and Fear expressions. The decompositions reveal the correct constituent AUs. It should be noted that in these experiments as well as in the expression synthesis experiments, in order to be able to visualize the reconstructed faces we concatenated the SIFT features extracted from each patch with the intensity difference image (the difference between the neutral face and the expressive face in that patch). So to visualize we add the reconstructed intensity difference feature to the neutral face.

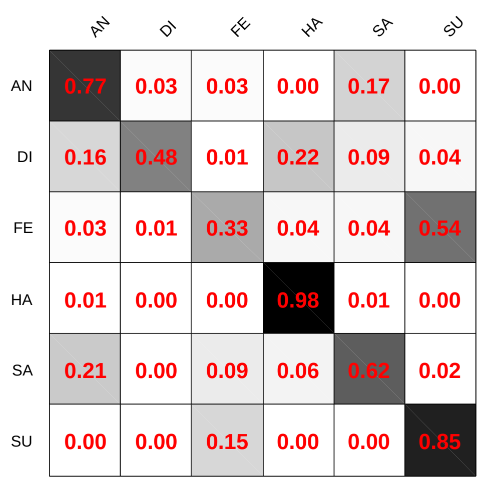
5.6.3 Expression Recognition

In this section we report the results of expression recognition using the learned structure-preserving dictionaries for both the CK+ and Bosphorus datasets. We adopt the leave-one-subject-out cross-validation configuration to maximize the amount of training and testing data. Figure 5.8 shows the confusion matrices for both datasets. It should be noted that for the CK+ dataset we only use the apex of



	AN	DI	FE	HA	SA	SU
AN	0.78	0.11	0.00	0.00	0.04	0.07
DI	0.02	0.95	0.00	0.00	0.00	0.03
FE	0.08	0.00	0.56	0.16	0.00	0.20
HA	0.00	0.00	0.00	1.00	0.00	0.00
SA	0.11	0.11	0.00	0.00	0.64	0.14
SU	0.00	0.00	0.00	0.01	0.00	0.99

(a) CK+



	AN	DI	FE	HA	SA	SU
AN	0.77	0.03	0.03	0.00	0.17	0.00
DI	0.16	0.48	0.01	0.22	0.09	0.04
FE	0.03	0.01	0.33	0.04	0.04	0.54
HA	0.01	0.00	0.00	0.98	0.01	0.00
SA	0.21	0.00	0.09	0.06	0.62	0.02
SU	0.00	0.00	0.15	0.00	0.00	0.85

(b) Bosphorus

Figure 5.8: Confusion matrix for emotion recognition on the (a) CK+ dataset, (b) Bosphorus dataset. The number of samples per classes for the CK+ dataset is: (AN:45, DI:59, FE:25, HA:69, SA:28, SU:79) and for the Bosphorus dataset is: (AN:71, DI:69, FE:70, HA:105, SA:66, SU:71).

each expression sequence for recognition. The average recognition rate on the CK+ dataset is 88.52% and on the Bosphorus dataset is 69.78%. It should be noted that while both the CK+ and the Bosphorus are datasets with posed expressions, but there are main discrepancies in their annotations. The CK+ was first FACS coded manually, then emotion labels were assigned based on FACS rules, while in the Bosphorus the subjects were asked to show the given emotion/AU and hence the emotion labels might not correspond to the given AU combinations. This explains the lower recognition rate on Bosphorus compared to CK+. Also in the Bosphorus dataset, there is a large degree of similarity among faces of some classes (e.g. Fear and Surprise).

To show the importance of incorporating high-level knowledge on grouping AUs, we compare our algorithm with two other algorithms on the CK+ dataset

Table 5.1: Comparing the averaged recognition rate using three algorithms on CK+.

Algorithms	**Recognition Rates**
Multi-layer multi-variable grouping	88.52%
Multi-layer single grouping	80.2%
Simple Lasso	69.68%

(Table 5.1). In the first algorithm, multi-layer single grouping [135], we removed the grouping information on the image matrix and so the sparse representation for each local descriptor extracted from the test face is obtained independent of others. In the second algorithm both grouping information on the dictionary and image matrix are ignored and a simple LASSO algorithm [99] is applied to learn sparse representation for each local descriptor. As Table 5.1 shows, removing each of the high-level grouping information decreases the recognition rate. These results again emphasize the importance of incorporating the high-level information for expression analysis.

Table 5.2 compares the average recognition rate of our algorithm with some recent advances in expression recognition for the CK/CK+ datasets. The CK dataset [122] is the old version of CK+ which has fewer subjects and sequences. Most of these algorithms did not follow leave-one-subject-out validation, but instead they divided the dataset randomly into training and testing parts and reported the results on the testing part. As the table shows, our result is comparable to the best reported results. However, it should be noted that as our algorithm is a generative approach, it is expected that its classification performance be lower than fully discriminative methods like Adaboost. But the proposed approach can also perform unknown expression decomposition and new expression synthesis which are

Table 5.2: Comparison with recent advances in expression recognition on CK dataset.

Recognition Methods	**Recognition Rates**
CERT [136] *	87.21%
Gabor+Adaboost+SVM [154]	93.3%
PGKNMF‡ [155]	83.5%
CAPP* +SVM [128] †	86.48%
RegRankBoost [147]	88%
Combined Features+Adaboost [134]	92.3%
Our approach	88.52%

$*$ results on 26 subjects in CK+ but not in CK

$\ddagger$ PGKNMF = Projected Gradient Kernel Non-negative Matrix Factorization

$\star$ CAPP = Canonical Appearance Features

$\dagger$ results on CK+ with leave-one-subject-out validation

not possible with pure discriminative algorithms. Moreover, our algorithm provides a general framework for decomposing an action in terms of its constituent basic components. Using a better feature representation for expressive faces as well as a machine learning algorithm for learning over the sparse code matrix, we can expect a higher recognition rate.

5.6.4 Expression Synthesis

Combining different AUs can result in new expressions on the face. Using the AU-dictionary, we perform this experiment to generate valid expressions through composition of some AUs. For this purpose, first we need valid models for sparse coefficients corresponding to each AU. We obtain these models for each subject by decomposing the sample of a particular AU onto the AU-dictionary. Then by selecting a subset of these AUs and assigning values to the corresponding coefficients in the sparse matrix we are able to synthesize new expressions. However, it

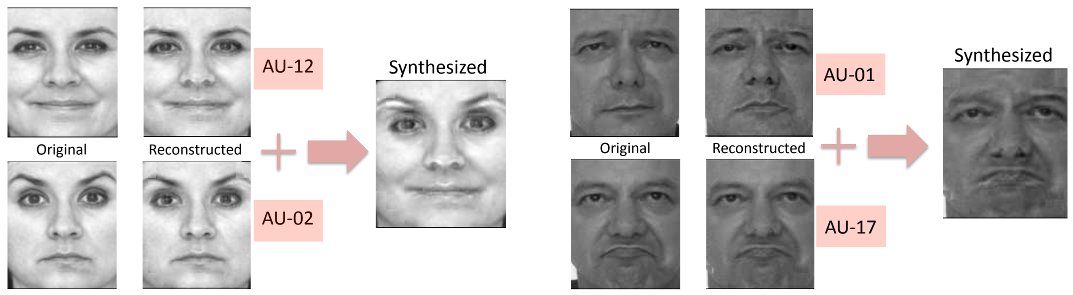


Figure 5.9: Expression synthesis using the AU-dictionary.

should be noted that this approach does not work for non-additive AUs (discussed in Section. 5.3.2). Figure 5.9 illustrates two examples of expression synthesis.

5.7 Summary

We presented a dictionary-based approach for facial expression analysis. Using the domain experts' knowledge provided in FACS, we learned an AU-dictionary. We also proposed an automatic algorithm to learn a structure-preserving dictionary without incorporating the experts' knowledge. Then the high-level knowledge regarding the AUs/structures composition is incorporated into this dictionary through a multi-layer grouping of dictionary atoms. Since we are also dealing with a multi-variable problem, we impose appropriate groupings over the test image matrix columns. We employed a multi-layer, multi-variable group sparse coding algorithm to impose these grouping constraints for structure-preserving sparse coding. This enables us to perform expression decomposition into the constituent AUs for a face with any unknown expression. Using the sparse code matrix we can also perform universal emotion recognition. The results indicate the improvements that this high-level knowledge brings to expression analysis. Moreover, since the proposed algorithm is a generative approach, we can also perform new expression synthesis. We show some preliminary results for expression synthesis.

This chapter presented a general framework for decomposing an action into its constituent components. We showed the potential of this algorithm for facial expression analysis, including expression decomposition, synthesis and recognition. The proposed algorithm can be generalized to recognition of human actions provided we have a good definition for human action units. This algorithm can be further improved by adding temporal sequence information as a new grouping layer. Incorporating temporal information enables us to predict the intensity of AUs and detect AUs that are combined sequentially to form an expression/action.

Chapter 6: Component-based Recognition of Faces and Facial Expressions

6.1 Introduction

Facial expressions arise owing to a person's internal emotional states, intentions, or social communications. On the one hand, these facial changes present important challenges for face recognition algorithms, where researchers are proposing various expression-invariant face recognition algorithms. On the other hand, these facial changes are the best cues for recognizing facial expressions. Understanding the users' emotions is a fundamental requirement of human-computer interaction systems (HCL) and facial expressions are important means of detecting emotions.

Many effective algorithms have been proposed for expression-invariant face recognition as well as facial expression recognition. While the main focus of expression-invariant face recognition algorithms is to mitigate the changes related to facial expressions [156–161], the goal of facial expression analysis algorithms is to automatically analyze and recognize facial motions and facial feature changes from visual information [6, 131, 134, 141]. Despite the connections between these two problems, there are only a few works that jointly address them.

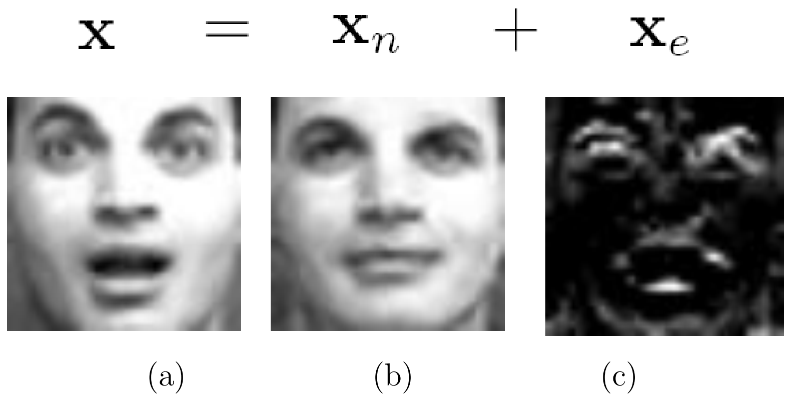


Figure 6.1: Facial component separation. Original face image (a) is viewed as the superposition of a neutral component (b) with a component containing the expression (c).

Proposed algorithms for joint face and facial expression recognition usually encode the identity and expression variability of faces in independent control parameters which are then used for recognition [162,163]. One popular class of algorithms is the bilinear model proposed by Tanenbaum *et al.* [164] and its generalization, tensor decomposition which offer an efficient way for modeling the bi-factor or multi-factor interactions. These algorithms have motivated some interesting face decomposition ideas [165–168]. After separating the identity and expression components of a face, joint expression-invariant face recognition and identity-independent expression recognition is achieved.

Motivated by the success of these bilinear/multilinear models for the decomposition of expressive faces, we propose a similar facial component separation algorithm based on the principle of sparse representation. We model an expressive face as a neutral face superimposed by a sparse image of deformations corresponding to the expression on the face (see Figure 6.1). Using this model, we propose a component separation method so that an expressive face image is decomposed into a sum of

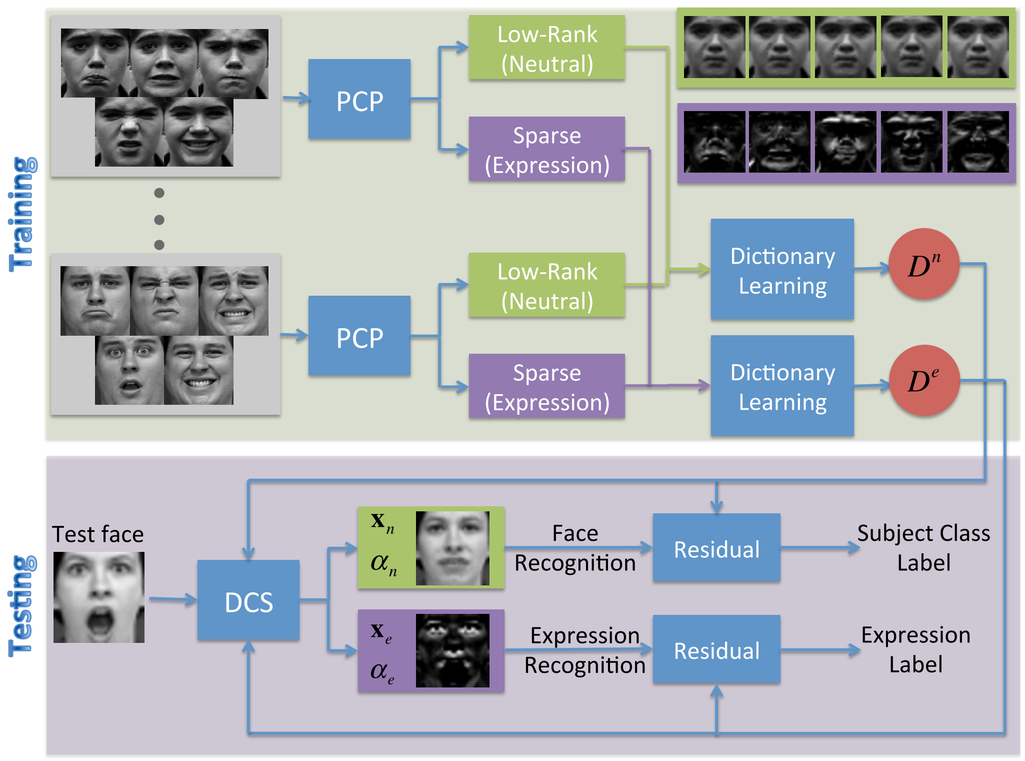


Figure 6.2: Overview of the proposed algorithm.

neutral and expression elements. Our formulation is based on finding sparse representations of these elements using dictionaries specifically suited to sparsify them. By taking advantage of sparse representations, we retain important salient features in the final estimated images. These separated components are then used for joint face and expression recognition.

Figure 6.2 illustrates an overview of the proposed joint face and expression recognition algorithm. In the training phase, the data-driven dictionaries are generated. Having multiple samples of expressive faces per subject [1], we first obtain the neutral and expressive parts by decomposing the training images into sparse and low-rank components using the well-known Principle Component Pursuit (PCP) algorithm [8]. We then find the best representation for each member in these com-

[1]neutral faces might be available as well

ponents by learning subject-specific dictionaries under strict sparsity constraints. In the testing phase, the expressive test face is first decomposed into its building components using the proposed dictionary-based component separation algorithm (DCS). The separated components along with the corresponding dictionaries are then used for recognition.

Contributions: Our main contributions are,

- We propose a dictionary-based component separation algorithm to decompose an expressive face into a neutral and an expression part.

- We propose a joint face and expression recognition scheme using the separated components, dictionaries and sparse coefficient vectors.

Outline of the chapter: We review related work in Section 6.2. The dictionary-based component separation algorithm is then presented in Section 6.3. Section 6.4 discusses how the joint recognition of face and expression is achieved using these separated components. Experimental results are presented in Section 6.5.

6.2 Related Work

Face and facial expression recognition have long been topics of interest for computer vision researchers. Face recognition is affected by different sources of variation such as nonrigid deformations due to facial expressions. Therefore, several algorithms have been proposed that offer expression-invariance into face recognition. These algorithms treat nonrigid deformations as noise and try to mitigate/remove their effects on the identity recognition. On the other hand, these nonrigid deformations on the faces are the main cues for facial expression analysis and recognition. The proposed algorithms for expression recognition consider the identity of faces as noise and attempt to make the algorithms independent of the face identity. The obvious connection between these two problems is the main motivation for joint recognition of faces and facial expressions[2]. Here we review some of the proposed algorithms for the individual problems as well as for jointly addressing these two related problems.

6.2.1 Expression-Invariant Face Recognition

A comprehensive review of the related works on face recognition can be found in [156, 169]. It has been noticed that facial expression usually affects the performance of face recognition algorithms. To deal with it several expression-invariant face recognition algorithms have been proposed. Expression-invariant face recognition is a challenging task owing to complex and varied nature of facial expressions. Some systems use video sequences to handle a wide range of facial expressions, since a video sequence contains more information than a single image [170, 171].

Image-based algorithms for expression-invariant face recognition can be categorized as subspace-based and optical flow-based approaches. Tsai *et al.* [161] provided a subspace-based analysis for face recognition which showed robustness to facial deformations. In [158], Naseem *et al.* used the concept that patterns from a single-object class lie on a linear subspace and proposed a linear regression classifi-

[2]It should be noted that here joint recognition implies that having an expressive test face we find both the face identity and the expression label. So, it is not the case that the solution to one problem helps to solve the other problem

cation algorithm in which a linear model was developed representing a probe image as a linear combination of class-specific galleries. Amberg *et al.* [159] proposed an expression-invariant algorithm for face recognition by fitting an identity/expression separated 3D morphable model to shape data.

Methods based on optical flow can establish a dense correspondence between pairs of faces and compute the face warping transformation. Martinez [172] used the optical flow between testing and sample images as a measure of how good each pixel is for face recognition. They proposed a weighting measure which gives more importance to the facial regions that are more similar between training and testing images and less importance to the less similar areas. Hsieh *et al.* [173] modified the regularization-based optical flow algorithm by imposing constraints on some given point correspondences to compute precise pixel displacement and intensity variation. They removed the expression from the face image by elastic image warping to recognize the subject with facial expression. Jorstad *et al.* [157] proposed a deformation as well as lighting insensitive metric to compare images and presented a framework to optimize over this metric for calculating a dense flow between images.

Recently, face recognition algorithms based on ideas of sparse representation and compressive sensing have been proposed [174,175] that are able to handle changing expression, pose and illumination. Nagesh *et al.* [160] proposed an expression-invariant face recognition algorithm based on compressive sensing. They represented the training images of a given subject using two feature images: one that captures the common features of the faces and one that captures the different expressions in all training images. Assuming that changes due to variation in expressions are sparse with respect to the whole image, they approximated the test image using these two feature images of the same subject.

6.2.2 Identity-Independent Expression Recognition

The goal of facial expression analysis is to create systems that can automatically analyze facial feature changes and map them to facial expressions. Since different faces have different structures and appearances, the main challenge for such systems is to be independent of the identity of the faces. There are several past and current efforts in modeling and recognizing the nonrigid deformations in facial components as a result of various expressions. Some of these approaches extract a set of appearance-based features, including raw gray scale intensities [140, 176], Gabor features [105, 106, 145, 177], Haar-like features [134, 147] and local binary patterns (LBP) [176, 178] from a still image or use some rules based on the deformation of local facial components and shape-based features [107, 132, 141, 179, 180] and then employ a machine learning approach, such as Adaboost [134], SVM [6, 181] and Neural Nets [182] for expression recognition.

Tong *et al.* [105, 106] proposed a unified probabilistic framework based on the dynamic Bayesian network to simultaneously and coherently represent the non-rigid motions and their image observations, as well as to capture the temporal evolution of the facial activities. Yang *et al.* [134] interpreted facial expressions by learning some compositional features based on local appearance changes on the face. They interpreted facial expressions by learning these compositional appearance features and showed the consistency of the built compositional features with respect to the action units (AUs).

Active shape models (ASM) and active appearance models (AAM) [183, 184], are used to approximate deformable face models using linear subspace analysis [6]. However, linear subspace methods are inadequate to represent the underlying structure of real data and so nonlinear manifold learning approaches have been proposed. Liao *et al.* [107] decomposed each expression using a basis of eight one-dimensional manifolds each learned offline using a tensor voting algorithm and sequences of labeled universal expression.

By observing that images of all possible facial deformations of an individual form a smooth manifold embedded in a high dimensional image space, Chang *et al.* [140] proposed a probabilistic video-based facial expression recognition method on manifolds. They represented a complete expression sequence as a path on the expression manifold and used a probabilistic model to perform expression recognition. Taheri *et al.* [141] modeled expressions as geodesic pathways on the Grassmann manifold. This representation enables expression models to be generalized across view changes.

Sparse representation have also been applied to facial expression recognition [145, 176]. Ying *et al.* [176] designed two classifiers in the sparse domain using two different sets of image features: raw gray scale pixel values and local binary patterns. The final expression recognition was then performed by fusing the results of the two classifiers. Mahoor *et al.* [145] presented a sparse learning approach for AU combination classification. They developed an overcomplete dictionary to efficiently and robustly recognize the combination of facial AUs using L1-norm minimization.

6.2.3 Joint Face and Facial Expression Recognition

As we mentioned earlier, expression-invariant face recognition and identity-independent expression recognition are two related problems and can be considered jointly. There are a few works that addressed this joint recognition problem. Colmenarez *et al.* [162] proposed a Bayesian framework for face and facial expression recognition. The algorithm finds a face model and facial expression that maximizes the likelihood of a given test image. Li *et al.* [163] employed the idea of separating geometry and texture information in a face image to perform joint face and expression recognition. They modeled the two types of information by projecting them into separate Principal Component Analysis (PCA) spaces which were specially designed to capture the distinctive features among different individuals. While a combination of texture and geometry attributes were employed for face recognition, study of geometry enabled expression classification. The geometry and texture separation was performed by fitting a generic mask to the face and warping the texture on it.

Vasilescu *et al.* [165] used a tensor decomposition known as the N-mode Singular Value Decomposition (SVD) to separate the identity, pose, illumination and expression of a given face. Wang *et al.* [166] used Higher-Order Singular Value Decomposition (HOSVD) to model the mapping between individuals and expressions. They learned the expression subspace and person subspace from a corpus of images and performed simultaneous face and expression recognition as a result of facial expression decomposition. A similar idea has also been applied to 3D faces. Mpiperis *et al.* [167] used a bilinear model to express the 3D facial surface as the interaction of expression and identity components. By fitting an elastically deformable model to unknown faces and decoupling the identity and expression they performed face recognition invariant to facial expressions and facial expression recognition with unknown identity. Lee *et al.* [168] learned nonlinear mappings between a conceptual embedding space and facial expression image space and decomposed the mapping space using multilinear analysis.

In this work, we decompose an expressive face into neutral and expression components and then perform joint face and expression recognition. After obtaining the individual components, we can extract any feature from them and apply any algorithm for face/expression recognition. However, since the proposed algorithm is based on sparsity ideas and we need to obtain neutral and expression dictionaries in the training phase, we adopt the sparse representation-based recognition algorithm [174] while taking the special structures of the dictionaries into account.

6.3 Decomposition of Expressive Faces

In this section, we present our proposed Dictionary-based Component Separation (DCS) algorithm for decomposing an expressive face into neutral and expression parts.

6.3.1 Dictionary-based Separation of Expressive Face Components

We identify an $l \times q$ grayscale face image as an N-dimensional vector, $\mathbf{x}$, which can be obtained by stacking its columns, where $N = l \times q$. A face image $\mathbf{x}$ containing an expression can be viewed as a superposition of a neutral face component $\mathbf{x}_n$ with a facial expression component $\mathbf{x}_e$. In other words

$$\mathbf{x} = \mathbf{x}_n + \mathbf{x}_e. \tag{6.1}$$

We assume that $\mathbf{x}_n$ is sparse in a dictionary $\mathbf{D}_n$, and similarly, $\mathbf{x}_e$ is sparse in a dictionary $\mathbf{D}_e$. Given $M_n, M_e \geq N$, the dictionaries $\mathbf{D}_n \in \mathbb{R}^{N \times M_n}$ and $\mathbf{D}_e \in \mathbb{R}^{N \times M_e}$ are chosen such that they provide sparse representations of neutral and expression contents, respectively. That is, we assume there are sparse coefficient vectors $\alpha_n \in \mathbb{R}^{M_n \times 1}$ and $\alpha_e \in \mathbb{R}^{M_e \times 1}$ so that $\mathbf{x}_n = \mathbf{D}_n \alpha_n$ and $\mathbf{x}_e = \mathbf{D}_e \alpha_e$.

One can recover the face image $\mathbf{x}$ by estimating the components $\mathbf{x}_n$ and $\mathbf{x}_e$ via α_n and α_e by solving the following optimization problem:

$$\begin{aligned}\hat{\alpha}_n, \hat{\alpha}_e = \arg \min_{\alpha_n, \alpha_e} & \lambda \|\alpha_n\|_1 + \lambda \|\alpha_e\|_1 \\ & + \frac{1}{2} \|\mathbf{x} - \mathbf{D}_n \alpha_n - \mathbf{D}_e \alpha_e\|_2^2, \end{aligned} \tag{6.2}$$

where for an N-dimensional vector $\mathbf{x}$, $\|.\|_q$ denotes the ℓ_q-norm, $0 < q < \infty$, defined as

$$\|\mathbf{x}\|_q = \left(\sum_{i=1}^{N} |x_i|^q \right)^{\frac{1}{q}}.$$

The two components are the corresponding representations of the two parts and can be obtained by $\hat{\mathbf{x}}_n = \mathbf{D}_n \hat{\alpha}_n$ and $\hat{\mathbf{x}}_e = \mathbf{D}_e \hat{\alpha}_e$. This notion of separating an image into different morphologies using sparse representations is often known as Morphological Component Analysis (MCA) [185]. Figure 6.2 shows an example of this separation in the testing part of the algorithm.

6.3.2 Iterative Shrinkage Algorithm

Various methods can be used to obtain the solution of (6.2) [186, 187]. In this section, we derive a fast convergent iterative shrinkage algorithm based on Separable Surrogate Functionals (SSF) to solve the separation problem posed in (6.2) [187–189]. For simplicity, we assume that $\mathbf{D} = [\mathbf{D}_n, \mathbf{D}_e]$. The objective function in (6.2) can then be re-written as

$$f(\alpha) = \lambda \|\alpha\|_1 + \frac{1}{2}\|\mathbf{x} - \mathbf{D}\alpha\|_2^2 \tag{6.3}$$

where α contains both the neutral and expression parts. Let

$$d(\alpha, \alpha_0) = \frac{c}{2}\|\alpha - \alpha_0\|_2^2 - \frac{1}{2}\|\mathbf{D}\alpha - \mathbf{D}\alpha_0\|_2^2, \tag{6.4}$$

where α_0 is an arbitrary vector of length N and the parameter c is chosen such that d is strictly convex. This constraint is satisfied by choosing

$$c > \|\mathbf{D}^T\mathbf{D}\|_2 = \lambda_{\max}(\mathbf{D}^T\mathbf{D}),$$

where $\lambda_{\max}(\mathbf{D}^T\mathbf{D})$ is the maximal eigenvalue of the matrix $\mathbf{D}^T\mathbf{D}$.

Adding (6.4) to (6.3) gives the following surrogate function

$$f(\alpha) = \lambda\|\alpha\|_1 + \frac{1}{2}\|\mathbf{x} - \mathbf{D}\alpha\|_2^2 + \frac{c}{2}\|\alpha - \alpha_0\|_2^2 - \frac{1}{2}\|\mathbf{D}\alpha - \mathbf{D}\alpha_0\|_2^2. \tag{6.5}$$

This surrogate function $\tilde{f}(\alpha)$ can be rewritten as

$$\tilde{f}(\alpha) = A + \frac{\lambda}{c}\|\alpha\|_1 + \frac{1}{2}\|\alpha - \mathbf{x}_0\|^2, \tag{6.6}$$

where

$$\mathbf{x}_0 = \frac{1}{c}\mathbf{D}^T(\mathbf{x} - \mathbf{D}\alpha_0) + \alpha_0$$

and A is some constant. Let $(a)_+$ denote the function $\max(a, 0)$ and sign(x) be the signum function defined as

$$\operatorname{sign}(x) = \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{for } x = 0 \\ 1 & \text{for } x > 0. \end{cases}$$

Given that

$$\mathcal{S}_\lambda(\mathbf{x}) = \operatorname{sign}(\mathbf{x})(|\mathbf{x}| - \lambda)_+ \tag{6.7}$$

is the element-wise soft-thresholding operator with threshold λ, the global minimizer of the surrogate function is given by

$$\begin{aligned}\alpha_{sol} &= \mathcal{S}_{\lambda/c}(\mathbf{x}_0)\\ &= \mathcal{S}_{\lambda/c}\left(\frac{1}{c}\mathbf{D}^T(\mathbf{x}-\mathbf{D}\alpha_0)+\alpha_0\right). \end{aligned} \tag{6.8}$$

It was shown in [188] that the iterations

$$\alpha^{k+1} = \mathcal{S}_{\lambda/c}\left(\frac{1}{c}\mathbf{D}^T(\mathbf{x}-\mathbf{D}\alpha^k)+\alpha^k\right) \tag{6.9}$$

converge to the minimizer of the function f in (6.3) where the superscript k indicates that it is the value for the kth iteration. We decompose the above iteration into sub-iterations that essentially solve (6.2),

$$\alpha_n^{k+1} = \mathcal{S}_{\lambda/c}\left(\frac{1}{c}\mathbf{D}_n^T(\mathbf{x}-\mathbf{D}_n\hat{\alpha}_n^k-\mathbf{D}_e\hat{\alpha}_e^k)+\hat{\alpha}_n^k\right) \tag{6.10}$$

$$\alpha_e^{k+1} = \mathcal{S}_{\lambda/c}\left(\frac{1}{c}\mathbf{D}_e^T(\mathbf{x}-\mathbf{D}_n\hat{\alpha}_n^k-\mathbf{D}_e\hat{\alpha}_e^k)+\hat{\alpha}_e^k\right). \tag{6.11}$$

The algorithm for recovering the two separated components of a facial image by minimizing (6.2) is summarized in Figure 6.3. In step 3 of the algorithm in Figure 6.3, $\|.\|_\infty$ denotes the ℓ_∞-norm. For an N dimensional vector $\mathbf{x}$, it is defined as $\|\mathbf{x}\|_\infty = \max(|x_1|, \cdots, |x_N|)$.

Once the representation coefficients of the two components of $\mathbf{x}$ are estimated, we obtain the final estimate of $\mathbf{x}_n$ and $\mathbf{x}_e$ as $\hat{\mathbf{x}}_n = \mathbf{D}_n\hat{\alpha}_n$ and $\hat{\mathbf{x}}_e = \mathbf{D}_e\hat{\alpha}_e$, respectively.

```
**Input:** $\mathbf{x}, c$.
**Initialization:** Initialize $k = 1$ and set
$\alpha_n^0 = \mathbf{0}$, $\alpha_e^0 = \mathbf{0}$ and $\mathbf{r}^0 = \mathbf{x} - \mathbf{D}_n\alpha_n^0 - \mathbf{D}_e\alpha_e^0$, and $\lambda^0 = \frac{1}{2}\left(\|\mathbf{D}_n^T\mathbf{x}\|_\infty + \|\mathbf{D}_e^T\mathbf{x}\|_\infty\right)$.
**repeat:**
1. Update the estimate of $\alpha_n$ and $\alpha_e$ as
       $\alpha_n^k = \mathcal{S}_{\lambda^k}\left(\frac{1}{c}\mathbf{D}_n^T(\mathbf{r}^{k-1}) + \alpha_n^{k-1}\right)$
       $\alpha_e^k = \mathcal{S}_{\lambda^k}\left(\frac{1}{c}\mathbf{D}_e^T(\mathbf{r}^{k-1}) + \alpha_e^{k-1}\right)$.
2. Update the residual as
       $\mathbf{r}^k = \mathbf{x} - \mathbf{D}_n\alpha_n^k - \mathbf{D}_e\alpha_e^k$.
3. Update the shrinkage parameter as
       $\lambda^k = \frac{1}{2}\left(\|\mathbf{D}_n^T\mathbf{r}^k\|_\infty + \|\mathbf{D}_e^T\mathbf{r}^k\|_\infty\right)$.
**until:** stopping criterion is satisfied.
**Output:** The two representation vectors $\hat{\alpha}_n = \alpha_n^k$ and $\hat{\alpha}_e = \alpha_e^k$.
```

Figure 6.3: The DCS iterative shrinkage algorithm to solve (6.2).

6.3.3 Obtaining Initial Dictionaries

Finding dictionaries that represent the neutral and expression components of faces is critical as it affects how well the components are separated through successive iterations. In this section, we propose a method based on sparse and low-rank approximation to determine the initial dictionaries.

Suppose that we are given C distinct subject classes and a set of m training images per class containing various expressions. The expression label for each training image is known and is one of the E available expression classes. Let

$$\mathbf{B}_i = [\mathbf{x}_i^1, \cdots, \mathbf{x}_i^m] \in \mathbb{R}^{N\times m} \tag{6.12}$$

be an $N \times m$ matrix of training images corresponding to the ith subject class. Using (6.1), the training matrix $\mathbf{B}_i$ in (6.12) can be written as

$$\mathbf{B}_i = \mathbf{B}_i^n + \mathbf{B}_i^e, \tag{6.13}$$

where $\mathbf{B}_i^n = [\mathbf{x}_{in}^1, \cdots, \mathbf{x}_{in}^m]$ and $\mathbf{B}_i^e = [\mathbf{x}_{ie}^1, \cdots, \mathbf{x}_{ie}^m]$ are the matrices containing the neutral and expression face components, respectively. Since the neutral components $\{\mathbf{x}_{in}^j\}$ are common parts in all the training images of a given subject, these components are very similar to each other[3], therefore $\mathbf{B}_i^n$ is a low-rank matrix. On the other hand, the facial expression components $\{\mathbf{x}_{ie}^j\}$ are often sparse as they contain small deformations that are applied to the neutral face components $\{\mathbf{x}_{in}^j\}$ which generate expressive face images, $\{\mathbf{x}_i^j\}$. As a result, $\mathbf{B}_i^e$ is a sparse matrix.

Given a matrix $\mathbf{B}_i$, one can decompose it into a low-rank matrix $\mathbf{B}_i^n$ and a sparse matrix $\mathbf{B}_i^e$ by solving the following optimization problem

$$\begin{aligned}(\hat{\mathbf{B}}_i^n, \hat{\mathbf{B}}_i^e) = \min_{\mathbf{B}_i^n, \mathbf{B}_i^e} \operatorname{rank}(\mathbf{B}_i^n) + \eta \|\mathbf{B}_i^e\|_0 \\ s.t.\ \ \mathbf{B}_i = \mathbf{B}_i^n + \mathbf{B}_i^e,\end{aligned} \tag{6.14}$$

where $\|\mathbf{A}\|_0$ denotes the ℓ_0 norm which counts the number of non-zero entries in $\mathbf{A}$ and $\eta > 0$ is a parameter that trades off the rank of the solution $\mathbf{B}_i^n$ versus the sparsity of $\mathbf{B}_i^e$. Various methods have been proposed in the literature that allow one to solve the above optimization problem [8, 160, 190]. In this work, we adopt the

[3]except some possible variations due to slight misalignment between images and other changes such as illumination variation

PCP algorithm [8] to solve (6.14). In the training part of Figure 6.2, we show some sample images from the output of the PCP algorithm. Note that the low-rank part essentially captures the common structure (neutral part) of training samples and the sparse component captures various expressions that are present in the training set.

6.3.4 Component Dictionary Learning

Let

$$\mathbf{B}^n = [\mathbf{B}_1^n, \mathbf{B}_2^n, \cdots, \mathbf{B}_C^n] \in \mathbb{R}^{N \times mC}$$

and

$$\mathbf{B}^e = [\bar{\mathbf{B}}_1^e, \bar{\mathbf{B}}_2^e, \cdots, \bar{\mathbf{B}}_E^e] \in \mathbb{R}^{N \times \bar{m}E}$$

be the concatenation of neutral and expression component matrices, respectively. It should be noted that for the expression matrix, $\mathbf{B}^e$, we also permute the elements of matrix and put the elements with the same expression labels together. Hence, while $\mathbf{B}_i^n$ has the neutral components of the ith subject images, $\bar{\mathbf{B}}_k^e$ has the expression components of the images with the kth expression label (there are $\bar{m}$ of such images per expression and $mC = \bar{m}E$).

Then, one can find the best representation for each member in $\mathbf{B}^n$ under strict sparsity constraints by solving the following optimization problem per subject

$$(\hat{\mathbf{D}}_i^n, \hat{\mathbf{\Gamma}_i^n}) = \arg\min_{\mathbf{D}_i^n, \mathbf{\Gamma}_i^n} \|\mathbf{B}_i^n - \mathbf{D}_i^n \mathbf{\Gamma}_i^n\|_F^2$$

$$s.t. \ \forall l \quad \|\boldsymbol{\gamma}_l^n\|_0 \leq T_0, \tag{6.15}$$

where $\boldsymbol{\gamma}_l^n$, $l \in \{1, \cdots, m\}$ represents a column of $\mathbf{\Gamma}_i^n$ and T_0 is a sparsity parameter. Here, the Frobenius norm of a matrix $\mathbf{A} \in \mathbb{R}^{M \times N}$ is defined as

$$\|\mathbf{A}\|_F = \left[\sum_{i=1}^{M} \sum_{j=1}^{N} |\mathbf{A}^2(i,j)|\right]^{\frac{1}{2}}.$$

Similarly, one can find the best representation for each member in $\mathbf{B}^e$ by solving

$$(\hat{\mathbf{D}}_k^e, \hat{\mathbf{\Gamma}_k^e}) = \arg\min_{\mathbf{D}_k^e, \mathbf{\Gamma}_k^e} \|\bar{\mathbf{B}}_k^e - \mathbf{D}_k^e \mathbf{\Gamma}_k^e\|_F^2$$

$$s.t. \ \forall l \quad \|\boldsymbol{\gamma}_l^e\|_0 \leq T_0, \tag{6.16}$$

where $\boldsymbol{\gamma}_l^e$, $l \in \{1, \cdots, \bar{m}\}$ represents a column of $\mathbf{\Gamma}_k^e$. Then, final dictionaries are defined as

$$\hat{\mathbf{D}}^n = [\hat{\mathbf{D}}_1^n, \cdots, \hat{\mathbf{D}}_C^n]$$

and

$$\hat{\mathbf{D}}^e = [\hat{\mathbf{D}}_1^e, \cdots, \hat{\mathbf{D}}_E^e].$$

One of the most well-known algorithms for learning such dictionaries is the K-SVD algorithm [191]. The K-SVD algorithm alternates between sparse-coding and dictionary update steps. First, a dictionary $\mathbf{D}$ with ℓ_2 normalized columns is initialized. Then, the main iteration is composed of the following two stages:

- *Sparse coding*: In this step, $\mathbf{D}$ is fixed and the following optimization problem is solved to compute the representation vector $\boldsymbol{\gamma}_l$ for each example $\mathbf{x}_l$, $l \in \{1, \cdots, m\}$ or $l \in \{1, \cdots, \bar{m}\}$, i.e.

$$\min_{\boldsymbol{\gamma}_l} \|\mathbf{x}_l - \mathbf{D}\boldsymbol{\gamma}_l\|_2^2 \text{ s. t. } \|\boldsymbol{\gamma}_l\|_0 \leq T_0.$$

  Since the above problem is NP-hard, approximate solutions are usually sought. Any standard technique [192] can be used but a greedy pursuit algorithm such as orthogonal matching pursuit [193] is often employed due to its efficiency.

- *Dictionary update*: In this stage, the dictionary update is performed atom-by-atom in an efficient way. It has been observed that the K-SVD algorithm converges in a few iterations.

Due to its simplicity and efficiency, we adapt the K-SVD algorithm to learn dictionaries for the neutral and expression components.

6.4 Joint Face/Expression Recognition

Given an expressive test face, $\mathbf{x}_t$, we first decompose it into neutral and expressive components using the DCS algorithm proposed in Section 6.3. Then an obvious next step is to use the sparse coefficient vectors, α_n, α_e, obtained for the neutral and expression parts of the face to perform joint face and expression recognition. Since the subject labels for the $\mathbf{D}^n$ atoms are available from the training images, we predict the subject class label for $\mathbf{x}_t$ based on the reconstruction error of $\mathbf{x}_n$ using the dictionary $\mathbf{D}^n$ as

$$c_s = \arg\min_{c_s} \|\mathbf{x}_n - \mathbf{D}^n \delta_{c_s}(\alpha_n)\|, \tag{6.17}$$

where $\delta_{c_s}(\alpha_n)$ is obtained by setting all the coefficients in α_n to zeros except those having the c_sth subject label. So we classify $\mathbf{x}_t$ by assigning it to the class c_s that has the lowest residual error.

The same procedure can be used for expression recognition. Now having the expression labels for the $\mathbf{D}^e$ atoms, we predict the expression label for $\mathbf{x}_t$ based on the reconstruction error of $\mathbf{x}_e$ using the dictionary $\mathbf{D}^e$ as follow

$$c_e = \arg\min_{c_e} \|\mathbf{x}_e - \mathbf{D}^e \delta_{c_e}(\alpha_e)\| \tag{6.18}$$

where $\delta_{c_e}(\alpha_e)$ is obtained by setting all the coefficients in α_e to zero except those having the c_eth expression label. So we assign $\mathbf{x}_t$ to the expression class c_e that has the lowest residual error.

While the proposed recognition procedure is effective in using the sparse codes obtained from the face decomposition step, it ignores the structural information available for the dictionaries. Since the label information is available for $\mathbf{D}^n$ and $\mathbf{D}^e$, we can obtain new sparse codes for each of the extracted components by projecting it into its corresponding dictionary while using the group structure of the dictionary as a constraint for the sparse decomposition. Thus, the obtained sparse codes are optimized for the recognition purpose.

Having the extracted neutral component of the test face, $\mathbf{x}_n$, and the neutral dictionary, $\mathbf{D}^n$, we want to sparsely decompose $\mathbf{x}_n$ using $\mathbf{D}^n$ having the dictionary subject labels as constraint. If $\mathbf{x}_t$ belongs to c_sth subject, our goal is to force the new sparse code, β_n, to be zero everywhere except for those coefficients having the c_sth subject label (corresponding to the dictionary atoms with the c_sth subject label). For this purpose, we perform the following optimization,

$$\min_{\beta_n} \frac{1}{2}\|\mathbf{x}_n - \mathbf{D}^n\beta_n\|_2^2 + \lambda_1\|\beta_n\|_1 + \lambda_2\sum_{g=1}^{C} w_g^n\|\beta_{nG_g}\|_2, \tag{6.19}$$

where β_n is divided into C non-overlapping groups $\beta_{nG_1}, \beta_{nG_2}, ..., \beta_{nG_C}$. The grouping is based on the subject labels of the dictionary atoms and C is the total number of subjects available in the gallery. λ_1, λ_2 and $\{w_g^n\}'s$ are weighting coefficients that are set appropriately. This objective function is similar to the sparse group Lasso penalty [194], so we use a proper implementation of spars group Lasso [195] to optimize this objective function and obtain the sparse code.

The new sparse code corresponding to the expression component is also obtained using the same approach and by optimizing the following objective function,

$$\min_{\beta_e} \frac{1}{2}\|\mathbf{x}_e - \mathbf{D}^e\beta_e\|_2^2 + \lambda_1\|\beta_e\|_1 + \lambda_2\sum_{g=1}^{E} w_g^e\|\beta_{eG_g}\|_2, \tag{6.20}$$

Given a test sample $\mathbf{x}_t$ and C training matrices $\mathbf{B}_1, \cdots, \mathbf{B}_C$ where each $\mathbf{B}_i \in \mathbb{R}^{N\times m}$ contains m training samples for ith subject. The expression label for each training image is known and is one of the E available expression classes. There are $\bar{m}$ sample images for each expression label.

Training:
1. Given a matrix $\mathbf{B}_i$, decompose it into a low-rank matrix $\mathbf{B}_i^n$ and a sparse matrix $\mathbf{B}_i^e$ by solving (6.14) using the PCP algorithm.
2. Let

$$\mathbf{B}^n = [\mathbf{B}_1^n, \mathbf{B}_2^n, \cdots, \mathbf{B}_C^n]$$

and

$$\mathbf{B}^e = [\bar{\mathbf{B}}_1^e, \bar{\mathbf{B}}_2^e, \cdots, \bar{\mathbf{B}}_E^e]$$

be the concatenation of neutral and permuted expression component matrices, respectively.
3. Learn the best dictionaries for the neutral and expression components by solving (6.15) and (6.16), respectively using the K-SVD algorithm.

Testing:
1. Given an expressive face, $\mathbf{x}_t$, decompose it into neutral and expressive components using the DCS algorithm outlined in Figure 6.3.
2. Obtain the new sparse codes β_n, β_e optimized for the face and expression recognition by decomposing the extracted components onto their corresponding dictionaries using equations (6.19) and (6.20), respectively.
3. Use the sparse coefficient vectors, β_n, β_e to perform joint face and expression recognition using the minimum residual rules in equation (6.21).

Figure 6.4: Joint face and expression recognition algorithm.

where the grouping is based on the expression labels of $\mathbf{D}^e$ atoms and E is the total number of expression labels. After computing the new sparse codes, the identity and expression label for the test image are obtained using (6.17) and (6.18) where $\{\alpha_n, \alpha_e\}$ are replaced with $\{\beta_n, \beta_e\}$.

$$c_s = \arg\min_{c_s} \|\mathbf{x}_n - \mathbf{D}^n \delta_{c_s}(\beta_n)\|, \qquad (6.21)$$
$$c_e = \arg\min_{c_e} \|\mathbf{x}_e - \mathbf{D}^e \delta_{c_e}(\beta_e)\|$$

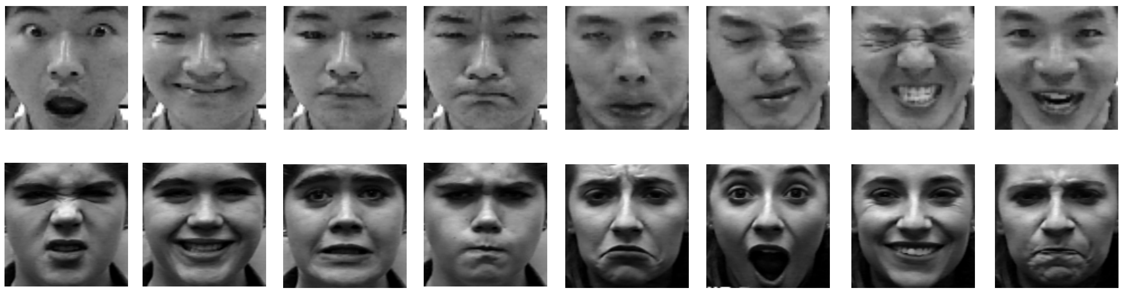


Figure 6.5: Some sample expressive faces in the two datasets. Top: the CMU dataset [5], Bottom: the CK+ dataset [6].

We summarize the proposed joint face and expression recognition algorithm in Figure 6.4.

6.5 Experimental Results

We evaluate our algorithm using two face expression datasets: (1) CMU AMP face expression dataset [5] and (2) Extended Cohn-Kanade face expression dataset (CK+) [6]. The CMU dataset contains 975 images (13 subjects with 75 images per subject) with different facial expressions. The CK+ dataset contains 593 expression sequences from 123 subjects. Only 327 out of 593 sequences have emotion labels from each of the six universal emotion categories (Anger, Disgust, Fear, Happy, Sad and Surprise). We compare our face recognition results with the results of B-JSM algorithm proposed in [160] which is an expression-invariant face recognition algorithm, also with the results from SRC [174] algorithm. For expression recognition results, we compare with recent results reported on CK+ dataset [6]. We also evaluate our algorithm with respect to two dictionary learning algorithms, KSVD [191] and

FDDL [196]. We learn dictionaries for the training data using these algorithms[4] and then perform face and expression recognition using the original test image (without the component separation step). These comparison emphasizes the importance of the proposed component separation algorithm for face and expression recognition. Figure 6.5 shows several examples of faces from these two datasets. All experiments are done on a Linux machine with 4GB of RAM using MATLAB.

6.5.1 Implementation details

In all the datasets, images are well-cropped and aligned. We resize the faces to 32×32. From the discussion in section 6.3, the parameter c should be chosen such that $c > \lambda_{\max}(\mathbf{D}_n\mathbf{D}_n^T + \mathbf{D}_e\mathbf{D}_e^T)$. This can be satisfied by choosing $c > 2$. In particular, the value we used is $c = 3$. We change the threshold value of λ^k during each iteration according to

$$\lambda^k = \frac{1}{2}\left(\|\mathbf{D}_n^T\mathbf{r}^k\|_\infty + \|\mathbf{D}_e^T\mathbf{r}^k\|_\infty\right)$$

and stop the iterations when $\lambda^k \le T$, where $T \approx 2.1$. The value for the regularization parameter η in (6.14) is set equal to $\frac{1}{\sqrt{\max(N,m)}}$ [8]. The parameter T_0 in the K-SVD algorithm is empirically determined and is set equal to $\frac{m}{2}$ for the neutral dictionary and $\frac{\bar{m}}{2}$ for the expression dictionary.

[4]for KSVD, we learn dictionaries per subject/expression and the final dictionary is obtained by concatenating the individual dictionaries. Therefore, we have the subject/expression labels for the dictionary atoms

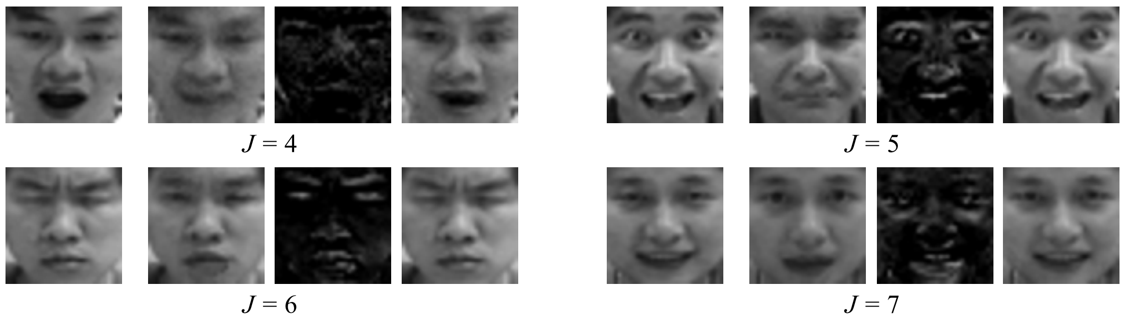


Figure 6.6: Some examples of expressive face decomposition using DCS on the selected samples from the CMU dataset for various values of J. In these images, the first image is the original input image. The second and the third images are the separated neutral and expression components, respectively. The fourth image is the sum of the two extracted components which is very similar to the original input image.

6.5.2 Experiments on the CMU dataset

For this dataset, we follow the experimental set-up presented in [160] to perform face recognition. We randomly select J images per subject to form the training set. The remaining faces per subjects are used for the face recognition experiment. We perform validation for $J = [4, ..., 8]$ with 10 trials each. Figure 6.6 has some examples of expressive face decomposition using the DCS algorithm on the selected samples from this dataset and for various values of J. As more number of images per subject are used for the training set, the learned dictionaries become more representative of the data and therefore better recognition rates are obtained, as shown in Table 6.1.

We compare the face recognition results from our algorithm with those of B-JSM algorithm [160] and sparse representation-based face recognition (SRC) algorithm [174] in Table 6.1. We report our results using both recognition schemes

Table 6.1: Recognition rate (%) on the CMU dataset with 10 trials for each J.

J	**DCS with (6.21)**			**DCS with (6.17,6.18)**			**B-JSM [160]**			**SRC [174]**		
	High	**Low**	**Avg**	**High**	**Low**	**Avg**	**High**	**Low**	**Avg**	**High**	**Low**	**Avg**
4	100	100	100	100	98.13	99.30	100	97.48	98.95	100	97.68	98.90
5	100	100	100	100	99.86	99.96	100	99.67	99.91	100	99.12	99.80
6	100	100	100	100	100	100	100	99.69	99.97	100	98.76	99.75
7	100	100	100	100	100	100	100	100	100	100	98.30	99.74
8	100	100	100	100	100	100	100	100	100	100	99.31	99.87

proposed in section 6.4. As the table shows when we use (6.21) for recognition, the proposed algorithm has 100% recognition rates for all values of J. Using the simpler recognition scheme (6.17,6.18) the results are slightly lower but still superior to two other algorithms. Since in the simpler recognition scheme we do not enforce any constraint regarding the structure of the dictionaries in the sparse coding, we can expect lower performance when the number of samples for each class is few (small value of J). However, as the value of J increases, the performance improves. This emphasizes the importance of the recognition step.

It should be noted that the testing phase in our algorithm is very fast. When $J = 7$, for each test face (of size 32×32) it takes about 1.4 cputime to decompose it into the constituent components using DCS algorithm and then obtain the residuals for e.g. 13 subjects. However, for the B-JSM algorithm the recognition phase is slow since it needs to perform the optimization between each gallery face and the test image, which for 13 subjects (and using the same setup as provided in the paper [160]) takes about 37.6 cputime, on the same machine.

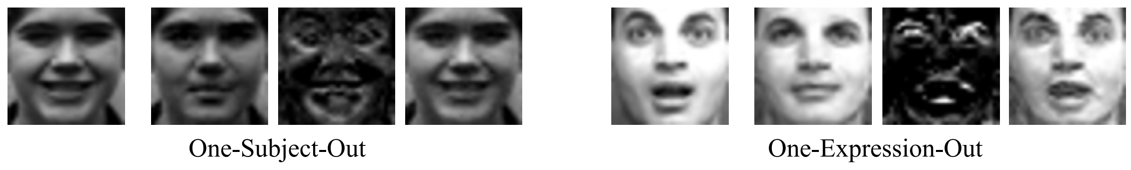


Figure 6.7: Two examples of expressive face decomposition using the DCS algorithm for one-subject-out (S2) and one-expression-out (S3) set-ups on the CK+ dataset. In both figures, the first image is the original input image. The second and the third images are the separated neutral and expression components, respectively. The fourth image is the sum of the two extracted components which is very similar to the original input image.

6.5.3 Experiments on the CK+ dataset

One of the important advantages of our algorithm over B-JSM and the other expression-invariant face recognition algorithms is that we can perform expression recognition with no additional cost. To show the performance of joint face and expression recognition, we use the CK+ dataset. This dataset is used mainly for expression recognition, but we perform both face and expression recognition on it. Since in this dataset, the number of sequences per subjects varies a lot (some subjects only have one or two labeled expression sequences and some may have as many as six expression sequences) we select a subset of the dataset in which the subjects have at least four different expression sequences. This helps us to have a balanced dictionary which is necessary for dictionary-based algorithms. The selected subset has 25 subjects.

We perform these experiments in three different set-ups. In the first set-up (S1), we randomly select 3 sequences per subject for training and leave the rest for testing. We repeat this process ten times and finally report the average face and expression recognition results. In the second set-up (S2) we perform one-subject-out expression recognition where we remove one subject with all its sequences from the dataset for testing and train on the rests. We also consider this experimental set-up for the whole dataset (106 subjects) and obtain the results for expression recognition, for the purpose of comparing with other algorithms. Finally in the third set-up (S3), we perform one-expression-out face recognition to evaluate the effect of various expressions on the face recognition performance. In all cases, we select the four initial frames of each sequence as neutral faces and the four last frames (apex) as expressive faces and run our algorithm on these images. The test images are selected as the four last frames of the test sequences. Figure 6.7 shows two examples of expressive face decomposition using DCS algorithm for one-subject-out (S2) and one-expression-out (S3) set-ups on this dataset.

We compare the face recognition results using S1 and S3 set-ups with the results from B-JSM, KSVD and FDDL algorithms. As Table 6.2 shows, in almost all the cases our algorithm gives higher recognition rates. FDDL algorithm, which is based on learning a discriminative dictionary, performs close (even slightly better in S3 set-up) to DCS algorithm. So considering the fact that the dictionaries learnt in the training step of the DCS algorithm are not discriminative, this emphasizes the importance of component separation step in DCS algorithm. Moreover, DCS shows

Table 6.2: Face recognition rates (%) on the CK+ dataset with S1 and S3 train-test set-ups.

Set-ups	**DCS**	**B-JSM** [160]	**KSVD** [191]	**FDDL** [196]
S1	99.14 ± 1.4	85.2 ± 5.01	85.6 ± 4.8	98.8 ± 1.6
S3	95.1 ± 6.7	81.5 ± 6.15	91.2 ± 3.1	95.3 ± 3.7

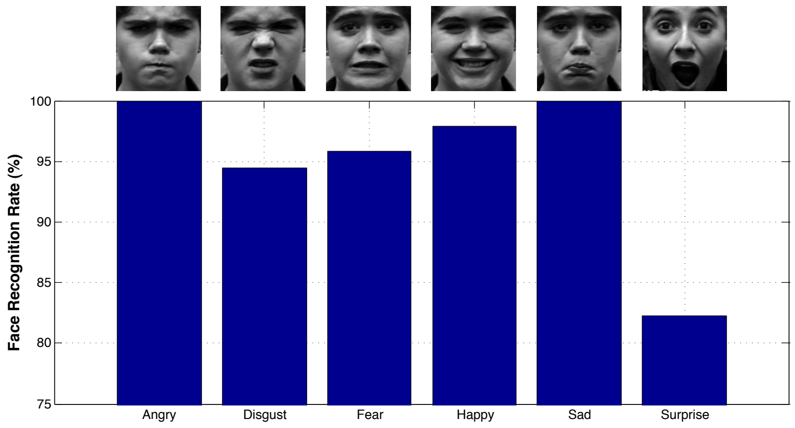


Figure 6.8: Effects of various expressions on the face recognition results on the CK+ dataset using S3 set-up. Each bar shows the face recognition rate we obtain when all the faces with corresponding expressions are kept out for testing and the rest are used for training.

superiority to FDDL in expression recognition (Table 3) which is more challenging task compared to face recognition on this dataset.

Figure 6.8 shows the effects of various expressions on the face recognition results using the S3 set-up. As the figure shows, while angry and sad faces are the easiest expressive faces to recognize (since these expressions are more subtle compared to others and so they present less challenges for face recognition), the surprise face is the most challenging one for recognition.

We also compare the results of expression recognition using our algorithm with those of KSVD and FDDL as well as some recent methods for expression recognition on the CK/CK+ datasets including a joint face and expression recognition method [168]. The CK dataset [197] is the old version of CK+ which has fewer subjects and sequences. Most of these algorithms performed the recognition by dividing the CK dataset randomly into training and testing parts and only [6] has the results with

Table 6.3: Comparison with recent advances in expression recognition on CK+ dataset.

Recognition Methods	**Recognition Rates (%)**
SVM+MLR* [181]	91.5
NN+GMM [182]	71
CAPP⋆ +SVM [6] †	86.48
RegRankBoost [147]	88
Combined Features+Adaboost [134]	92.3
Decomposable Generative Model [168]	70.85
KSVD-S1	49.2
KSVD-S2-selected subset	64.6
FDDL-S1	73.7
FDDL-S2-selected subset	73.6
DCS-S1	**81.64**
DCS-S2-selected subset	**86.8**
DCS-S2-whole dataset	**89.21**

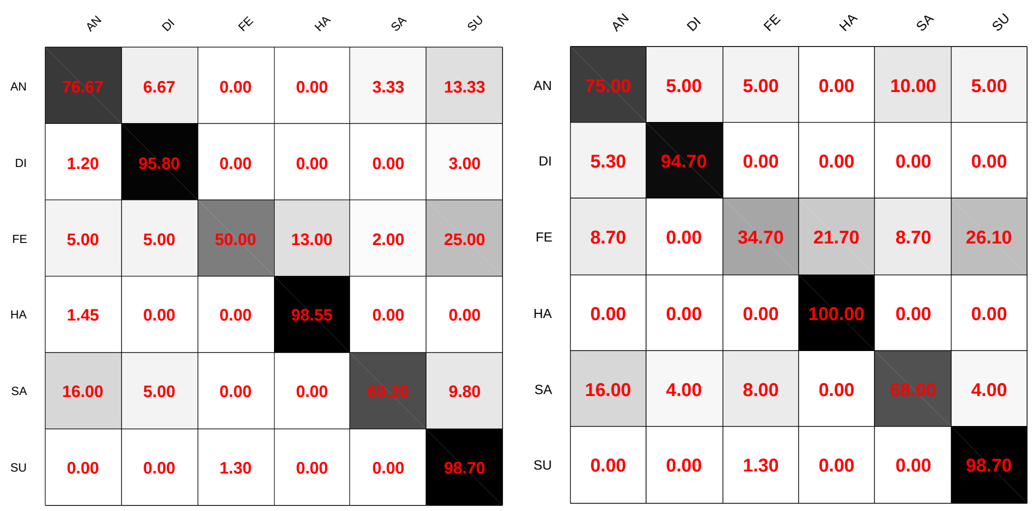
∗ MLR = Multinomial Logistic Ridge Regression

⋆ CAPP = Canonical Appearance Features

† results on CK+ with leave-one-subject-out validation

one-subject-out cross-validation on the CK+. We compare the results from both S1 and S2 set-ups in Table 6.3. As the table shows, our results are better than the results of KSVD and FDDL algorithms which as we mentioned before this proves the importance of component separation for face and expression recognition. Compared to other reported results for expression recognition on CK/CK+, while our result (DCS-S2-whole dataset) is among the top reported results, it is not the best one. But it should be noted that while most of these algorithms extract several features from the expressive faces and use trained classifiers such as SVM, Adaboost and Neural Networks, our algorithm only uses the extracted deformation component of the face as a holistic image with a simple residual-based classification.

We also evaluate our expression recognition results for different expressive



	AN	DI	FE	HA	SA	SU
AN	76.67	6.67	0.00	0.00	3.33	13.33
DI	1.20	95.80	0.00	0.00	0.00	3.00
FE	5.00	5.00	50.00	13.00	2.00	25.00
HA	1.45	0.00	0.00	98.55	0.00	0.00
SA	16.00	5.00	0.00	0.00	69.20	9.80
SU	0.00	0.00	1.30	0.00	0.00	98.70

	AN	DI	FE	HA	SA	SU
AN	75.00	5.00	5.00	0.00	10.00	5.00
DI	5.30	94.70	0.00	0.00	0.00	0.00
FE	8.70	0.00	34.70	21.70	8.70	26.10
HA	0.00	0.00	0.00	100.00	0.00	0.00
SA	16.00	4.00	8.00	0.00	68.00	4.00
SU	0.00	0.00	1.30	0.00	0.00	98.70

Figure 6.9: Confusion matrices for expression recognition on CK+ using one-subject-out cross-validation. **left**: results using our approach, **right**: result from [6]

faces. Figure 6.9 shows the confusion matrix for the whole CK+ dataset with one-subject-out cross-validation (S2). The figure also shows the confusion matrix reported in [6] for the same set-up [5]. As the results show, both algorithms have difficulty recognizing the fear expression. These results can be improved by adding some other types of features, such as shape features [6], through joint sparse representation.

6.6 Summary

We proposed joint face and facial expression recognition using a dictionary-based component separation algorithm. Considering an expressive face as a superposition of a neutral face with expression component, we proposed an algorithm to decompose an expressive test face into its building components. For this purpose, we

[5]The original matrix has the results for 'Contempt' expression as well. Since we removed this expression from our experiments due to non-enough sequences, we modified their results appropriately to have a fair comparison

first generate two data-driven dictionaries, one for neutral components and the other one for the expression components. Knowing that the neutral component of the test face has sparse representation in the neutral dictionary and the expression part can be sparsely represented using the expression dictionary, we decompose the test face into these morphological components. The elements of the test face along with the dictionaries are then used for face and expression recognition. For this purpose, the separated components are sparsely decomposed using dictionaries while the grouping structures of the dictionaries are enforced into the sparse decomposition results. The results for face recognition are very good and the expression recognition results are among the top results. These results can be further improved by incorporating some facial features such as the shape of facial components to boost the expression recognition results.

Chapter 7: Directions for Future Work

The problems addressed in this dissertation and the methods proposed to solve them suggest several interesting future research directions. In this chapter we outline a few directions for future research work.

7.1 Expression Flow Modeling

One limitation of our work on facial expression analysis on the shape-space is the need for having landmarks on the face. We can eliminate this requirement by using the dense flow field corresponding to various expression sequences. We can employ two possible approaches for modeling the expression flow field. The first approach is to model flow fields corresponding to various expression sequences as dynamic processes driven by spatio-temporal models. For any point on the face, the motion corresponding to that point can be modeled as

$$X(t_1) = F(t_1, t_0, x, y)X(t_0)$$

where t_0 is taken to be the starting time. Once we have learned F for all t_1, x, and y, then the expression motion is completely characterized. Therefore to be able to

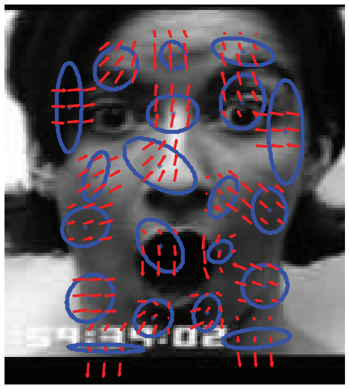


Figure 7.1: Uniform flow regions on an expressive face.

learn F, we consider a parametric model for it and a good model is to limit the F to belong to affine Lie group. Moreover, to make the problem more tractable, instead of modeling the motion at each point, we model the motion over a region of the face with uniform motion pattern. This leads to a spatial hybrid model, in which we assume K affine models over the expressive face. Figure 7.1 shows a face with the learned uniform flow regions on it. The flow at each region is modeled using affine Lie group [198].

The second approach for modeling the expression flow is to find a non-analytic expression flow manifold for various facial expressions. It is known that faces across various expressions can be modeled on image articulation manifold (IAM) [199], since it provides a powerful model for such ensembles: a collection of N-pixel images, with each image indexed by K degrees of freedom, can be modeled as a K-dimensional nonlinear manifold embedded in $\mathbb{R}^N$. Then several problems such as parameter estimation, supervised and semi-supervised classification, and novel view synthesis can all be cast as navigation to appropriate point/regions on an IAM and this requires constructing *transport operators* that traverse the IAM. The approach described in the previous paragraph provides an algebraic method for IAM transport which exploits the geometric relationships between the images comprising an IAM. A limitation of this approach and other algebraic transport methods is that they are limited to a small class of IAMs with a well-defined algebraic structure (such as a Lie group structure); such a structure occurs only in some special cases, such as affine articulation. On the other hand, optical flow between pair of images is a natural instance of transport on an IAM [7]. We argue that set of optical flows from neutral face to various expressive faces forms a low-dimensional smooth manifold and we can call it *expression flow manifold*. Such a manifold facilitates tasks such as pose-invariant expression analysis and novel-expression synthesis. An illustration of this idea can be seen in Fig. 7.2.

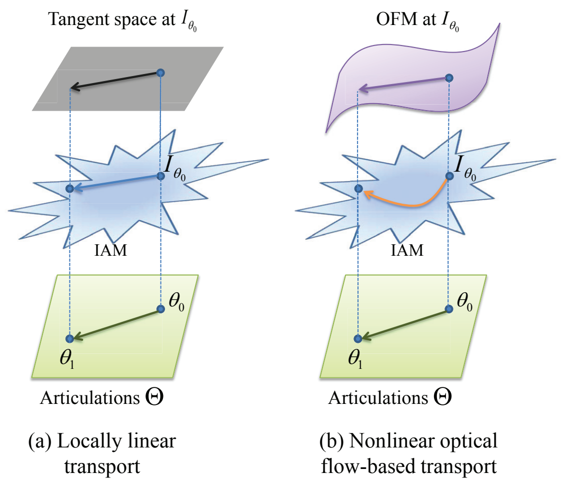

Tangent space at I_{θ_0}
OFM at I_{θ_0}
I_{θ_0}
IAM
θ_0
θ_1
Articulations Θ
(a) Locally linear transport
(b) Nonlinear optical flow-based transport

Figure 7.2: Image articulation manifolds (IAMs) are non-differentiable; therefore, locally linear models such as tangent spaces provide an inaccurate approximation to the manifold geometry. In contrast, the optical flow manifold (OFM) associated with a point on the IAM accurately captures the intrinsic curved geometric structure of the IAM. The image is taken from [7].

7.2 Structure-Preserving Sparse Decomposition for Actions/Activities

Recognizing human actions in still images has many potential applications in image indexing and retrieval. One straightforward solution for this problem is to use the whole image to represent an action and treat action recognition as a general image classification problem [200–202]. Although such methods have achieved promising performance, they do not explore the semantically meaningful components of an action, such as human poses, body parts involved and the objects that are closely related to the action [203]. Structure-preserving sparse decomposition algorithm discussed in Chapter 5 proposes a general framework for decomposing an action into its constituent components considering their semantic information and compositional rules. If we could define proper attributes and parts for each action, then the proposed algorithm can be generalized to general class of action recognition.

Group activity recognition is a challenging problem mainly due to the inherent difficulties in modeling inter-person interactions. For individual actions, body parts share common motions due to human articulation constraints. Similar to the individual actions, for structured group activities collaborative players can have correlated motions. Such interactions between players and the temporal constraints among their motions can be used to describe a group activity. One can also incorporate spatial constraints such as orientation and distance. Using the proposed structure-preserving sparse decomposition algorithm, a complex activity can be described as a set of semantic units connected using spatial and temporal constraints.

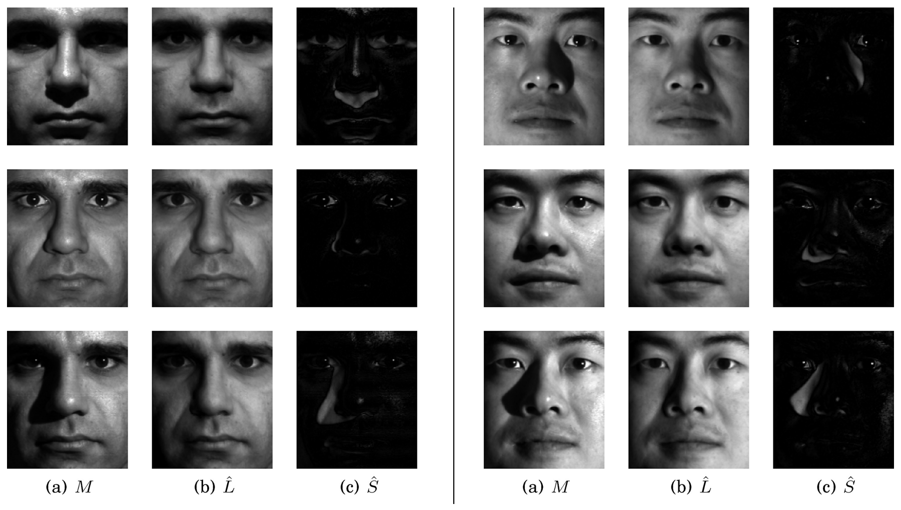


Figure 7.3: Removing shadow, specularities, and saturations from face images [8].

7.3 Taking the PIE off the face

As we discussed earlier pose, illumination and expression (PIE) impose challenges for face/object recognition. Various algorithms have been proposed to remove these variations from faces. An important class of algorithms is based on separating style and content [164, 204, 205]. The main goal of these algorithms is to decompose a test face into its style and content components having enough samples of faces with different variations from multiple subjects. But the main limitation of these algorithms is that they often need the same set of possible variations for different subjects in order to perform the separation task.

The component separation algorithm proposed in Chapter 6 is not specific to expression decomposition, but it can also be applied to separate other sources of variation from the test face. The important challenge here is to learn proper dictionaries for each component and the advantage is that there is no need to have all possible variations for every subject. Robust principal component analysis (RPCA) [8] shows promising results in separating the variations due to illumination, specularities and saturations from the faces (Fog. 7.3). So we propose using DCS algorithm to take PIE off the test face.

Bibliography

[1] S. Biswas, G. Aggarwal, and R. Chellappa. Robust estimation of albedo for illumination-invariant matching and shape recovery. *TPAMI*, 31(5):884–899, 2009.

[2] Lei Zhang and D. Samaras. Face recognition from a single training image under arbitrary unknown lighting using spherical harmonics. *TPAMI*, 28(3):351–363, 2006.

[3] A. Srivastava and E. Klassen. Bayesian and geometric subspace tracking. *Advances in Applied Probability*, 36(1):43–56, 2004.

[4] Yingli Tian, Takeo Kanade, and Jeffrey F. Cohn. *Facial Expression Recognition.* Springer, 2011.

[5] X. Liu, T. Chen, and B.V.K. Vijaya Kumar. Face authentication for multiple subjects uisng eigenflow. *pattern recognition*, 2003.

[6] P. Lucey, J.F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The extended cohn-kanade dataset (CK+): A complete dataset for action unit and emotion-specified expression. In *CVPR Workshop*, 2010.

[7] Aswin C. Sankaranarayanan, Chinmay Hegde, Sriram Nagaraj, and Richard G. Baraniuk. Go with the flow: Optical flow-based transport operators for image manifolds. In *Allerton*, 2011.

[8] E. Candes, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):137, 2011.

[9] Y. Tian, T. Kanade, and J.F. Cohn. *Facial Expression Recognition*, chapter 11. Springer, 2011.

[10] P. Ekman and W.V. Friesen. *The facial action coding system: a technique for the measurement of facial movement.* Consulting Psychologists Press., 1978.

[11] Ravi Ramamoorthi. Modeling illumination variation with spherical harmonics. In *Face Processing: Advanced Modeling Methods*. 2006.

[12] B. K. P. Horn and M. J. Brooks. *Shape from Shading.* Cambridge Massachusetts: MIT Press, 1989.

[13] Ruo Zhang, Ping sing Tsai, James Edwin Cryer, and Mubarak Shah. Shape from shading: A survey. *TPAMI*, 21(8):690–706, 1999.

[14] S. Zhou, G. Aggarwal, R. Chellappa, and D. Jacobs. Appearance characterization of linear lambertian objects, generalized photometric stereo and illumination-invariant face recognition. *TPAMI*, 29(2):230–245, 2007.

[15] Azeem Lakdawalla and Aaron Hertzmann. Shape from video: Dense shape, texture, motion and lighting from monocular image streams. In *Photometric Analysis for Comp. Vision.*, 2007.

[16] Yilei Xu and A.K. Roy-Chowdhury. Integrating motion, illumination, and structure in video with applications in illumination-invariant tracking. *TPAMI*, 29(5):793–806, 2007.

[17] Ronen Basri and David Jacobs. Lambertian reflectance and linear subspaces. *TPAMI*, 25(2):218–233, 2003.

[18] Zhen Wen, Zicheng Liu, and Thomas Huang. Face relighting with radiance environment maps. In *CVPR*, volume 2, 2003.

[19] Yilei Xu, A. Roy-Chowdhury, and K. Patel. Pose and illumination invariant face recognition in video. In *CVPR*, 2007.

[20] Angelos Barmpoutis, Ritwik Kumar, B. C. Vemuri, and A. Banerjee. Beyond the lambertian assumption: A generative model for ABRDF fields of faces using anti-symmetric tensor splines. In *CVPR*, 2008.

[21] Tim Weyrich, Wojciech Matusik, Hanspeter Pfister, Bernd Bickel, Craig Donner, Chien Tu, Janet Mcandless, Jinho Lee, Addy Ngan, Henrik Wann, and Jensen Markus Gross. Analysis of human faces using a measurement-based skin reflectance model. *ACM Trans. Graphics*, 25(3):1013–1024, 2006.

[22] Y. Wang, Z. Liu, G. Hua, Z. Wen, Z. Zhang, and D. Samaras. Face re-lighting from a single image under harsh lighting conditions. In *CVPR*, 2007.

[23] Neel Joshi and David J. Kriegman. Shape from varying illumination and viewpoint. In *ICCV*, 2007.

[24] Carlos Hernandez Esteban, George Vogiatzis, and Roberto Cipolla. Multiview photometric stereo. *TPAMI*, 30(3):548–554, 2008.

[25] Jongwoo Lim, Jeffrey Ho, Ming hsuan Yang, and David Kriegman. Passive photometric stereo from motion. In *ICCV*, 2005.

[26] Li Zhang, Brian Curless, Aaron Hertzmann, and Steven M. Seitz. Shape and motion under varying illumination: Unifying structure from motion, photometric stereo, and multi-view stereo. In *ICCV*, 2003.

[27] Denis Simakov, Darya Frolova, and Ronen Basri. Dense shape reconstruction of a moving object under arbitrary, unknown lighting. In *ICCV*, pages 1202–1209, 2003.

[28] R. J. Woodham. Photometric method for determining surface orientation from multiple images. *Optical Engineerings*, 19(1):139–144, 1980.

[29] K. Ikeuchi. Determining surface orientation of specular surfaces by using the photometric stereo method. *TPAMI*, 3(6):661–669, 1981.

[30] Wen Yi Zhao and Rama Chellappa. Symmetric shape-from-shading using self-ratio image. *Intl. J. Computer Vision*, 45(1):55–75, 2001.

[31] JJ. Atick, P.A. Griffin, and A.N. Redlich. Statistical approach to SFS: Reconstruction of 3D face surfaces from single 2D images. *Neural Computation*, 8:1321–1340, 1996.

[32] Roman Dovgard and Ronen Basri. Statistical symmetric shape from shading for 3D structure recovery of faces. In *ECCV*, 2004.

[33] W. A. P. Smith and E. R. Hancock. Recovering facial shape using a statistical model of surface normal direction. *TPAMI*, 28(12):1914–1930, 2006.

[34] B. K. P. Horn. Determining lightness from an image. *Computer Graphics, Image Processing*, 3:277–299, 1974.

[35] E. H. Land and J. J. McCann. Lightness and retinex theory. *J. of the Optical Society of America*, 61(1):1–11, 1971.

[36] M. F. Tappen, E. H. Adelson, and W. T. Freeman. Estimating intrinsic component images using non-linear regression. In *CVPR*, 2006.

[37] Volker Blanz and Thomas Vetter. Face recognition based on fitting a 3D morphable model. *TPAMI*, 25(9):1063–1074, 2003.

[38] Yang Wang, Lei Zhang, Zicheng Liu, Gang Hua, Zhen Wen, Zhengyou Zhang, and Dimitris Samaras. Face relighting from a single image under arbitrary unknown lighting conditions. *TPAMI*, 31(11):1968–1984, 2009.

[39] S. Biswas and R. Chellappa. Pose-robust albedo estimation from a single image. In *CVPR*, pages 2683 – 2690, 2010.

[40] Tianli Yu, Ning Xu, and Narendra Ahuja. Recovering shape and reflectance model of non-Lambertian objects from multiple views. In *CVPR*, 2004.

[41] Gaurav Aggarwal, Ashok Veeraraghavan, and Rama Chellappa. 3D facial pose tracking in uncalibrated videos. In *PReMI*, 2005.

[42] S.O. Ba and J.M. Odobez. A probabilistic head pose tracking evaluation in single and multiple camera setups. In *CLEAR, Evaluation and Workshop*, 2007.

[43] Marco La Cascia, Stan Sclaroff, and Vassilis Athitsos. Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3D models. *TPAMI*, 22(4):322–336, 2000.

[44] Tim K. Marks, John R. Hershey, and Javier R. Movellan. Tracking motion, deformation, and texture using conditionally gaussian processes. *TPAMI*, 32(2):348–363, 2010.

[45] E. Murphy-Chutorian and M.M. Trivedi. Head pose estimation in computer vision: A survey. *TPAMI*, 31(4):607–626, 2008.

[46] Y. Tanabe, T. Inui, and Y. Onodera. *Group Theory and Its Applications in Physics*. Springer, 1990.

[47] Zhanfeng Yue, Wenyi Zhao, and Rama Chellappa. Pose-encoded spherical harmonics for face recognition and synthesis using a single image. *EURASIP Journal on Advances in Signal Processing*, 2008.

[48] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, 1999.

[49] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter. A 3D face model for pose and illumination invariant face recognition. In *IEEE Intl. Conf. AVSS*, 2009.

[50] Daniel F. DeMenthon and Larry S. Davis. Model-based object pose in 25 lines of code. *Intl. J. Computer Vision*, 15(1-2):123–141, 1995.

[51] Russ B. Altman. A probabilistic algorithm for calculating structure: Borrowing from simulated annealing. Technical report, Stanford University, 1990.

[52] Russ B. Altman and James F. Brinkley. Probabilistic constraint satisfaction with structural models. In *Symposium on Computer Applications in Medical Care*, 1993.

[53] Peter S. Maybeck. *Stochastic models estimation and control.* Academic Press, 1979.

[54] Terence Sim, Simon Baker, and Maan Bsat. The CMU pose, illumination, and expression database. *TPAMI*, 25(12):1615–1618, 2003.

[55] S.O. Ba and J.M Odobez. A Rao-blackwellized mixed state particle filter for head pose tracking in meetings. In *ACM MMMP*, 2005.

[56] A. C. Sankaranarayanan, A. Srivastava, and R. Chellappa. Algorithmic and architectural optimizations for computationally efficient particle filtering. *IEEE Trans. on Image Processing*, 17(5):737–748, 2008.

[57] J. Shi and C. Tomasi. Good features to track. In *CVPR*, 1994.

[58] B. Anderson and J.B. Moore. *Optimal Filtering.* Prentice Hall, 1979.

[59] S. Hong, S. S. Chin, P. M. Djurić, and M. Bolić. Design and implementation of flexible resampling mechanism for high-speed parallel particle filters. *J. VLSI Signal Processing*, 44(1-2):47–62, 2006.

[60] V. Caglioti and A. Giusti. On the apparent transparency of a motion blurred object. *IJCV*, 2008.

[61] A. Sellent, M. Eisemann, B. Goldlucke, D. Cremers, and M. Magnor. Motion field estimation from alternate exposure images. *TPAMI*, 2011.

[62] G. Boracchi. Estimating the 3d direction of a translating camera from a single. *Pattern Recognition Letter*, 2009.

[63] S. Dai and Y. Wu. Motion from blur. In *CVPR*, 2008.

[64] G. Klein and T. Drummond. A single-frame visual gyroscope. In *BMVC*, 2005.

[65] C. Paramanand and A. N. Rajagopalan. Inferring image transformation and structure from motion-blurred images,. In *BMVC*, 2010.

[66] C. Paramanand and A. N. Rajagopalan. Depth from motion and optical blur with unscented Kalman filter. *TIP*, 2012.

[67] Y. Tai, P. Tan, and M. S. Brown. Richardson-lucy deblurring for scenes under projective motion path. *TPAMI*, 2011.

[68] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce. Non-uniform deblurring for shaken images. *CVPR*, 2010.

[69] S. Z. Li, Dong Yi, and Zhen Lei. Discriminant image filter learning for face recognition with local binary pattern like representation. In *CVPR*, 2012.

[70] M. Yang, L. Zhang, J. Yang, and D. Zhang. Robust sparse coding for face recognition. In *CVPR*, 2011.

[71] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. In *TPMAI*, 2009.

[72] M. Yang and L. Zhang. Gabor feature based sparse representation for face recognition with gabor occlusion dictionary. In *ECCV*, 2010.

[73] Chih-Fan Chen, Chia-Po Wei, and Y.-C.F. Wang. Low-rank matrix recovery with structural incoherence for robust face recognition. In *CVPR*, 2010.

[74] Kui Jia, Tsung-Han Chan, and Yi Ma. Robust and practical face recognition via structured sparsity. In *ECCV*, 2012.

[75] P. H. Hennings-Yeomans, S. Baker, and B.V. Kumar. Simultaneous super-resolution and feature extraction for recognition of low resolution faces. In *CVPR*, 2008.

[76] Haichao Zhang, Jianchao Yang, Yanning Zhang, Nasser M. Nasrabadi, and Thomas S. Huang. Close the loop: Joint blind image restoration and recognition with sparse representation prior. In *ICCV*, 2011.

[77] M. Nishiyama, H. Takeshima, J. Shotton, T. Kozakaya, and O. Yamaguchi. Facial deblur inference to improve recognition of blurred faces. In *CVPR*, pages 1115–1122, June 2009.

[78] R. Gopalan, S. Taheri, P. Turaga, and R. Chellappa. A blur-robust descriptor with applications to face recognition. *TPAMI*, 2012.

[79] H.C. Andrews and B.R. Hunt. *Digital Image Restoration*. Prentice Hall Signal Processing Series, 1977.

[80] I. Stainvas and N. Intrator. Blurred face recognition via a hybrid network architecture. In *ICPR*, pages 805–808, Sep. 2000.

[81] Jan Flusser, Jiří Boldyš, and Barbara Zitová. Invariants to convolution in arbitrary dimensions. *Journal of Mathematical Imaging and Vision*, 2000.

[82] J. Flusser and T. Suk. Degraded image analysis: an invariant approach. *IEEE TPAMI*, 1998.

[83] Jan Flusser, Jirí Boldys, and Barbara Zitová. Moment forms invariant to rotation and blur in arbitrary number of dimensions. *IEEE TPAMI*, 2003.

[84] T. Ahonen, E. Rahtu, V. Ojansivu, and J. Heikkila. Recognition of blurred faces using local phase quantization. In *ICPR*, 2008.

[85] Ville Ojansivu and Janne Heikkilä. A method for blur and affine invariant object recognition using phase-only bispectrum. In *ICIAR*, 2008.

[86] P. Vageeswaran, K. Mitra, and R. Chellappa. Blur and illumination robust face recognition via set-theoretic characterization. *TIP*, 2013.

[87] S. Cho, Y. Matsushita, and S. Lee. Removing non-uniform motion blur from images. In *ICCV*, 2007.

[88] J. G. Nagy and D. P. O'Leary. Restoring images degraded by spatially variant blur. *SIAM J. Sci. Comput*, 1998.

[89] Q. Shan, W. Xiong, and J. Jia. Rotational motion deblurring of a rigid object from a single image. In *ICCV*, 2007.

[90] R. Vio, J. Nagy, L. Tenorio, and W. Wamsteker. Multiple image deblurring with spatially variant psfs. *Astronomy and Astrophysics*, 2005.

[91] J. W. Miskin and D. J. C. MacKay. Ensemble learning for blind image separation and deconvolution. *Advances in Independent Component Analysis. Springer-Verlag*, 2000.

[92] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. *SIGGRAPH*, 2006.

[93] A. Gupta, N. Joshi, L. Zitnick, M. Cohen, and B. Curless. Single image deblurring using motion density functions. In *ECCV*, 2012.

[94] M. Hirsch, S. Harmeling, S. Sra, and B. Schölkopf. Efficient filter flow for space-variant multiframe blind deconvolution. In *CVPR*, 2010.

[95] D. Krishnan, T. Tay, and R. Fergus. Blind deconvolution using a normalized sparsity measure. In *CVPR*, 2011.

[96] Sunghyun Cho, Hojin Cho, Yu-Wing Tai, and Seungyong Lee. Non-uniform motion deblurring for camera shakes using image registration. In *SIGGRAPH Talks*, 2011.

[97] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce. Non-uniform deblurring for shaken images. *IJCV*, 2011.

[98] L. Yuan, J. Sun, L. Quan, and H.-Y. Shum. Image deblurring with blurred/noisy image pairs. *SIGGRAPH*, 2007.

[99] R. Tibshirani. Regression shrinkage and selection via the LASSO. *J. Royal. Statist.*, 1996.

[100] P. Phillips, H. Moon, S. Rizvi, and P. Rauss. The FERET evaluation methodology for face-recognition algorithms. *TPAMI*, 2000.

[101] Anil K. Jain and Stan Z. Li. *Handbook of Face Recognition*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[102] Tomas Simon, Nguyen Minh, Fernando De la Torre, and Jeff Cohn. Action unit detection with segment-based SVMs. In *CVPR*, 2010.

[103] Peng Yang, Qingshan Liu, and Metaxas Dimitris. Exploring facial expression with compositional features. In *CVPR*, 2010.

[104] Ying-Li Tian, Takeo Kanade, and Jeffrey F. Cohn. Recognizing action units for facial expression analysis. *TPAMI*, 23:97–115, 1999.

[105] Zhiwei Zhu and Qiang Ji. Robust real-time face pose and facial expression recovery. *CVPR*, 1:681–688, 2006.

[106] Yan Tong, Jixu Chen, and Qiang Ji. A unified probabilistic framework for spontaneous facial action modeling and understanding. *TPAMI*, 32(2):258–274, 2010.

[107] Wei-Kai Liao and G. Medioni. 3D face tracking and expression inference from a 2D sequence using manifold learning. In *CVPR*, 2008.

[108] Y. Chang, M. Vieira, M. Turk, and L. Velho. Automatic 3D facial expression analysis in videos. In *IEEE International Workshop on Analysis and Modeling of Faces and Gestures*, October 2005.

[109] T.H. Wang and J.J. Lien. Facial expression recognition system based on rigid and non-rigid motion separation and 3D pose estimation. In *Pattern Recognition*, volume 42, pages 962–977, 2009.

[110] Ognjen Rudovic, Ioannis Patras, and Maja Pantic. Coupled gauusian process regression for pose-invariant facial expression recognition. In *ECCV*, pages 350–363, 2010.

[111] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models-their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, January 1995.

[112] Y. Chang, C. Hu, R. Feris, and M. Turk. Manifold based analysis of facial expression. *Image and Vision Computing*, 24(6):605–614, 2006.

[113] Chan-Su Lee and Ahmed M. Elgammal. Nonlinear shape and appearance models for facial expression analysis and synthesis. In *ICPR06*, pages II: 497–502, 2006.

[114] E. Begelfor and M. Werman. Affine invariance revisited. In *CVPR*, pages 2087–2094, 2006.

[115] D. G. Kendall. Shape manifolds, procrustean metrics, and complex projective spaces. *Bull. London Math. Soc.*, 16:18–121, 1984.

[116] Abhishek Bhattacharya and Rabi Bhattacharya. Nonparametric statistics on manifolds with applications to shape spaces. In *IMS Collections*, volume 3, pages 282–301. 2008.

[117] P. Turaga, A. Veeraraghavan, and R. Chellappa. Statistical analysis on stiefel and grassmann manifolds with applications in computer vision. In *CVPR*, 2008.

[118] Yasuko Chikuse. *Statistics on Special Manifolds*. Springer, 2003.

[119] Alan Edelman, Tomas A. Arias, and Steven T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl*, 20:303–353, 1998.

[120] V. Patrangenaru, X. Liu, and S. Sugathadasa. A nonparametric approach to 3D shape analysis from digital camera image - I. *Journal of Multivariate Analysis*, 101:11–31, January 2010.

[121] A. Savran, N. Alyüz, H. Dibeklioğlu, O. Çeliktutan, B. Gökberk, B. Sankur, and L. Akarun. Bosphorus database for 3D face analysis. In *Workshop on Biometrics and Identity Management (BIOID)*, 2008.

[122] Takeo Kanade, Jeffrey F. Cohn, and Yingli Tian. Comprehensive database for facial expression analysis. In *FGR*, pages 46–53, 2000.

[123] G. Lipori. Manual annotations of facial fiducial points on the Cohn-Kanade database. LAIV laboratory, University of Milan, web url: http://lipori.dsi.unimi.it/download.html.

[124] P. Thomas Fletcher, Conglin Lu, Stephen M. Pizer, and Sarang Joshi. Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE Transactions on Medical Imaging*, 23:995–1005, 2004.

[125] Xavier Pennec. Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements. *Journal of Mathematical Imaging and Vision*, 25(1):127–154, 2006.

[126] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE*, 290:2323–2326, 2000.

[127] Ashok Veeraraghavan, Anij Srivastava, Amit K. Roy Chowdhury, and Rama Chellappa. Rate-invariant recognition of humans and their activities. *IEEE Transactions on Image Processing*, 18(6):1326–1339, 2009.

[128] P. Lucey, J.F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The extended cohn-kanade dataset (CK+): A complete dataset for action unit and emotion-specified expression. In *CVPR Workshop*, 2010.

[129] Jihun Ham and Daniel D. Lee. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *ICML*, pages 376–383, 2008.

[130] Zicheng Liu, Ying Shan, and Zhengyou Zhang. Expressive expression mapping with ratio images. In *SIGGRAPH*, pages 271–276, 2001.

[131] Zhihong Zeng, Maja Pantic, Glenn I. Roisman, and Thomas S. Huang. A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *TPAMI*, 2009.

[132] M.F. Valstar, M. Mehu, Bihan Jiang, M. Pantic, and K. Scherer. Meta-analysis of the first facial expression recognition challenge meta-analysis of the first facial expression recognition challenge meta-analysis of the first facial expression recognition challenge. *IEEE Transactions on Systems, Man, and Cybernetics*, 2012.

[133] M.F. Valstar and M. Pantic. Biologically vs. logic inspired encoding of facial actions and emotions in video. In *IEEE Int'l. Conf. on Multimedia and Expo*, 2006.

[134] Peng Yang, Qingshan Liu, and Metaxas Dimitris. Exploring facial expression with compositional features. In *CVPR*, 2010.

[135] Xi Chen, Qihang Lin, Seyoung Kim, Jaime G. Carbonell, and Eric P. Xing. An efficient proximal gradient method for general structured sparse learning. *Journal of Machine Learning*, 2011.

[136] Gwen Littlewort, Jacob Whitehill, Tingfan Wu, Ian R. Fasel, Mark G. Frank, Javier R. Movellan, and Marian Stewart Bartlett. The computer expression recognition toolbox (cert). In *FG*, 2011.

[137] Yunfeng Zhu, F. De la Torre, Jeffrey F. Cohn, and Yu-Jin Zhang. Dynamic cascades with bidirectional bootstrapping for action unit detection in spontaneous facial behavior. *TAC*, 2011.

[138] Peng Yang, Qingshan Liu, Xinyi Cui, and D.N. Metaxas. Facial expression recognition using encoded dynamic features. *CVPR*, 2008.

[139] J. Xiao, S. Baker, I. Matthews, and T. Kanade. Real-time combined 2D+3D active appearance models. In *CVPR*, pages 535–542, 2004.

[140] Y. Chang, C. Hu, and M. Turk. Probabilistic expression analysis on manifolds. In *CVPR*, 2004.

[141] S. Taheri, P. Turaga, and R. Chellappa. Towards view-invariant expression analysis using analytic shape manifolds. In *Automatic Face and Gesture Recognition (FG)*, 2011.

[142] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-SVD: Design of dictionaries for sparse representation. *Trans. on Signal Processing*, 2006.

[143] Qiang Qiu, Zhuolin Jiang, and Rama Chellappa. Sparse dictionary-based representation and recognition of action attributes. In *ICCV*, 2011.

[144] Zhuolin Jiang, Zhe Lin, and Larry S. Davis. Learning a discriminative dictionary for sparse coding via label consistent K-SVD. In *CVPR*, 2011.

[145] M.H. Mahoor, M. Zhou, K.L. Veon, M. Mavadati, and J.F Cohen. Facial action unit recognition with sparse representation. In *FG*, 2011.

[146] Guoshen Yu, Guillermo Sapiro, and Stéphane Mallat Mallat. Solving inverse problems with piecewise linear estimators: From Gaussian mixture models to structured sparsity. *CoRR*, 2010.

[147] Peng Yang, Qingshan Liu, and Dimitris N. Metaxas. RankBoost with L1 regularization for facial expression recognition and intensity estimation. In *ICCV*, 2009.

[148] Jerome H. Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1), 2010.

[149] Lukas Meier, Sara Van De Geer, and Peter Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society*, 2008.

[150] M. Lobo, L. Vandenberghe, S. Boyd, , and H. Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228, 1998.

[151] X. Chen, Q. Lin, S. Kim, J. Pena, J. G. Carbonell, and E. P. Xing. An efficient proximal-gradient method for single and multi-task regression with structured sparsity. Technical report, CMU, 2010.

[152] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal Imaging Sci.*, 2(1):183–202, 2009.

[153] Shenghua Gao, Liang-Tien Chia, and Ivor W. Tsang. Multi-layer group sparse coding - for concurrent image classification and annotation. In *CVPR*, 2011.

[154] M.S. Bartlett, G. Littlewort, M. Frank, C. Lainscsek, I. Fasel, and J. Movellan. Recognizing facial expression: Machine learning and application to spontaneous behavior. In *CVPR*, 2005.

[155] S. Zafeiriou and M. Petrou. Nonlinear non-negative component analysis algorithms. *TIP*, 2010.

[156] W. Zhao, R. Chellappa, A. Rosenfeld, and P. J. Phillips. Face recognition: A literature survey. *ACM Computing Surveys*, pages 399–458, 2003.

[157] A. Jorstad, D. Jacobs, and A. Trouv. A deformation and lighting insensitive metric for face recognition based on dense correspondences. In *CVPR*, pages 2353–2360, Jun. 2011.

[158] I. Naseem, R. Togneri, and M. Bennamoun. Linear regression for face recognition. *TPAMI*, 32:2106– 2112, 2010.

[159] B. Amberg, R. Knothe, and T. Vetter. Expression-invariant 3D face recognition with a morphable model. In *FG*, 2008.

[160] P. Nagesh and B. Li. A compressive sensing approach for expression-invariant face recognition. In *CVPR*, 2009.

[161] P.-H. Tsai and T. Jan. Expression-invariant face recognition system using subspace model analysis. In *IEEE Conf. Systems, Man and Cybernetics*, volume 2, 2005.

[162] Antonio Colmenarez, Brendan Frey, and Thomas S. Huang. A probabilistic framework for embedded face and facial expression recognition. In *CVPR*, 1999.

[163] Xiaoxing Li, Greg Mori, and Hao Zhang. Expression-invariant face recognition with expression classification. In *Canadian Conf. on Computer and Robot Vision*, 2006.

[164] J. Tenenbaum and W. Freeman. Separating style and content with bilinear models. In *Neural Computation*, 2000.

[165] M. A. O. Vasilescu and D. Terzopoulos. Multilinear subspace analysis of image ensembles. In *CVPR*, 2003.

[166] H. Wang and N. Ahuja. Facial expression decomposition. In *ICCV*, 2003.

[167] I. Mpiperis, S. Malassiotis, and M.G. Strintzis. Bilinear models for 3D face and facial expression recognition. *IEEE Transactions on Information Forensics and Security*, 3, 2008.

[168] Chan-Su Lee and Ahmed Elgammal. Facial expression analysis using nonlinear decomposable generative model. In *FG*, 2005.

[169] S.Z. Li and A.K. Jain. *Handbook of face recognition*. Springer, 2005.

[170] D. O. Gorodnichy. Video-based framework for face recognition in video. In *Can. Conf. Computer and Robot Vision*, 2005.

[171] U. Park, H. Chen, and A. Jain. 3D model-assisted face recognition in video. In *Can. Conf. Computer and Robot Vision*, 2005.

[172] A. M. Martinez. Recognizing expression variant faces from a single sample image per class. In *CVPR*, 2003.

[173] C.K. Hsieh, S.H. Lai, and Y.C. Chen. Expression-invariant face recognition with constrained optical flow warping. *Transaction on Multimedia*, 11, 2009.

[174] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *TPAMI*, 2008.

[175] J. Huang, X. Huang, and D. Metaxas. Simultaneous image transformation and sparse representation recovery. In *CVPR*, 2008.

[176] Z. Ying, Z. Wang, and M. W. Huang. Facial expression recognition based on fusion of sparse representation. *Lecture Notes in Computer Science*, 6216, 2010.

[177] M.S. Bartlett, G.C. Littlewort, M.G. Frank, C. Lainscsek, I. Fasel, and J.R. Movellan. Automatic recognition of facial actions in spontaneous expressions. *Journal of Multimedia*, 2006.

[178] C. Shan, S. Gong, and P.W. McOwan. Robust facial expression recognition using local binary patterns. In *ICIP*, 2005.

[179] Ying-Li Tian, Takeo Kanade, and Jeffrey Cohn. Recognizing lower face action units for facial expression analysis. In *FG*, pages 484 – 490, March 2000.

[180] M. Pantic and I. Patras. Dynamics of facial expression: Recognition of facial actions and their temporal segments from face profile image sequences. *SMC-B*, 36(2):433–449, 2006.

[181] G. Ford. Fully automatic coding of basic expressions from video. Technical report, Machine Perception Lab, Institute for Neural Computing, UCSD, 2002.

[182] Zhen Wen and T.S. Huang. Capturing subtle facial motions in 3D face tracking. In *ICCV*, 2003.

[183] T. Cootes, C. Taylor, D. Cooper, and J. Graham. Active shape models - their training and application. *Comput. Vis. Image Understand*, 61:18–23, 1995.

[184] T. Cootes, G. J. Edwards, and C. Taylor. Active appearance models. *TPAMI*, 23:681–685, 2001.

[185] J.-L Starck, Michael Elad, and David L. Donoho. Image decomposition via the combination of sparse representations and a variational approach. *IEEE Transactions on Image Processing*, 14(10):1570–1582, 2005.

[186] Sylvain Sardy, Andrew G. Bruce, and Paul Tseng. Block coordinate relaxation methods for nonparametric wavelet denoising. *Journal of Computational and Graphical Statistics*, 9(2):361–379.

[187] M. Elad. *Sparse and Redundant Representations: From theory to applications in Signal and Image processing.* Springer, 2010.

[188] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Commun. Pure Appl. Math.*, 57:1413–1541, 2004.

[189] M. Zibulevsky and M. Elad. L1-L2 optimization in signal and image processing. *Signal Processing Magazine, IEEE*, 27(3):76 –88, May 2010.

[190] V. Chandrasekaran, S. Sanghavi, P. Parrilo, and A. Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2):572596, 2011.

[191] M. Aharon, M. Elad, and A. M. Bruckstein. The K-SVD: an algorithm for designing of overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Process.*, 54(11):4311–4322, 2006.

[192] S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Sci. Comp.*, 20(1):33–61, 1998.

[193] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. *1993 Conference Record of the 27th Asilomar Conference on Signals, Systems and Computers*, pages 40–44, Pacific Grove, CA, 1993.

[194] J. Friedman, T. Hastie, and R. Tibshirani. A note on the group Lasso and a sparse group Lasso. Technical report, Department of Statistics, Stanford University, 2010.

[195] J. Liu, S. Ji, and J. Ye. *SLEP: Sparse Learning with Efficient Projections.* Arizona State University, 2009.

[196] M. Yang, L. Zhang, X. Feng, and D. Zhang. Fisher discrimination dictionary learning for sparse representation. In *ICCV*, 2011.

[197] T. Kanade, J. F. Cohn, and Y. Tian. Comprehensive database for facial expression analysis. In *FG*, 2000.

[198] Ruonan Li and Rama Chellappa. Group motion segmentation using a spatio-temporal driving force model. In *CVPR*, 2010.

[199] C. Grimes and D. Donoho. Image manifolds which are isometric to euclidean space. *Math. image and Vision*, 23(1):5–24, 2005.

[200] V. Delaitre, I. Laptev, and J. Sivic. Recognizing human actions in still images: A study of bag-of-features and part-based representations. In *BMVC*, 2010.

[201] N. Ikizler-Cinbis, R. G. Cinbis, and S. Sclaroff. Learning actions from the web. In *ICCV*, 2009.

[202] B. Yao, A. Khosla, and L. Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *CVPR*, 2011.

[203] B. Yao, X. Jiang, A. Khosla, A.L. Lin, L. Guibas, and L. Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *ICCV*, 2011.

[204] Chan su Lee and Ahmed Elgammal. Facial expression analysis using nonlinear decomposable generative models. In *International Workshop on Analysis and Modeling of Faces and Gestures*, 2005.

[205] Chan su Lee and Ahmed Elgammal. Separating style and content on a nonlinear manifold. In *CVPR*, 2004.